Dynamic Properties of Rotating Structures: Modelling and Visualisation of Simulation and Experimental Results

by

Götz von Groll

Supervisors:
Prof. Dr. D.J. Ewins and Prof. Dr. P. Hagedorn

May 1995

A thesis submitted in partial fulfilment of the requirements for the degree of
Diplom-Ingenieur (Mechanik) of the
Technische Hochschule Darmstadt.

Fachbereich Mechanik
64289 Darmstadt, Germany

The work was accomplished at the

Department of Mechanical Engineering
Dynamics Section
Imperial College of Science, Technology and Medicine
London SW7, UK

Acknowledgements

I am deeply grateful to Dr. Izhak Bucher, who provided me with invaluable guidance and assistance and supported me strongly over the course of this Diplomarbeit.

I would like to thank Professor Ewins for making my stay in the Dynamics Section possible and for his sustained interest as well as his many suggestions throughout the project.

I would like to thank Professor Hagedorn for supporting the idea of doing my Diplomarbeit abroad and his readiness to accept and mark this work.

Furthermore, I would like to thank Philipp Schmiechen for his help and advice on many matters, particularly though for the background theory and layout of the disc animation program.

Many thanks are also due to the Ph.D. students in Room 564, who created a very friendly and pleasant atmosphere and helped me with many little but essential things like sorting out my computer when I crashed it again.

Abstrakt

Die vorliegende Diplomarbeit behandelt einige Aspekte der mathematischen Modellierung von rotierenden Strukturen und der Visualisierung ihrer dynamischen Eigenschaften. Es wird auf die Methode der finiten Elemente unter Berücksichtigung der für rotierende Strukturen relevanten gyroskopischen Kräfte eingegangen. Ein bestehendes Finite Element Programm wird zur Modellierung von unsymmetrischen Strukturen erweitert. Es ergeben sich dabei Bewegungsgleichungen mit periodischen Koeffizienten, auf deren Lösung mit dem Ansatz nach Hill ebenfalls eingegangen wird. Des weiteren wird gezeigt, wie sich Schwingungen von Scheiben und Wellen in ihre räumlichen Bestandteile zerlegen lassen und sich dadurch mehr Aufschluß über die Art der gemessenen oder simulierten Schwingungen gewinnen läßt. Bei den Scheibenschwingungen liegt der Schwerpunkt auf der Animation der Superposition von verschiedenen Schwingungskomponenten.

Abstract

This present Diplomarbeit is concerned with some aspects of mathematical modelling of rotating structures and the visualisation of the dynamic properties. The Finite Element Method is used for modelling rotating structures taking into consideration the gyroscopic forces, which are of particular relevance here. An existing Finite Element code is expanded to model asymmetric structures. The resulting equations of motion with periodic coefficients are treated using Hill's approach of balancing the harmonic terms. Furthermore, the decomposition of vibration in discs and rotor shafts into their spatial components is shown, thus improving the understanding of the simulated or measured vibrations. The main aspect of the disc vibration is the animation of the superposition of individual vibration components.

Table of Contents

1. Introduction

This Diplomarbeit is concerned with the mathematical modelling of rotating structures, and the visualisation of the dynamic properties as simulated or measured results. Almost every machine has rotating parts, therefore analysis tools that predict dynamic forces or vibration phenomena associated with rotating structures are very important for the designer. The analyst investigating unexpected high levels of vibration will find these tools essential for diagnosing the cause for this, and the experimenter can plan an experiment more accurately due an enhanced knowledge of the nature of the vibration that is to be determined with the measurement.

Two types of rotating structures are dealt with in this thesis:

- The rotor: The rotor consists of a flexible shaft which can carry discs and is mounted on flexible bearings. An expansion of the investigation is the addition of non-axisymmetric elements to the rotor shaft.
- The discs: The disc is treated as flexible, neglecting gyroscopic effects.

The emphasis lies clearly on the rotor structure as it provides the most general basis in rotordynamics.

The work is structured as follows:

After the introduction, the second chapter contains some aspects of mathematical modelling of rotating structures using finite elements. The first part of this chapter shows the derivation of the homogenous equation of motion for an axisymmetric, flexible rotor shaft carrying rigid discs, and the transformation between stationary and rotating frames of reference. The results of the program are then compared with the results of the recently developed finite element program LISA [1] and with measured data.

The analysis is then expanded in the following sections of chapter 2 by unbalance response and incorporation of non-axisymmetric elements. The presents of asymmetry in the rotor and stator results in time varying coefficients in the equations of motion, which calls for a modified way of solving these linear differential equations. Hill's approach to balance terms containing the same harmonics is used and the resulting solution of the hyper-system is discussed here.

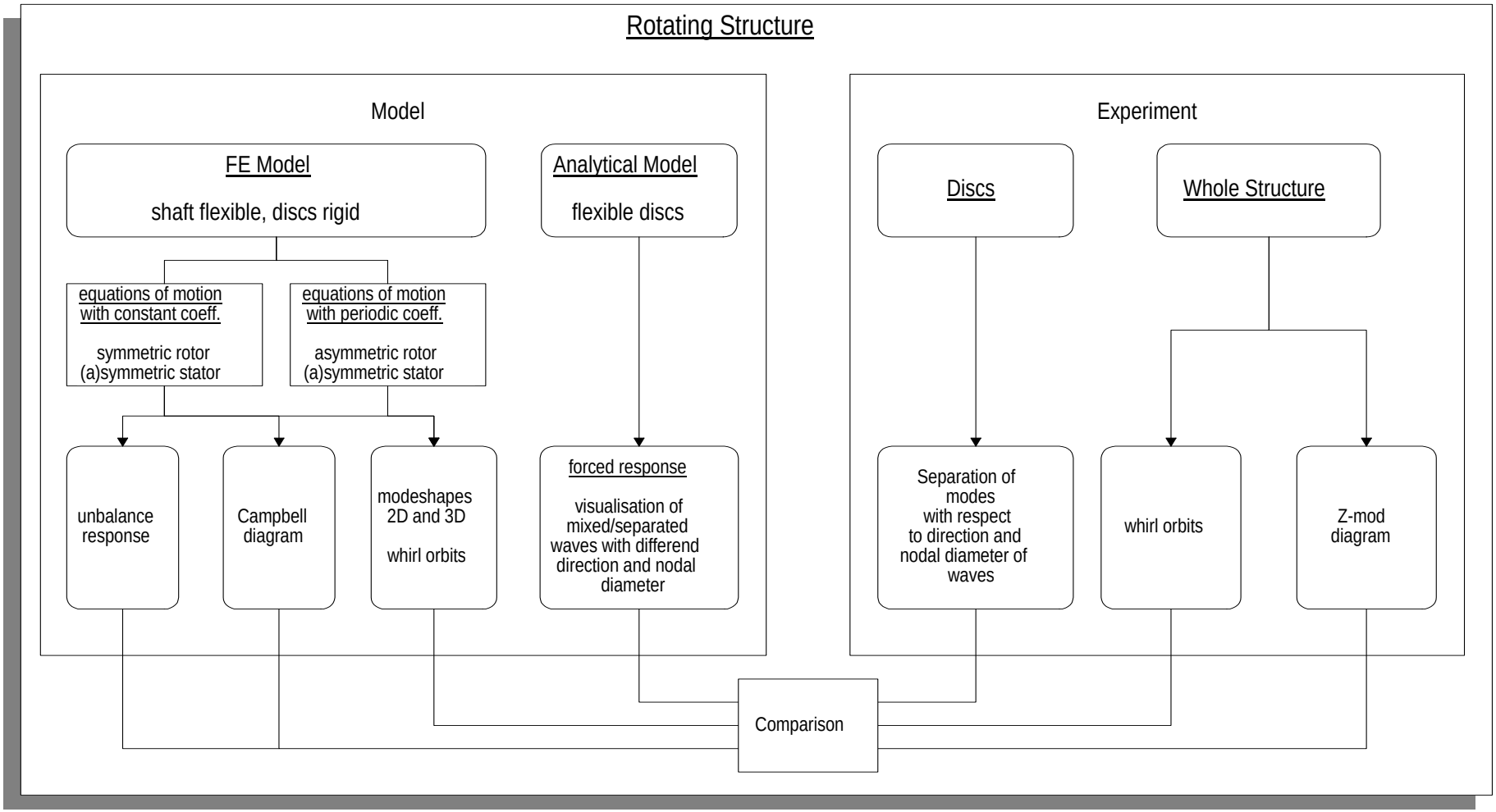
The third chapter deals with the visualisation of the dynamic properties using the results from the FE analysis given. Displays of modeshapes,

