



**University of Thessaly**

**Department of Mechanical Engineering**

THESIS

**A framework for modal identification and finite element model  
updating using output-only vibration measurements**

**ARVANITIS SPYRIDON**

Volos

July 2018

## Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα της διπλωματικής μου εργασίας τον Καθηγητή κ. Κ. Παπαδημητρίου, για την πολύτιμη βοήθεια και καθοδήγησή του κατά τη διάρκεια εκπόνησης της εργασίας. Επιπλέον, οφείλω ευχαριστίες στα υπόλοιπα δύο μέλη της τριμελούς εξεταστικής επιτροπής, τον Καθηγητή κ. Σπύρο Καραμάνο και την Διδάσκουσα κα. Ελένη Καμούτση. Επί προσθέτως θα ήθελα να ευχαριστήσω τον κύριο Πανέτσο για τα δεδομένα και τις πληροφορίες που μου έστειλε ώστε να μπορέσω να προχωρήσω με την εργασία καθώς και τον Κώστα Αργύρη που μας παραχώρησε τον πρόγραμμα του ώστε να μπορέσουμε να δουλέψουμε.

Ακόμη, θα ήθελα να απευθύνω ένα μεγάλο ευχαριστώ στους φίλους και στην οικογένεια μου, για την αμέριστη συμπαράσταση και υποστήριξη τους καθ' όλη τη διάρκεια των σπουδών μου.

## **Abstract**

This thesis is intended to illustrate the process followed to achieve the updating of finite element models of structures. The way to obtain and calculate the data to be used as inputs from a model updating program is presented. Such data are the mass and stiffness matrices of the finite element model, as well as the identifiable frequencies and modeshapes that arise from the real measurements analysis. The calculation and the saving of the required matrices of the model are made with the help of a commercial finite element program named SAP2000 and with the creation of a code in matlab. The code interacts with the commercial finite element program in order to parameterize the different parts of the model and extract the necessary stiffness and mass matrices at desired substructuring level. The frequencies and modeshapes result from the analysis of measurements via a Modal identification program and are compared with those of the finite element model's. Finally, reference is made to the model reduction method and to the procedure to be followed in order to achieve it. With the help of a code created in matlab, the input data are calculated in order to be used in a model reduction program. The framework is demonstrated using an application on a bridge structure.

## CONTENTS

|           |   |           |
|-----------|---|-----------|
| <b>1.</b> | <b>Introduction .....</b>   | <b>6</b>  |
| 1.1.      | The need for Model updating.....  | 6         |
| 1.2.      | The importance of Model reduction .....   | 8         |
| 1.3.      | Outline of this work .....  | 9         |
| <b>2.</b> | <b>Model Updating .....</b>   | <b>11</b> |
| 2.1       | Introduction .....  | 11        |
| 2.2       | Structural Model Class .....  | 12        |
| 2.3       | Substructuring and FE Model Parameterization .....  | 12        |
| 2.4       | Linear Relation Between Model Matrices and Parameters.....  | 13        |
| 2.5       | An Introduction to Model Updating Based on Modal Properties .....   | 13        |
| 2.5.1     | Introduction.....   | 13        |
| 2.5.2     | Formulation.....  | 14        |
| <b>3.</b> | <b>The process of extraction of model's stiffness and mass matrices(presentation of the sotware) .....</b>        | <b>15</b> |
| 3.1       | The FE Model.....   | 15        |
| 3.2       | Software for acquisition of stiffness and mass matrices of model's components.....                                | 17        |
| 3.2.1     | The acquisition of global stiffness matrix(before the parameterization process), $K_o, K_i$ .....                 | 19        |
| 3.2.2     | The acquisition of global mass matrix(before the parameterization process), $M_o, M_i$ .....                      | 25        |
| <b>4.</b> | <b>Modal Identification of a bridge using vibration measurements.....</b>   | <b>28</b> |
| 4.1       | Measurements data.....  | 28        |
| 4.2       | Identification of mode shapes and modal frequencies based on FE model.....  | 30        |
| <b>5.</b> | <b>Model reduction.....</b>   | <b>56</b> |
| 5.1       | Introduction.....   | 56        |
| 5.2       | The kind of the degrees of freedom of a component.....  | 56        |
| 5.3       | Calculation of the necessary degrees of freedom needed in model reduction (the presentation of the software)..... | 57        |
| 5.4       | Conclusion.....   | 60        |

|                             |           |
|-----------------------------|-----------|
| <b>6. Conclusions .....</b> | <b>60</b> |
| <b>Appendix A.....</b>      | <b>61</b> |
| <b>Appendix B.....</b>      | <b>69</b> |
| <b>Appendix C.....</b>      | <b>70</b> |
| <b>Appendix D.....</b>      | <b>81</b> |
| <b>Bibliography.....</b>    | <b>95</b> |

## LIST OF FIGURES

|   |    |
|---|----|
| Figure 3.1. The bridge Model made in SAP2000 .....  | 15 |
| Figure 3.2. Structure's components .....  | 16 |
| figure 4.1. nm008 configuration .....   | 29 |
| Figure 4.2. nm009 configuration.....  | 29 |
| Figure 4.3. The location of the sensors .....   | 30 |
| Figure 4.4. The measured frequencies from time history nm008 .....                            | 32 |
| Figure 4.5. Mode shape based on measurements ( $f=1.13\text{Hz}$ ) .....                      | 32 |
| Figure 4.6. Oscillation of the bridge model for $f=1.076257\text{Hz}$ (front view) .....      | 33 |
| Figure 4.7. Oscillation of the bridge model for $f=1.076257\text{Hz}$ (top view) .....        | 33 |
| Figure 4.8. Mode shape based on measurements ( $f=1.5004\text{ Hz}$ ) .....                   | 34 |
| Figure 4.9. Oscillation of the bridge model for $f=1.283569\text{Hz}$ (front view) .....      | 34 |
| Figure 4.10. Oscillation of the bridge model for $f=1.283569\text{ Hz}$ (cross section) ..... | 35 |
| Figure 4.11. Mode shape based on measurements ( $f=1.13\text{ Hz}$ ).....                     | 35 |
| Figure 4.12. Oscillation of the bridge model for $f=1.915901\text{ Hz}$ (original view) ..... | 36 |
| Figure 4.13. Oscillation of the bridge model for $f=1.915901\text{ Hz}$ (cross section) ..... | 36 |
| Figure 4.14. Oscillation of the bridge model for $f=1.1993078\text{ Hz}$ (front view) .....   | 37 |
| Figure 4.15. Oscillation of the bridge model for $f=1.1993078\text{Hz}$ (top view) .....      | 37 |
| Figure 4.16. Mode shape based on measurements ( $f=2.5511\text{ Hz}$ ) .....                  | 38 |
| Figure 4.17. Oscillation of the bridge model for $f=2.462727\text{Hz}$ (front view).....      | 38 |
| Figure 4.18. Oscillation of the bridge model for $f=2.462727\text{ Hz}$ (Cross Section) ..... | 39 |
| Figure 4.19. Mode shape based on measurements ( $f=3.2928\text{ Hz}$ ) .....                  | 39 |
| Figure 4.20. Oscillation of the bridge model for $f=3.148780\text{Hz}$ (front view).....      | 40 |
| Figure 4.21. Oscillation of the bridge model for $f=3.148780\text{Hz}$ (cross section) .....  | 40 |
| Figure 4.22. Mode shape based on measurements ( $f=3.9256\text{ Hz}$ ) .....                  | 41 |
| Figure 4.23. Oscillation of the bridge model for $f=3.644488\text{ Hz}$ (original view) ..... | 41 |
| Figure 4.24. Oscillation of the bridge model for $f=3.644488\text{ Hz}$ (cross section) ..... | 42 |
| Figure 4.25. Oscillation of the bridge model for $f=3.644488\text{ Hz}$ (front view).....     | 42 |
| Figure 4.26. Mode shape based on measurements ( $f=4.4405\text{ Hz}$ ) .....                  | 43 |
| Figure 4.27. Oscillation of the bridge model for $f=3.8759\text{ Hz}$ (front view) .....      | 43 |

|   |    |
|---|----|
| Figure 4.28. Oscillation of the bridge model for $f=3.8759$ Hz (cross section) .....  | 44 |
| Figure 4.29. Mode shape based on measurements ( $f=4.724$ Hz) .....                   | 44 |
| Figure 4.30. Oscillation of the bridge model for $f=4.48316$ (front view) .....       | 45 |
| Figure 4.31. Oscillation of the bridge model for $f=4.48316$ (top view) .....         | 45 |
| Figure 4.32. Oscillation of the bridge model for $f=4.48316$ (original view) .....    | 45 |
| Figure 4.33. Mode shape based on measurements ( $f=5.1036$ Hz) .....                  | 46 |
| Figure 4.34. Oscillation of the bridge model for $f=4.54937$ Hz (front view) .....    | 46 |
| Figure 4.35. Oscillation of the bridge model for $f=4.54937$ Hz (original view) ..... | 47 |
| Figure 4.36. Oscillation of the bridge model for $f=4.54937$ Hz (cross section) ..... | 47 |
| figure 4.37. Mode shape based on measurements ( $f=5.4675$ Hz) .....                  | 48 |
| Figure 4.38. Oscillation of the bridge model for $f=4.76772$ Hz (cross section) ..... | 48 |
| Figure 4.39. Oscillation of the bridge model for $f=4.76772$ Hz (front view) .....    | 48 |
| Figure 4.40. Identifiable mode shapes of the arced part .....                         | 49 |
| Figure 4.41. Mode shape for $f=5.91396$ Hz.....                                       | 50 |
| Figure 4.42. Mode shape for $f=5.91401$ Hz.....                                       | 50 |
| Figure 4.43. Mode shape for $f=6.02791$ Hz.....                                       | 50 |
| Figure 4.44. Mode shape for $f=6.05633$ Hz.....                                       | 50 |
| Figure 4.45. Mode shape for $f=6.13507$ Hz.....                                       | 51 |
| Figure 4.46. . Mode shape for $f=6.14033$ Hz .....                                    | 51 |
| Figure 4.47. Mode shape for $f=6.15619$ Hz .....                                      | 51 |
| Figure 4.48. Mode shape based on measurements ( $f=7.4562$ Hz) .....                  | 52 |
| Figure 4.49. Oscillation of the bridge model for $f=6.71095$ Hz (cross section) ..... | 52 |
| Figure 4.50. Oscillation of the bridge model for $f=6.71095$ Hz (front view) .....    | 52 |
| Figure 4.51. Oscillation of the bridge model for $f=6.71095$ Hz (front view) .....    | 53 |
| Figure 4.52. Oscillation of the bridge model for $f=7.14691$ Hz (front view).....     | 53 |
| Figure 4.53. Oscillation of the bridge model for $f=7.14691$ Hz (original view).....  | 54 |
| Figure 5.1. The bridge Model.....   | 57 |

# **1. Introduction**

## **1.1. The need for Model updating**

In modern analysis of structural dynamics, much effort is devoted to the derivation of accurate models of structures. Availability of an accurate dynamic finite element model of a structure is very important to design engineers as it allows them to improve the dynamic design of the structure at computer level resulting in an optimized design apart from savings in terms of money and time. The first step is the derivation of an analytical model, usually finite element model, based on the assumed equations of motion. But there may be some inaccuracies or uncertainties that may be associated with a finite element model. The discretisation error, arising due to approximation of a continuous structure by a finite number of individual elements, is inherent to the finite element technique. Other inaccuracies may be due to the assumptions and simplifications made by the analyst with regards to the choice of elements, modeling of boundary conditions, joints, etc.

These assumptions and simplifications have as a result that when tests are performed to validate the analytical model, inevitably their results, notably natural frequencies and modeshapes, do not coincide with the expected results from the theoretic model. Clearly one would like to have a better model, based on both the theoretical and the experimental results. In order to face the problems of inaccuracy in analytical models, researchers have turned their attention to the development of modeling methods based on experimental observation. This area, known as system identification, has been particularly vibrant in the control engineering community over the past 40 years. The model to be identified may be a parametric or a non-parametric model and, in addition, it may be non-linear. Once the model structure and model order have been chosen, the estimation of parameters follows.

In structural dynamics, experimental modal analysis may be considered as a special area of system identification for the determination of modal data (natural frequencies, mode shapes, generalized masses and loss factors) from vibration tests. The modal testing and modal extraction methods (Ewins 2000, Mc Connell

1995) are also well developed for obtaining a reliable estimate of the modal data. Given the availability of an accurate data acquisition and measuring equipment the measured test data, though may not be precise, is generally considered to be more accurate than analytical model predictions. This has formed the basis for adjustment or correction of a finite element model, in the light of measured test data, which is referred as model updating. The purpose of model updating is to modify the mass, stiffness and damping parameters of the numerical model in order to obtain better agreement between numerical results and test data. If the updated model is to be used predictively, for untested loading conditions or modified structural configurations, then it is important that the improved agreement in results is achieved by correcting the inaccurate modelling assumptions and not by making other (physically meaningless) alterations to the model. Comprehensive reviews of structural parameter identification methods can be found in (Mottershead and Friswell 1993; Doebling et al. 1996).

Structural model updating is an inverse problem according to which a model of a structure, usually a finite element model, is adjusted so that either the calculated time histories, frequency response functions, or modal parameters best match the corresponding quantities measured or identified from the test data. This inverse process aims at providing updated models and their corresponding uncertainties based on the data. These updated models are expected to give more accurate response predictions to future loadings, as well as allow for an estimation of the uncertainties associated with such response predictions. In practice, the inverse problem of model updating is usually ill-conditioned due to insensitivity of the response to changes in the model parameters, and non-unique (Udwadia and Sharma 1978; Berman 1989; Katafygiotis and Beck 1998; Katafygiotis et al. 2000) because of insufficient available data relative to the large number of model parameter needed to describe the desired model. Structural model updating is an inverse problem according to which a model of a structure, usually a finite element model, is adjusted so that either the calculated time histories, frequency response functions, or modal parameters best match the corresponding quantities measured or identified from the test data. This inverse process aims at providing updated models and their corresponding uncertainties based on the data. These updated models are expected to give more accurate response predictions to future loadings,



as well as allow for an estimation of the uncertainties associated with such response predictions.

## **1.2. The importance of Model reduction**

There are several definitions of model order reduction, and it depends on the context which one is preferred. Originally, MOR was developed in the area of systems and control theory, which studies properties of dynamical systems in application for reducing their complexity, while preserving their input-output behavior as much as possible. The field has also been taken up by numerical mathematicians. Nowadays, model order reduction is a flourishing field of research, both in systems and control theory and in numerical analysis. This has a very healthy effect on MOR as a whole, bringing together different techniques and different points of view, pushing the field forward rapidly.

Such simplification is needed in order to perform simulations within an acceptable amount of time and limited storage capacity, but with reliable outcome. In some cases, we would even like to have on-line predictions of the behaviour with acceptable computational speed, in order to be able to perform optimizations of processes and products.

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

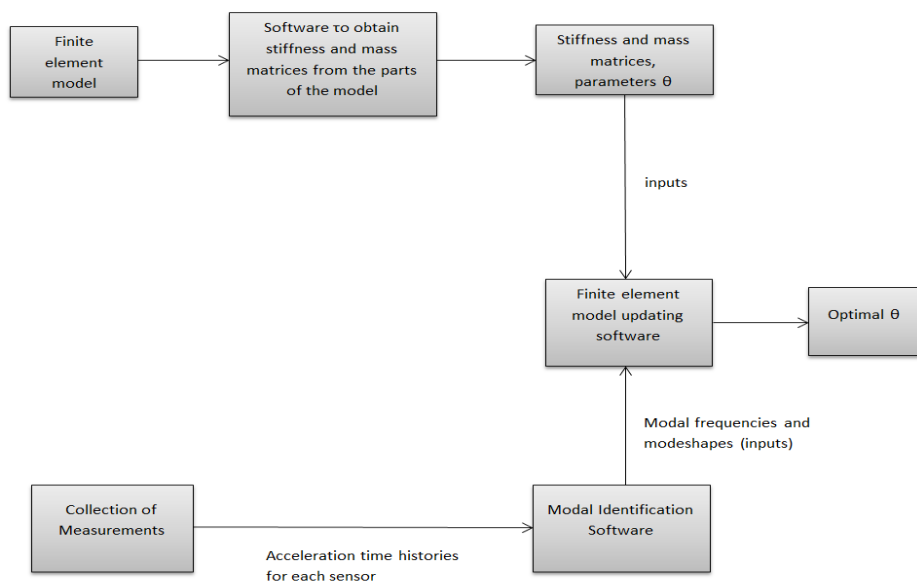
### 1.3. Outline of this work

The study carried out in order to device this work, was aimed at obtaining the necessary information for the execution of the model updating . To achieve model updating, the mass and stiffness matrices coming from the analysis of a model in SAP2000 are required as well as the modeshapes and model frequencies of actual construction.

Initially, the mass and stiffness matrices are calculated and extracted from the different parts of the finite element model. Achieving this goal is done by creating a matlab code that interacts with the SAP2000 finite element program and extracts and stores the model's matrices. The parameterization of the various model members is made by introducing parameters  $\theta$  that are related to the mass and stiffness matrices and are the ones that will be valued in model updating so that their optimal price improves the model.

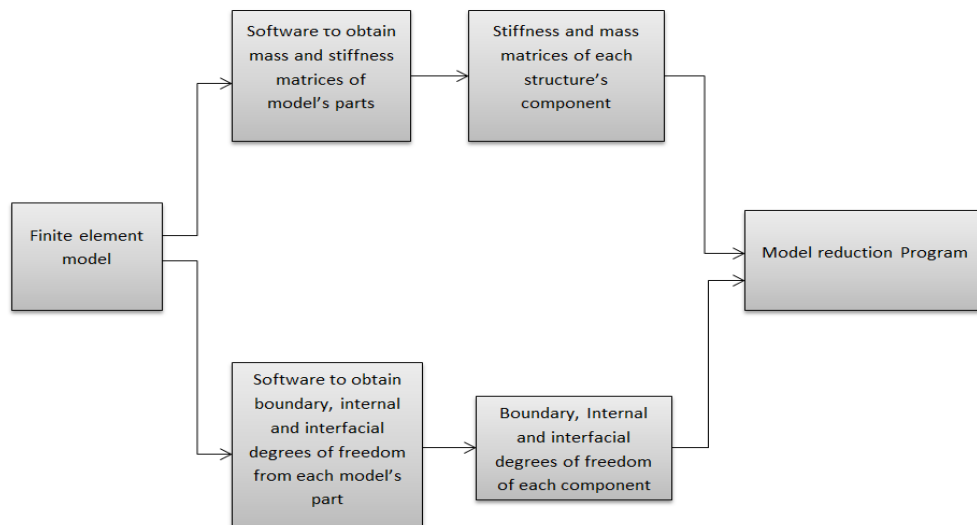
Afterwards, from the measurements that have been given by the real bridge model and specifically from the its deck, a range of frequencies and modeshapes has been found. This was done with the help of a program that was given to us. Through this range of measured frequencies and modeshapes, we identified these that match to these of the finite element model.

Then, a flow diagram illustrates the model updating prepared process and the data needed to make it.



Finally, reference is made to the model reduction method. Reference is made to its utility as well as to the data it is necessary to achieve model reduction. These data are related with the mass and stiffness matrices of each section of the model as well as with the internal, boundary and interfacial degrees of freedom of each component of the structure.

The process followed to obtain the input data that are going to be used in a model reduction program is depicted by the below flow chart.



Subsequently, the chapters 3, 4, 5, show the ways to acquire the input data in terms of model updating and model reduction respectively.

## **2. Model Updating**

### **2.1 Introduction**

Structural design and analysis generally requires a mathematical model representing the physical behaviour of the structure. The finite element (FE) method is the most appropriate tool for such modeling in structural engineering today. However it is often observed that the initial FE model is a poor reflection of structure, particularly in the field of structural dynamics. Inaccuracies arise because of a number of simplifying assumptions and idealizations that have to be made in FE modeling. In the recent years various model updating methods have been developed to update the initial model using experimental data. If accurately measured data are available then these data could be used to improve the numerical model in general, and the uncertain parameters of the model in particular.

The methods may be split according to the type of measured data they use and model parameters that are updated. The measured data may be in form of frequency response function (FRF) data or natural frequencies and mode shapes. The updating process may estimate physical parameters, complete mass, damping and stiffness matrices or groups of individual matrix elements. Other aspects of model updating, such as parameter uniqueness, efficient computation, parameterization, ill-conditioning and the use of incomplete data, are being investigated. The measured data will always be incomplete because the measurements will only be taken at a relatively small number of locations and over a limited frequency range.

## 2.2 Structural Model Class

Consider a parameterized class of linear structural models (e.g. a class of finite element models) used to model the dynamic behaviour of the structure. The structural model class involves a set of model parameters  $\theta$ . The equation of motion of such systems is:

$$M(\theta)\ddot{u}(t) + C(\theta)\dot{u}(t) + K(\theta)u(t) = f(t)$$

Where  $M(\theta), C(\theta), K(\theta)$  are the global mass, damping and stiffness matrices respectively.  $u(t)$  is the displacements and  $f(t)$  is the vector of forces.

The parameter set  $\theta$  is the set of free model parameters to be estimated using the measured data. The parameter set  $\theta$  is usually associated with geometrical, material, stiffness or mass properties and boundary conditions.

Here we associate  $\theta$  with modulus of elasticity and density of the model.

## 2.3 Substructuring and FE Model Parameterization

Using finite element model analysis, one derives the element stiffness and the mass matrices, the stiffness and the mass matrices of the substructures formed by a group of elements, and finally the global stiffness and the mass matrices. These matrices depend on the properties of the structure, like modulus of elasticity and mass density. These properties are selected for updating and are included in  $\theta$ .

In finite element analysis, the global stiffness and mass matrices formulation are taking the following form:

$$K(\theta) = \sum_{i=1}^N K_i^l(\theta) \quad \text{and} \quad M(\theta) = \sum_{i=1}^N M_i^l(\theta)$$

$K_i^l(\theta), M_i^l(\theta), N$  are the local stiffness, mass matrices and number of structure's elements respectively.

The linear relation between global stiffness and mass matrices and  $\theta$  is listed below.

## 2.4 Linear Relation Between Model Matrices and Parameters

$$K(\theta) = K_0 + \sum_{i=1}^{N_\theta} K_i(\theta) \quad \text{and} \quad M(\theta) = M_0 + \sum_{i=1}^{N_\theta} M_i(\theta)$$

$K_i$  and  $M_i$  are assembled from element stiffness and mass matrices that depend linearly on  $\theta$ .  $K_0$  and  $M_0$  are assembled from element stiffness and mass matrices that do not depend on  $\theta$ .  $N_\theta$  is the number of parameters used to parameterize the model or the number of the model's parts that are parameterized.

$K_i$  and  $M_i$  as well as the set of  $\theta$  are used as inputs in model updating programs.

## 2.5 An Introduction to Model Updating Based on Modal Properties

### 2.5.1 Introduction

The problem of identifying the parameters of a structural model using dynamic data has received much attention because of its importance in structural model updating, structural health monitoring and structural control. The estimate of the parameter values involves uncertainties that are due to limitations of the mathematical models used to represent the behavior of the real structure, the presence of measurement error in the data, and insufficient excitation and response bandwidth. Structural identification and finite element model updating methodologies are often based on modal data. The optimal structural models resulting from such method can be used for response and reliability predictions, structural health monitoring and control.

### 2.5.2 Formulation

Let  $D = \{\hat{\omega}_r^{(k)}, \hat{\phi}_r^{(k)} \in R^{N_0}, r = 1, \dots, m, k = 1, \dots, N_D\}$  be the measured data from a structure, consisting of modal frequencies  $\hat{\omega}_r^{(k)}$  and modeshape components  $\hat{\phi}_r^{(k)}$  at  $N_0$  measured DOFs where  $m$  is the number of observed modes and  $N_D$  is the number of modal data sets available.

Consider a parameterized class of linear structural models  $M$  used to model the dynamic behaviour of the structure and let  $\theta \in R^{N_\theta}$  be the set of free structural model parameters to be identified using the measured modal data. Let also  $\{\omega_r(\theta), \phi_r(\theta) \in R^{N_d}, r = 1, \dots, m\}$  where  $N_d$  is the number of model degrees of freedom (DOF), be the predictions of the modal frequencies and modeshapes obtained for a particular value of the parameter set  $\theta$  by solving the eigenvalue problem corresponding to the model mass and stiffness matrices  $M(\theta)$  and  $K(\theta)$  respectively, that is,

$$[K(\theta) - \omega_r^2(\theta)M]\phi_r(\theta) = 0$$

The objective in a modal-based structural identification methodology is to estimate the values of the parameter set  $\theta$  so that the modal data  $\{\omega_r(\theta), \phi_r(\theta), r = 1, \dots, m\}$  predicted by the linear class of models best matches, in some sense, the experimentally obtained modal data in  $D$ . In this thesis we produced the necessary data in order to be used later for model updating.

### **3. The process of extraction of model's stiffness and mass matrices(presentation of the software)**

#### **3.1. The FE Model**

Based on the linear relation between model matrices and parameters , a bridge model made in SAP2000 is examined in order to be parameterized.

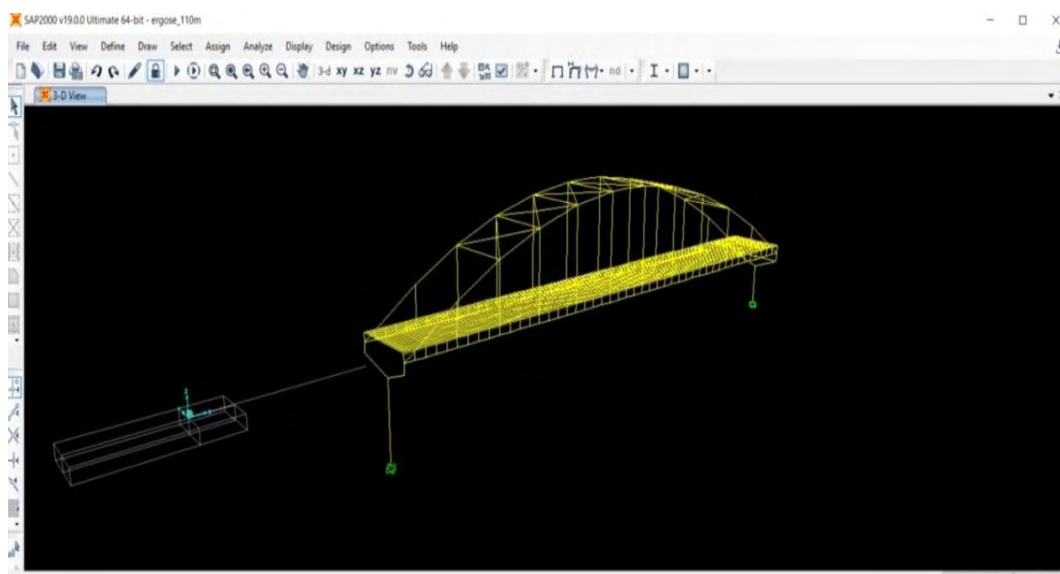


Figure 3.1. The bridge Model made in SAP2000

For this examination we separate the model into 4 components as it is shown in figure 3.2. Another separation can be done according to which parts we want to parameterize.



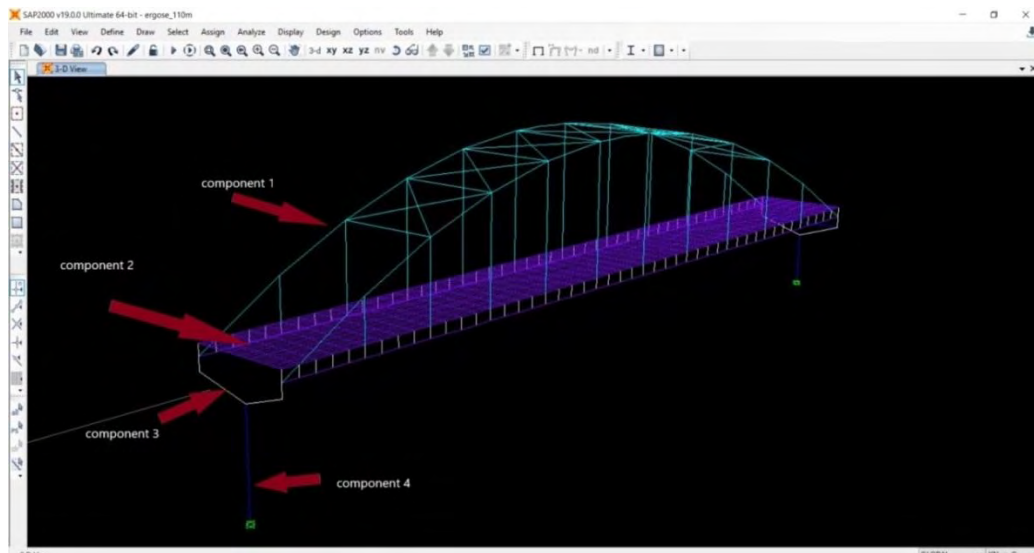


Figure 3.2. Structure's components

- Component 1:arched part (aquamarine color)
- Component 2:deck (purple color)
- Component 3:piers(white color)
- Component 4:columns(blue color)

The separation of the bridge's parts based on the materials that constitute these parts.