Campbell diagrams, unbalance responses and the animation of whirl orbits are all treated. Particular emphasis lies on analysis allowing for the discrimination of the whirl direction, as the nature of forward and backward whirl is quite different in terms of stress cycles and internal damping, and therefore are the cause for different vibration phenomena.

The second part of chapter 3 deals with the animation of vibration of flexible rotating discs. This is connected with the ROSTADYN project [2] where one of the experiments is set up to excite a specific combination of waves to have defined initial conditions. The nature of vibration patterns as a superposition of waves leads to the analysis of separating these individual components from measured vibration signals. An animation tool allowing to visualise individual vibration components or any superposition of those, with the purpose of enhancing the perception of a particular vibration pattern selected for the experiment, is introduced.

In the fourth chapter some aspects of the experimental side are mentioned briefly. A main point is the signal processing required to produce Z-mod diagrams from a spectrogram to make a direct comparison with a Campbell diagram or unbalance response predicted by FE computation possible. A second point is the analysis of the measured data necessary to decompose whirling of shafts into its forward and backward components and thus creating Z-mod and Campbell diagrams which show forward and backward components separately. As mentioned before, this supports further investigation of the vibration measured.

In summary, the objective of this work is to process information generated by different means, be it some form of numerical simulation or experimental measurements, and bring it to a stage where a comparison between modelling, simulation, and experiment is possible. Modeshapes, Campbell diagrams or unbalance response predicted by the FE program can be compared with experimentally-measured whirl orbits, Z-mod diagrams and unbalance responses.


Rotating Structure
Model
FE Model
shaft flexible, discs rigid
Analytical Model
flexible discs
equations of motion
with constant coeff.
symmetric rotor
(a)symmetric stator
equations of motion
with periodic coeff.
asymmetric rotor
(a)symmetric stator
unbalance
response
Campbell
diagram
modeshapes
2D and 3D
whirl orbits
forced response
visualisation of
mixed/separated
waves with differend
direction and nodal
diameter
Experiment
Discs
Whole Structure
Separation of
modes
with respect
to direction and
nodal diameter of
waves
whirl orbits
Z-mod
diagram
Comparison

A note about the history of the FE program:

I did not start this FE program from scratch: I used an existing code as a basis which was written in the Dynamics Section by Dr. Izhak Bucher. The program already consisted of assembling the global mass, stiffness and gyroscopic matrices from axisymmetric shaft elements and discs and springs attached to nodes, as described in section 2.2; it was then expanded by including unbalance response and the possibility of using non-axisymmetric elements for modelling. The FE program somehow grew to well over 5000 lines of code, even though it was written in MATLAB, and would take up some 80 pages if all files were included in the Appendix. So only the most essential files are listed to enable one or the other interested look for how some formulations are realised, as some functions are referred to in the main text. In this thesis I face the task at some points of explaining something in writing that can best be seen in a computer session, so a disk of the program is enclosed, together with the animation tool for rotating discs. The disk consists of MATLAB scripts and functions written under the 4.2c version, so difficulties can arise with older versions as some commands, especially in connection with the graphical interfaces, will not be recognised.

2. Finite Element Model for Rotating Structures

2.1 Features of the Model

In this chapter, a rotating structure consisting of a rotating, flexible shaft with rigid discs mounted on stationary bearings is treated. Assuming the shaft to be slender, it is modelled as a Rayleigh beam, i.e. shear deformation effects are neglected but the gyroscopic moment terms and the moment terms due to the inertia of rotation of the cross section are included. This leads to a stiffer model of the rotor when compared with a Timoshenko model and the obtained results of natural frequencies predicted are therefore slightly high. The model allows for the defining of rigid boundary conditions in all possible combinations, which shall yield the cancellation of the corresponding rows in the M, K, G, C, and F (unbalance force, or any other excitation vector) matrices for that particular degree of freedom. Bearings can be included in the model with their stiffness and damping properties. Both shaft elements and discs can be either solid or hollow. The discs are treated as rigid elements. Furthermore, it is possible to attach more than one disc to a node and more than one shaft element between nodes to achieve higher flexibility and accuracy in how certain parts of a rotor are modelled. The basic shaft elements are modelled as cylindrical shaped elements. Rectangular cross areas and other shapes are non-axisymmetric elements and are dealt with in section 2.5.

2.2 The Derivation of the FE Matrices

2.2.1 The derivation in stationary co-ordinates

The following paragraph summarises the derivation of the finite element matrices for the mass, stiffness, and gyroscopic matrices of a rotating shaft, modelled as a Rayleigh beam.

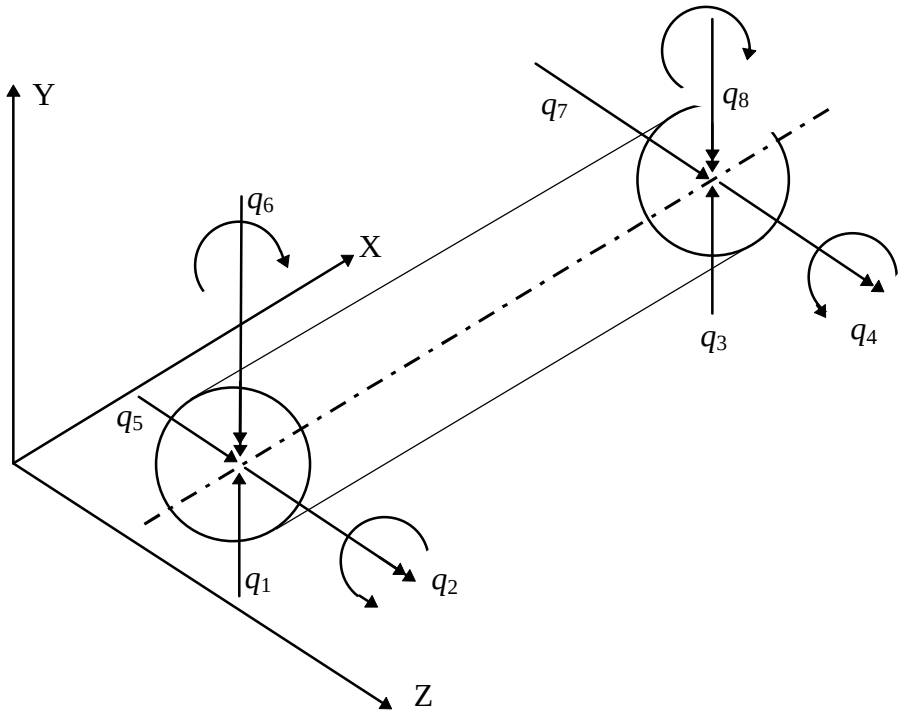


Figure 2.1 Finite Element with selection of co-ordinates

Only a brief run through the derivation of the final matrices of the rotor elements will be given here showing the choice of co-ordinates and shape functions used. The full details are given in Appendix A, where the derivation is carried out for the general case of non-axisymmetric elements, also showing intermediate results for symmetric elements.