- Arched part: Material name A992Fy50(Sap2000's name)
- Deck: Material name MAT
- Piers: Material name concrete\_nomass
- Columns: Material name concrete\_abut

The definition of stiffness and mass matrices for a component entails that the parameters which are related to the other parameterized components will get zero value. For example, if we want to extract only deck's stiffness and mass matrices we have to zero modulus of elasticity and mass density from the other three components.

### 3.2. Software for acquisition of stiffness and mass matrices of model's components

Studying the SAP2000 bridge model, a code is created in order to extract the stiffness and mass matrices from each part of the structure.

The code is written in MATLAB (version 2015a) and creates an interface between Sap2000 and MATLAB. This interaction allows to change the model's properties through MATLAB.

- The interface is possible via a specific part of the code that is presented below:

```
% pass data to Sap2000 as one - dimensional arrays
feature ('COM_SafeArraySingleDim', 1);
% pass non - scalar arrays to Sap2000 API by reference
feature ('COM_PassSafeArrayByRef', 1);
%start the interface
SapObject = actxserver('CSI.SAP2000.API.SapObject');
% create Sap2000 object
SapModel = SapObject . SapModel ;
% start the application
ret = SapObject.ApplicationStart();
```

These commands are used in order to create a bridge between two programs. Thus the interface is done.

- Next step is to open the model that we are studying.

```
% give the name of Sap2000 model file
name=input('dwse onoma arxeiou: ' , 's')
% the file is on the desktop
ret=SapObject.SapModel.File.OpenFile(['C:\Users\Spyros\Desktop\',' ',name, '.sdb'])
```

- The SAP2000 file was loaded and the next step is to define the material properties for each component. The bridge model is constituted from four components, so the user has to define the material properties for each part.

```
% % % % % insert material properties for each material in
sap2000
number_mat=input('number of structure's material/components'
)
for i=1:number_mat
```

```

prompt = {'material name:', 'modulus of
elasticity:', 'coefficient of thermal expansion:', 'poisson
ratio:', 'density:'};
title = 'material properties';

r = inputdlg(prompt, title, [1 60]);
mat_name(i)=r{1};
E(i)= str2num(r{2});
A(i) = str2num(r{3});
Nu(i) = str2num(r{4});
density(i)=str2num(r{5});

end

```

The modulus of elasticity, the poisson ratio, the coefficient of thermal expansion and mass density are the necessary inputs. These inputs are based on values that were given by the maker of the model . The coefficient of thermal expansion and the poisson ratio are parameters that do not contribute to stiffness and mass matrices. However these parameters are necessary for the two programs interface.

- After that, these inputs are passed into the finite element model. We insert these data again in the model in case they were changed in a previous model analysis.

```

for k=1:number_mat

    Enew(k)=E(k);
    density_new(k)=density(k);
    %      pass material properties to Sap2000

    ret=SapModel.PropMaterial.SetMPIsotropic(char(mat_name{k}), E
(k), Nu(k), A(k));
    %      pass mass density to Sap2000

    ret=SapModel.PropMaterial.SetWeightAndMass(char(mat_name{k})
,1, density(k));
end

```

### 3.2.1 The acquisition of global stiffness matrix(before the parameterization process) , $K_0, K_i$

All the necessary information is known in order to start the parameterization and the extraction of the model's matrices. Initially, the global stiffness is calculated(the global mass matrix and  $M_i$  are calculated after the calculation of  $K_i$ ) through the model's analysis. At this point, it is worth mentioning that SAP2000 extracts stiffness and mass matrices as text files saved in the place where the sap file is. TXK file gives the lower half of the symmetric stiffness matrix and .TXM file gives the lower half of the symmetric mass matrix. There are three columns in the files. The first and the second one give the position of each stiffness/mass value (matrix's row and column). The third column is the stiffness/mass value. The software imports these data.

```
unlock the model
ret=SapModel.SetModelIsLocked(0);
% define load case in order to extract model matrices.
%We have to define a load case dead or modal for the
extraction of the matrices.
%that is asked by sap2000.There is no difference in the
matrices' values
ret = SapModel.Analyze.SetSolverOption_1(1, 0,1, 'DEAD')
% analyze the model
ret = SapModel.Analyze.RunAnalysis();
% import data from M file
Mfilename=[name, '.TXM'];

mass = importdata(Mfilename);
mass_matrix=mass.data;
% import data from Stiffness file
Sfilename=[name, '.TXK'];
s = importdata(Sfilename);
% Kt has 3 columns ,number of row and column,stiffness value
Kt=s.data;
```

Kt is a matrix which includes three columns. The third column indicates the stiffness value and the first two contains the number of the row and column to which the stiffness value corresponds. The data from global mass matrix also are imported in order to be used later.

- The global stiffness matrix does not have its normal form thus we have to bring it to its symmetric form. The process is easy while we seek for a non-zero element of the third column of the Kt matrix. When we detect it, we save its coordinates in matrices i and j respectively. If an element of the third column of Kt is zero the i,j matrices obtain a zero value. The i matrix contains the numbers of the rows and j the number of the columns for every non zero element. After that, in a matrix called Ktotal the stiffness values of Kt are imported in the positions which are indicated by the matrices i,j . We create the lower half of the matrix and afterwards we bring it to its normal form.

```
% unlock the model
ret=SapModel.SetModelIsLocked(0);
% define load case in order to extract model matrices.We
have to define a load case dead or modal for the extraction
of the matrices. that is asked by sap2000. There is no
difference in the matrices' values
ret = SapModel.Analyze.SetSolverOption_1(1, 0,1, 'DEAD')
% analyze the model
ret = SapModel.Analyze.RunAnalysis();
% import data from M file
Mfilename=[name, '.TXM'];

mass = importdata(Mfilename);
mass_matrix=mass.data;
% import data from Stiffness file
Sfilename=[name, '.TXK'];
s = importdata(Sfilename);
% Kt has 3 columns ,number of row and column, stiffness value

Kt=s.data;
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % bring the stiffness matrix to its
original form % % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% matrix length
mhkos=length(Kt)
% first 2 columns are the number of the row and column and
the third the stiffness value
% the last number of the column is the size of the matrix
given that the
% diagonal elements of the global stiffness matrix are non
zero thus the
% last element of the matrix describe its size
% read the coordinates of the last element
last1=Kt(end,2);
Ktotal=sparse(zeros(last1));

% we find the non zero elements and keep their coordinates
```

```

for p=1:mhkos
    if Kt(p,3)~=0
%         if Kt(p,3)=0 then the values of i(p),j(p) are zero
        i(p)=Kt(p,1);
        j(p)=Kt(p,2);
    end
end
%
for m=1:mhkos
%     if Kt(p,3)=0 then the values of i(p),j(p) are zero so
for the non
%     zero i,j (where the non zero values on stiffness are)
we save the
%     value of Kt in Ktotal
%     So we create the lower half of the symmetric stiffness
matrix
    if i(m) j(m) ~=0
        Ktotal(i(m),j(m))=Kt(m,3);
    end
end
% we bring stiffness matrix to its final form by add the
upper symmetric half without the diagonal
Ktotal=(Ktotal+(tril(Ktotal,-1)).')

```

- Next step is to define the materials/components that are going to be parameterized. The user decides which part/material wants to parameterize and this materials is saved in a matrix.

```

for c=1:number_mat
prompt = {'1:','2:'};
opts.Interpreter = 'tex';
answer = questdlg('Do you want to parameterized this
material/component?' ,mat_name{c}, 'YES','NO','.')
switch answer
    case 'YES'
        disp([answer ' done.'])
        ans = 1;
    case 'NO'
        disp([answer ' ok.'])
        ans = 2;
end
if ans ==1
%     the meter counts how many materials are going to be
%     parameterized.if meter<number_mat then Ko exists
    meter=meter+1
%     we keep the number of material which is going to be
parameterized
% this number corresponds to a material depending on the
order that the material is given by the user

    param(meter)=c
end
end

```

- After the parameterized materials are defined,  $K_0$  is calculated ( $K_0$  is assembled from element stiffness matrix that does not depend on  $\theta$ ). If the variable meter is equal to the number of structure's material then all components are going to be parameterized and there is no  $K_0$ . By contrast, if the variable meter is lower than the number of structure's materials then some parts were not parameterized. The param matrix corresponds to the parameterized materials. The value of modulus of elasticity of each parameterized material is set to zero in order to extract  $K_0$ .

```
for u=1:meter
%     E=0 for every parameterized material though this
sap2000 command

ret=SapModel.PropMaterial.SetMPIsotropic(char(mat_name{param
(u)}),0,Nu(param(u)),A(param(u)));
end
```

This methodology is executed if the below condition is satisfied

```
if meter~=number_mat&&meter~=0
```

- The model is analyzed in order to extract  $K_0$ . The parts with zero modulus of elasticity do not give a stiffness matrix. The extraction of  $K_0$ , as well as the figuration of its normal form, follow the same methodology with the calculation of global stiffness matrix.

```
% run model analysis on order to extract stiffness
matrix.this stiffness matrix is Ko
ret = SapModel.Analyze.RunAnalysis();
% import data from the stiffnes text files
Sfilename=[name, '.TXK'];
s = importdata(Sfilename);
K_o=s.data;
mhkos0=length(K_o);

Ko=sparse(zeros(last1));

%     bring Ko to its normal form
for dd=1:mhkos0
if K_o(dd,3)~=0
metrhths0=dd
i(dd)=K_o(dd,1);
j(dd)=K_o(dd,2);
end
end
for nn=1:metrhths0
if i(nn)j(nn)~=0
Ko(i(nn),j(nn))=K_o(nn,3);
```

```

        end
    end
    % final form of Ko
    Ko=(Ko+(tril(Ko,-1)).');

end

```

If all materials/components are chosen by the user to be parameterized then the  $K_0$  matrix is a zero matrix.

```

if meter==number_mat
    Ko=sparse(zeros(last1));
end

```

Thus far, we have defined the global stiffness matrix and the  $K_0$  (stiffness matrix that does not depend on  $\theta$ ).

- The next step is to calculate  $K_i$  the stiffness matrices that depend linearly on  $\theta$  as well as the values of  $\theta_i$ . The value of  $\theta$  is different to zero only for the component from which we want to extract the stiffness matrix, the other values of  $\theta$  are zero. The value of  $\theta$  changes in every loop depending on the material that we examine. For example if we want to parameterize the  $j$  material  $\theta_j \neq 0$  while  $\theta_i = 0$  for  $i=1....N$  and  $i \neq j$

The process of extraction of  $K_i$  is the same with that for extraction and calculation of global stiffness matrix.

```

for q =1:meter

    ret=SapModel.SetModelIsLocked(0);
    % keep the number of material that is parameterized in
    this loop
    number_mat1=q
    % give value of  $\theta$  for the material
    prompt = {'the value of parameter  $\theta$ :'};
    title = 'value of parameter  $\theta$  for ';
    th = inputdlg(prompt,mat_name{param(q)},[1 60]);
    % b is  $\theta$ 
    b(q)=str2num(th{1})
    % we kept the parameterized material.  $\theta$  for others
    materials is going to be zero. this changes in every loop.
    only the material that is
    % analyzed (in every loop) has no zero value of  $\theta$ .
    for l=1:meter
        if l~=q

```



```

        b(1)=0;
    end
    % multiple the modulus of elasticity of each material with  $\theta$ 
    Enew(param(1))=E(param(1))*b(1);

end

%for every material pass the value of new modulus of
elasticity in sap2000
    for t=1:meter

ret=SapModel.PropMaterial.SetMPIsotropic(char(mat_name{param
(t)}), Enew(param(t)),Nu(param(t)),A(param(t)));
    end
%analyze the model
    ret = SapModel.Analyze.RunAnalysis();
%import stiffness matrix and bring it to its normal form
    Sfilename=[name, '.TXK'];
    s = importdata(Sfilename);
    K_new=s.data;
    mtkos1=length(K_new);
    last2=K_new(end,2);
    K_i=sparse(zeros(last2));
    sum=0

    k=0
    K_new_end=s.data(:,end);
    for r=1:mtkos1
        if K_new(r,3)~=0
            metrhths=r;
            i(r)=K_new(r,1);
            j(r)=K_new(r,2);

        end
    end

    for n=1:metrhths
        if i(n)j(n)~=0
            K_i(i(n),j(n))=K_new(n,3);
        end
    end

    K_i=(K_i+(tril(K_i,-1)).');
    % The Ki matrices are created

    Ki{q,1}=mat_name(param(q));
    % if there is Ko then the K of the spesific part is the K_i
    that was
    % calculated above minus Ko.
    Ki{q,2}=(K_i-Ko);
    Ki{q,2}=Ki{q,2}/b(q);
    % the values of  $\theta$ 
    thita(q)=b(q);

end
end

```

The cell arrays of  $\mathbf{K}_i$  contain the  $K_i$  matrices of every parameterized material/component. Also, the values of  $\theta$  that were given for each material were saved in a matrix called `thita`. The stiffness matrix is calculated for each material and saved in `matlab`.

### 3.2.2 The acquisition of global mass matrix(before the parameterization process) , $M_0, M_i$

The changes in modulus of elasticity do not affect the mass matrix given that, it depends on mass density. The global mass matrix's data were imported in the code when we tried to calculate the global stiffness matrix.

- At this point, global mass matrix is going to be transformed to its normal form.

[illegible]

Regarding the code, the `mass_matrix` is a matrix in which the data from mass text file were imported and saved. It is constituted by three columns. The third one is the mass value, the first and the second columns are the matrix's number of the row and column respectively. The `M` matrix is a diagonal matrix which contains the mass values of the global mass matrix.

- The next step is the calculation of  $M_0$ (if it exists). The existence of  $M_0$  depends on the value of the meter variable.

```
% %%%%%%%%% calculation of Mo%%%%%%%%
if meter~=number_mat&&meter~=0

    for ee=1:meter
        ret=SapModel.SetModelIsLocked(0);

ret=SapModel.PropMaterial.SetWeightAndMass(char(mat_name{par
am(ee)}),1, 0)
    end
    ret = SapModel.Analyze.RunAnalysis();
    Mfilename=[name, '.TXM'];

    mass0= importdata(Mfilename);
    mass_matrix0=mass0.data;
    mtkos_m0=length(mass_matrix0);

    Mo=sparse(zeros(last_m));
    metrhts00=0
    for ww=1:mtkos_m0
        if mass_matrix0(ww,3)~=0
            x0(ww)=mass_matrix0(ww,1);
            z0(ww)=mass_matrix0(ww,2);
            metrhts00=ww
        end
    end
    for uu=1:metrhts00
        if x0(uu),z0(uu)~=0
            Mo(x0(uu),z0(uu))=mass_matrix0(uu,3);
        end
    end
end
if meter==number_mat
    Mo=sparse(zeros(last_m));
end
```

- After the calculation of  $M_0$ , follow the extraction and the calculation of  $M_i$ . The mass matrix depends the mass density of each component, thus a new parameter  $\varphi$  must be defined. This parameter is related to mass density. The value of  $\varphi$  is different from zero only for the component/material from which we want to extract the mass matrix, the other values of  $\varphi$  are set zero. The value of  $\varphi$  changes in every loop as it depends on the material that we examine. For example, if we want to parameterize the  $j$  material  $\varphi_j \neq 0$  while  $\varphi_i = 0$  for  $i=1 \dots N$  and  $i \neq j$ .

```

% % % % % % % % % % % % % % calculation of Mi % % % % % % % %
% % % % % % % %
if meter~=0
    for kk=1:meter
        ret=SapModel.SetModelIsLocked(0);
        % define the value of f where f is  $\varphi$ 
        prompt = {'the value of parameter  $\varphi$ :'};
        title = 'value of parameter  $\varphi$  for ';
        ff = inputdlg(prompt,mat_name{param(kk)},[1 60]);

        f(kk)=str2num(ff{1})
        % the value  $\varphi$  that is not zero is the only one that
        relating with the material that is parameterized in this
        loop
        for ll=1:meter
            if ll~=kk
                f(ll)=0;
            end
            % multiple the density with  $\varphi$ 
            density_new(param(ll))=density(param(ll))*f(ll);

        end
        for tt=1:meter
            % pass the parameterized densities in Sap2000

ret=SapModel.PropMaterial.SetWeightAndMass(char(mat_name{par
am(tt)}),1, density_new(param(tt)));
        end
        % analyze the model
        ret = SapModel.Analyze.RunAnalysis();
        % extract Mi and bring them to their normal form
        Mfilename=[name, '.TXM'];

        mass1= importdata(Mfilename);
        mass_matrixNew=mass1.data;
        mhkos_m1=length(mass_matrixNew);
        last_m1=mass_matrixNew(end,2);
        M_new=sparse(zeros(last_m1));
        metrhths1=0
        for yy=1:mhkos_m1
            if mass_matrixNew(yy,3)~=0
                xn(yy)=mass_matrixNew(yy,1);
                zn(yy)=mass_matrixNew(yy,2);
                metrhths1=yy
            end
        end
        for gg=1:metrhths1
            if xn(gg),zn(gg)~=0
                M_new(xn(gg),zn(gg))=mass_matrixNew(gg,3);
            end
        end
        Mi{kk,1}=mat_name(param(kk));
        Mi{kk,2}=(M_new-Mo);
        Mi{kk,2}=Mi{kk,2}/f(kk);
        fi(kk)=f(kk)
    end
end
end

```

With regard to the code,  $M_{new}$  corresponds to the mass matrix of the parameterized component/material. The cell arrays of  $M_i$  contain the mass matrices for each parameterized component. Also, the values of  $\varphi$  are saved in a matrix called  $f_i$ .

The model matrices have been calculated and can be saved in .mat files in order to be used for model updating.

## **4. Modal Identification of a bridge using vibration measurements.**

### **4.1 Measurements data**

Based on the finite element model of a bridge model, developed with the Static and Dynamic Analysis Program Sap2000v19, the analytical values of frequencies and mode shapes of the arched bridge were calculated and used for more accurate identification of frequencies, velocities and damping rates of measured oscillation time histories.

Two sets of measurements were used and analyzed in order for the mode shapes to be identified (figures 4.1 and 4.2). The location of the accelerometers on the structure plays a key role for obtaining optimum measurements (figure 4.3). Obtaining measurements that provide maximum information regarding the dynamic modal characteristic

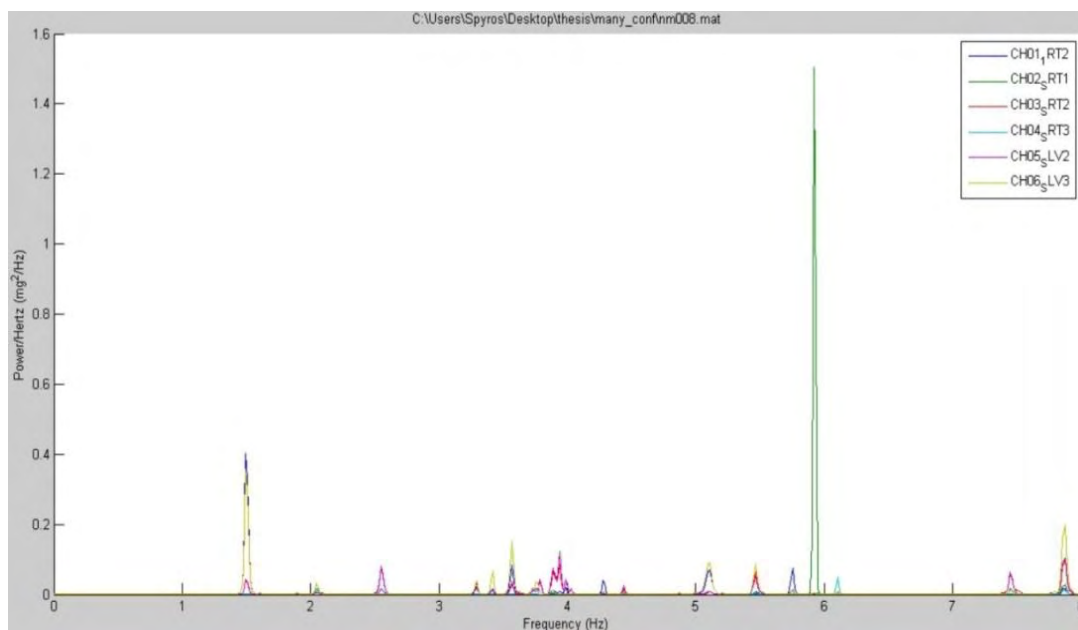


Figure 4.1. nm008 configuration

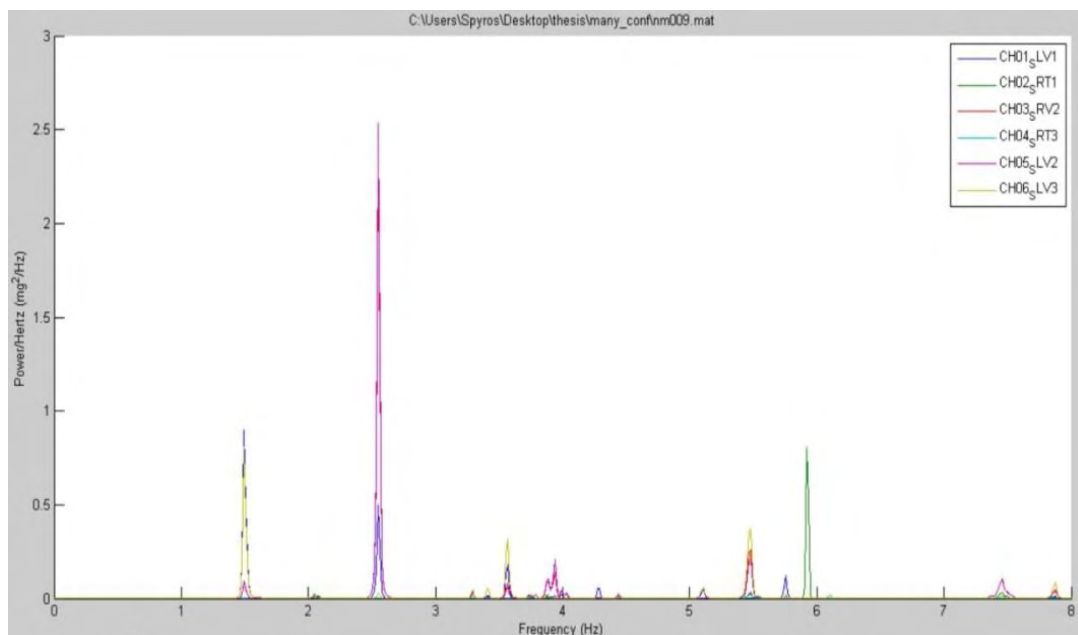


Figure 4.2. nm009 configuration

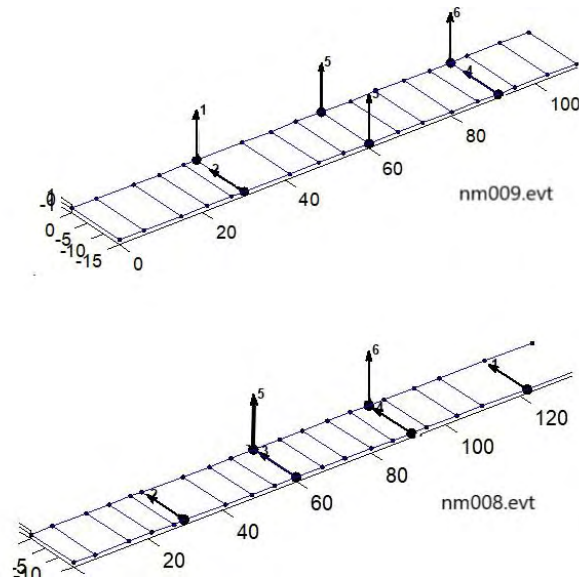


Figure 4.3. The location of the sensors

## 4.2 Identification of mode shapes and modal frequencies based on FE model

The identified modal frequencies based on the finite element model's modal analysis (the first fourteenth identified frequencies).

Table 1. Frequencies based on Sap2000 modal analysis and on measurements

| Frequencies based on<br>Sap2000 Model(Hz) | Identified measured<br>frequencies(Hz) |
|---|--|
| 1.076257                                  | 1.1332                                 |
| 1.283569                                  | 1.5004                                 |
| 1.9159                                    | 2.05                                   |
| 1.993078                                  | -                                      |
| 2.462727                                  | 2.5511                                 |

|   |  |
|---|--|
| 3.148780  | 3.2928   |
| 3.644488  | 3.925  |
| 3.8759  | 4.4405   |
| 4.48316   | 4.73   |
| 4.54937   | 5.1036   |
| 4.76772   | 5.4675   |
| 5.91396<br>5.91401<br>6.02791<br>6.05633<br>6.13507<br>6.14033<br>6.15619 | 5.925,6.1003(probably<br>corresponds to one of<br>these frequencies) |
| 6.35409   | -  |
| 6.71095   | 7.4562   |
| 7.14691   | 7.8703   |

These frequencies were identified based on the FE model through a range of measurements. The figure 3.6 shows the explored peaks of the time response spectrum of the arched bridge (for time history nm008).



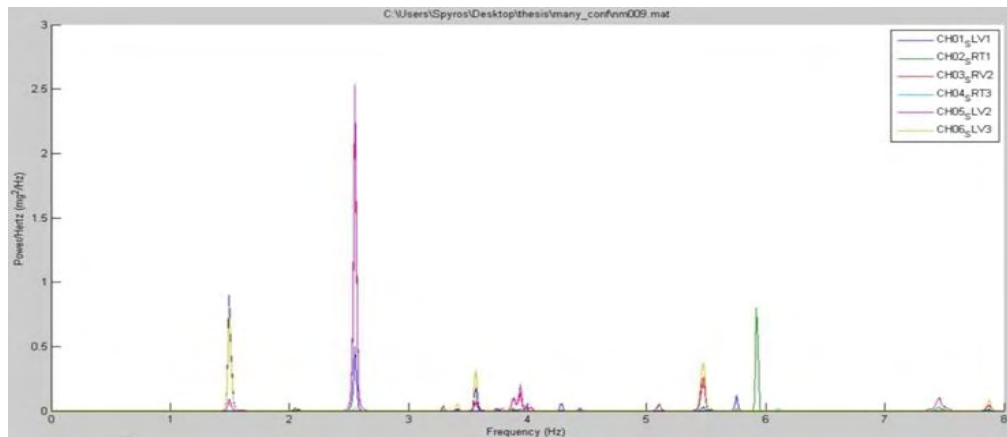


Figure 4.4. The measured frequencies from time history nm008

The matching of frequencies is based on the form of mode shapes. The comparison of mode shapes for the above sets of frequencies is illustrated below:

### 1. Modal frequency $f=1.1332\text{Hz}$

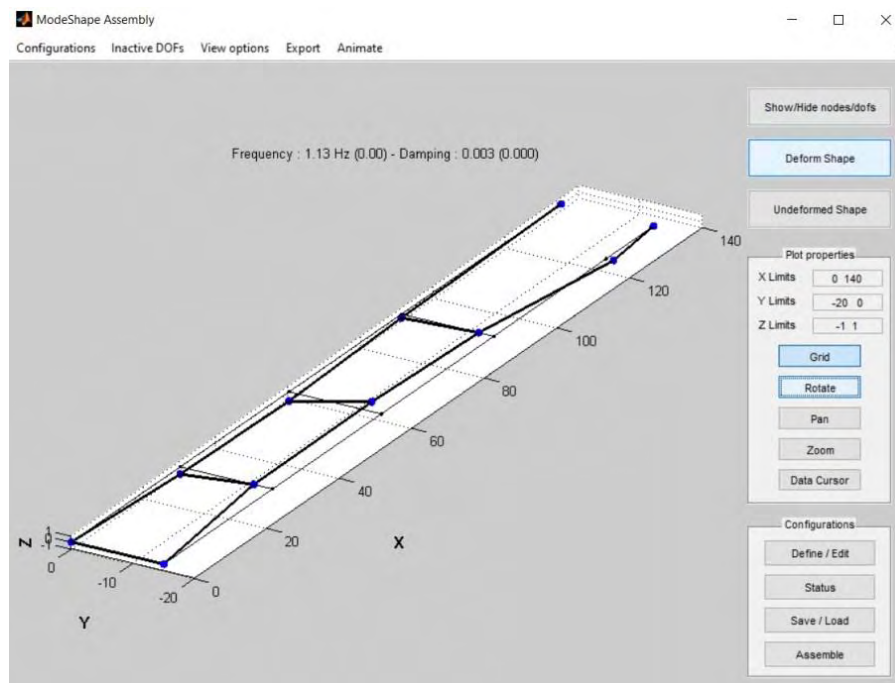


Figure 4.5. Mode shape based on measurements ( $f=1.13\text{Hz}$ )

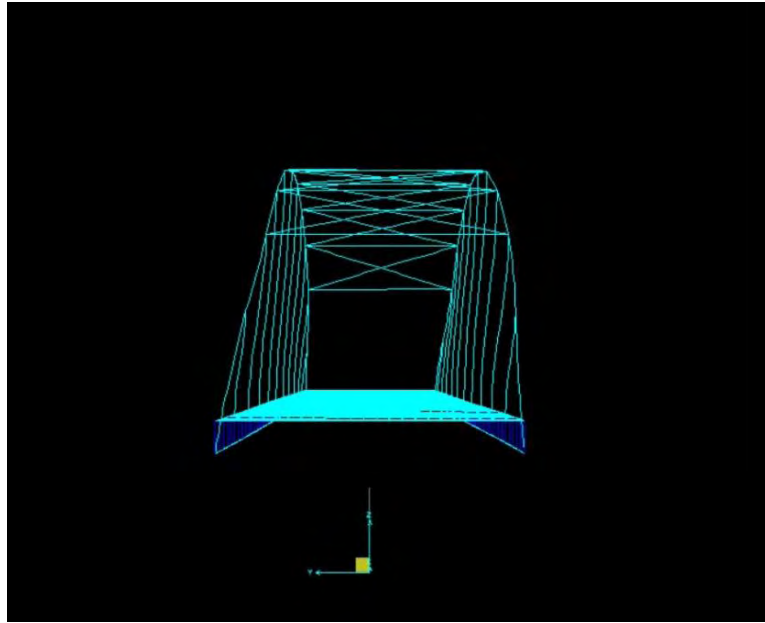


Figure 4.6. Oscillation of the bridge model for  $f=1.076257\text{Hz}$  (front view)

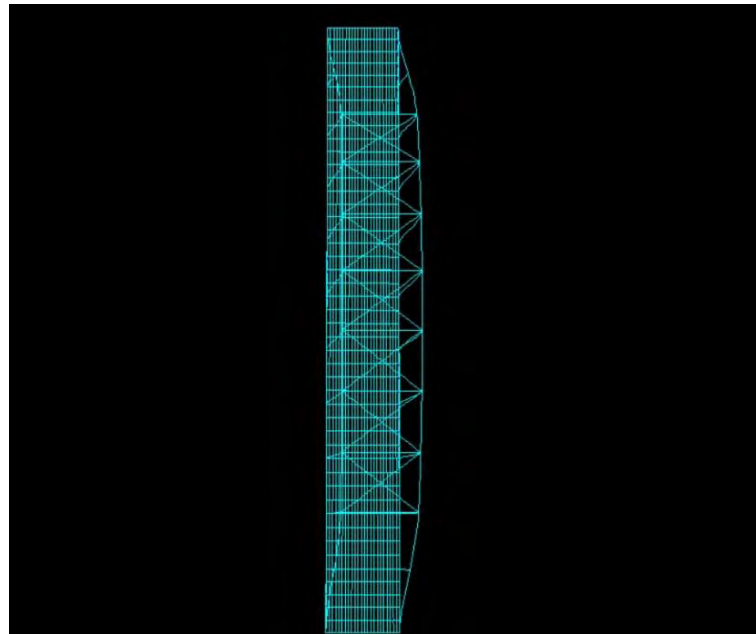


Figure 4.7. Oscillation of the bridge model for  $f=1.076257\text{Hz}$  (top view)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=1.1332\text{Hz}/1.076257\text{Hz}=1.053$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

## 2. Modal frequency $f=1.5004\text{Hz}$

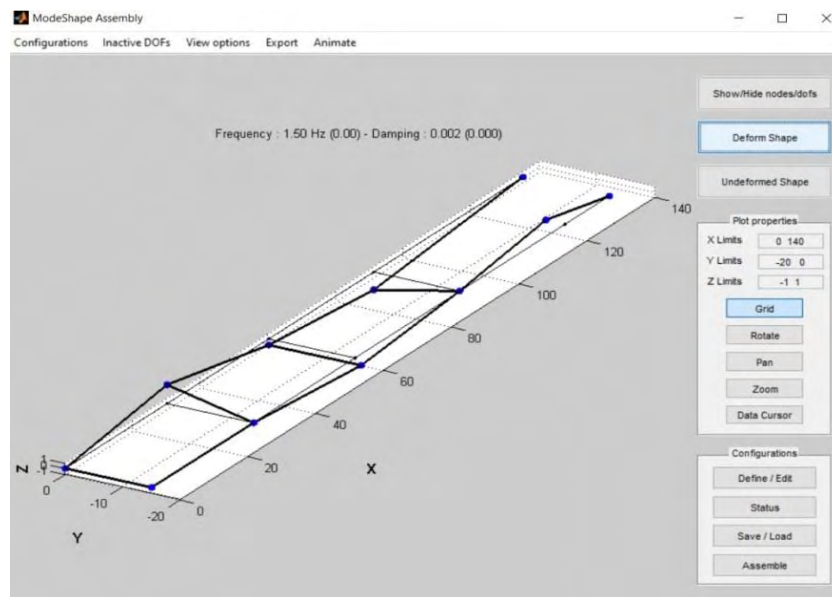


Figure 4.8. Mode shape based on measurements ( $f=1.5004\text{ Hz}$ )

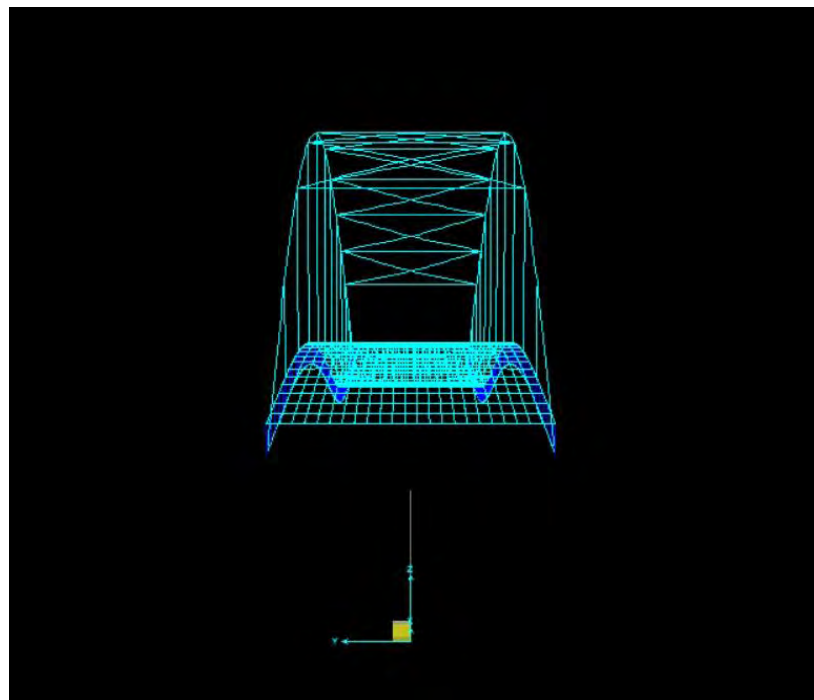


Figure 4.9. Oscillation of the bridge model for  $f=1.283569\text{Hz}$  (front view)

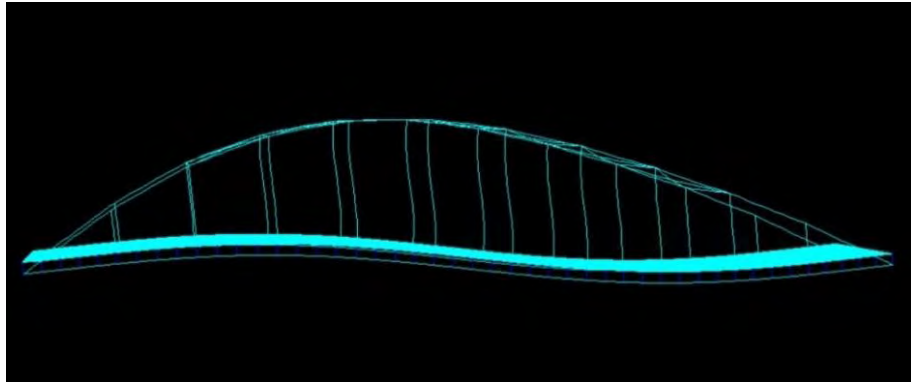


Figure 4.10. Oscillation of the bridge model for  $f=1.283569$  Hz (cross section)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=1.5004\text{Hz}/1.283569\text{Hz}=1.1686$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

### 3. Modal frequency $f=2.05$ Hz

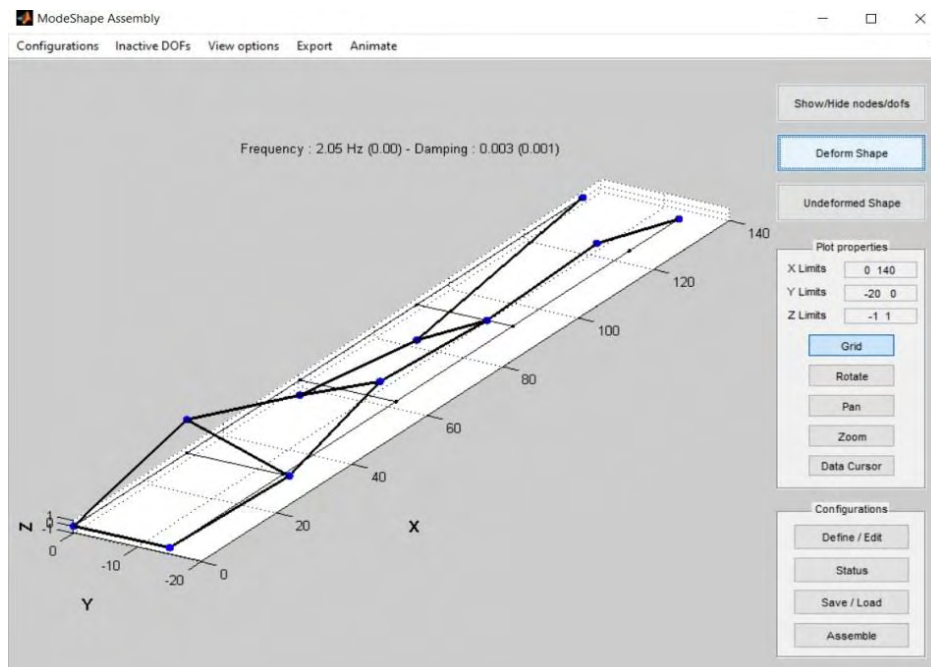


Figure 4.11. Mode shape based on measurements ( $f=1.13$  Hz)

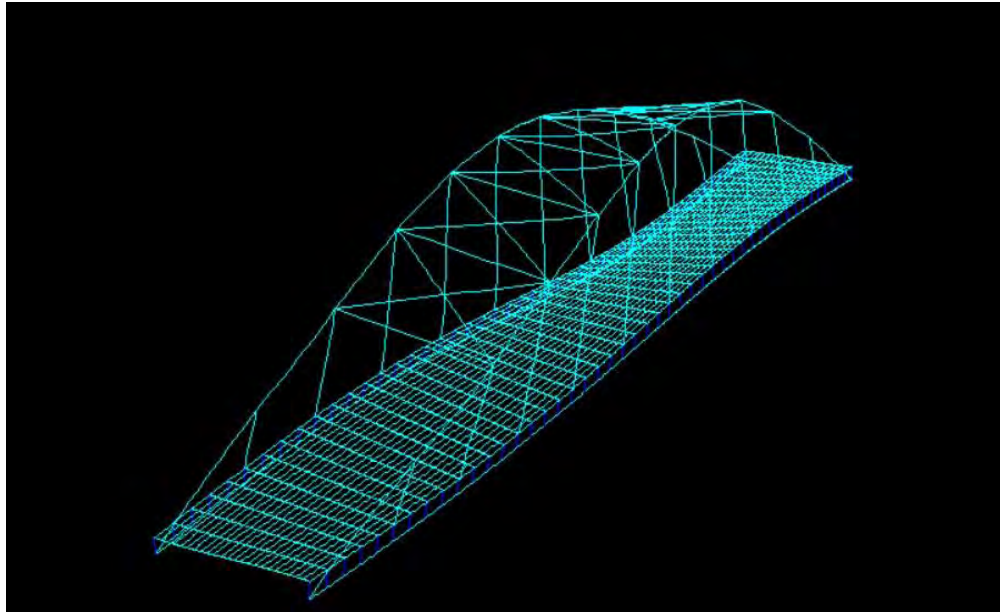


Figure 4.12. Oscillation of the bridge model for  $f=1.915901$  Hz (original view)

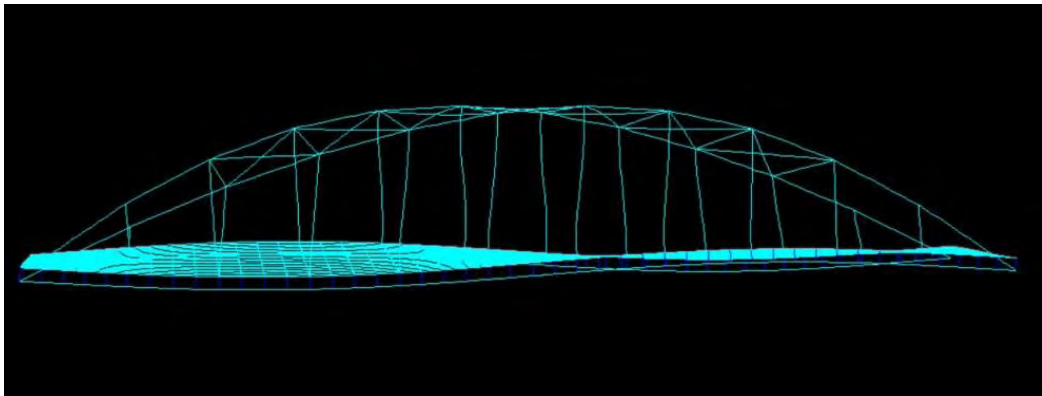


Figure 4.13. Oscillation of the bridge model for  $f=1.915901$  Hz (cross section)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=2.05\text{Hz}/1.915901\text{Hz}=1.07$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

#### 4. The modal frequency 1.993078 Hz

Based on finite element model, the modal frequency 1.993078 Hz does not match any of the measurements due to the form of the mode shape.

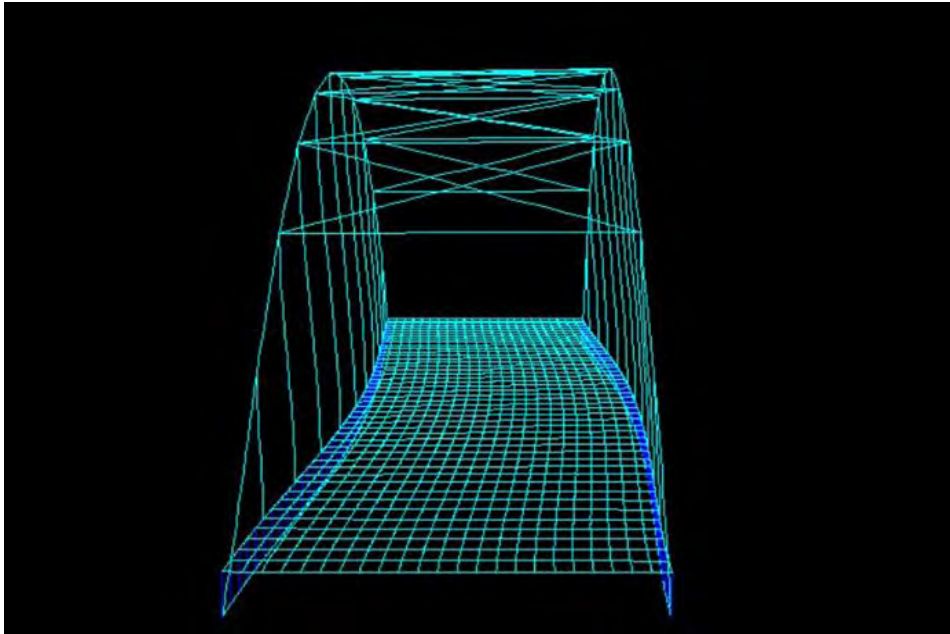


Figure 4.14. Oscillation of the bridge model for  $f=1.1993078$  Hz (front view)

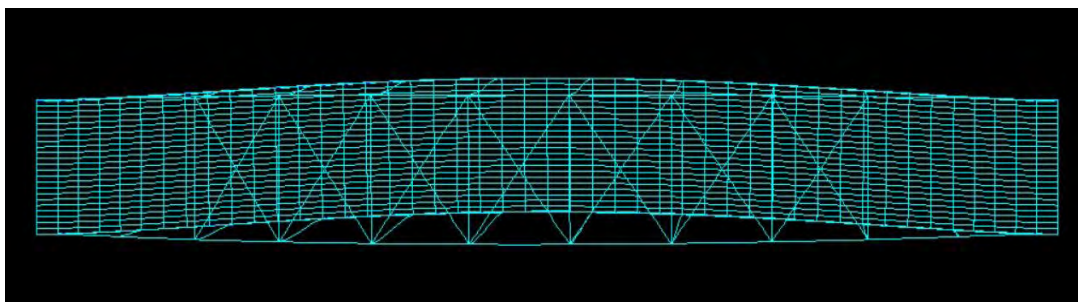


Figure 4.15. Oscillation of the bridge model for  $f=1.1993078$ Hz (top view)

## 5. Modal frequency $f=2.5511$ Hz

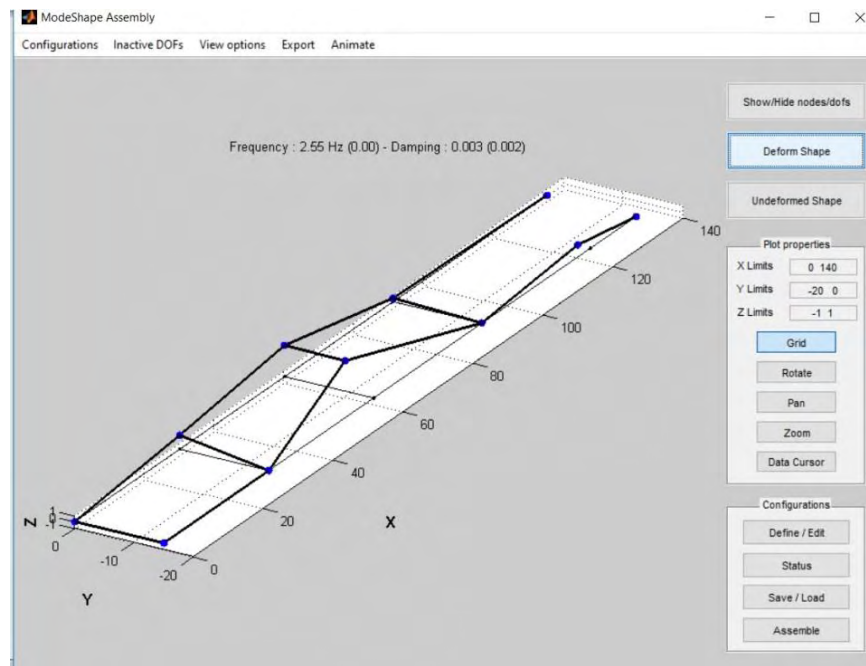


Figure 4.16. Mode shape based on measurements ( $f=2.5511$  Hz)

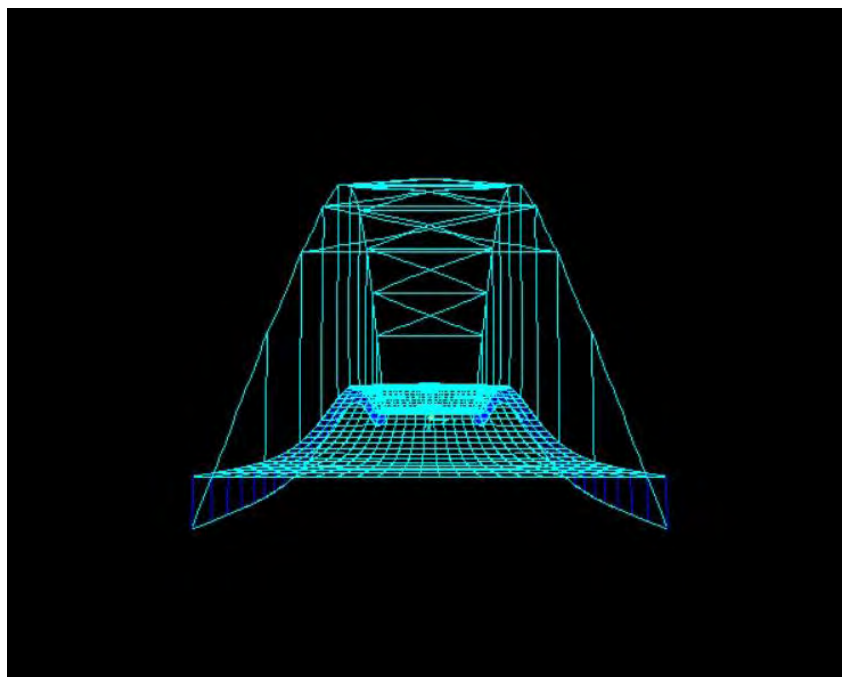


Figure 4.17. Oscillation of the bridge model for  $f=2.462727$ Hz (front view)



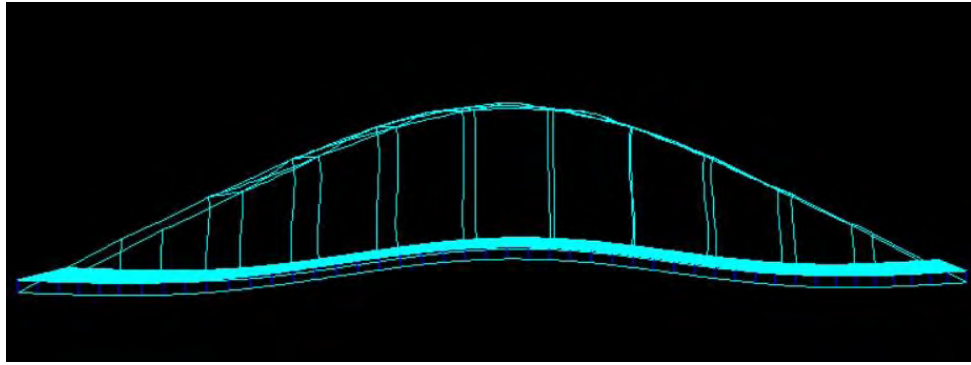


Figure 4.18. Oscillation of the bridge model for  $f=2.462727$  Hz (Cross Section)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model. ( $f_{\text{measurements}}/f_{\text{FEM}}=2.5511\text{Hz}/2.462727\text{Hz}=1.03588$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

## 6. Modal frequency $f=3.2928$ Hz

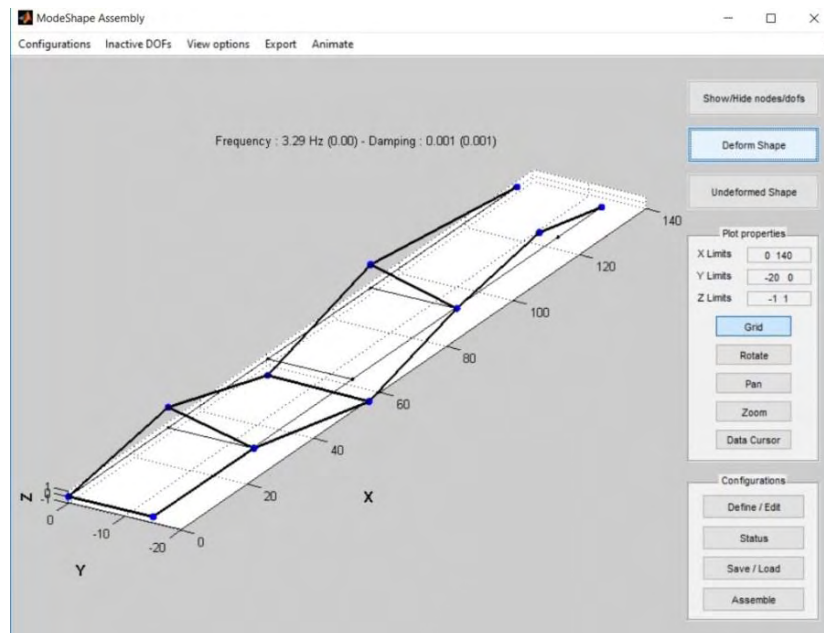


Figure 4.19. Mode shape based on measurements ( $f=3.2928$  Hz)



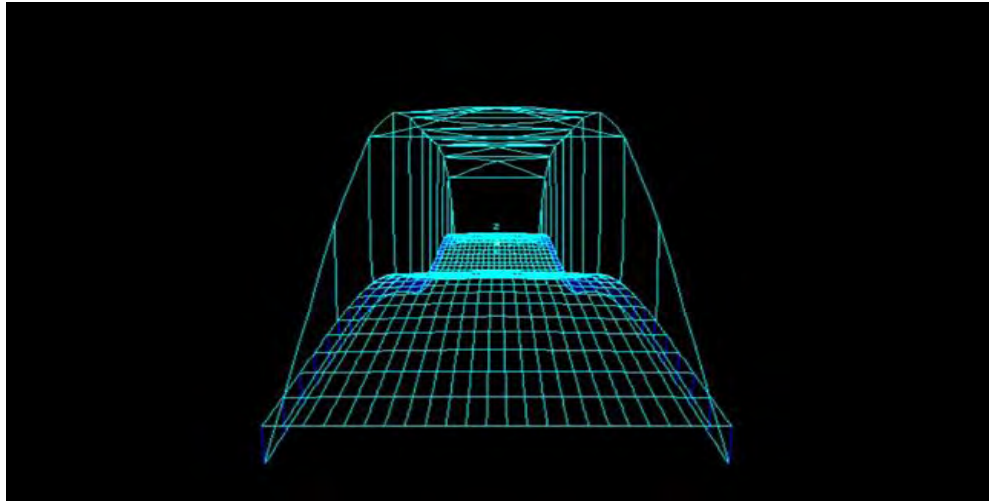


Figure 4.20. Oscillation of the bridge model for  $f=3.148780\text{Hz}$  (front view)

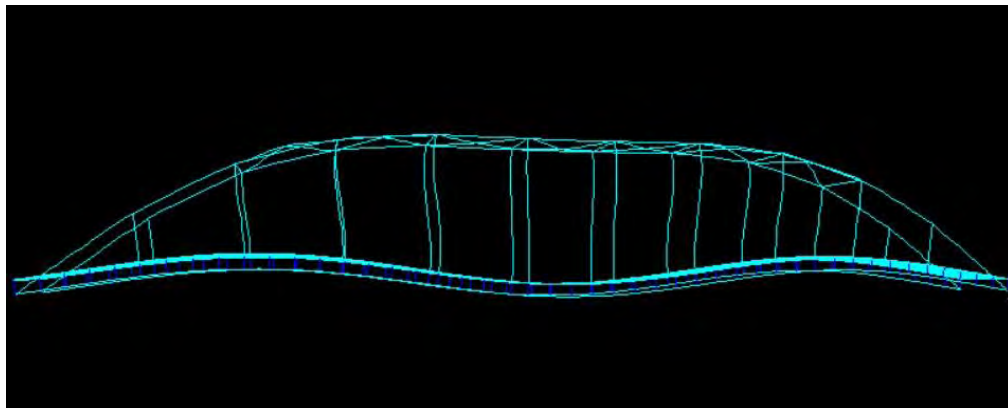


Figure 4.21. Oscillation of the bridge model for  $f=3.148780\text{Hz}$  (cross section)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=3.2928\text{Hz}/3.148780\text{Hz}=1.03588$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