The deflections in the Y and Z directions are functions of the same interpolation functions:

$$\text{Y direction: } v = \sum_{i=1}^{4} \Psi_i(\xi)\, q_i(t) \tag{2.1}$$

$$\text{Z direction: } w = \sum_{i=1}^{4} \Psi_i(\xi)\, q_{i+4}(t) \tag{2.2}$$

The shape functions [3, 4] are (using the non-dimensional parameter $\xi = \frac{x}{L}$):

$$\Psi_1(\xi) = 1 - 3\xi^2 + 2\xi^3$$

$$\Psi_2(\xi) = L(\xi - 2\xi^2 + \xi^3)$$

$$\Psi_3(\xi) = 3\xi^2 - 2\xi^3$$

$$\Psi_4(\xi) = L(-\xi^2 + \xi^3) \tag{2.3}$$

Using a Raleigh beam model, the angular deflection is related to the linear deflection by:

$$\alpha = \frac{\partial v}{\partial x} \quad , \qquad \beta = -\frac{\partial w}{\partial x} \tag{2.4}$$

This leads to the formulation of the kinetic energy of one element [3]:

$$T = \frac{L}{2}\int_0^1 \rho A\left(\left(\frac{\partial v}{\partial t}\right)^2 + \left(\frac{\partial w}{\partial t}\right)^2 + I_\mathrm{d}\left(\left(\frac{\partial \alpha}{\partial t}\right)^2 + \left(\frac{\partial \beta}{\partial t}\right)^2\right) + \Omega I_\mathrm{p}\left(\beta\frac{\partial \alpha}{\partial t} - \alpha\frac{\partial \beta}{\partial t}\right)\right)\mathrm{d}\,\xi \tag{2.5}$$

with I_d and I_p being the diametrical and polar area moments of inertia, respectively.

The potential energy for a shaft element is:

$$V = \frac{L}{2}\int_0^1 EI\left(\left(\frac{1}{L^2}\cdot\frac{\partial^2 v}{\partial \xi^2}\right)^2 + \left(\frac{1}{L^2}\cdot\frac{\partial^2 w}{\partial \xi^2}\right)^2\right)d\xi \tag{2.6}$$

In case there is a constant axial force F acting on the shaft, the following term is to be added to the potential energy:

$$V_F = \frac{L}{2}\int_0^1 F\left(\left(\frac{1}{L}\cdot\frac{\partial v}{\partial \xi}\right)^2 + \left(\frac{1}{L}\cdot\frac{\partial w}{\partial \xi}\right)^2\right)\mathrm{d}\xi \tag{2.7}$$

Extracting the elements of the mass, stiffness and gyroscopic terms for an axisymmetric shaft element [5]:

The mass matrix is computed by: $$M_{ij} = \frac{\partial^2 T}{\partial\left(\frac{\partial q_i}{\partial t}\right)\partial\left(\frac{\partial q_j}{\partial t}\right)} \tag{2.8}$$

The gyroscopic matrix is computed by: $$G_{ij} = -2\frac{\partial^2 T}{\partial\left(\frac{\partial q_i}{\partial t}\right)\partial q_j} \tag{2.9}$$

The stiffness matrix is computed by: $$K_{ij} = \frac{\partial^2 V}{\partial q_i \partial q_j} \tag{2.10}$$

Assembling these matrices, the following finite element equation of motion can be derived:

$$\begin{bmatrix} M_e^s & 0 \\ 0 & M_e^s \end{bmatrix}\begin{Bmatrix} \ddot{y}^s \\ \ddot{z}^s \end{Bmatrix} + \begin{bmatrix} D_e^s + C_e^s & \Omega G_e^s \\ -\Omega G_e^s & D_e^s + C_e^s \end{bmatrix}\begin{Bmatrix} \dot{y}^s \\ \dot{z}^s \end{Bmatrix} + \begin{bmatrix} K_e^s & \Omega D_e^s \\ -\Omega D_e^s & K_e^s \end{bmatrix}\begin{Bmatrix} y^d \\ z^d \end{Bmatrix} = \begin{Bmatrix} f_y^s \\ f_z^s \end{Bmatrix}$$

Equation (2.11)

where $\{y^{sT}\ z^{sT}\} = q = \{ q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7\ q_8\}$, $D^s = \eta K^s$ represents the internal hysteretic damping with the shaft and C^s is proportional viscous damping.

The matrices for one particular element (per direction in either Y or Z, *d* is the diameter of the element) are [3]:

$$M_e^s = \frac{\rho A}{3360L}\begin{bmatrix} 1248L^2 + 252d^2 & & & & \textit{symmetric} \\ L(176L^2 + 21d^2) & 4L^2(8L^2 + 7d^2) & & & \\ 432L^2 - 252d^2 & L(104L^2 - 21d^2) & 1248L^2 + 252d^2 & & \\ -L(104L^2 - 21d^2) & -L^2(24L^2 + 7d^2) & -L(176L^2 + 21d^2) & 4L^2(8L^2 + 7d^2) & \end{bmatrix}$$

$$G_e^s = \frac{\rho A d^2}{240L}\begin{bmatrix} 36 & & & \textit{sym} \\ 3L & 4L^2 & & \\ -36 & -3L & 36 & \\ 3L & -L^2 & -3L & 4L^2 \end{bmatrix} \quad K_e^s = \frac{EI}{L^3}\begin{bmatrix} 6 & & & \textit{sym} \\ 3L & 2L^2 & & \\ -6 & -3L & 6 & \\ 3L & L^2 & -3L & 2L^2 \end{bmatrix}$$

Equations (2.12)

A rigid disc added to a particular node has the following equation of motion:

$$\begin{bmatrix} M_e^d & 0 \\ 0 & M_e^d \end{bmatrix}\begin{Bmatrix} \ddot{y}^d \\ \ddot{z}^d \end{Bmatrix} + \begin{bmatrix} 0 & G_e^d \\ -G_e^d & 0 \end{bmatrix}\begin{Bmatrix} \dot{y}^d \\ \dot{z}^d \end{Bmatrix} = \begin{Bmatrix} f_y^d \\ f_z^d \end{Bmatrix}, \qquad (2.13)$$

where $M_e^d = \begin{bmatrix} m & 0 \\ 0 & J_d \end{bmatrix}$, $G_e^d = \begin{bmatrix} 0 & 0 \\ 0 & -J_p \end{bmatrix}$,

and the diametrical and polar mass moments of inertia are

$J_d = \frac{m}{4}\left(\frac{d^2}{4} + \frac{h^2}{3}\right)$ and $J_p = \frac{m \cdot d^2}{8}$, respectively.

h is the disc thickness, and *m* is the disc mass.