## 7. Modal frequency $f=3.9256$ Hz

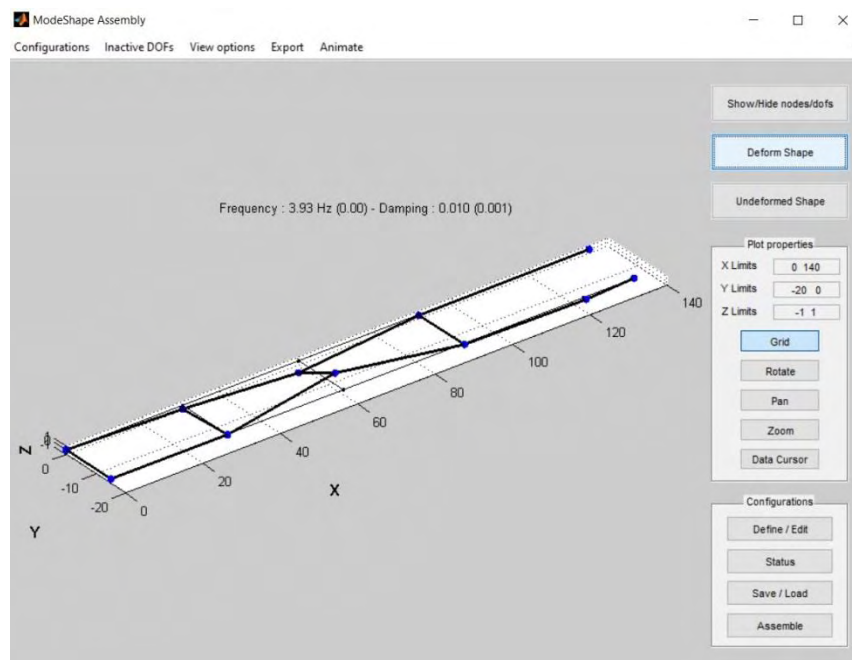


Figure 4.22. Mode shape based on measurements ( $f=3.9256$  Hz)

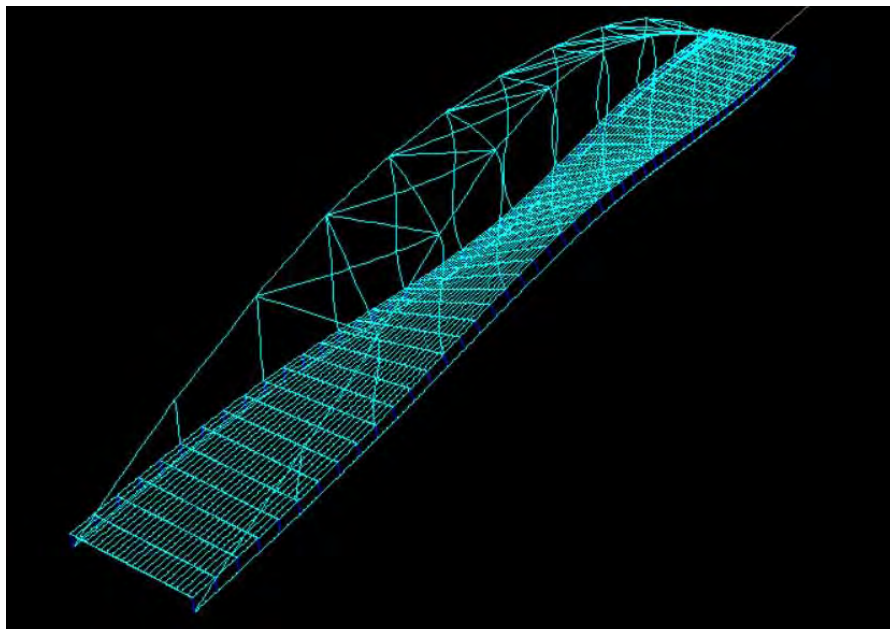


Figure 4.23. Oscillation of the bridge model for  $f=3.644488$  Hz (original view)

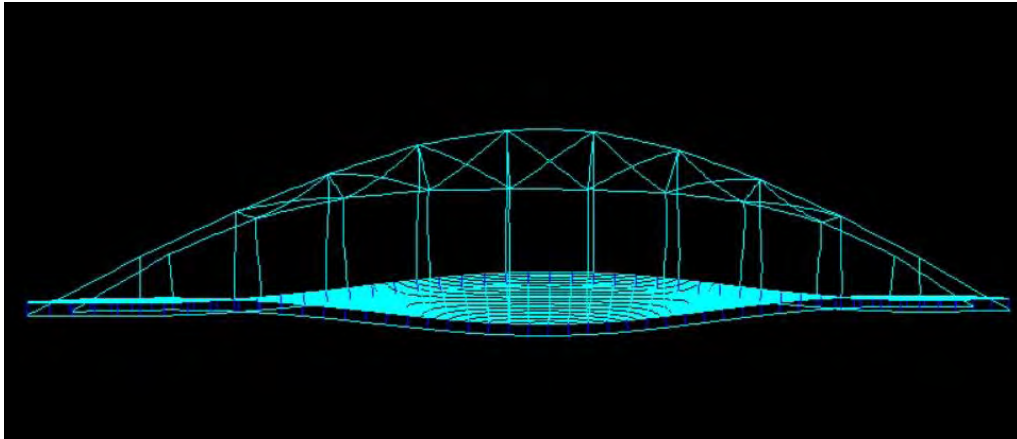


Figure 4.24. Oscillation of the bridge model for  $f=3.644488$  Hz (cross section)

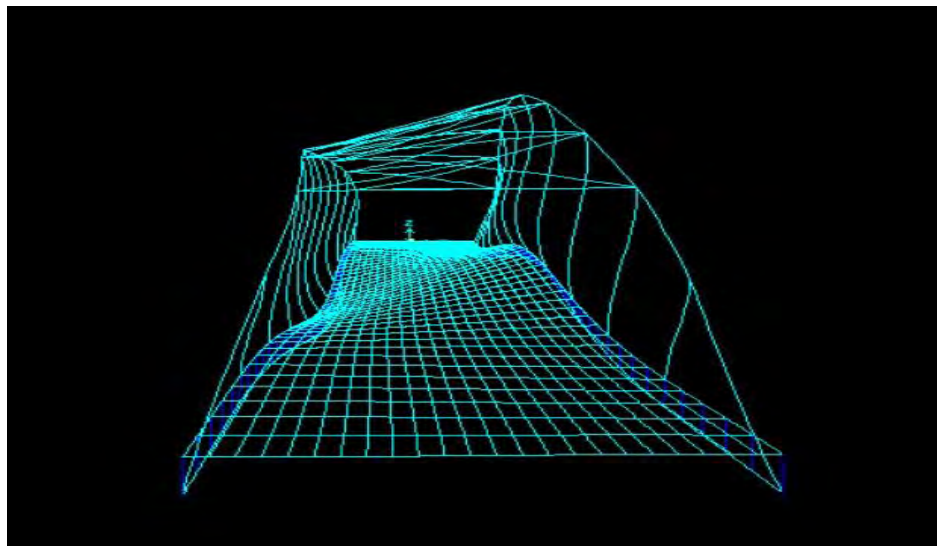


Figure 4.25. Oscillation of the bridge model for  $f=3.644488$  Hz (front view)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=3.9256\text{Hz}/3.64488\text{Hz}=1.0771$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

## 8. Modal frequency $f=4.4405$ Hz

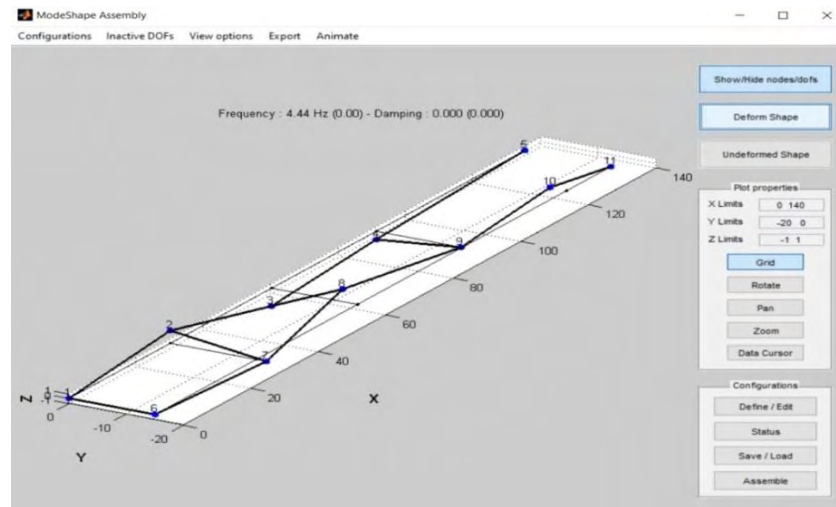


Figure 4.26. Mode shape based on measurements ( $f=4.4405$  Hz)

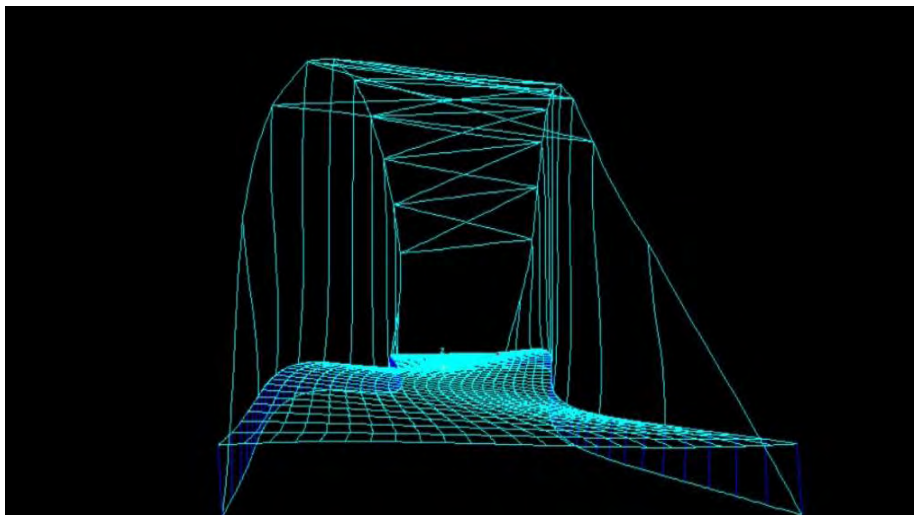


Figure 4.27. Oscillation of the bridge model for  $f=3.8759$  Hz (front view)

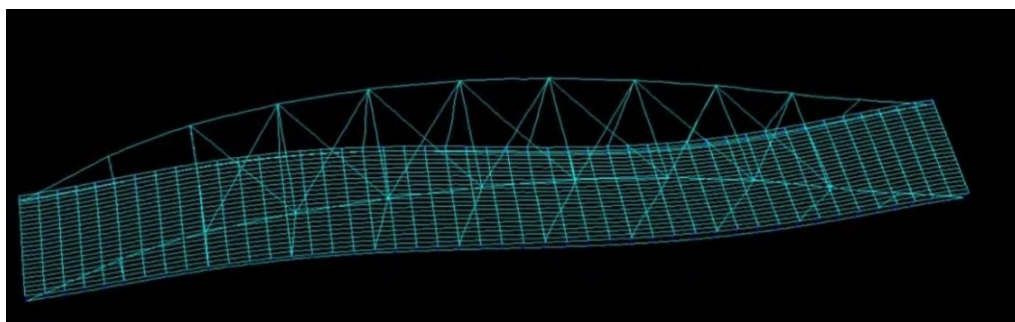


Figure 4.28. Oscillation of the bridge model for  $f=3.8759$  Hz (top view)

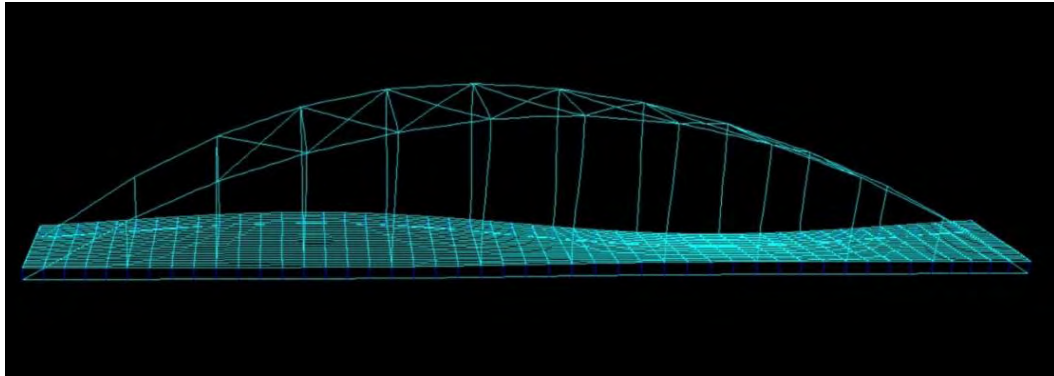


Figure 4.28. Oscillation of the bridge model for  $f=3.8759$  Hz (cross section)

The approach of the mode shape is not absolutely accurate. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=4.4405$  Hz/ $3.8759$ Hz= $1.1457$ ). More measurements should be taken in order to verify the mode shape.

## 9. Modal frequency $f=4.724$ Hz

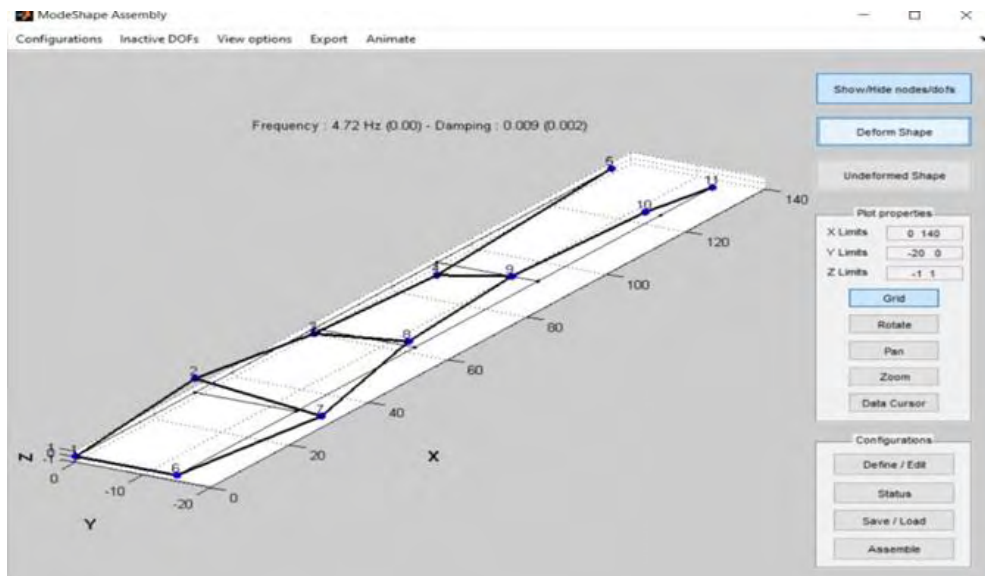


Figure 4.29. Mode shape based on measurements ( $f=4.724$  Hz)



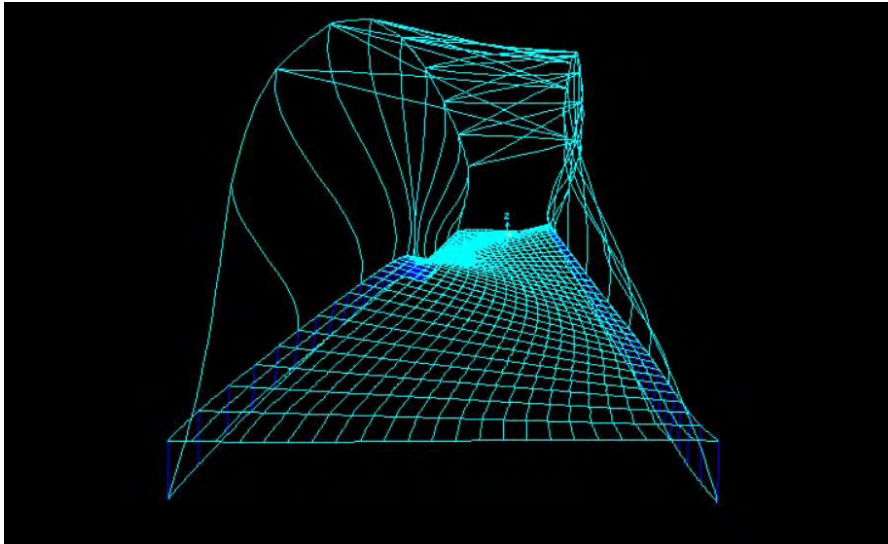


Figure 4.30. Oscillation of the bridge model for  $f=4.48316$  (front view)

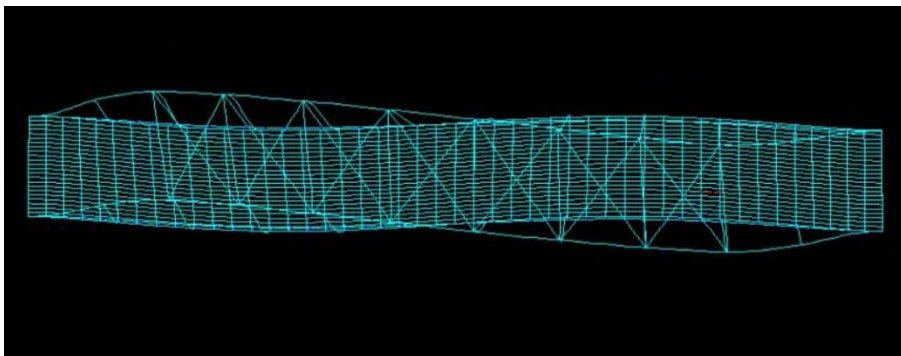


Figure 4.31. Oscillation of the bridge model for  $f=4.48316$  (top view)

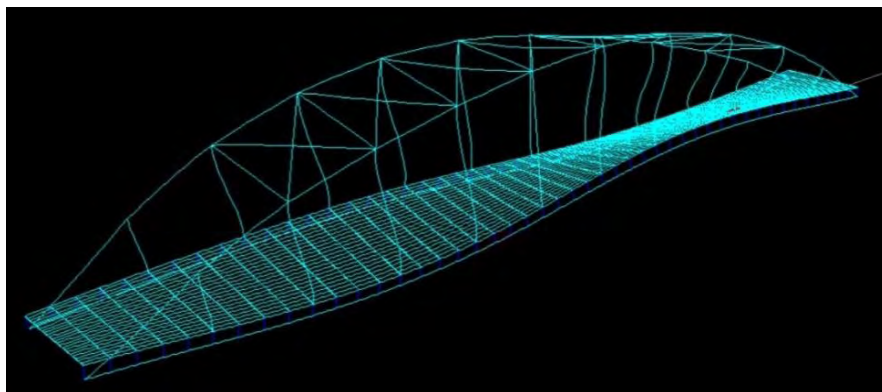


Figure 4.32. Oscillation of the bridge model for  $f=4.48316$  (original view)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=4.724\text{Hz}/4.48316\text{ Hz}=1.0537$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

# 10. Modal frequency $f=5.1036\text{ Hz}$

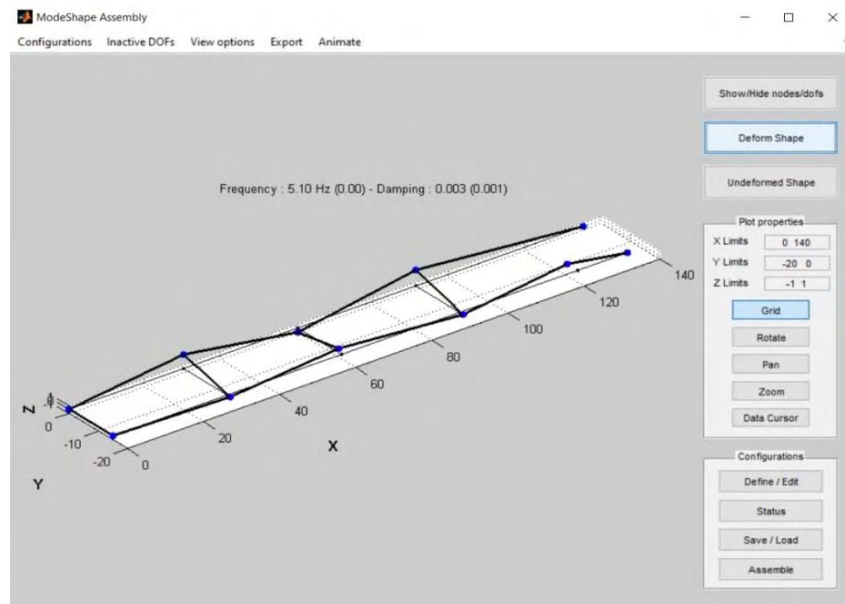


Figure 4.33. Mode shape based on measurements ( $f=5.1036\text{ Hz}$ )

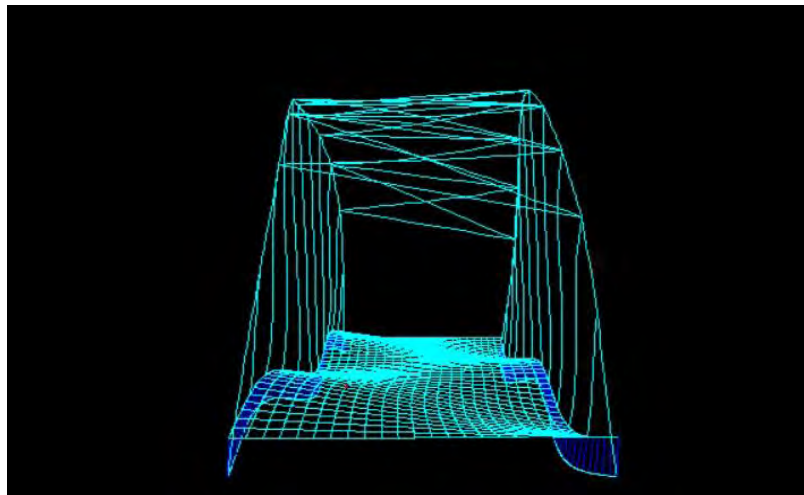


Figure 4.34. Oscillation of the bridge model for  $f=4.54937\text{ Hz}$  (front view)

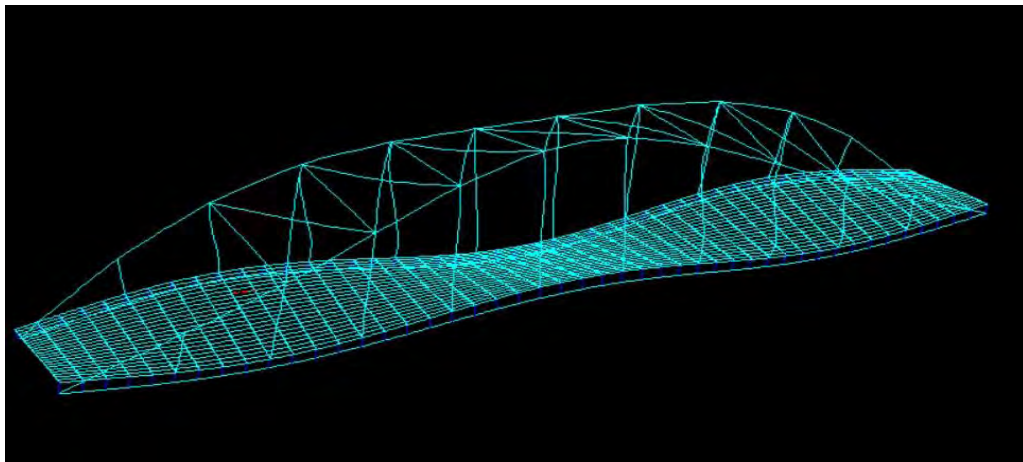


Figure 4.35. Oscillation of the bridge model for  $f=4.54937\text{Hz}$  (original view)

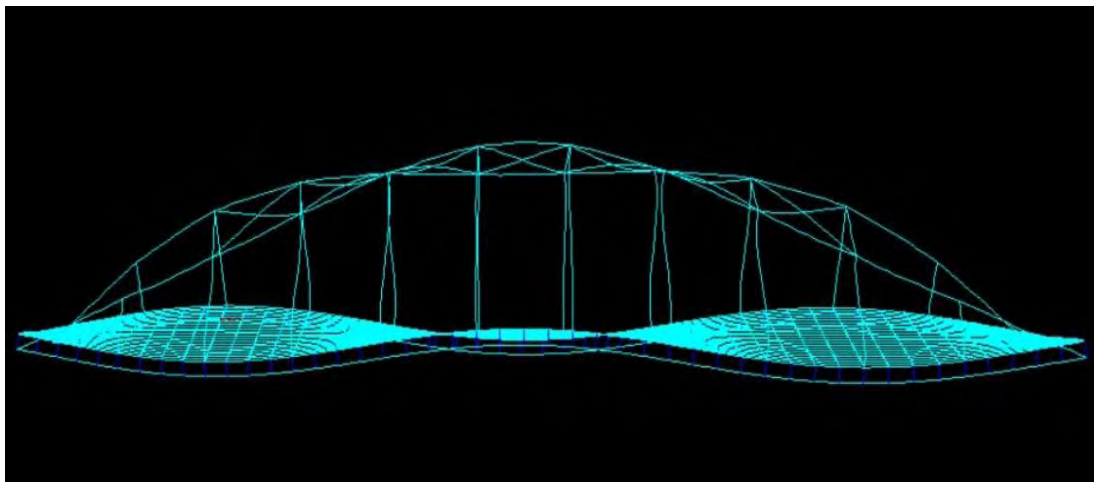


Figure 4.36. Oscillation of the bridge model for  $f=4.54937\text{Hz}$  (cross section)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=5.1036\text{Hz}/4.54937\text{Hz}=1.1218$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.



## 11. Modal frequency $f=5.4675$ Hz



figure 4.37. Mode shape based on measurements ( $f=5.4675$  Hz)

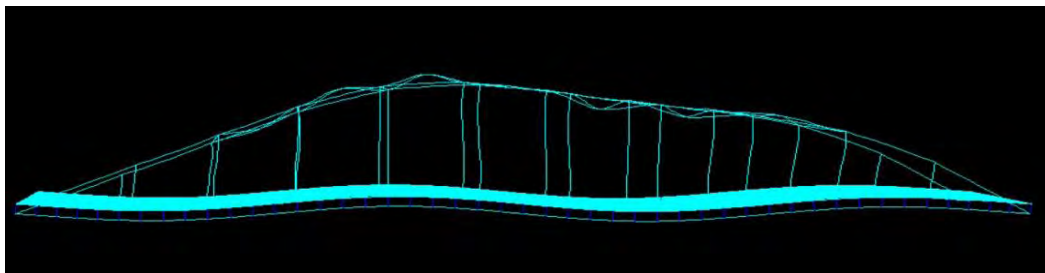


Figure 4.38. Oscillation of the bridge model for  $f=4.76772$ Hz (cross section)

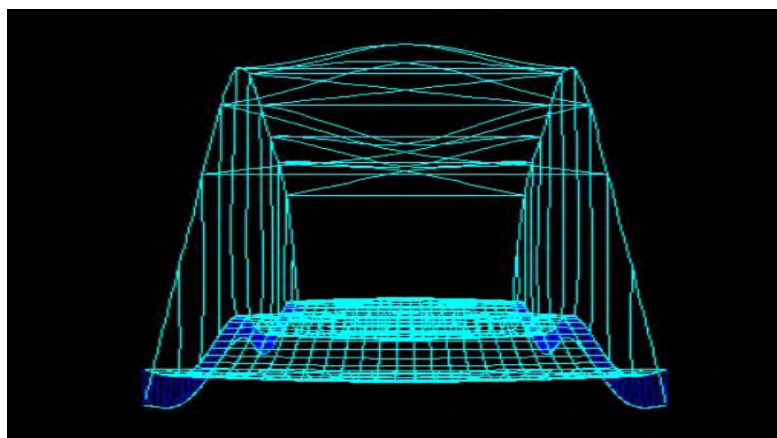


Figure 4.39. Oscillation of the bridge model for  $f=4.76772$ Hz (front view)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are close. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=5.4675\text{Hz}/4.76772\text{Hz}=1.1468$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

**12. The modal frequencies 5.925 and 6.1003 Hz** leads to a distortion of the arched part of the bridge.

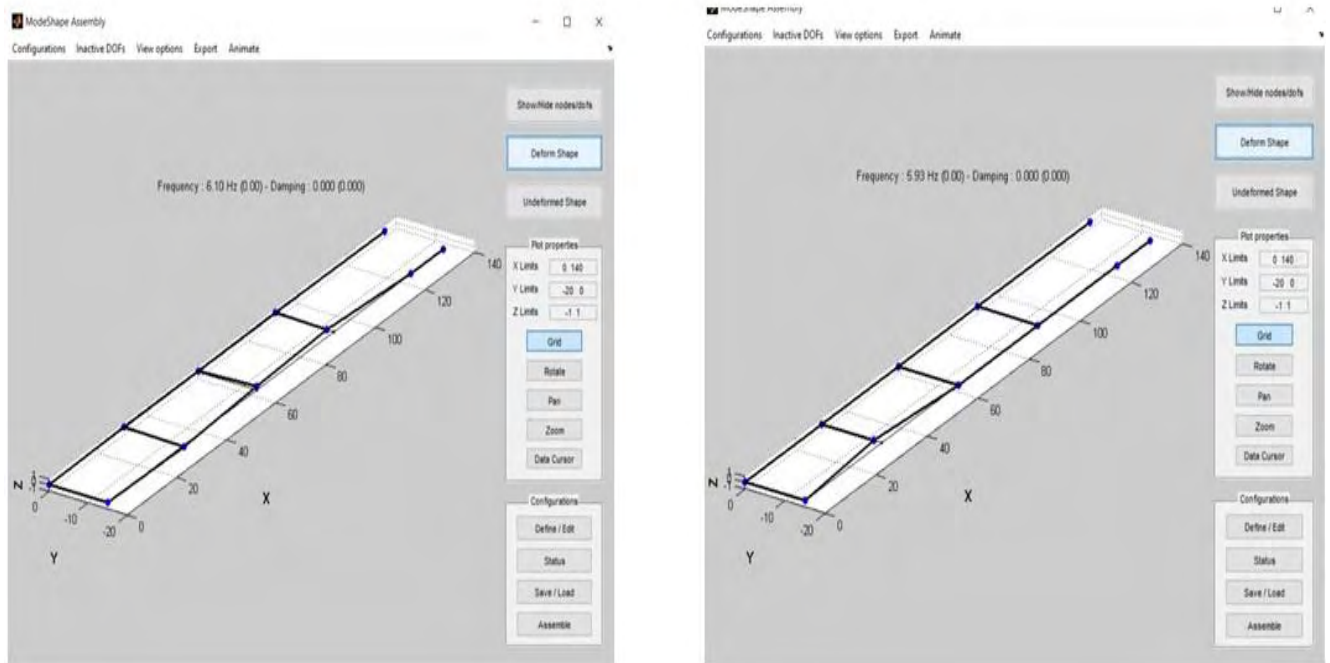


Figure 4.40. Identifiable mode shapes of the arced part

The representations of the mode shapes do not give us a clear image of the distortion of the bridge. Considering the mode shapes of the finite element model for frequencies close to the measured ones, we realize that 5.925 and 6.1003Hz constitute local model frequencies.

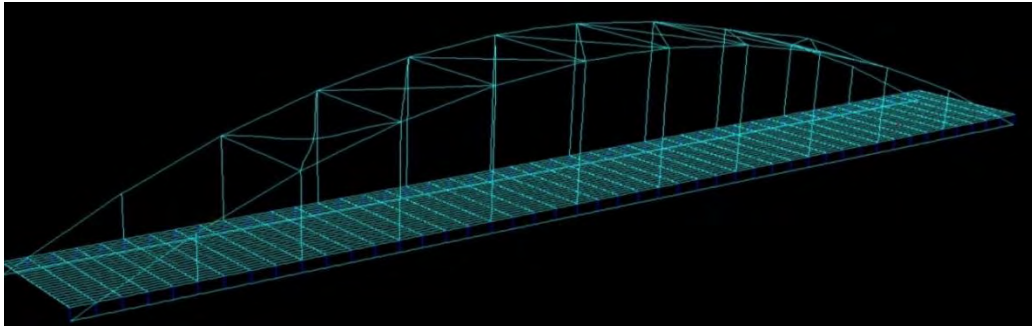


Figure 4.41. Mode shape for  $f=5.91396$  Hz

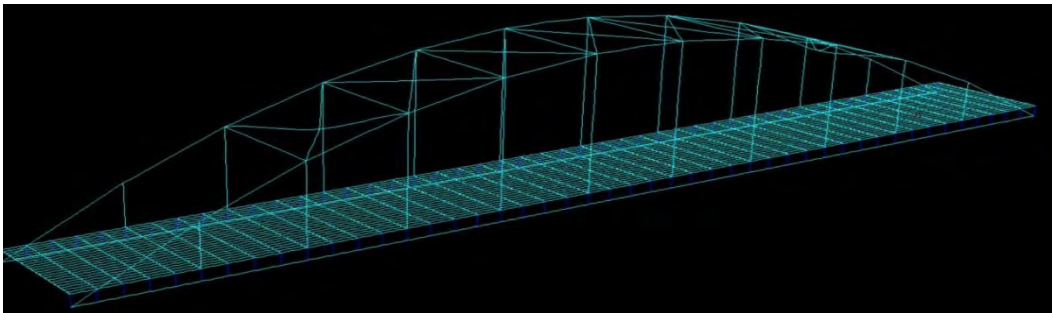


Figure 4.42. Mode shape for  $f=5.91401$  Hz

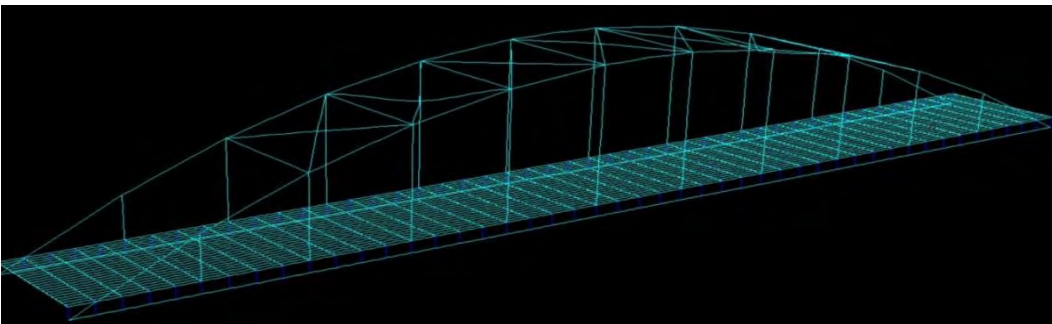


Figure 4.43. Mode shape for  $f=6.02791$  Hz

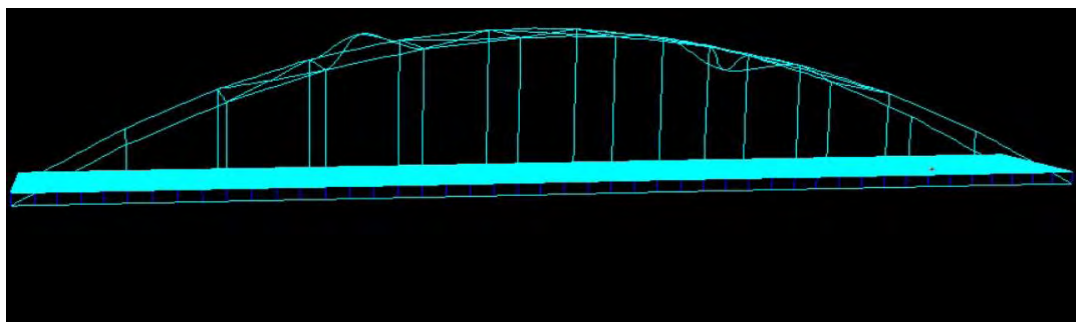


Figure 4.44. Mode shape for  $f=6.05633$  Hz

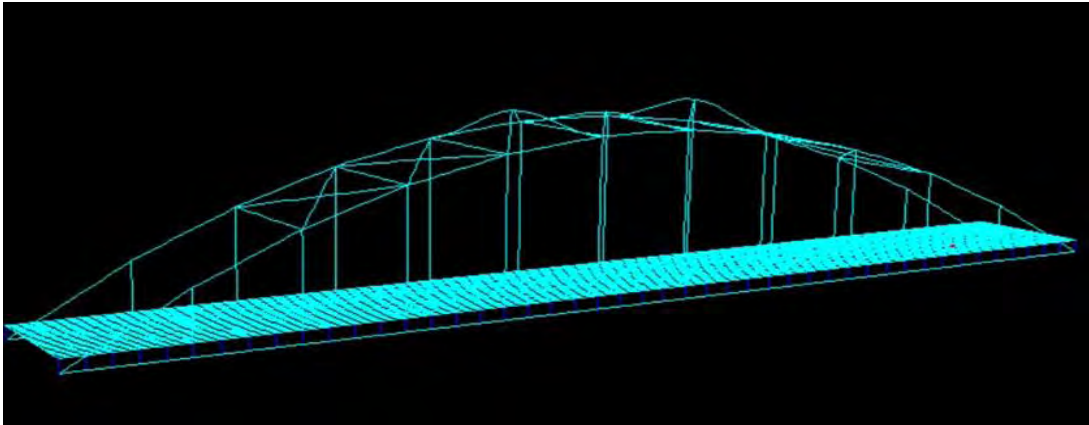


Figure 4.45. Mode shape for  $f=6.13507$  Hz

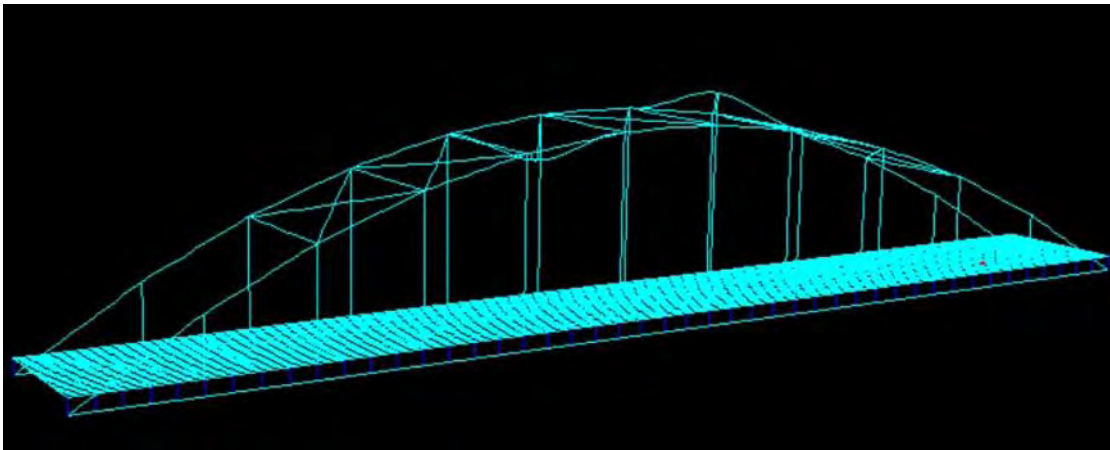


Figure 4.46. . Mode shape for  $f=6.14033$  Hz

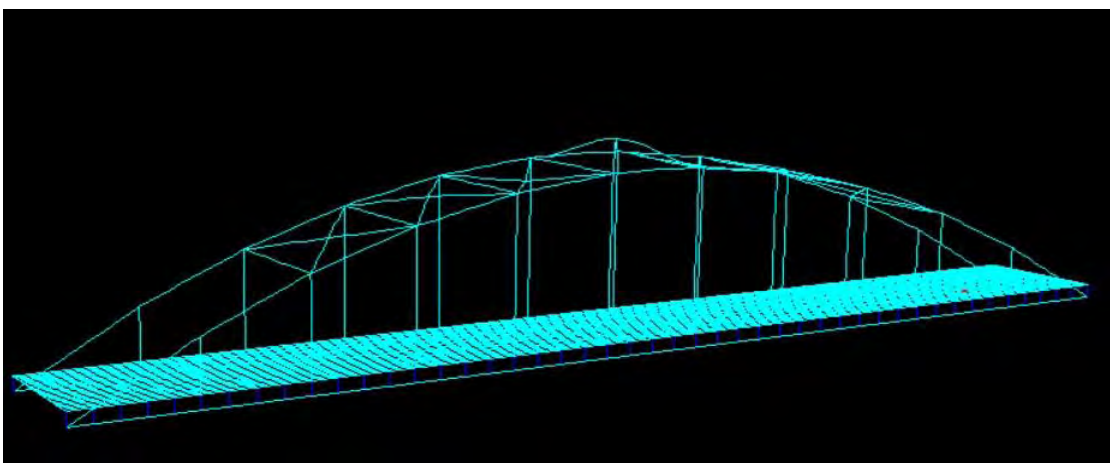


Figure 4.47. Mode shape for  $f=6.15619$  Hz

### 13. Modal frequency $f=7.4562$ Hz

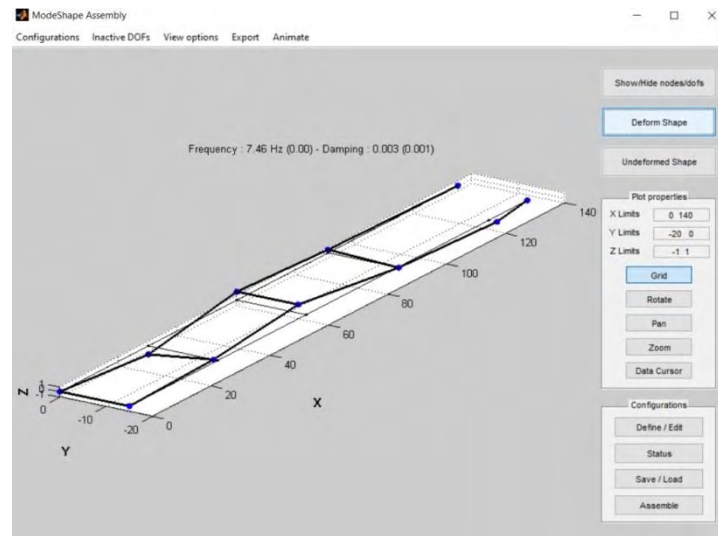


Figure 4.48. Mode shape based on measurements ( $f=7.4562$  Hz)

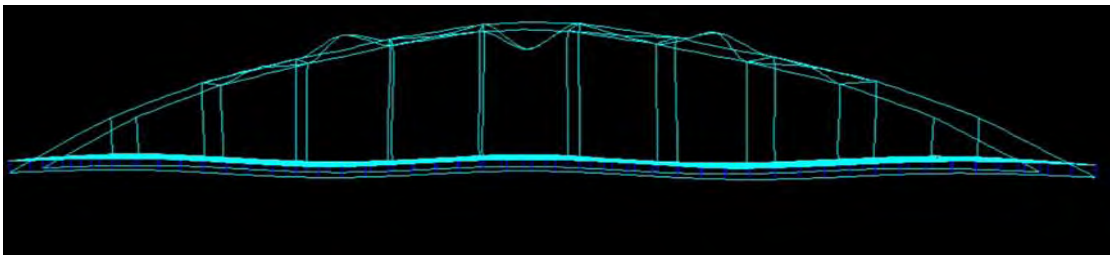


Figure 4.49. Oscillation of the bridge model for  $f=6.71095$  Hz (cross section)

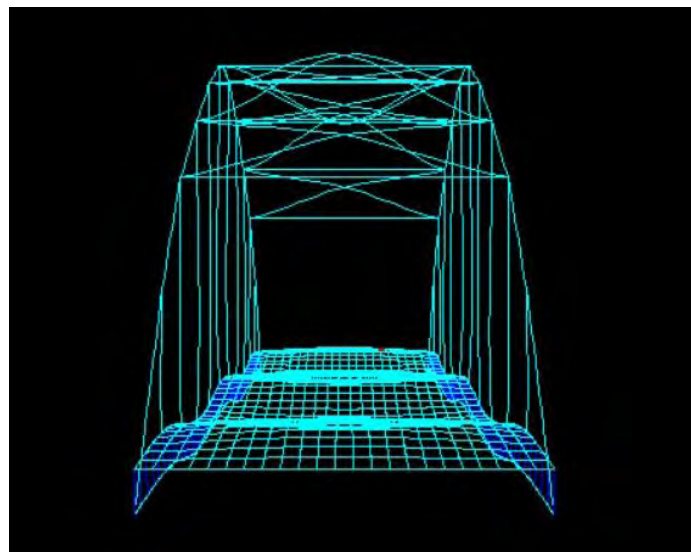


Figure 4.50. Oscillation of the bridge model for  $f=6.71095$  Hz (front view)



The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model. ( $f_{\text{measurements}}/f_{\text{FEM}}=7.4562\text{Hz}/6.71095\text{Hz}=1.11$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

#### 14. Modal frequency $f=7.8703$ Hz

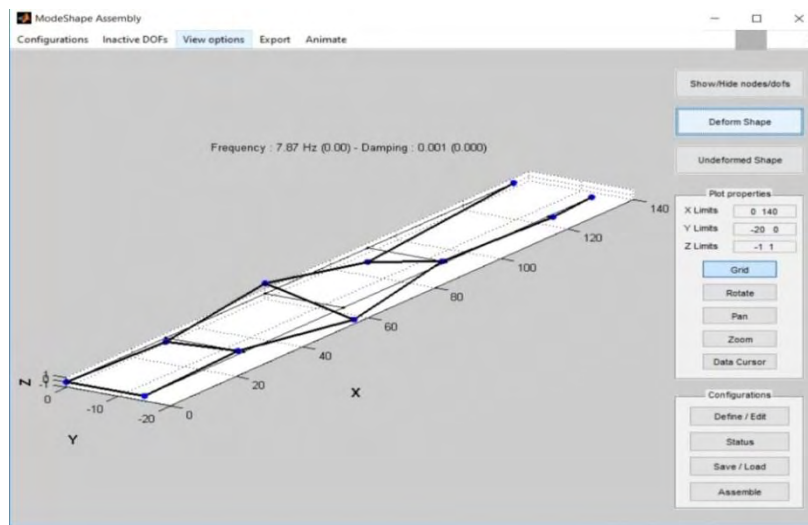


Figure 4.51. Oscillation of the bridge model for  $f=6.71095$  Hz (front view)

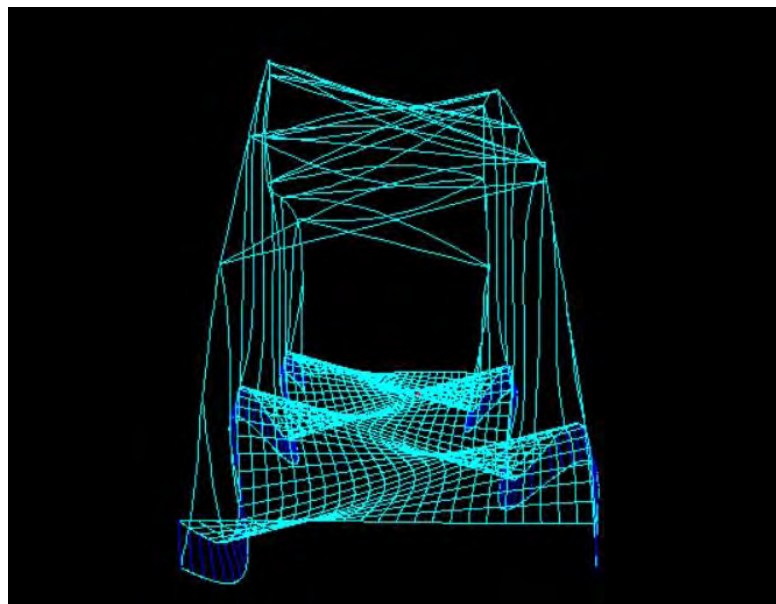


Figure 4.52. Oscillation of the bridge model for  $f=7.14691$  Hz (front view)

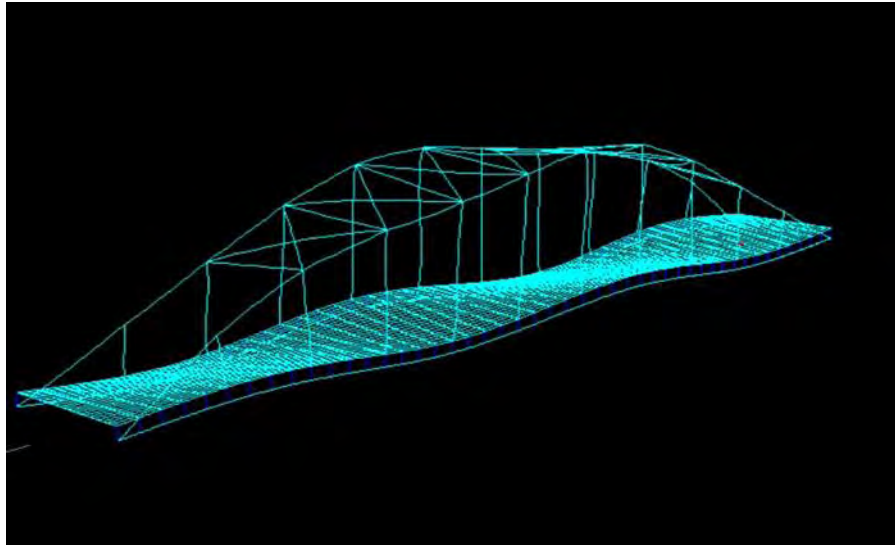


Figure 4.53. Oscillation of the bridge model for  $f=7.14691$  Hz (original view)

The approach of the mode shape of the finite element model was successful given that the two mode shapes are similar. The measured frequency is close to that of the finite element model ( $f_{\text{measurements}}/f_{\text{FEM}}=7.8703\text{Hz}/7.14691\text{Hz}=1.1$ ). Our estimation regarding the distribution of the components of the mode shape was accurate.

## Conclusion

The aim was the identification of the basic mode shapes of the bridge's deck and arched part. As expected, the mode shapes were transverse, longitudinal, flexural and torsional. Despite the fact that a small number of sensors were used our estimation in the basic mode shapes was accurate. The first fifteen measured frequencies were examined and the majority corresponds to the finite element model's frequencies.

- $f=1.1332$  Hz is a transverse modal frequency and gives the first identified mode shape.
- $f=1.5004$  Hz is a longitudinal modal frequency and gives the second identified mode shape

- $f=2.05$  Hz is a torsional modal frequency and gives the third identified mode shape.
- $f=2.55$  Hz is a flexural modal frequency and gives the fourth identified mode shape.
- $f=3.29$  Hz is a flexural modal frequency and gives the fifth identified mode shape.
- $f=3.925$  Hz is a transverse modal frequency and gives the sixth identified mode shape.
- $f=4.4403$  Hz is a transverse modal frequency and gives the seventh identified mode shape.
- $f=4.73$  Hz is a transverse modal frequency and gives the eighth identified mode shape.
- $f=5.1036$  Hz is a transverse modal frequency and gives the ninth identified mode shape.
- $f=4675$  Hz is a longitudinal modal frequency and gives the tenth identified mode shape.
- $f=5.25$  Hz and  $6.1003$  Hz are local frequencies and gives the eleventh and twelfth identified mode shape respectively.
- $f=7.4562$  Hz is a flexural model frequency and gives the thirteenth identified mode shape.
- $f=7.8703$  Hz is a transverse model frequency and gives the fourteenth identified mode shape.

The small differences in prices and distributions of the mode shapes of the analytical model with those derived from measurements are due to the well-prepared and correct instrumental monitoring with very sensitive sensors as well as the good model simulation. The measured frequencies and mode shapes gave a good estimation of the model's mode shapes. These modal data and the model's matrices are used in model updating programs.



## **5. Model reduction**

### **5.1 Introduction**

Many modern mathematical models of real-life processes pose challenges when used in numerical simulations, due to complexity and large size (dimension). Model order reduction aims to lower the computational complexity of such problems, for example, in simulations of large-scale dynamical systems and control systems. By a reduction of the model's associated state space dimension or degrees of freedom, an approximation to the original model is computed which is commonly referred to as a reduced order model. Reduced order models are useful in settings where it is often unfeasible to perform numerical simulations using the complete full order model.

Many software requires as inputs the stiffness and mass matrices of the model's components as well as the interface degrees of freedom of the components, the internal degrees of freedom of each component and the boundary degrees of freedom for each component. Thus we develop a software in order to extract this information.

### **5.2 The kinds of the degrees of freedom of a component**

Degrees of freedom(DOFs) are a set of independent displacements/rotations that completely define the displaced position of the mass with respect to its initial position. It is the number of parameters that determine the state of a system.

The DOFs of a structure are divided into internal and boundary DOFs. The boundary DOFs of a component are the DOFs that exist in the interface of a component with the other components. The internal DOFs of a component are the DOFs that exist where there is no interface with other components.

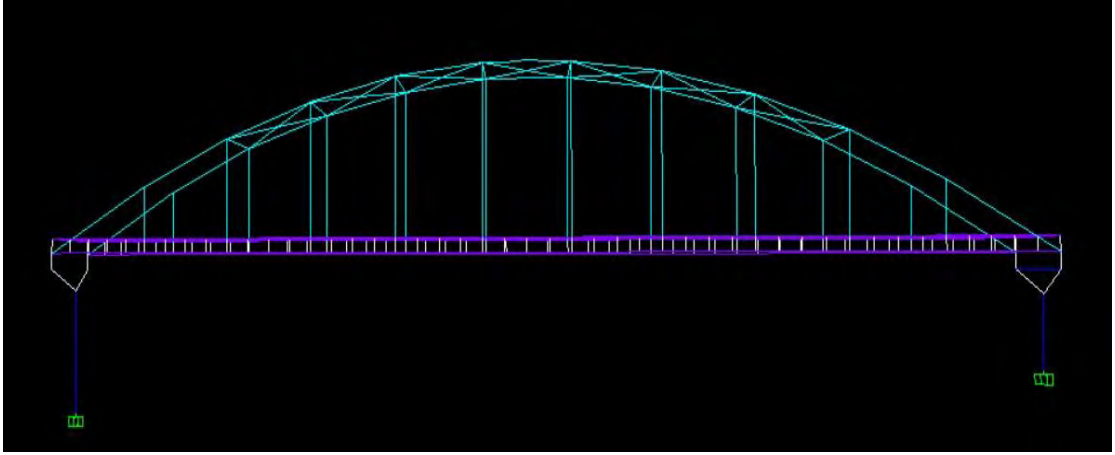


Figure 5.1. The bridge Model

For example the deck's boundary DOFs(as we see in figure 4.59) are in the interface among the deck,the arched part and the piers. The internal DOFs are where there is no interface.

### 5.3 Calculation of the necessary degrees of freedom needed in model reduction( The presentation of the software)

We have calculated the mass and stiffness matrices from each part of the model using the previous code. Now, the next information we need is the boundary, internal and interfacial degrees of freedom from each part of the structure.

The nodes' displacements in the interface are the same for each part that is involved. Thus, by comparing the stiffness matrices of each part we can find the boundary DOFs. The common positions where the non zero stiffness values are, reveal the boundary DOFs. The internal DOFs of the component are the rest DOFs that are not involved in this comparison.( the diagonal of the stiffness matrix contains non-zero elements in the degrees of freedom of the nodes of the component).

The internal, boundaries and interface DOFs are calculated based on the comparison of  $K_i$  (stiffness matrices). Internal, boundaries, interfacial DOFs and  $K_i$  are important data used as inputs in model reduction softwares in order for

a simplified model to be created. This helps us avoid the computational complexity of huge structures.