Adding linear bearings or dampers which include stiffness and damping terms involves adding the following equation:

$$\begin{bmatrix} c_{yy} & c_{yz} \\ c_{zy} & c_{zz} \end{bmatrix}\begin{Bmatrix} \dot{y}^d \\ \dot{z}^d \end{Bmatrix} + \begin{bmatrix} k_{yy} & k_{yz} \\ k_{zy} & k_{zz} \end{bmatrix}\begin{Bmatrix} y^d \\ z^d \end{Bmatrix} = \begin{Bmatrix} f_y^b \\ f_z^b \end{Bmatrix} \qquad (2.14)$$

2.2.2 Transformation of Matrices between rotating and stationary frame of reference

To express the element or global *M*, *K*, *G* matrices in a rotating (instead of a stationary) frame of reference, it is not necessary to reformulate the energy expressions and go through the whole derivation again. Once the *M*, *K*, *G* matrices have been derived in the stationary frame of reference, the transformation of the matrices is a convenient way of expressing the equation of motion in rotating co-ordinates. Let *r* be the vector of displacement whose co-ordinates are in a rotating frame of reference. The differentiation of *r* with respect to time as observed in the stationary frame is thus:

$$\dot{r} = \left.\frac{\mathrm{d}r}{\mathrm{d}t}\right|_{\mathrm{XYZ}} = \left.\frac{\mathrm{d}r}{\mathrm{d}t}\right|_{\mathrm{X'Y'Z'}} + u \times r \tag{2.15}$$

with XYZ denoting the stationary and X'Y'Z' the rotating frame of reference where X and X' coincide. u is the vector specifying the speed of rotation of X'Y'Z' with respect to XYZ, hence

$$u = \begin{Bmatrix} u_{\mathrm{X}} \\ u_{\mathrm{Y}} \\ u_{\mathrm{Z}} \end{Bmatrix} = \begin{Bmatrix} \Omega \\ 0 \\ 0 \end{Bmatrix} \tag{2.16}$$

The second derivative of *r* is

$$\ddot{r} = \left.\frac{\mathrm{d}^2 r}{\mathrm{d}t^2}\right|_{\mathrm{XYZ}} = \left.\frac{\mathrm{d}^2 r}{\mathrm{d}t^2}\right|_{\mathrm{X'Y'Z'}} + 2u \times \left.\frac{\mathrm{d}r}{\mathrm{d}t}\right|_{\mathrm{X'Y'Z'}} + u \times u \times r \tag{2.17}$$

To keep the following matrices small, they are only shown in an illustrative manner for the four co-ordinates at one node. Thus the vector *q* used here contains the following co-ordinates (see Figure 2.1):

$$q = \begin{Bmatrix} q_1 \\ q_6 \\ q_5 \\ q_2 \end{Bmatrix} \tag{2.18}$$

In this case the vector r now consists of generalised co-ordinates, very similar to vector q described above but with the only difference that these co-ordinates are now co-rotating with the shaft.

$$q = R\,r\,. \tag{2.19}$$

where R is the matrix of rotation

$$R = \begin{bmatrix} \cos(\Omega t) & 0 & -\sin(\Omega t) & 0 \\ 0 & \cos(\Omega t) & 0 & -\sin(\Omega t) \\ \sin(\Omega t) & 0 & \cos(\Omega t) & 0 \\ 0 & \sin(\Omega t) & 0 & \cos(\Omega t) \end{bmatrix}. \tag{2.20}$$

The equation of motion in stationary co-ordinates is:

$$M\,\ddot{q} + G\,\dot{q} + K\,q = 0 \tag{2.21}$$

To convert the equation of motion into rotating co-ordinates, q ,$\dot{q}$, and $\ddot{q}$ have to be substituted with the expressions r ,$\dot{r}$, and $\ddot{r}$ shown above. Furthermore, the M, K, and G matrices have to be transformed to rotating co-ordinates:

$$M_{\mathrm{r}} = R^{-1} M\,R \tag{2.22}$$

The cross-product appearing in the expressions for $\dot{r}$ and $\ddot{r}$ (Equations (2.15) and (2.17)) can be written in matrix notation [6]:

$$u \times A = U \cdot A, \text{ with } U = \begin{bmatrix} 0 & 0 & -\Omega & 0 \\ 0 & 0 & 0 & -\Omega \\ \Omega & 0 & 0 & 0 \\ 0 & \Omega & 0 & 0 \end{bmatrix} \tag{2.23}$$

The equation of motion in rotating co-ordinates can then be written as:

$$M_{\mathrm{r}}\,\ddot{r} + \left(G_{\mathrm{r}} + 2\,M_{\mathrm{r}}U\right)\dot{r} + \left(\left(G_{\mathrm{r}} + M_{\mathrm{r}}U\right)U + K_{\mathrm{r}}\right)r = 0 \tag{2.24}$$

and rewritten

$$M_{\mathrm{r}}\,\ddot{r} + G_{\mathrm{rot}}\,\dot{r} + K_{\mathrm{rot}}\,r = 0\ . \tag{2.25}$$

2.3 Test of the Program Code

2.3.1 Comparison between the analytical solution for a rotating shaft and the FE results

The comparison was carried out for a simple rotating shaft primarily to test the gyroscopic terms in the program. For a rotational speed of $\Omega = 0$, the results should be exactly those of an Euler Bernoulli beam, and then slightly diverging as speed of rotation increases. The range of speed chosen was from 0 to 60 000 rpm, the shaft being simply supported at both ends, has a length of 1 m, a diameter of 0.10 m, Young's modulus of $2 \cdot 10^{11}$ GPa and a density of 8000 kg/m^3 . By defining a thick shaft, the gyroscopic effects are expected to be more pronounced, therefore emphasising any possible mistakes in the code.

The governing equation of free motion for a Rayleigh beam model is [7]:

$$EI\frac{\partial^4 u}{\partial z^4} + \rho A\frac{\partial^4 u}{\partial t^2} - \rho I\left(\frac{\partial^4 u}{\partial z^2 \partial t^2} - 2\Omega\frac{\partial^3 u}{\partial z^2 \partial t}\right) = 0 \tag{2.26}$$

with u being the complex displacement $u = v + \mathrm{i} \cdot w$.

Assuming the solution form:

$$u(x,t) = \sum_{n=1}^{\infty} \psi_n(x) \cdot e^{\mathrm{i}\omega_n t} \tag{2.27}$$

substituting equation (2.27) into (2.26) gives

$$EI\psi_n'''' - 2\rho A d_0 \Omega \omega_n \psi_n'' + \rho A \omega_n{}^2(-\psi + d_0{}^2 \psi_n'') = 0 \tag{2.28}$$

with d_0 being the diameter of gyration, $d_0 = \sqrt{\frac{I}{A}}$.

For a simply-supported shaft, one finds:

$$\psi_n(x) = \sin(n\pi x / l). \tag{2.29}$$

Substituting equation (2.29) into (2.28) :

$$(1 + d_n{}^2)\omega_n{}^2 - 2\Omega d_n{}^2 \omega_n - \omega_{nEB}{}^2 = 0 \tag{2.30}$$

where $d_n{}^2 = (n\pi d_0 / l)^2$ and $\omega_{nEB}{}^2 = (n\pi / l)^4 (EI / \rho A)$,

so that solving (2.30) determines the natural frequencies:

$$\omega_{n_{1,2}} = \frac{\Omega d_n{}^2 \pm \sqrt{\Omega^2 d_n{}^4 + (1+d_n{}^2)\omega_{nEB}{}^2}}{1+d_n{}^2} \tag{2.31}$$

Three models for the finite element calculations are used, their only difference being the number of elements they consist of, namely 10, 12, and 20. The comparison is done for the first 10 natural frequencies. The model with 10 finite elements had an error over the whole speed range of below 1% for the first six frequencies, the maximum errors for the seventh to tenth were 1.4%, 2.2%, 3.2%, and 11% respectively. For a shaft model with only 10 elements, the 10^{th} natural frequency is the very limit of the model, and one would therefore expect quite some error in the higher frequencies. When using two elements more, i.e. a 12 element shaft model, the maximum errors are considerably lower: For the first seven frequencies they are below 0.7%, and the eighth to tenth were 1.2%, 1.8% and 2.5%. For a model with 20 shaft elements, all ten frequencies have a maximum error of below 0.4%. The errors here also reflect the known effect that the finite element model is always stiffer than the analytical one, all natural frequencies calculated with finite elements were higher than the analytical results. For the modeshapes and natural frequencies of interest one can conclude that the program code supplies very accurate results when a sufficient number of elements are chosen.