We develop a software in order to extract the necessary DOFs based on the comparison of stiffness matrices. The software is listed below:

```
% % %% % %% % %% % %% % %% % %% % %% % %% % %% % %% % %% % %% % %%
% %
% % % Interface,internal,boundary DOFS for fixed joints% % %% % %%
% %% % %%
% % % % % % % % % % % % %% % %% % %% % %% % %% % %% % %% % %%
% %
% % % % % the non define variables are resulted from the code
sap_mat
% % % % % %% % % % % % %% % % % % %
% % % % % % % interface dofs% % % % % %
% % % % % %% % % % % % %% % % % % %
% last2 is the total dofs or the size of the K matrix
step=1;
counter=1;
parts=1;
plus=0;
for iel=step:number_mat-1;
    for ie=(step+1):number_mat;
        for jk=1:last2;
            % the interface dofs exists where the digonals of the
stiffness
            % matrices of two components are non-zero
            if SMI{iel}(jk,jk)~=0&&SMI{ie}(jk,jk)~=0;
                dofs{parts}(1,counter)=jk;
                counter=counter+1;
                plus=plus+1
            end
        end
        if counter~=1
            name1=mat_name(iel);
            name2=mat_name(ie);
            pavla='-';
            str=strcat(name1,pavla,name2);
            interface_dofs{parts,1} = str;
            interface_dofs{parts,2} =dofs{parts}

            parts=parts+1
        end
        counter=1
    end
    step=step+1
end
% % % % % %% % % % % % %% % % % % %
% % % % % % % boundary dofs% % % % % %
% % % % % %% % % % % % %% % % % % %
step=1;
counter=1;
parts=1;
l1=0;
for iel=step:number_mat;
    for ie=1:number_mat;
        if ie~=step
```

```

        for jk=1:last2;
            % the boundary dofs exists where the digonals
of the stiffness
% matrices of the components are non-zero.Sometimes
boundary and
% interface dofs are the same
        if SMi{iel}(jk,jk)~=0&&SMi{ie}(jk,jk)~=0
            b_dofs{parts,1}=mat_name(iel);
            b_dofs{parts,2}(1,counter)=jk;
            counter=counter+1;
        end
    end
    if counter~=1
        l1(parts)=counter-1
    end
end
end
counter=1
parts=parts+1
step=step+1
end
% We compare all the stiffness matrices from all the components in
order to find the boundary dofs so
% same elements are appeared more than ones.We keep the unique
dofs
for iel=1:number_mat
    boundary_dofs{iel,1}=mat_name(iel)
    boundary_dofs{iel,2}=unique( b_dofs{iel,2})
end
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % internal dofs% % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % %
    parts=1
    counter=1
    for ie=1:number_mat
        for jk=1:last2
            if SMi{ie}(jk,jk)~=0
% find all dofs(total for each component)
                all_dofs{parts,1}=mat_name(ie);
                all_dofs{parts,2}(1,counter)=jk;
                counter=counter+1;
            end
        end
    end
    if counter~=1
        parts=parts+1
    end
    counter=1
end
% number_mat=number of materials(materials with same properties
and different name are received as different)/components
for l=1:number_mat
% the dofs that are different from the boundary dofs are the
internal
% dofs
    internal_dofs{l}=setdiff(all_dofs{l,2},boundary_dofs{l,2})
end

```

The variables `mat_name`, `number_mat`, `last2` and `SMi` are calculated in the main software which is presented in section 2 and in Appendix A.

## **5.4 Conclusion**

The degrees of freedom of each part of the structure as well as the calculated stiffness and mass matrices consist important data (inputs) for model reduction. Softwares dealing with model reduction use these data as inputs in order for a simplified model to be created.

# **6. Conclusions**

## **Model updating**

By using the code we developed and by analyzing the modeshapes and modal frequencies that come from measurements, we have acquired the necessary data to be introduced into a model updating program. We have acquired the mass and stiffness matrices and the values of the parameters  $\theta$  that are related with the finite element model as well as the measurement results. The aim is to use the model updating to solve the eigenvalue problem, previously mentioned, and the its results, for the appropriate value of  $\theta$ , to be close to those of the measurements

## **Model reduction**

With the help of the program developed for the bridge model, the internal, boundary and interfacial degrees of freedom of each part of the model were calculated. These degrees of freedom along with the mass and stiffness matrices of each parameterized part of the structure are the input data into a model reduction program

## Appendix A: Software for extraction of the model's stiffness and mass matrices in SAP2000

### Part 1: Inputs

```
% % % % % insert material properties for each material in sap2000
number_mat=input('number of structure material/components ' )
for i=1:number_mat

prompt = {'material name:', 'modulus of elasticity:', 'coefficient
of thermal expansion:', 'poisson ratio:', 'density:'};
title = 'material properties';

r = inputdlg(prompt,title,[1 60]);
mat_name{i}=r{1};
E(i)= str2num(r{2});
A(i) = str2num(r{3});
Nu(i) = str2num(r{4});
density(i)=str2num(r{5});
end
```

### Part 2: Main body

```
% pass data to Sap2000 as one - dimensional arrays
feature ('COM_SafeArraySingleDim', 1);
% pass non - scalar arrays to Sap2000 API by reference
feature ('COM_PassSafeArrayByRef', 1);
% create Sap2000 object
% ginetai to interface
SapObject = actxserver('CSI.SAP2000.API.SapObject');

SapModel = SapObject . SapModel ;
ret = SapObject.ApplicationStart();

% give the name of sap2000 file that is studies
name=input('dwse onoma arxeiou: ' , 's')

ret=SapObject.SapModel.File.OpenFile(['C:\Users\Spyros\Desktop\'
,name, '.sdb'])
ret=SapModel.SetModelIsLocked(0);

for k=1:number_mat

    Enew(k)=E(k);
    density_new(k)=density(k);
%     pass material properties

ret=SapModel.PropMaterial.SetMPIsotropic(char(mat_name{k}),E(k),Nu
(k),A(k));
```

```

%      pass mass density

ret=SapModel.PropMaterial.SetWeightAndMass(char(mat_name{k}),1,
density(k));
end

%  unlock the model
ret=SapModel.SetModelIsLocked(0);
%  define load case in order to extract model matrices.
%We have to define a load case dead or modal for the extraction of
the matrices.
%that is asked by sap2000. There is no difference in the matrices'
values
ret = SapModel.Analyze.SetSolverOption_1(1, 0,1,  'DEAD')
%  analyze the model
ret = SapModel.Analyze.RunAnalysis();
% import data from M file
Mfilename=[name, '.TXM'];

mass = importdata(Mfilename);
mass_matrix=mass.data;
% import data from Stiffness file
Sfilename=[name, '.TXK'];
s = importdata(Sfilename);
% Kt has 3 columns ,number of row and column,stiffness value
Kt=s.data;
% % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % %
% % % % % % % % % % % bring the stiffness matrix to its original
form % % % % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % %
% matrix length
mhkos=length(Kt)
% first 2 columns are the number of the row and column and the
third the stiffness value
% the last number of the column is the size of the matrix given
that the
% diagonal elements of the global stiffness matrix are non zero
thus the
% last element of the matrix describe its size
% read the coordinates of the last element
last1=Kt(end,2);
Ktotal=sparse(zeros(last1));

% we find the non zero elements and keep their coordinates

for p=1:mhkos
    if Kt(p,3)~=0
%         if Kt(p,3)=0 then the values of i(p),j(p) are zero
            i(p)=Kt(p,1);
            j(p)=Kt(p,2);
        end
    end
%
for m=1:mhkos
    % if Kt(p,3)=0 then the values of i(p),j(p) are zero so for
the non
    % zero i,j(where the non zero values on stiffness are) we save
the

```







```

title = 'value of parameter  $\theta$  for ';
th = inputdlg(prompt,mat_name{param(q)},[1 60]);
%      b is  $\theta$ 
b(q)=str2num(th{1})

%      we kept the parameterized material.  $\theta$  for others materials
is going
%      to be zero.this changes in every loop.only the material
that is
%      analyzed (in every loop) has no zero value of  $\theta$ .
for l=1:meter
    if l~=q
        b(l)=0;
    end
% multiple the modulus of elasticity of each material with  $\theta$ 
    Enew(param(l))=E(param(l))*b(l);

end

%      for every material pass the value of new modulus of
elasticity in
%      sap2000
for t=1:meter

ret=SapModel.PropMaterial.SetMPIsotropic(char(mat_name{param(t)}),
Enew(param(t)),Nu(param(t)),A(param(t)));
end
%      analyze the model
ret = SapModel.Analyze.RunAnalysis();
%      import stiffness matrix and bring it to its normal form
Sfilename=[name,'.TXK'];
s = importdata(Sfilename);
K_new=s.data;
mhkos1=length(K_new);
last2=K_new(end,2);
K_i=sparse(zeros(last2));
sum=0

k=0
K_new_end=s.data(:,end);
for r=1:mhkos1
    if K_new(r,3)~=0
        metrhths=r;
        i(r)=K_new(r,1);
        j(r)=K_new(r,2);
    end
end

for n=1:metrhths
    if i(n)j(n)~=0
        K_i(i(n),j(n))=K_new(n,3);
    end
end

K_i=(K_i+(tril(K_i,-1)).');
% The Ki matrices are created

Ki{q,1}=mat_name(param(q));

```



```

for ww=1:mhkos_m0
    if mass_matrix0(ww,3)~=0
        x0(ww)=mass_matrix0(ww,1);
        z0(ww)=mass_matrix0(ww,2);
        metrhths0=ww
    end
end
for uu=1:metrhths00
    if x0(uu),z0(uu)~=0
        Mo(x0(uu),z0(uu))=mass_matrix0(uu,3);
    end
end
end
if meter==number_mat
    Mo=sparse(zeros(last_m));
end
% % % % % % % % % % % % % % calculation of Mi % % % % % % % % % % %
% % % %
if meter~=0
    for kk=1:meter
        ret=SapModel.SetModelIsLocked(0);
        % define the value of f where f is  $\phi$ 
        prompt = {'the value of parameter  $\phi$ :'};
        title = 'value of parameter  $\phi$  for ';
        ff = inputdlg(prompt,mat_name{param(kk)},[1 60]);

        f(kk)=str2num(ff{1})
        % the value  $\phi$  that is not zero is the only one that relating
        with the material that is parameterized in this loop
        for ll=1:meter
            if ll~=kk
                f(ll)=0;
            end
        % multiple the density with  $\phi$ 
            density_new(param(ll))=density(param(ll))*f(ll);
        end
        for tt=1:meter
        % pass the parameterized densities in Sap2000

ret=SapModel.PropMaterial.SetWeightAndMass(char(mat_name{param(tt)
}),1, density_new(param(tt)));
        end
        % analyze the model
        ret = SapModel.Analyze.RunAnalysis();
        % extract Mi and bring them to their normal form
        Mfilename=[name, '.TXM'];
        mass1= importdata(Mfilename);
        mass_matrixNew=mass1.data;
        mhkos_m1=length(mass_matrixNew);
        last_m1=mass_matrixNew(end,2);
        M_new=sparse(zeros(last_m1));
        metrhths1=0
        for yy=1:mhkos_m1
            if mass_matrixNew(yy,3)~=0
                xn(yy)=mass_matrixNew(yy,1);
                zn(yy)=mass_matrixNew(yy,2);
                metrhths1=yy
            end
        end
    end
end

```



## **Appendix B: Explanation on how to use the SAP2000 API and how the SAP2000 API functions are documented.**

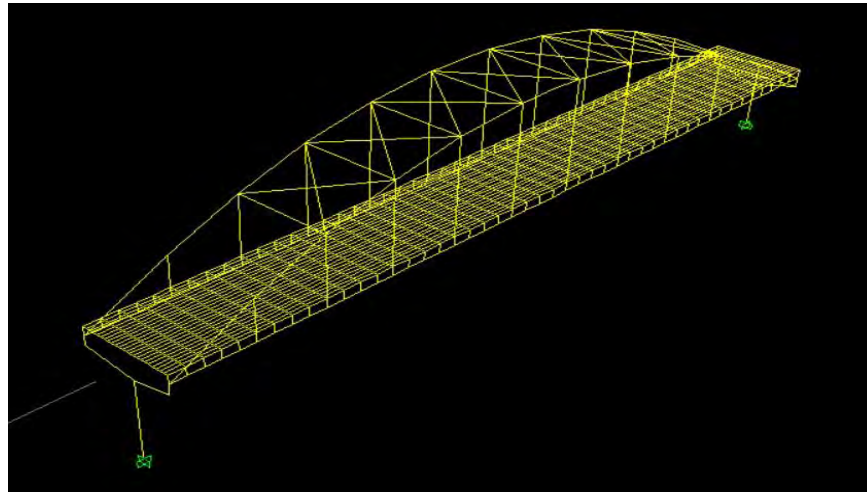
All the used SAP2000 API functions are listed here

- `SapObject = actxserver('CSI.SAP2000.API.SapObject')` . This function starts the interface between the two programs.
- `SapModel= SapObject.SapModel` Creates the SapModel object
- `SapObject.ApplicationStart`. This function starts the Sap2000 application.
- `SapObject.SapModel.File.OpenFile`. This function opens an existing Sap2000 file. The file name must have an sdb, \$2k, s2k, xls, or mdb extension. Files with sdb extensions are opened as standard Sap2000 files. Files with \$2k and s2k extensions are imported as text files. Files with xls extensions are imported as Microsoft Excel files. Files with mdb extensions are imported as Microsoft Access files.
- `SapObject.SapModel.SetModelIsLocked`. This function unlocks the model. With some exceptions, definitions and assignments can not be changed in a model while the model is locked. If an attempt is made to change a definition or assignment while the model is locked and that change is not allowed in a locked model, an error will be returned.
- `SapObject.SapModel.PropMaterial.SetMPIsotropic`. This function sets the material directional symmetry type to isotropic, and assigns the isotropic mechanical properties(The modulus of elasticity, Poisson's ratio, The thermal coefficient)
- `SapObject.SapModel.PropMaterial.SetWeightAndMass`. This is either 1 or 2, indicating what is specified by the Value item.
  - 1 = Weight per unit volume is specified
  - 2 = Mass per unit volume is specified

If the weight is specified, the corresponding mass is program calculated based on the specified weight. Similarly, if the mass is specified, the corresponding weight is program calculated based on the specified mass.

## **Appendix C: Instructions for the identification of Model's properties in SAP2000**

### **A. The identification of component's material and their properties in SAP2000**



**Figure C.1.** The original model

#### **Step 1: view the colors**

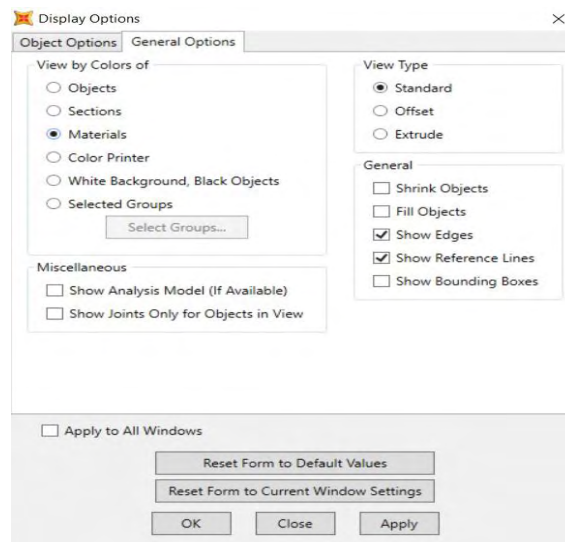
To identify the different parts of the structure the user can click on Display options toolbar button>General options. The options on this form can be used to selectively display various features associated with objects in the model.

**View by Color of** drop-down list. This list can be used to specify that the model be displayed using Colors associated with Objects (as assigned using the Options menu > Graphics Colors > Display command), Section properties (as defined using the Define menu > Section Properties subcommands), Material properties (as defined using the Define menu > Materials command), Groups (see next bullet), Frame Design Type (as determined from the object, section, and material properties), or full color or black and white "color" using a color printer. By clicking on materials the user can see the different materials in the structure(this assumes that every material has different color).(figure C.2)

By selecting view by color of materials the user can see all the different materials used by the model.(figure C.3)



(a)



(b)

Figure C.2. Display Options

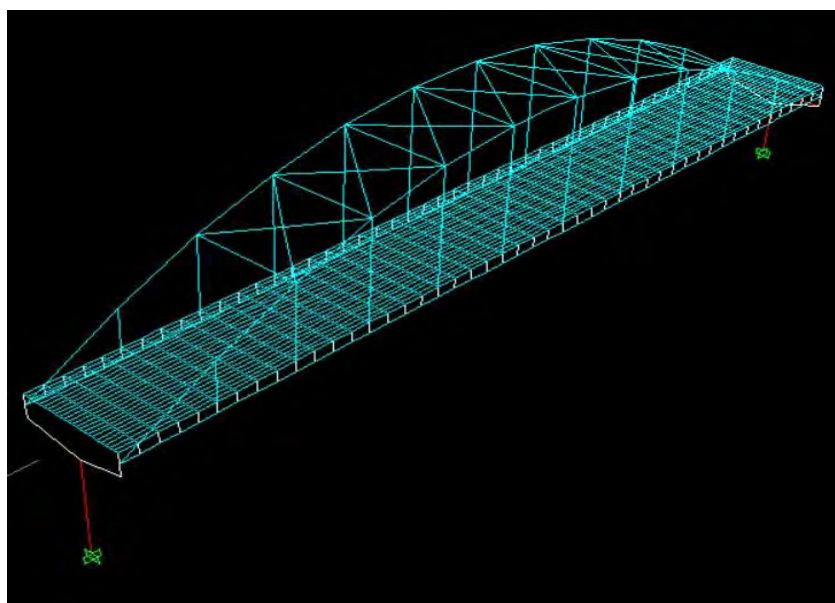


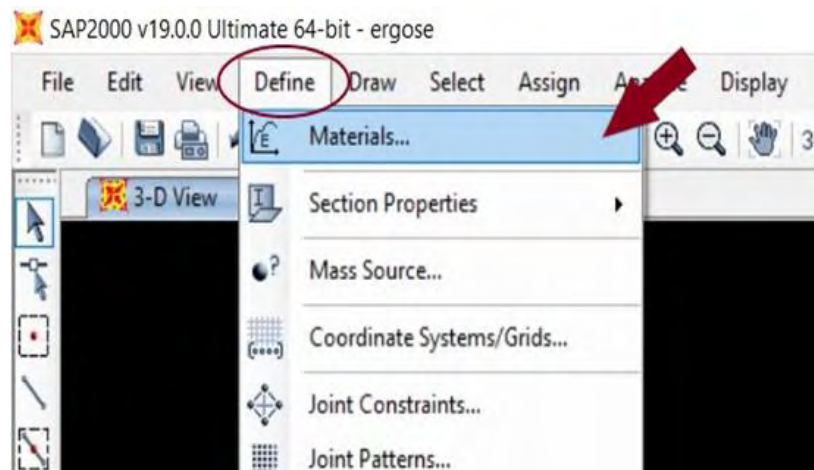
Figure C.3. The materials of the structure (viewed by color)

It is obvious that there are three different materials in the structure.

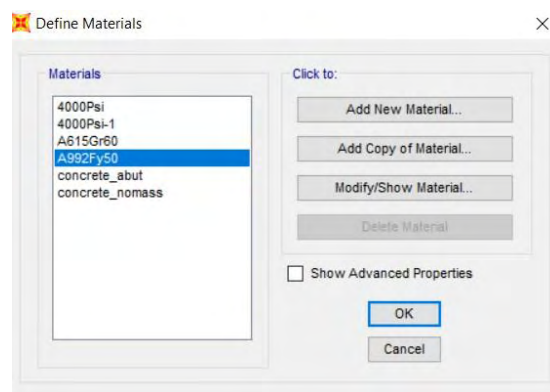


## **Step 2: Identify the used materials**

In the Define menu>Material the user can see the materials that exist in Sap2000. In Modify/Show Material section, he can see the properties of the materials and understand which material is used in the model(depending on its color).(figures C.4 , C.5)



**Figure C.4.** The Define Menu



**Figure C.5.** The Materials

**Material Property Data**

**General Data**

Material Name and Display Color: A992Fy50

Material Type: Steel

Material Notes: Modify/Show Notes...

**Weight and Mass**

Weight per Unit Volume: 75,9729

Mass per Unit Volume: 7,849

Units: KN, m, C

**Isotropic Property Data**

Modulus of Elasticity, E: 1,999E+08

Poisson, U: 0,3

Coefficient of Thermal Expansion, A: 1,170E-05

Shear Modulus, G: 76903069,

**Other Properties for Steel Materials**

Minimum Yield Stress, Fy: 344737,9

Minimum Tensile Stress, Fu: 448159,3

Expected Yield Stress, Fye: 379211,7

Expected Tensile Stress, Fue: 492975,2

☐ Switch To Advanced Property Display

OK Cancel

**Figure C.6.** The properties of A992Fy50 material

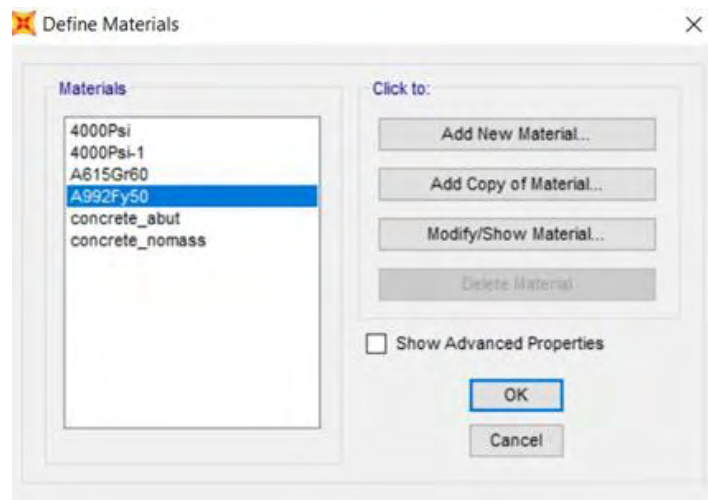
## B. Separation of parts in SAP2000

### Step 1: Select the parts

Sometimes we have to separate some parts of a structure in order to examine them. As we saw in figure 1 the deck and the arched part consists a component. We want to separate these two components in SAP2000 in order to calculate the stiffness and mass matrix of each part. For example we select to modify the deck.

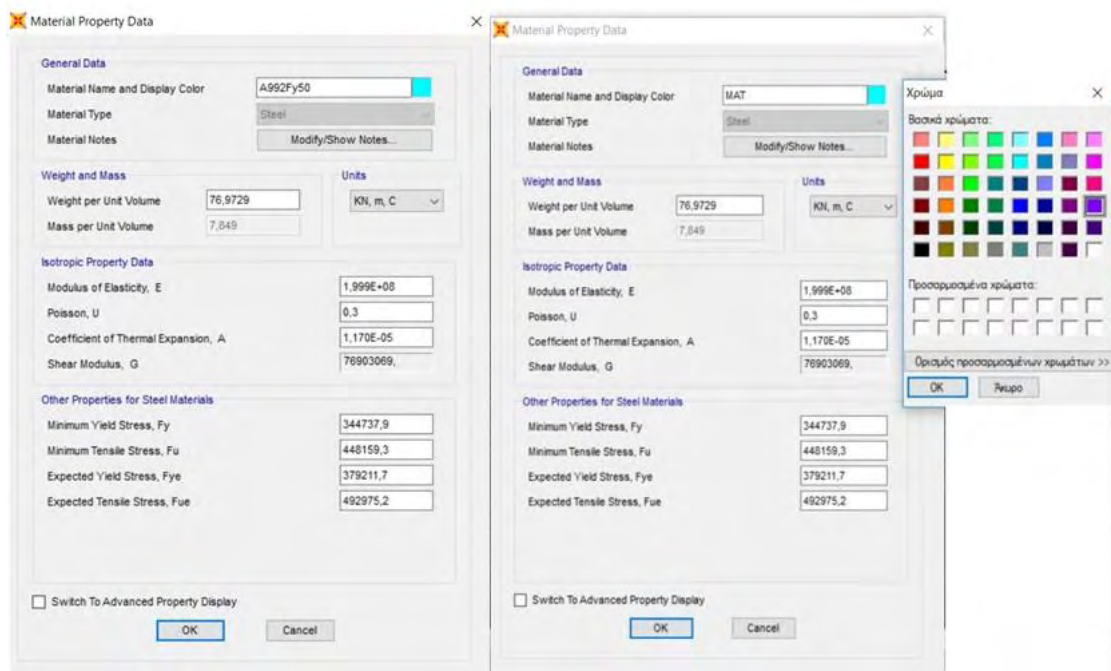
### Step 2: Separation process

The modified part must have the same properties as it had before the modification. Thus, we create a copy of the deck's material. In the Define menu>Material>Add copy of Material a copy of the material is created.(figure C.7)



**Figure C.7.** The deck's material is A992Fy50 so we create a copy of it

The properties of the copied material are the same with the original one (figures C.8).



**Figure C.8.** The properties of the copied material

The copied material should have different name and color from the original one in order to distinguish them.(we select material name MAT and purple color)

Now we change the deck's material from A992Fy50 to MAT. This can be done by selecting an element from the deck ( each element has a frame) . In section property there is the name of element's frame. We can click on the frame's name and after that we change its material to MAT(figures C.9, ).

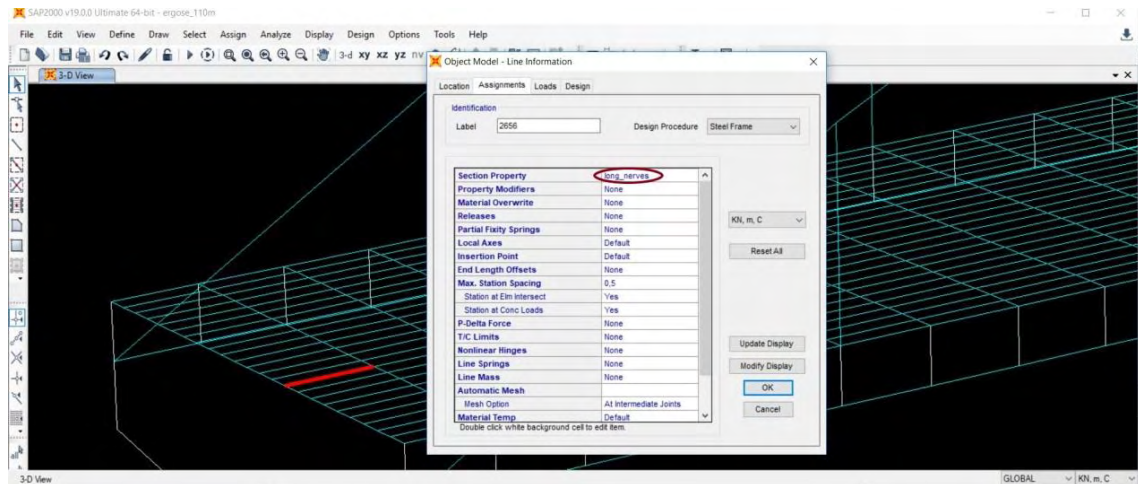


Figure C.9. Click on this element and after that on its frame called long\_nerves

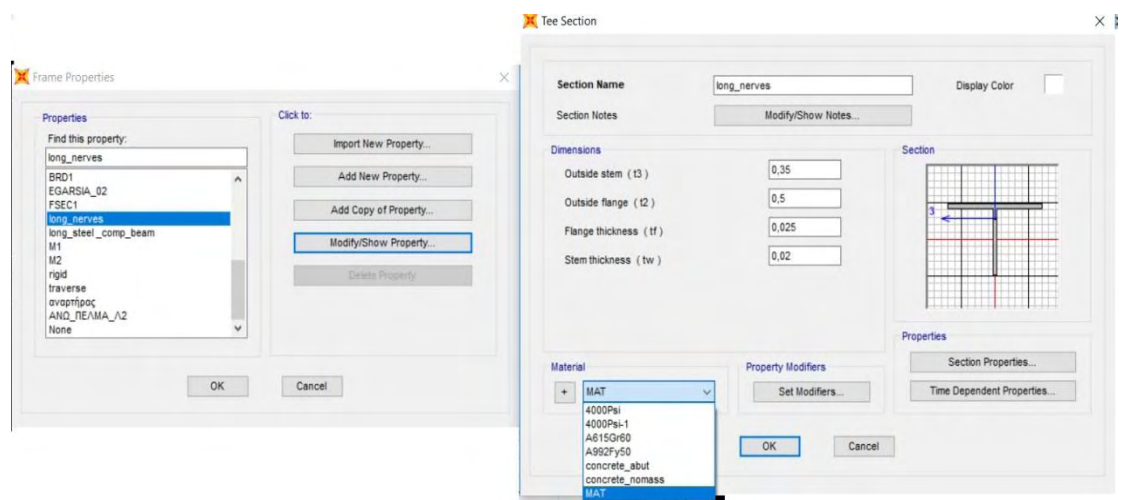


Figure C.10. For this frame we click on Modify/Show Property section and change the material to MAT.