2.3.2 Comparison with other FE results using more complex structures

A very first step to verify the model is to compare it with results from other FE programs. In this case the simple models described in [8, chapter 3] and [9, chapter 4] were used and the results given there compared with this FE program. Relative error was less than 1% for the first ten natural frequencies.

In addition to this, there has been the possibility of testing the FE program with results from the state-of-the-art FE program called LISA (Large Interactive Structures Analysis), written by Andreas Reister at the University of Kaiserslautern as part of the MARS project [1].

Since LISA models the shaft as Timoshenko beam elements and this program models it as a Rayleigh beam, the natural frequencies given by LISA can be expected to be slightly lower because of the lower stiffness the modelling as Timoshenko beam invokes. However, with a relatively simple and slender

shaft as used in this example, the effect should not be very strong. As can be seen in the comparison of the results listed in Appendix B, this expectation is nicely met. For the first ten modeshapes, the deviations of the calculated results of the two programs is less than 1.3%.

The corresponding matrices in LISA format and the necessary routines to convert the format for the elements used by LISA into the format used by this FE program are included in Appendix B.

Furthermore, measurements were taken from a rig that is used for the ROSTADYN project [2]. The first two bending modes of the rotor were measured at speeds of rotation from 0 to 2100 rpm in steps of 300 rpm and at 3000 and 3600 rpm. There is a good level of agreement between the measured data and the computed results (figures are included in Appendix B), the maximum deviation for the first mode is less than 1.25%, and for the second mode less than 1.6%.

2.4 Implementation of unbalance in the model

To allow for modelling a continuous unbalance distribution in the rotor, the approach taken here is to express unbalance in terms of eccentricity which changes linearly over the length of the elements. Unbalance in a shaft element in the model will thus be determined by the amount of eccentricity in the y- and z-directions at the left and right nodes of the element, represented here by y_l, y_r, z_l, and z_r respectively.

A linearly changing eccentricity in y- and z-directions can be expressed as

$$e_y(\xi) = y_l(1-\xi) + y_r\xi \quad \text{and} \quad e_z(\xi) = z_l(1-\xi) - z_r\xi \tag{2.32}$$

which results in a force distribution of

$$f_y(x) = e_y(\frac{x}{L}) \cdot \rho A\Omega^2 \quad \text{and} \quad f_z(x) = e_z(\frac{x}{L}) \cdot \rho A\Omega^2 \tag{2.33}$$

with A as constant cross section area, ρ as density, and Ω as speed of rotation.

The expressions for the forces in the z-direction are analogous to the ones in the y-direction, and y_l and y_r merely have to be replaced with z_l and z_r, respectively. To avoid repeating the same expressions twice the next steps are carried out in exemplary manner for the y-direction only.

With $F_y = f_y(x)\,\mathrm{d}x$, the work done by F_y over a virtual displacement δy is equated to the virtual work done by the generalised forces Q_{y_i} over the virtual displacement of the generalised co-ordinates q_i:

$$F_y\,\delta y = \sum_{i=1}^{4} Q_{y_i}\,\delta q_i \tag{2.34}$$

Using the same shape functions Ψ_i as in chapter 2.2, equations (2.3), the virtual displacement δy is

$$\delta y = \sum_{i=1}^{4} \Psi_i(\frac{x}{L})\,\delta q_i\,. \tag{2.35}$$

Substituting (2.35) in (2.34) and integrating over x yields

$$\left(\int_0^L \sum_{i=1}^{4} \Psi_i(\frac{x}{L})\,f_y(x)\,\mathrm{d}x\right)\delta q_i = \sum_{i=1}^{4} Q_{y_i}\,\delta q_i \tag{2.36}$$

and re-writing the equation above

$$\sum_{i=1}^{4}\left(\int_0^L \Psi_i(\frac{x}{L})\cdot f_y(x)\,\mathrm{d}x - Q_{y_i}\right)\delta q_i = 0\,. \tag{2.37}$$

Since δq_i is arbitrary it follows that

$$Q_{y_i} = \int_0^L \Psi_i(\frac{x}{L})\cdot f_y(x)\,\mathrm{d}x\,. \tag{2.38}$$

Solving the integrals above gives the following expressions for Q_{y_i}:

$$Q_{y_1} = \frac{L}{20}(7y_\mathrm{l} + 3y_\mathrm{r})\ ,\ Q_{y_2} = \frac{L^2}{60}(3y_\mathrm{l} + 2y_\mathrm{r})$$

$$Q_{y_3} = \frac{L}{20}(3y_\mathrm{l} + 7y_\mathrm{r})\ \text{and}\ Q_{y4} = -\frac{L^2}{60}(2y_\mathrm{l} + 3y_\mathrm{r}) \tag{2.39}$$

The equation of motion for unbalance is thus:

$$M\,\ddot{q} + G\,\dot{q} + K\,q = Q\cdot e^{\mathrm{i}\Omega t} \tag{2.40}$$

Often the unbalance is not given as a linear distribution but in a discrete form as an unbalance mass m_{un} with a certain eccentricity e_n located at one particular node n. The force vector for this sort of unbalance model is assembled in the following manner:

$$Q_n = e_n \cdot m_{un} \cdot \Omega^2 \tag{2.41}$$

2.5 Implementation of non-axisymmetric elements

The presence of non-axisymmetric parts in the rotor produces equations of motion with harmonic coefficients when these equations are expressed in stationary co-ordinates and equations with constant coefficients when expressed in rotating co-ordinates. However, when the stator is not isotropic in addition to the presence of non-axisymmetric rotor elements, the equations of motion have harmonic coefficients in either frame of reference. For this general case the equations of motion are derived here in stationary co-ordinates, the same used in chapter 2.1.

As mentioned before, in case of an isotropic stator the equations of motion in a rotating frame will have constant coefficients. For this special case the transformation from the equations of motion in stationary to ones in rotating co-ordinates is shown in the second part of section 2.5.2, where the new matrices are derived by means of transformation, without having to rewrite the kinetic and potential energies again.

Outlined here is the derivation of the equation of motion for a non-axisymmetric part attached to a certain node. The derivation for non-axisymmetric shaft elements follows the same procedure, the results are just a bit longer and are therefore not included here. The expression "non-axisymmetric part" is used here for "disc" like components, which are treated as rigid and are attached to one node, therefore contributing to the equation matrices at the four co-ordinates of this node only (2 translatory, 2 angular degrees of freedom). The expression "non-axisymmetric (shaft) element" is used here for elements of the rotor shaft, which are treated as flexible and therefore contribute to the equation matrices at eight co-ordinates (i.e. left and right nodes). The derivation of element matrices also involves integrating the shape functions over the length of the element, and the

results are therefore one step further away from immediate physical interpretation as compared with the results for "non-axisymmetric parts".