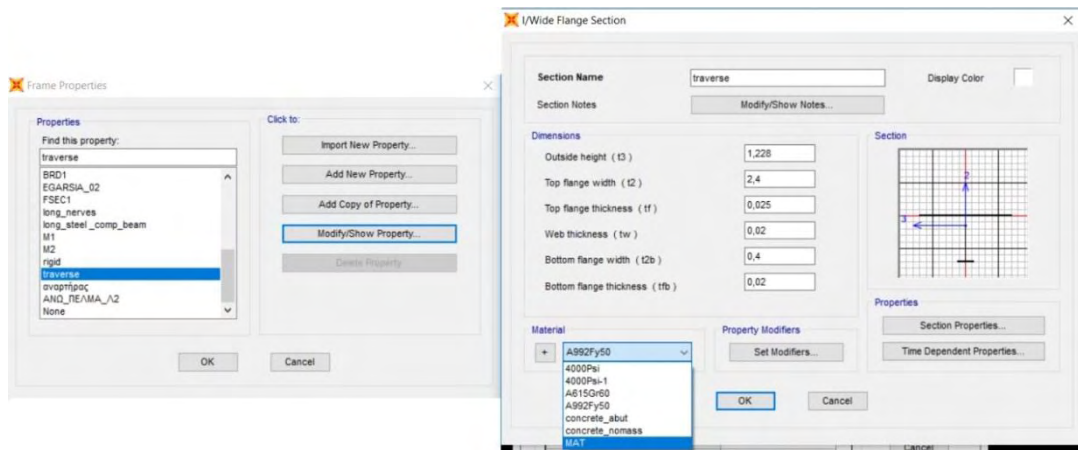


Figure C.13 The frame's material is changed to MAT

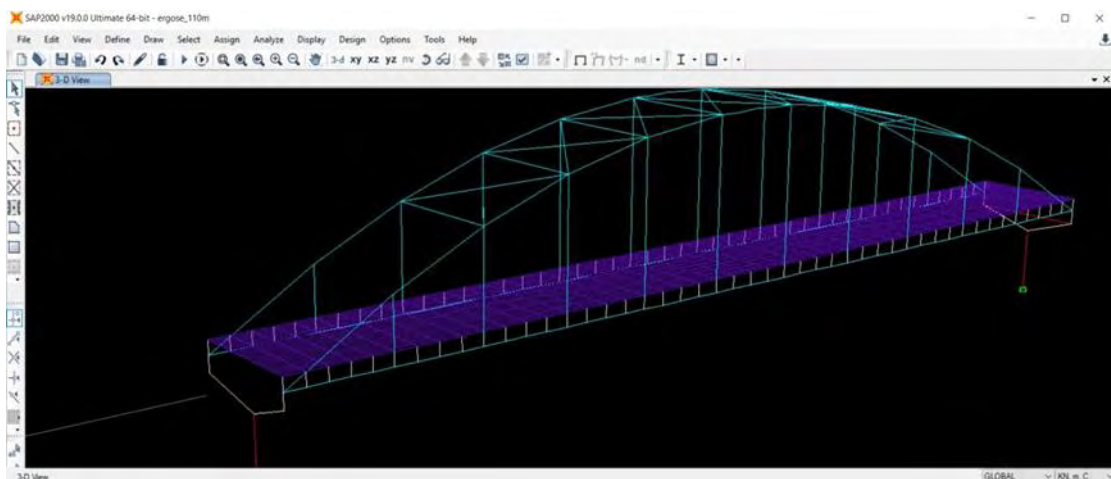


Figure C.14. The figure of the model after modification

One last set of elements should be modified in order to distinguish the deck from the arched part. The process is the same as previously.

The final form of the model is depicted in Figure C.15.

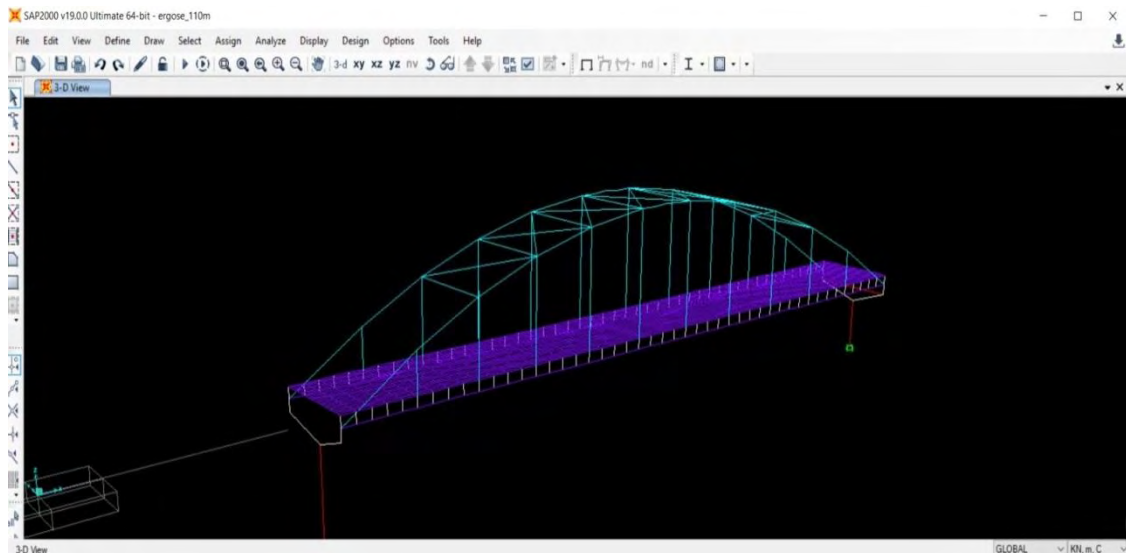


Figure C.15. Model's final form

In the case that we want to study only one element the process is quite different.

## C. Study one element

### Step 1: choose the element

Initially we have to choose which element we want to examine. For example, an element of the deck is selected (Figure C.16)

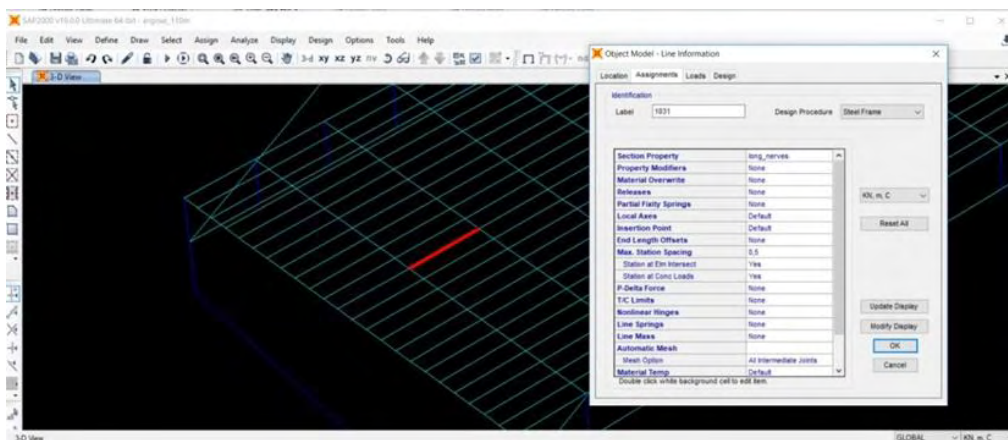


Figure C.16. The selected element

## Step 2: Modification of the element

After the selection of the element, we have to modify it. There many elements with the same properties with the selected one, thus the discrimination of the specific element is based on its frame's modification.

We have to create a copy of its frame by clicking on the frame's name and after that on the Add a Copy of Property section (Figures C.17,C.18).

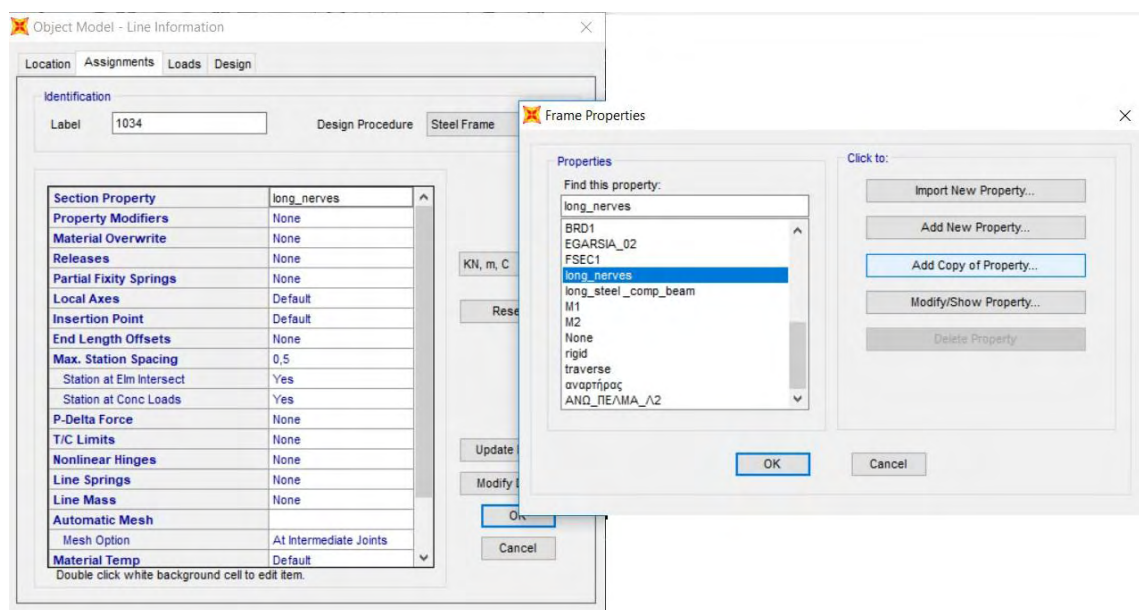


Figure C.17. The frame called long\_nerves is copied

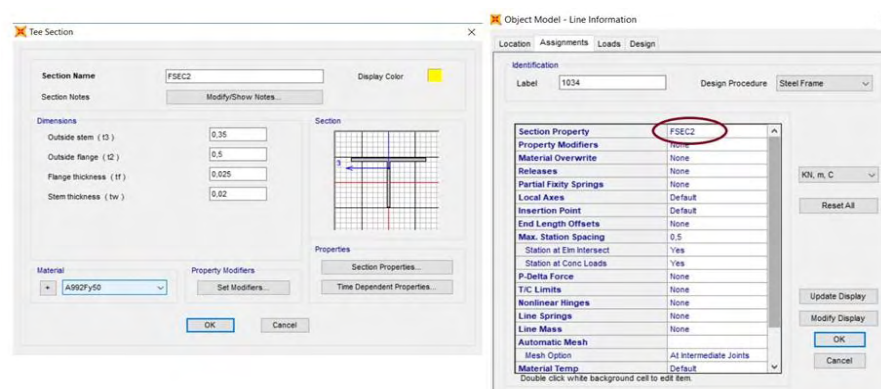


Figure C.18. FSEC2 is a copy of the frame called long\_nerves and replaces long\_nerves frame only in this element.



Next we have to create a copy of the material called A992Fy50 in order to replace it and the discrimination is done (Figure C.19)

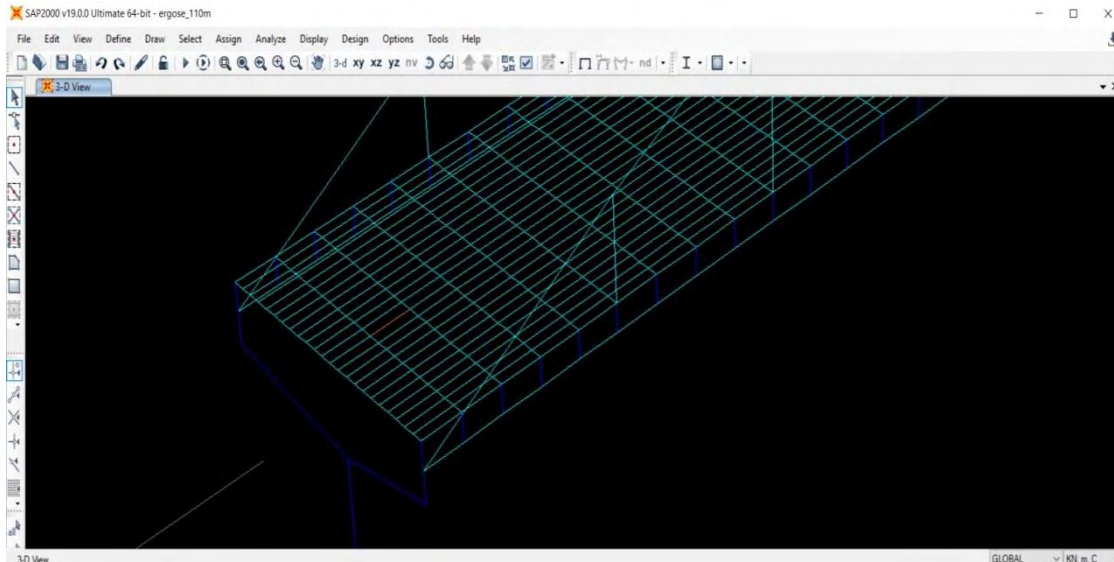


Figure C.19. The selected element after the discretization. (with orange color)

## Conclusion

The example that was presented in this appendix illustrates the parts separation's general idea in SAP2000. The user can separate one part from the others based on this methodology.

This separation helps us to examine each part of the structure. For example, after the deck's and arced part separation we have the ability to extract stiffness and mass matrices from both parts separately.

## Appendix D: Software for Experimental Modal Analysis

The software is written in MATLAB2011a and consists of four independent modulus which are: Data, Pre-Processing, Modal Identification, Post-Processing. The main menu of the software is depicted in figure D.1.

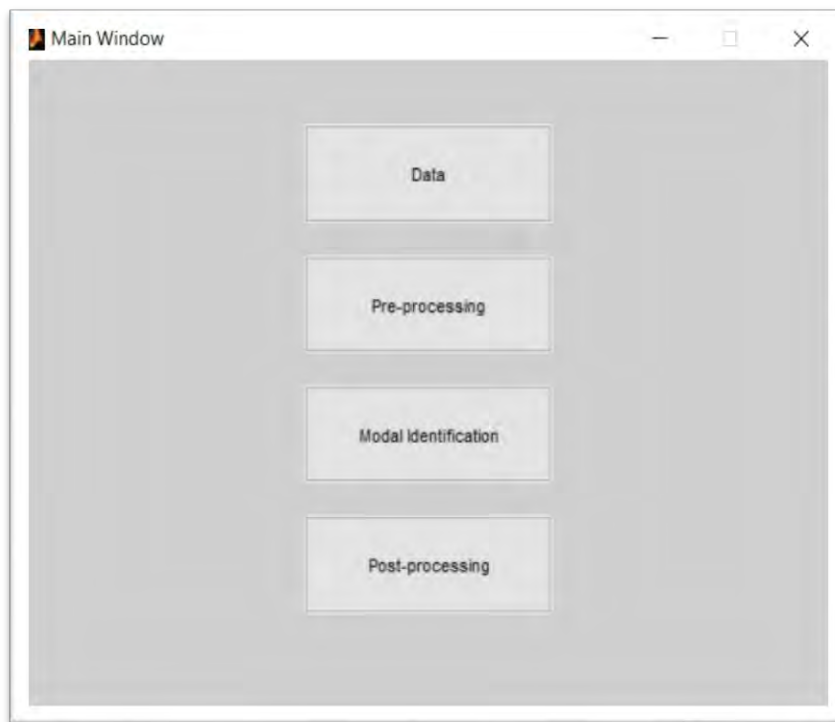


Figure D.1. Main menu

The Data module is used to load data from .mat files which the program is going to process. Also results from previous estimations can be loaded (Figure D.2).

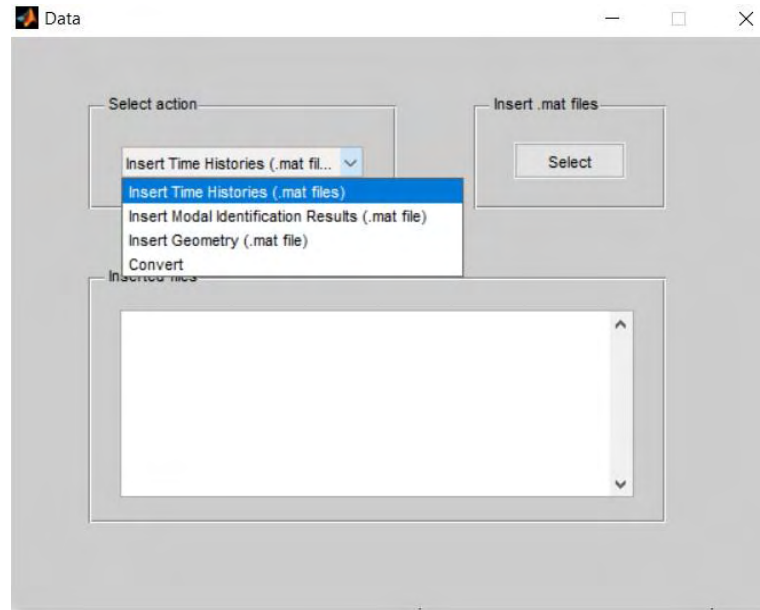


Figure D.2

More specifically the data can be:

- Experimental measurements of the acceleration(Time Histories) of multiple sensors. In the Insert Time Histories section the user must load a .mat file which contains the measured by sensors acceleration, the label of each sensor and the discretization time. The time histories of multiple sensors need to be arranged in the columns of a matrix named **accel**. Each column represents the measurement of the specific sensor. Furthermore the sensors label must be stored in a matrix named **channeltext** and time in a variable named **dt**. It is important that the same names to be followed otherwise the program will not run. An example of 6 sensors is shown in figure D.3.

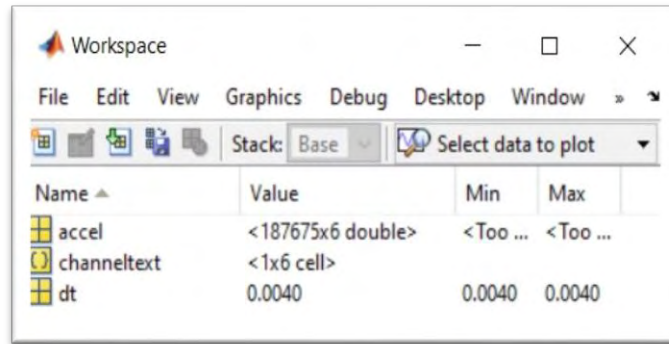


Figure D.3. accel=measurement acceleration, dt=time between measurements, channeltext=label of each sensor

It is possible to load more than one measurements. If we have to measure a large structure as a bridge with a small number of sensors we have to take multiple measurements from different positions. That is called sensor configuration.

- In the **Insert Modal Identification results** section saved results can be loaded from previous sessions in order to not be reproduced.
- In the **Insert Geometry** section the user can load a .mat file that contains variables which define the geometry of the structure. The geometry is a figure of the structure and consists of the nodes ,their degrees of freedom and lines(elements) which connect the nodes. The geometry .mat file must contain a matrix named **node coords** which contains the coordinates of each node, a matrix named **node\_dofs** which contains the degrees of freedom of each node, a matrix **el\_nodes** which is the element connectivity and a matrix named **reference\_dofs** which contains the common reference degrees of freedom. In every sensor configuration we change the location of the sensors but some of them should remain common. The names of matrices should be the same with the above ones in order for the software to run correctly. An example of a bridge geometry

.mat file for the previous measurements is illustrated in figure D.4.

| Name           | Value         | Min | Max |
|----------------|---------------|-----|-----|
| el_nodes       | <52x2 double> | 1   | 36  |
| node_coords    | <36x3 double> | -15 | 110 |
| node_dofs      | <36x3 double> | 1   | 108 |
| reference_dofs | [71,30,45]    | 30  | 71  |

Figure D.4. node\_coords=nodes coordinates, node\_dofs=degrees of freedom of each node, el\_nodes=nodes of each element,reference\_dofs=common sensors

- In **Convert** section measurements from another format can be converted in to a .mat file in order to be used by the software.

After the user loads the necessary .mat files, he or she continue with the Pre-Processing stage where the Power Spectral Density (PSD),the Singular Value Spectrum (SVS) of the ambient acceleration time histories and Time Histories(figure D.5, figure D.6,figure D.7 respectively)can be visually inspected. Such inspection can provide information about the modal frequencies and damping ratios of the structure. In this stage the user gives an estimation of the modal frequencies which are to going to be used in Modal Identification.

Each sensor configuration can be inspected individually in the pre-processing step, and specific channels of a configuration can be selected or de-selected from being used in the Modal Identification process. This feature serves to potentially remove an unwanted sensor from the analysis because of possible bad recording. It is worth mentioning that SVS main merit derives from the fact that it has the ability

to separate the noise from the signal, and that it can reveal closely spaced modes that are not apparent in the PSD.

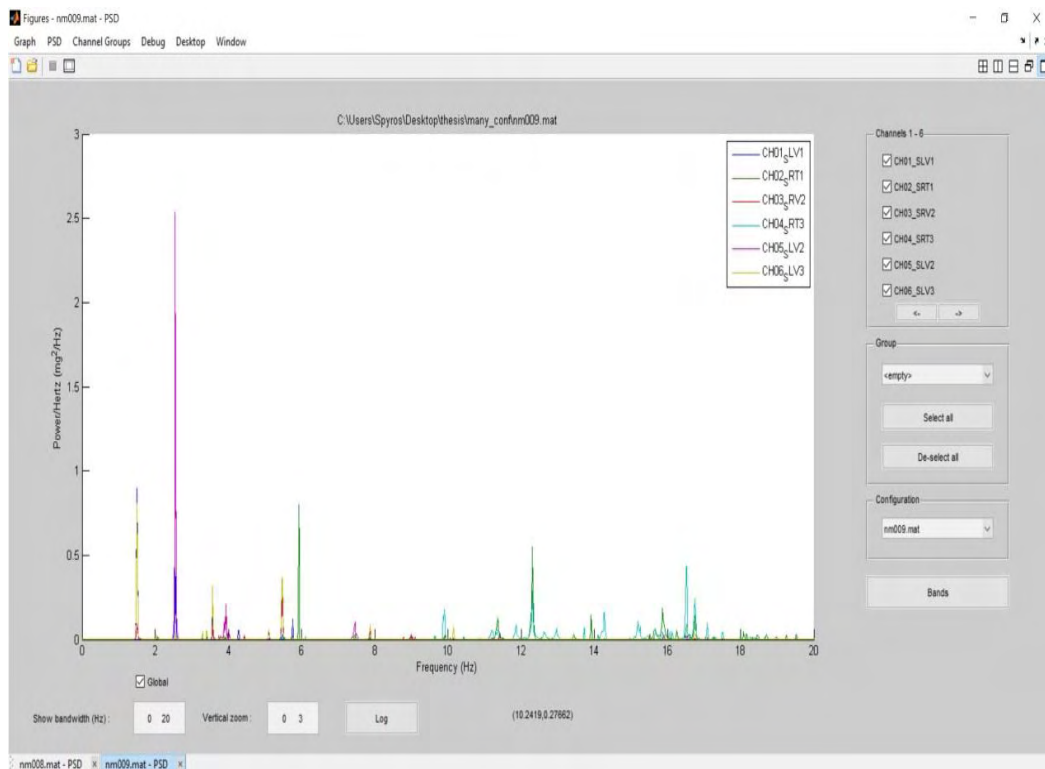


Figure D.5. PSD of a single configuration using each sensor.

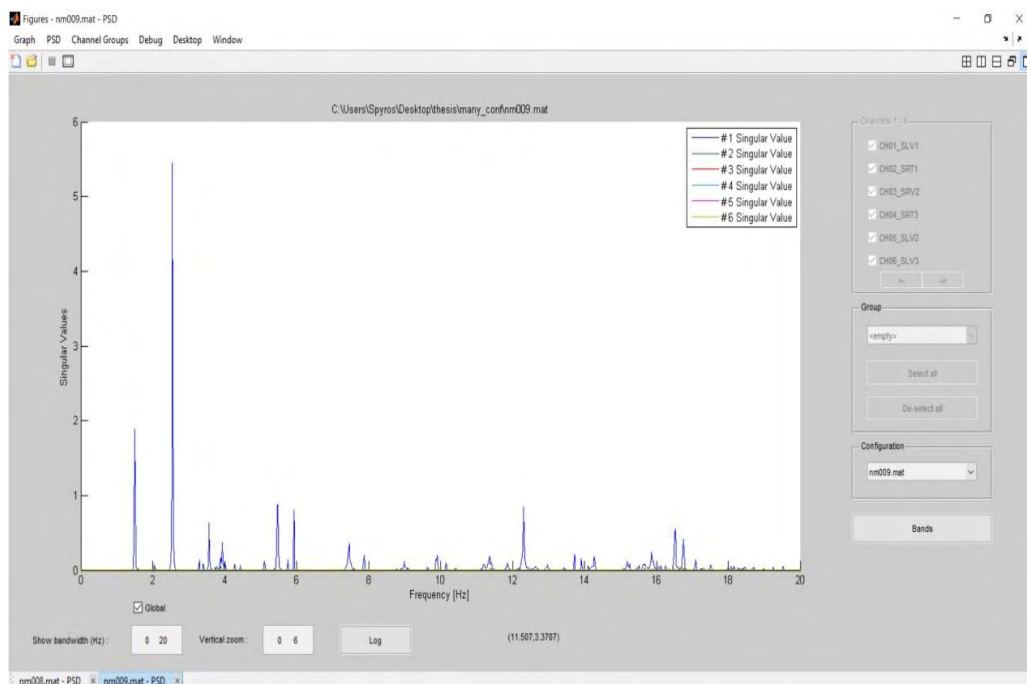


Figure D.6. SVS of a single configuration using each sensor.

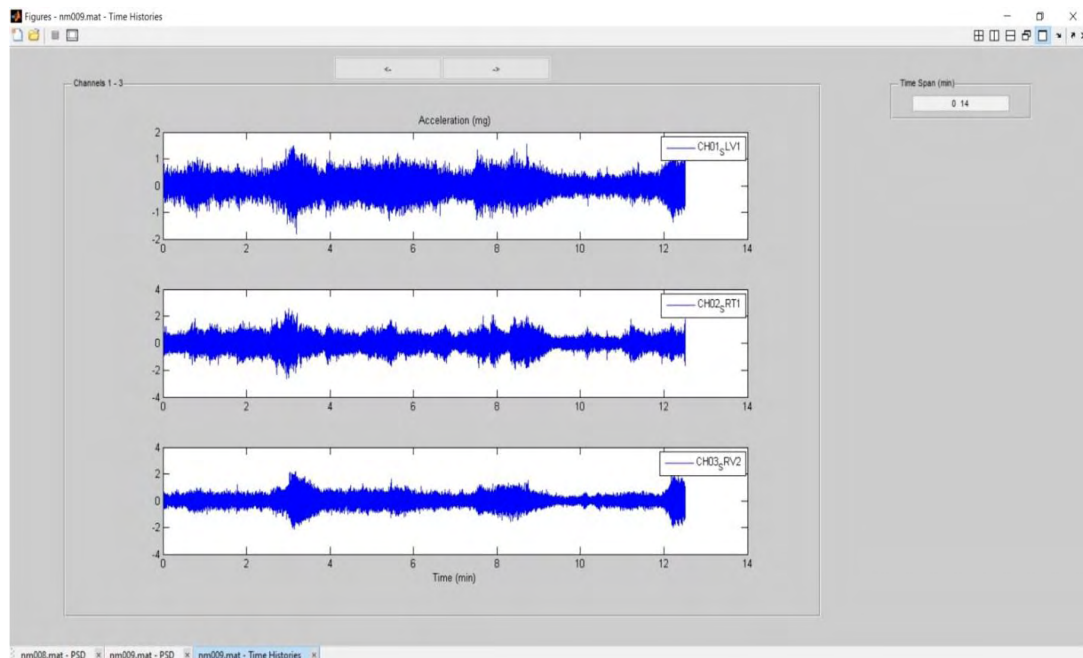


Figure D.7. Time Histories of a single configuration using each sensor.

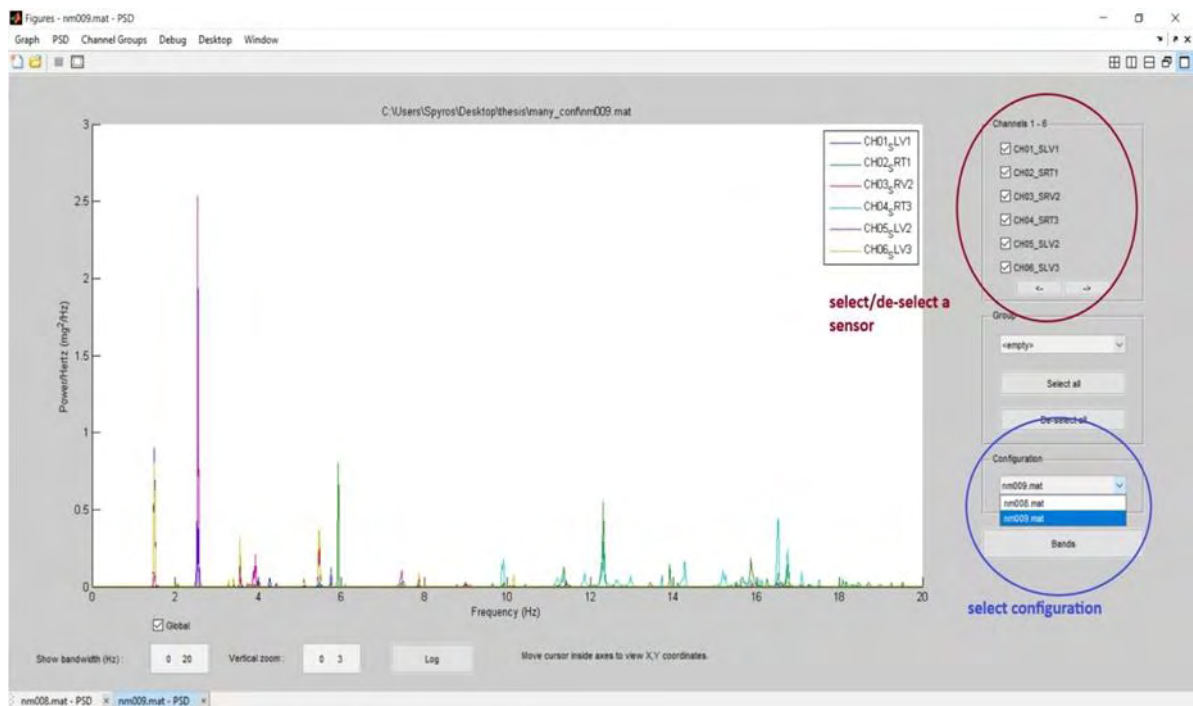


Figure D.8. Selection/de-selection of sensors and selection of configurations.