2.5.1 The co-ordinate transformation

XYZ is the fixed frame of reference and $\xi\eta\zeta$ the rotating body-fixed co-ordinates of the deflected, rotating shaft. The basic transformation matrices used are:

$$R_{\alpha,X} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$ rotating a vector around the X-axis

with $\alpha = \Omega t$,

$$R_{\beta,Y} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$ rotating a vector around the Y-axis

with $\beta = \frac{\partial Z}{\partial X}$,

$$R_{\gamma,Z} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ rotating a vector around the Z-axis

with $\gamma = -\frac{\partial Y}{\partial X}$, (2.42)

The rotation matrices are orthogonal, i.e. $R^T = R^{-1}$.

To express the rotation of the shaft in the body-fixed co-ordinates, the rotations of the frame of reference around three angles are necessary: The body co-ordinates $\xi\eta\zeta$ coincide with the fixed XYZ frame in the non-deflected position of the shaft for $t = 0$. The first rotation of the co-ordinate system around the X-axis with $\alpha = \Omega t$ creates the rotating frame of reference X'Y'Z' with X' = X. The second rotation around the Y'-axis by β moves the X'-axis to X" and Z' to Z", creating X"Y"Z" with Y" = Y'. To move the body co-ordinates from X"Y"Z" into their final position, the third rotation takes place around the Z" axis by the angle γ_2. When doing this transformation rigorously, the projection of angle γ_2 into the X-Y plane is the angle γ, which is the generalised co-ordinate . It can be seen from elementary geometry that $\tan(\gamma_2) = \tan(\gamma)\cdot\cos(\beta)$ with γ being the resulting angle in the X-Y plane. Since a co-ordinate transformation matrix is the inverse of its rotation matrix, the

overall transformation matrix *T*, transforming from the fixed to the body frame, can now be defined:

$$T = R_{\alpha,X} R_{\beta,Y} R_{\gamma_2,Z}" \tag{2.43}$$

The rotation vector ω, which describes the speed of rotation of the deflected shaft in body co-ordinates, is thus:

$$\omega = \begin{pmatrix} \omega_\xi \\ \omega_\eta \\ \omega_\varsigma \end{pmatrix} = T \cdot \omega_{XYZ} \ , \ \text{with} \ \omega_{XYX} = \begin{pmatrix} \omega_X \\ \omega_Y \\ \omega_Z \end{pmatrix} = \begin{pmatrix} \Omega \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix}, \tag{2.44}$$

Ω being the rotational speed of the motor along the X-axis, $\dot{\beta} = \frac{d\beta}{dt}$ (along the Y-axis) and $\dot{\gamma} = \frac{d\gamma}{dt}$ (along the Z-axis).

N.B. The formulation of the deflection angles of the shaft shown above is rigorous and thus yields a symmetric expression for the kinetic energy (see following section) with respect to the terms of the polar moment of inertia. There are examples in literature where the symmetric property of the kinetic energy was lost due to incomplete transformation, cf. [9].

2.5.2 Derivation of the element matrices

Stationary frame of reference:

The kinetic energy of an unsymmetric element, where the rotational part of the kinetic energy is expressed in rotating body co-ordinates, is:

$$T = \frac{1}{2}\left(m(\dot{v}^2 + \dot{w}^2) + J_\xi \omega_\xi{}^2 + J_\eta \omega_\eta{}^2 + J_\varsigma \omega_\varsigma{}^2\right). \tag{2.45}$$

Since the translational kinetic energy is not coupled with the rotational kinetic energy, the equations of motion can be derived separately for translational and rotational motion. Substituting into *T* the expressions for ω_ξ , ω_α , and ω_ς derived above, approximating the expression for the kinetic energy with its second-order Taylor series and introducing:

$$J_m = \frac{J_\eta + J_\varsigma}{2}, \ J_d = \frac{J_\eta - J_\varsigma}{2}, \text{ and } \ J_p = J_\xi, \tag{2.46}$$

the expression for the rotational part of the kinetic energy is:

$$T_{\text{angular}} = \frac{1}{2} J_{\text{m}}(\dot{\beta}^2 + \dot{\gamma}^2) + \frac{1}{2} J_{\text{p}}\Omega(\dot{\beta}\gamma - \dot{\gamma}\beta) + \frac{1}{2} J_{\text{p}}\Omega^2 + \frac{1}{2} J_{\text{d}}(\dot{\beta}^2 - \dot{\gamma}^2)\cos(2\Omega t) + \frac{1}{2} J_{\text{d}}\dot{\beta}\cdot\dot{\gamma}\cdot\sin(2\Omega t) \tag{2.47}$$

The linearised equation of motion for the angular co-ordinates can then be written as:

$$\begin{bmatrix} J_{\text{m}} + J_{\text{d}}\cos(2\Omega t) & J_{\text{d}}\sin(2\Omega t) \\ J_{\text{d}}\sin(2\Omega t) & J_{\text{m}} - J_{\text{d}}\cos(2\Omega t) \end{bmatrix} \begin{Bmatrix} \ddot{\beta} \\ \ddot{\gamma} \end{Bmatrix} + \Omega \begin{bmatrix} -2J_{\text{d}}\sin(2\Omega t) & J_{\text{p}} + 2J_{\text{d}}\cos(2\Omega t) \\ -J_{\text{p}} + 2J_{\text{d}}\cos(2\Omega t) & 2J_{\text{d}}\sin(2\Omega t) \end{bmatrix} \begin{Bmatrix} \dot{\beta} \\ \dot{\gamma} \end{Bmatrix} = 0 \tag{2.48}$$

The mass, stiffness, and gyroscopic matrices, appearing in an equation of motion of a rotor with non-axisymmetric elements, can be separated into matrices with constant parts (M_0, G_0, and K_0), matrices with the factor $\sin(2\Omega t)$ ($M_{2_{\text{s}}}$, $G_{2_{\text{s}}}$, and $K_{2_{\text{s}}}$), and matrices with the factor $\cos(2\Omega t)$ ($M_{2_{\text{c}}}$, $G_{2_{\text{c}}}$, and $K_{2_{\text{c}}}$).

The (real) sin- and cos- dependent matrices can be written in complex notation with matrices having $e^{\text{i}2\Omega t}$ and $e^{-\text{i}2\Omega t}$ as factors, e.g.

$$M_{2_{\text{c}}}\cos(2\Omega t) + M_{2_{\text{s}}}\sin(2\Omega t) = M_2 e^{\text{i}2\Omega t} + M_{-2} e^{-\text{i}2\Omega t} \tag{2.49}$$

with

$$M_2 = \frac{M_{2_c} - \text{i}\cdot M_{2_S}}{2} \qquad M_{-2} = \frac{M_{2_c} + \text{i}\cdot M_{2_S}}{2} \ . \tag{2.50}$$

So the global matrices can be expressed differently as:

$$\begin{aligned} M &= M_{-2}e^{-\text{i}2\Omega t} + M_0 + M_2 e^{\text{i}2\Omega t} \\ G &= G_{-2}e^{-\text{i}2\Omega t} + G_0 + G_2 e^{\text{i}2\Omega t} \\ K &= K_{-2}e^{-\text{i}2\Omega t} + K_0 + K_2 e^{\text{i}2\Omega t} \end{aligned} \tag{2.51}$$

Rotating frame of reference:

As described in section 2.2.2, the equation of motion can be transformed from the stationary to the rotating frame of reference. The results will be shown here for the non-axisymmetric part only.

From above, the matrices for stationary co-ordinates are:

$$M = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_m + J_d\cos(2\Omega t) & 0 & J_d\sin(2\Omega t) \\ 0 & 0 & m & 0 \\ 0 & J_d\sin(2\Omega t) & 0 & J_m - J_d\cos(2\Omega t) \end{bmatrix},\ K = 0,\ \text{and}$$

$$G = \Omega\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2J_d\sin(2\Omega t) & 0 & J_p + 2J_d\cos(2\Omega t) \\ 0 & 0 & 0 & 0 \\ 0 & -J_p + 2J_d\cos(2\Omega t) & 0 & 2J_d\sin(2\Omega t) \end{bmatrix} \qquad (2.52)$$

Performing the transformations as described in section 2.2.2, $M_{\mathrm{r}} = R^{-1}M\,R$, $G_{\mathrm{r}} = R^{-1}G\,R$, and $K_{\mathrm{r}} = R^{-1}K\,R$, the matrices for rotating co-ordinates are:

$$M_{\mathrm{r}} = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_\eta & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & J_\zeta \end{bmatrix},\ K_{\mathrm{r}} = 0,\ \text{and}$$

$$G_{\mathrm{r}} = \Omega\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J_p + J_\eta - J_\zeta \\ 0 & 0 & 0 & 0 \\ 0 & -J_p + J_\eta - J_\zeta & 0 & 0 \end{bmatrix} \qquad (2.53)$$

Repeating Equation (2.24) from section 2.2.2,

$$M_{\mathrm{r}}\ddot{r} + \left(G_{\mathrm{r}} + 2M_{\mathrm{r}}U\right)\dot{r} + \left(\left(G_{\mathrm{r}} + M_{\mathrm{r}}U\right)U + K_{\mathrm{r}}\right)r = 0 \qquad (2.24)$$

the equation of motion expressed in rotating co-ordinates is:

$$\begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_\eta & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & J_\zeta \end{bmatrix}\ddot{r} + \Omega\begin{bmatrix} 0 & 0 & -2m & 0 \\ 0 & 0 & 0 & J_{\mathrm{p}} - J_\eta - J_\zeta \\ 2m & 0 & 0 & 0 \\ 0 & -J_{\mathrm{p}} + J_\eta + J_\zeta & 0 & 0 \end{bmatrix}\dot{r}$$
$$+\Omega^2\begin{bmatrix} -m & 0 & 0 & 0 \\ 0 & J_{\mathrm{p}} - J_\zeta & 0 & 0 \\ 0 & 0 & -m & 0 \\ 0 & 0 & 0 & J_{\mathrm{p}} - J_\eta \end{bmatrix}r = 0$$

Equation (2.54)

However, it is not really necessary to actually perform the co-ordinate transformation of the matrices via the rotation matrix $R(t)$. Using the mass matrices as an example, it is shown how the transformed matrix can be assembled directly from the matrices in stationary co-ordinates:

$$\begin{aligned} M_{\mathrm{r}} &= R(t)^{-1} M(t) R(t) \\ &= R(t)^{-1} (M_{-2} e^{-\mathrm{i}2\Omega t} + M_0 + M_2 e^{\mathrm{i}2\Omega t}) R(t) \end{aligned} \tag{2.55}$$

From the fact that M_{r} is constant and that the equation above is true for all times, e.g. $t = 0$ $\left(R(0) = \mathrm{I}\right)$, M_{r} can be calculated by

$$M_{\mathrm{r}} = M_{-2} + M_0 + M_2 . \tag{2.56}$$

The calculations were carried out using MAPLE's symbolic math capabilities. A transcript of the MAPLE file with all intermediate results of the calculations is given in Appendix A.

2.5.3 Solving equations of motion with periodic coefficients

Formulation of the Problem:

The equation of motion of discretised systems like FE models can be written in matrix form [10]:

$$\mathrm{M}(t)\ddot{\mathrm{u}} + \mathrm{G}(t)\dot{\mathrm{u}} + \mathrm{K}(t)\mathrm{u} = \mathrm{F}(t) \tag{2.57}$$

where M(t), G(t), K(t) are periodic with period T, i.e.:

$$\mathrm{M}(t) = \mathrm{M}(t+T),\ \mathrm{G}(t) = \mathrm{G}(t+T),\ \mathrm{K}(t) = \mathrm{K}(t+T) \tag{2.58}$$

Equation (2.57) can be transformed into a set of first-order differential equations

$$\dot{\mathrm{x}}(t) - \mathrm{A}(t)\mathrm{x}(t) = \mathrm{g}(t) \tag{2.59}$$

According to Floquet's theory for linear differential equations with periodic coefficients [11], a solution of the following form can be found:

$$\mathrm{x}_k(t) = e^{\lambda_k t} \psi_k(t) \tag{2.60}$$

where $\psi_k(t)$ is the eigenvector associated with the complex eigenvalue:

$$\lambda_k = \alpha_k + \mathrm{i}\,\omega_k \tag{2.61}$$

ψ_k is also periodic in T. The period T is related to the speed of rotation in rotating machinery by:

$$\Omega = \frac{2\pi}{T} \tag{2.62}$$

A Fourier decomposition of $\psi_k(t)$ and $\mathrm{A}(t)$ yields:

$$\psi_k(t) = \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}j\Omega t} \quad \text{and} \quad \mathrm{A}(t) = \sum_{a=-\infty}^{\infty} \mathrm{A}_j e^{\mathrm{i}a\Omega t} \tag{2.63}$$

The Eigenvalue Problem:

Substituting equation (2.63) into the homogeneous part of equation (2.59):

$$\lambda_k \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}j\Omega t} + \sum_{j=-\infty}^{\infty} \mathrm{i}\, j\Omega \psi_{k,j} e^{\mathrm{i}j\Omega t} - \sum_{a=-\infty}^{\infty} \mathrm{A}_a e^{\mathrm{i}a\Omega t} \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}j\Omega t} = 0 \tag{2.64}$$

The equation above must be satisfied for all frequencies independently, thus all terms of equal frequency have to balanced. This can be written as an infinite hyper-eigenvalue problem

$$(\lambda_k \hat{\mathrm{I}} - \begin{bmatrix} \ddots & \cdots & & & & & \\ \cdots & \mathrm{A}_0 - \mathrm{i}2\Omega\mathrm{I} & \mathrm{A}_{-1} & \mathrm{A}_{-2} & \cdots & & \\ \cdots & \mathrm{A}_{+1} & \mathrm{A}_0 - \mathrm{i}\Omega\mathrm{I} & \mathrm{A}_{-1} & \mathrm{A}_{-2} & \cdots & \\ \cdots & \mathrm{A}_{+2} & \mathrm{A}_{+1} & \mathrm{A}_0 & \mathrm{A}_{-1} & \mathrm{A}_{-2} & \cdots \\ & \cdots & \mathrm{A}_{+2} & \mathrm{A}_{+1} & \mathrm{A}_0 + \mathrm{i}\Omega\mathrm{I} & \mathrm{A}_{-1} & \cdots \\ & & \cdots & \mathrm{A}_{+2} & \mathrm{A}_{+1} & \mathrm{A}_0 + \mathrm{i}2\Omega\mathrm{I} & \cdots \\ & & & & & \cdots & \ddots \end{bmatrix}) \begin{pmatrix} \vdots \\ \psi_{k,-2} \\ \psi_{k,-1} \\ \psi_{k,0} \\ \psi_{k,+1} \\ \psi_{k,+2} \\ \vdots \end{pmatrix} = \hat{0}$$

$$(\lambda_k \hat{\mathrm{I}} - \hat{\mathrm{A}}) \cdot \hat{\psi}_k = \hat{0} \tag{2.65}$$

Now, since the matrices M, G, and K are real, the components of the hyper matrix are related to each other by

$$\mathrm{A}_{-n} = \mathrm{A}^*_{+n} \tag{2.66}$$

where the * denotes the complex conjugate from which also follows that the fundamental component $\mathbf{A}_0$ is real and symmetric. This makes the whole matrix Hermitian.