The Show Bandwidth mode, which is found underneath the graph, gives the ability to zoom in a specific area (figure D.9).



Figure D.9

At the top of the graph there is a **Graph** mode which provides PSD,SVS and Time Histories diagrams and the **PSD** mode provides information about PSD for each sensor (figure D.10 and D.11).

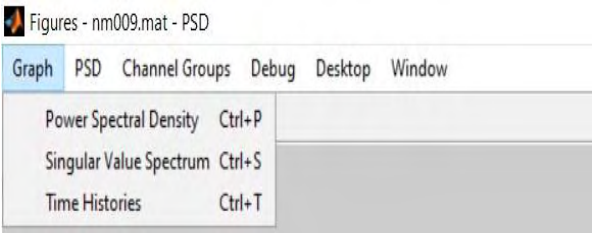


Figure D.10

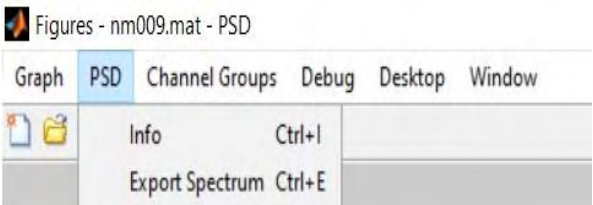


Figure D.11



After obtaining an estimate of the natural frequencies of the structure from observing the SVS or PSD(as it shown in figure D.5 and figure D.6) the user can define the frequency bands which contain a natural frequency .By clicking the Bands button the user can mention the space where one of the natural frequencies is. These estimations can be saved for other analysis. An example is depicted in figure D.12.

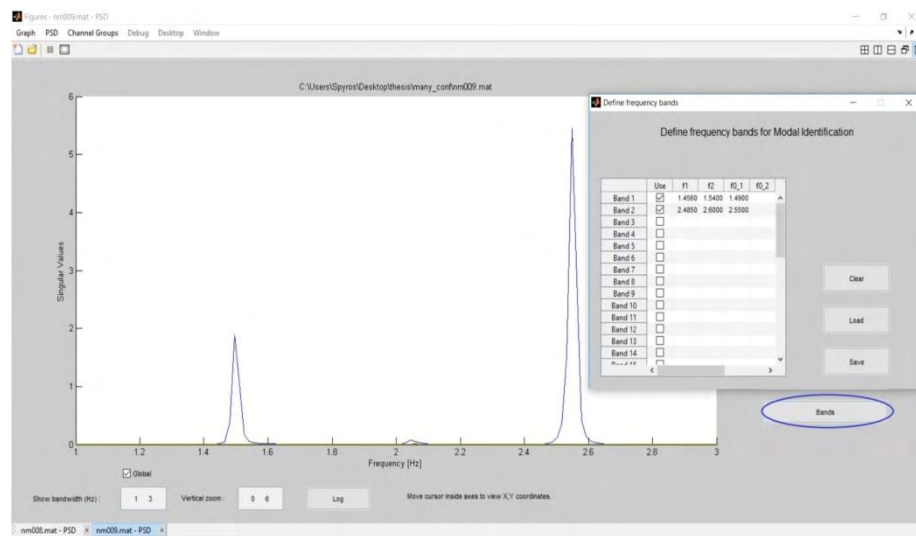


Figure D.12. Frequency bands(an example for two frequencies).

Next step is The Modal Identification which uses a Bayesian methodology in order to extract the modal frequencies, mode shapes, and modal damping ratios from the measured ambient acceleration time histories of each sensor configuration.(figure D.13)

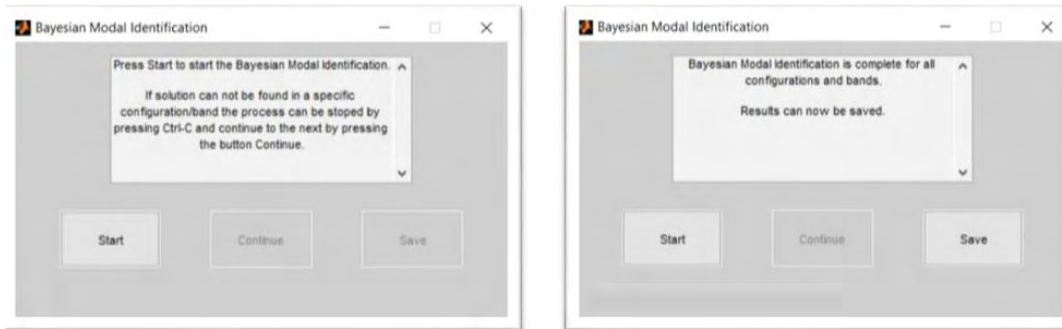


Figure D.13. Modal Identification

After obtaining the modal properties the next step is to visualize the model in Post-Processing.(figure D.14)

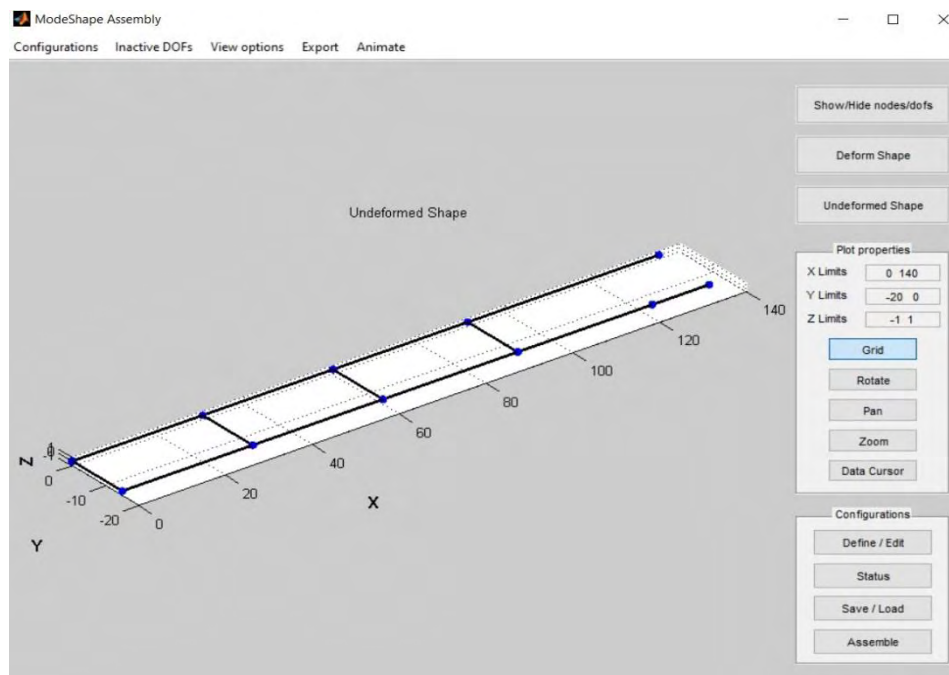


Figure D.14.The Shape is formed for two configurations nm008,nm009

In order to visualize the mode shapes it is necessary to combine all the local mode shapes identified from each configuration to produce the full mode shape at all measured degrees of freedom. The user, must also determine the measured points of the structure as the degrees of freedom in which each sensor measuring for each configuration. This can be done in the Define/Edit window (figure D.15)

In the first column the user selects (or de-selects) the wanted(or unwanted) sensors for each configuration, and in the second column defines the direction of each sensor. After that he selects the nodes where the sensors are and clicks in Select Nodes. These steps defines where the sensors are and in which direction they are measuring. This process can be saved in order to be used in another analysis (figure D.16).

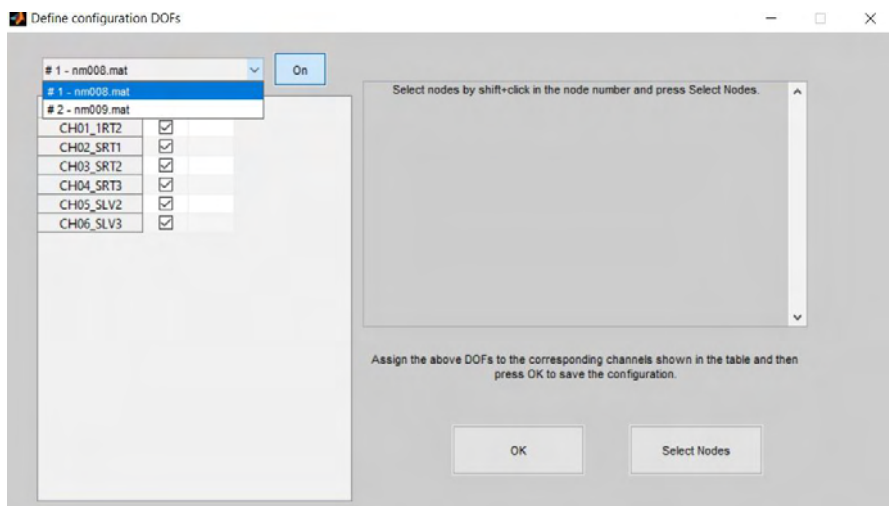


Figure D.15. Define/Edit menu

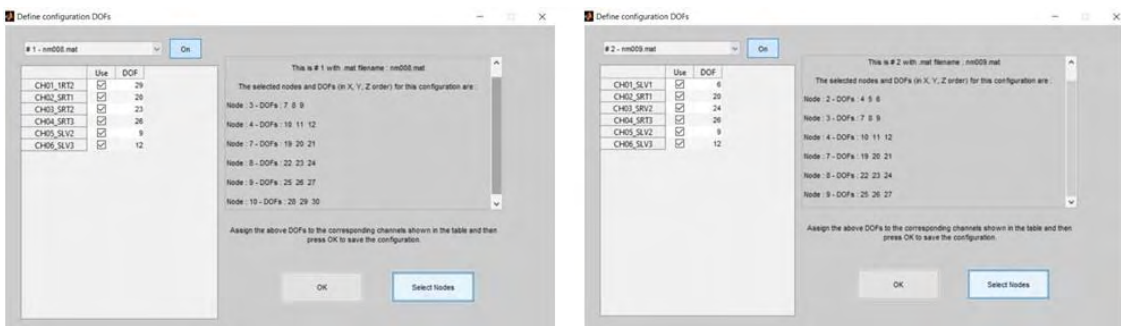


Figure D.16. Selected DOFs and selected nodes

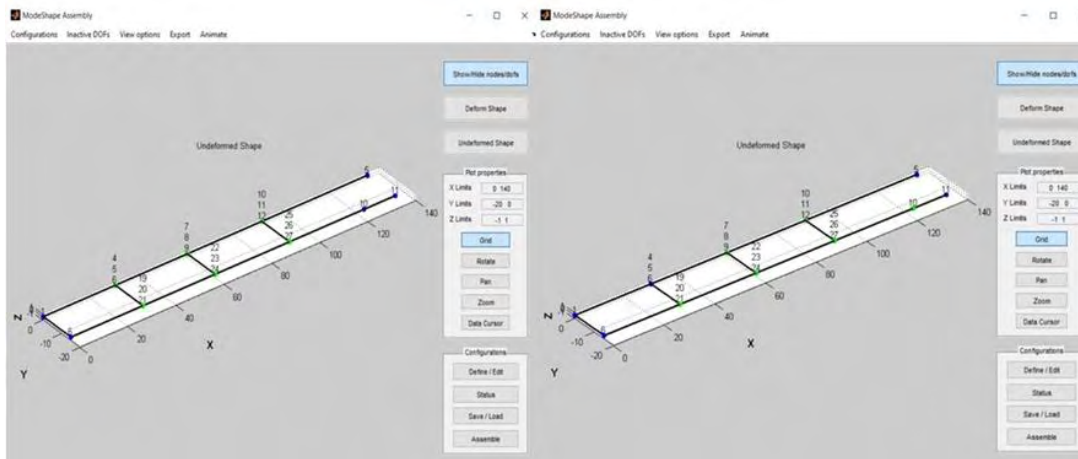


Figure D.17. Nodes where the sensors were placed for each configuration

The status of all the sensor configurations can be viewed from the Status button of the main Post-processing window.(figure D.18)

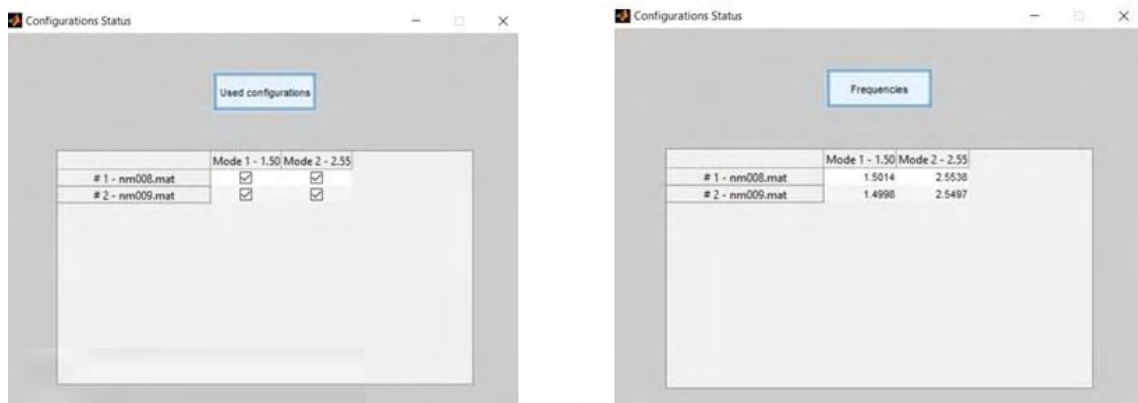


Figure D.18. Configurations Status

After the placing of the sensors the mode shapes can be assembled from the Assemble button and afterwards the user can view the mode shapes by clicking in the Deform Shape button (figure D.19).

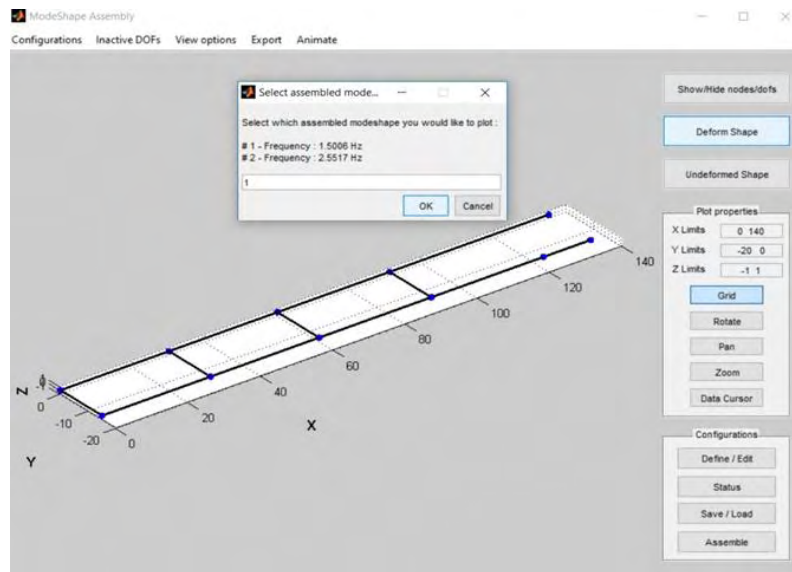


Figure D.19. Deform shape menu

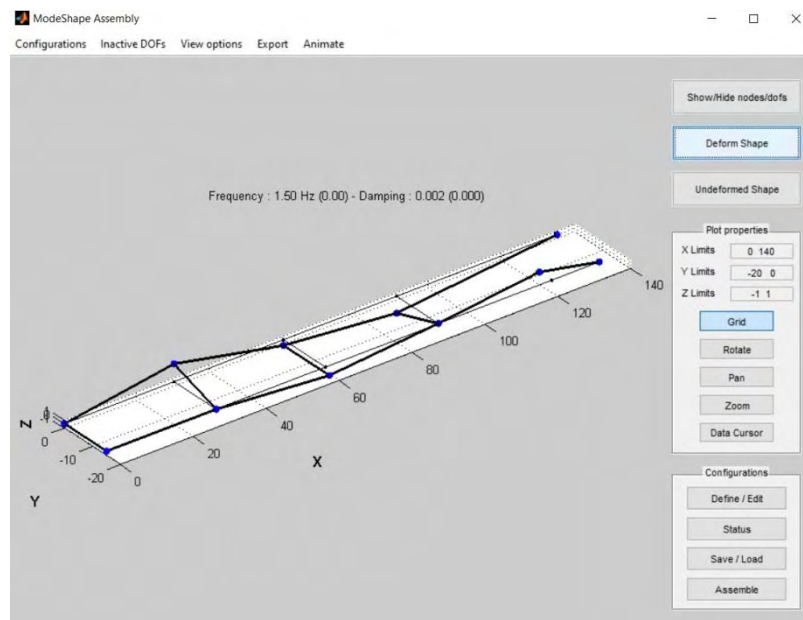


Figure D.20. Mode shape for frequency 1.50 Hz

Usually not all geometry points were measured by a sensor, and those points have no associated mode shape component. However, for visualization purposes we would like to associate those points with some other measured points in order for them to deform as well. This is done from the Inactive DOFs menu. The inactive DOFs deform as the means of the two associated DOFs (figure D.21).

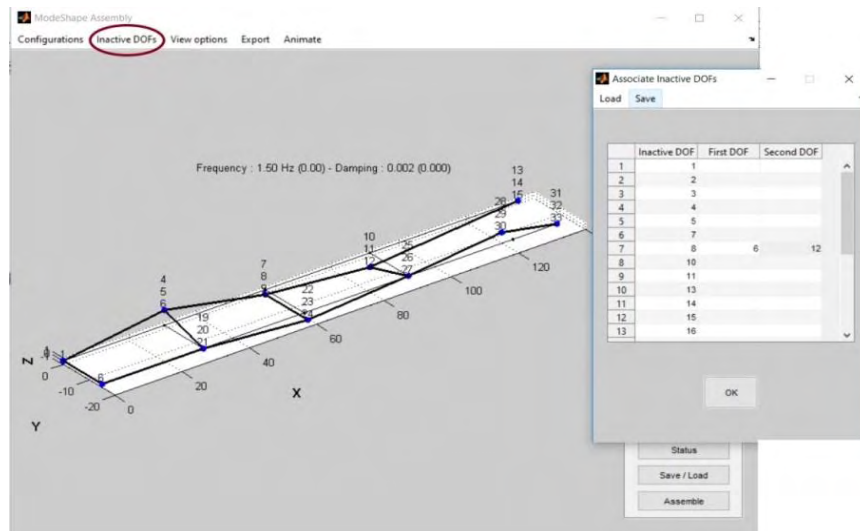


Figure D.21. The inactive DOF 8 becomes active while is associated with 6 and 12 (active DOFs)

The Configuration button provides information and the results, concerning all configuration. (figure D.22)

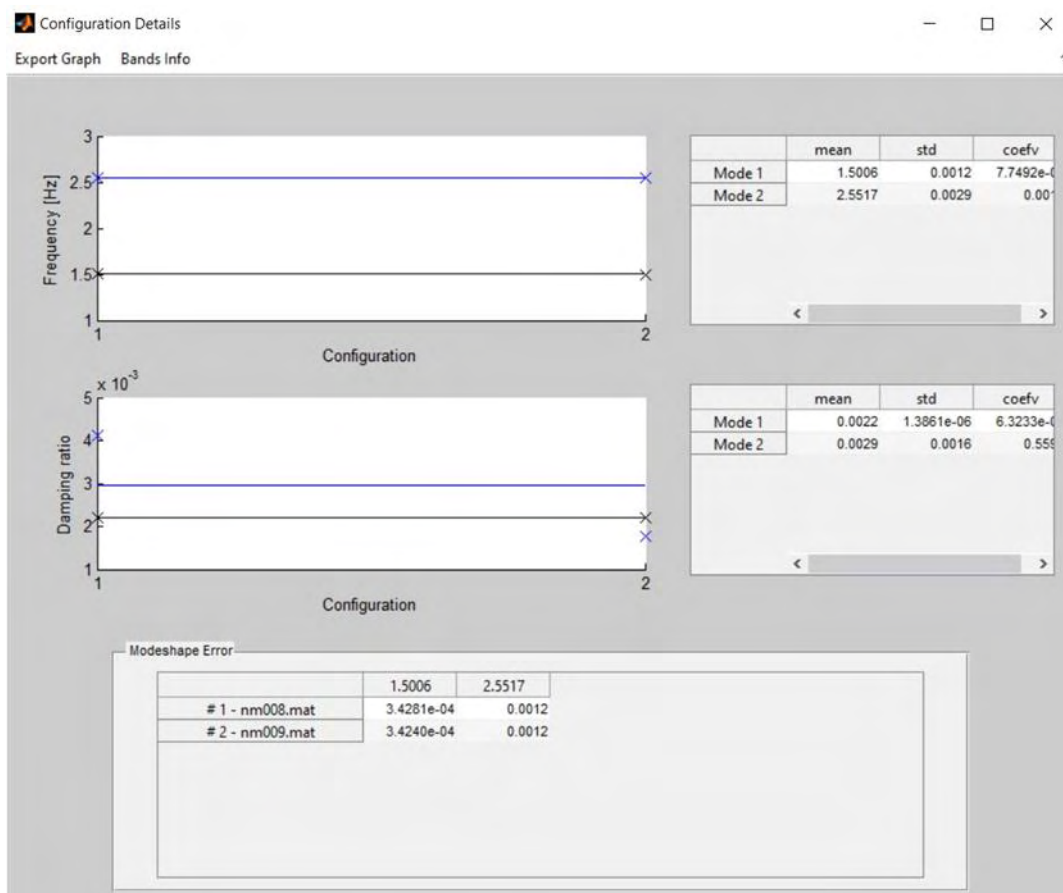


Figure D.22. Configuration Details

View options button shows/hides details of the model.(figure D.23)

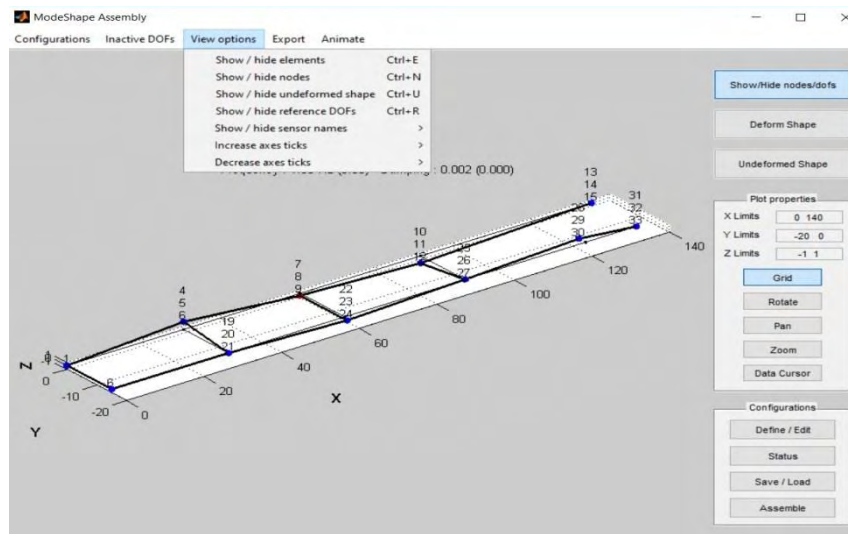


Figure D.23

After the analysis of the model by clicking the export button the user can export the mode shapes as a .mat file or as an image (figure D.23).

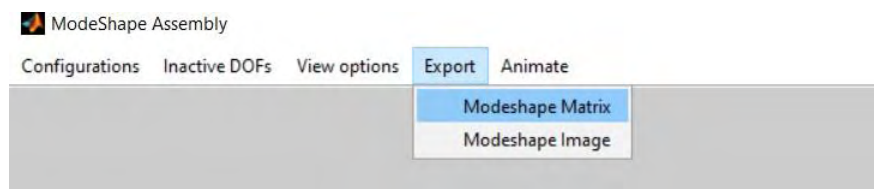


Figure D.24

The mode shapes have been saved in a .mat file. The form of this file is presented in figure D.25

| Workspace     |             |
|---------------|-------------|
| Name          | Value       |
| auout         | 1x5 struct  |
| inactive_dofs | 1x2 cell    |
| phi           | 33x2 double |

Figure D.25

## BIBLIOGRAPHY

- T. Kailath, Linear Systems, Prentice–Hall, Englewood Cliffs, NJ, 1980.
- R. E. Kalman, Mathematical description of linear dynamical systems, J. Soc. Indust. Appl. Math. Ser. A Control, 1 (1963)
- R. C. K. Lee, Optimal Estimation, Identification, and Control, Research Monographs 196, MIT Press, Cambridge, MA, 1964.
- H. H. Rosenbrock, Structural properties of linear dynamical systems, Int. J. Control, 20 (1974)
- E. Walter and L. Pronzato, Identification of Parametric Models from Experimental Data, translated from the 1994 French original and revised by the authors, with the help of John Norton, Comm. Control Engrg. Ser., Springer-Verlag, Berlin, Masson, Paris, 1997.
- Mottershead JE, Friswell MI. Model updating in structural dynamics: a survey. J Sound Vib 1993;
- Marwala T. Finite element model updating using computational intelligence techniques: applications to structural dynamics. Springer; 2010.
- Moaveni B, He X, Conte JP, De Callafon RA. Damage identification of a composite beam using finite element model updating. Comput Aided Civil Infrastruct Eng 2008;
- Christodoulou K, Papadimitriou C. Structural identification based on optimally weighted modal residuals. Mech Syst Signal Process 2007
- Craig Jr RR. Structural dynamics –an introduction to computer methods. New York: John Wiley & Sons; 1981
- Friswell, M.I.; Mottershead, J.E. (1995). Finite element model updating in structural dynamics. Solid Mechanics and Its Applications. **38**. MA: Kluwer Academic Publishers Group. pp. 1–286. ISBN 978-0-7923-3431-6.
- Marwala, Tshilidzi (2002). "Finite element model updating using wavelet data and genetic algorithm". Journal of Aircraft
- Marwala, Tshilidzi (2010). Finite Element Model Updating Using Computational Intelligence Techniques: Applications to Structural Dynamics.



G. H. Golub and C. F. Van Loan. Matrix Computations. Johns Hopkins University Press, Baltimore, Maryland, 3rd edition, 1996.

Y. Saad. Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia, 2nd edition, 2003.

Reynier, M. and Abou-Kandil, H., “Sensors Location for Updating Problems”, Mechanical Systems and Signal Processing, 1999,

Alvin, K.F., “Finite Element Model Update via Bayesian Estimation and Minimization of Dynamic Residuals”, American Institute of Aeronautics and Astronautics Journal, 1997,

Stephan, C. (2012). \Sensor placement for modal identi\_cation". Mechanical Systems and Signal Pro-cessing,

Qureshi, Z.H., Ng, T.S. and Goodwin, G.C. (1980). \Optimum experimental design for identi\_cation of distributed parameter systems". International Journal of Control,

Beck, J.L. and Katafygiotis, L.S., “Updating Models and Their Uncertainties. I: Bayesian Statistical Framework”, Journal of Engineering Mechanics (ASCE),

SAP2000 API

MATLAB API

Beck, J.L., “Statistical System Identification of Structures”, Proceedings of the 5th International Conference on Structural Safety and Reliability (ASCE), San Francisco, 1989,