The stability criterion for this problem is the same as for the standard problem, i.e. the real parts of the eigenvalues must be negative.

The system is time-invariant and consists of an infinite number of eigenvalues and eigenvectors, which are, however, not all linearly independent. Only 2*N* eigenvalues and vectors are necessary to describe the homogeneous solution fully, *N* being the dimension of the original system. The other eigenvalues and vectors do not hold any additional physical information. This redundancy can be shown as follows: let λ_k be an eigenvalue of $\hat{\mathrm{A}}$ and $\psi_k(t)$ the corresponding eigenvector and then the following equation holds:

$$\mathrm{u}_k(t) = e^{\lambda_k t} \cdot \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}j\Omega t} = e^{(\lambda_k + \mathrm{i}n\Omega)t} \cdot \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}(j-n)\Omega t} \tag{2.67}$$

It follows that

$$\lambda_n = \lambda_k + \mathrm{i}\, n\Omega \tag{2.68}$$

is also an eigenvalue of $\hat{\mathrm{A}}$ and the corresponding eigenvector is

$$\psi_n = \psi_k\, e^{-\mathrm{i}\, n\Omega}. \tag{2.69}$$

For clarification the dimensions of ψ_n, ψ_k and $\psi_{k,j}$ are pointed out again:

$$\psi_n, \psi_k \in \boldsymbol{C}^{\infty\times 1}, \text{ and } \psi_{k,j} \in \boldsymbol{C}^{N\times 1} \tag{2.70}$$

For practical computation it is necessary to truncate this infinite series after including a certain number of terms. The number of terms included for a wanted accuracy has to be weighed against a marked increase in problem and computation size. Truncation after *h* harmonics leads to the finite eigenvalue problem

$$(\lambda_k \hat{\mathrm{I}} - \hat{\mathrm{A}}) \cdot \hat{\psi}_k = \hat{0} \tag{2.71}$$

where the circumflex denotes hyper-quantities of size 2*N*(2*h*+1). According to Equation (2.67) there are only 2*N* linearly independent eigenvectors and eigenvalues. In [10] the authors assume that it is possible to reduce the problem size to 2*N* to begin with, and in doing so would result in an iterative procedure.

2.5.4 The formulation for systems with harmonic coefficients

In the case of a non-axisymmetric rotor, the mass, stiffness, and gyroscopic matrices have both constant parts and parts with second-order harmonic coefficients, i.e. $\mathrm{M}(t) = \mathrm{M}_{-2} e^{-i2\Omega t} + \mathrm{M}_0 + \mathrm{M}_2 e^{i2\Omega t}$, and in an analogous

manner, so do K and G. As shown in section 2.5.2, M_{-2} is the complex conjugate of M_2 , and the same holds for K_2 and G_2 . The Fourier decomposition of a time varying matrix into a constant matrix and two matrices with the two harmonic coefficients is exact for any non-axisymmetric rotor. By inverting the mass or stiffness matrix with two harmonic components to obtain the state-space formulation, as described in Equations (2.57) to (2.59), it is clear that from a practical point of view the resulting infinite series of such an inversion has to be truncated somewhere. For some systems with very small asymmetry a low-order approximation might be sufficient. This depends on the response amplification for higher harmonics, but for systems with larger asymmetry, e.g. a two-bladed propeller mounted onto the rotor, it might be necessary to carry out the calculations with quite a few terms to keep the truncation error within the desired accuracy. $\mathrm{M}(t)^{-1}$ can be developed as power series:

$$\mathrm{M}(t)^{-1} = (\mathrm{I} - (\tilde{\mathrm{M}}_{-2}e^{-\mathrm{i}2\Omega t} + \tilde{\mathrm{M}}_{2}e^{\mathrm{i}2\Omega t}) + (\tilde{\mathrm{M}}_{-2}e^{-\mathrm{i}2\Omega t} + \tilde{\mathrm{M}}_{2}e^{\mathrm{i}2\Omega t})^2 - (\ldots)^3 + \ldots)\mathrm{M}_0{}^{-1} \quad (2.72)$$

or, rewritten in a different notation:

$$\mathrm{M}(t)^{-1} = \ldots + \breve{\mathrm{M}}_{-2}e^{-\mathrm{i}2\Omega t} + \breve{\mathrm{M}}_0 + \breve{\mathrm{M}}_2 e^{\mathrm{i}2\Omega t} + \breve{\mathrm{M}}_4 e^{\mathrm{i}4\Omega t} + \ldots \; . \quad (2.73)$$

The development for the matrix with constant coefficients, $\breve{\mathrm{M}}_0$ is given below just to show how the truncation error affects the accuracy of all matrices, not just the ones with higher-order coefficients:

$$\breve{\mathrm{M}}_0 = (\mathrm{I} + 2\left|\tilde{\mathrm{M}}_2\right|^2 + 6\left|\tilde{\mathrm{M}}_2\right|^4 + \mathrm{hot})^{-1}\mathrm{M}_0{}^{-1} \quad (2.74)$$

Thus, the accuracy of this inversion depends strongly on the number of higher-order terms included. The convergence is better the smaller the asymmetry effect is, namely the infinity-norm or 1-norm of the matrix M_2 .

However, in this case of mass, stiffness, and gyroscopic matrices consisting of only three harmonic components, the problem can be transformed into the inversion of a matrix with constant coefficients, which is exact. The inversion of a matrix with harmonic coefficients can be avoided by substituting an assumed solution

$$\mathrm{u}_k(t) = e^{\lambda_k t}\psi_k(t) \quad \text{and} \quad \psi_k(t) = \sum_{j=-\infty}^{\infty} \psi_{k,j} e^{\mathrm{i}j\Omega t} \quad (2.75)$$

directly into Equation (2.57), where the series for M(t), K(t), and G(t) are not infinite but carry only the -2, 0, and 2 components. Sorting the resulting

equation with respect to the sought eigenvalues λ_k, the following matrices due to inertia, gyroscopic and stiffness effects are obtained:

$$(\mathrm{M}' + \mathrm{G}' + \mathrm{K}') \sum_{j=-\infty}^{\infty} \psi_{k,j} = 0, \text{ with} \tag{2.76}$$

$$\mathrm{M}' = \sum_{j=-\infty}^{\infty} \lambda_k{}^2 (\mathrm{M}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{M}_0 + \mathrm{M}_2 e^{\mathrm{i}2\Omega t}) e^{\mathrm{i}j\Omega t} \tag{2.77}$$

$$\mathrm{G}' = \sum_{j=-\infty}^{\infty} \lambda_k \begin{pmatrix} \mathrm{G}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{G}_0 + \mathrm{G}_2 e^{\mathrm{i}2\Omega t} \\ +\mathrm{i}\,2\Omega j (\mathrm{M}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{M}_0 + \mathrm{M}_2 e^{\mathrm{i}2\Omega t}) \end{pmatrix} e^{\mathrm{i}j\Omega t} \tag{2.78}$$

$$\mathrm{K}' = \sum_{j=-\infty}^{\infty} \begin{pmatrix} \mathrm{K}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{K}_0 + \mathrm{K}_2 e^{\mathrm{i}2\Omega t} \\ +\mathrm{i}\Omega j (\mathrm{G}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{G}_0 + \mathrm{G}_2 e^{\mathrm{i}2\Omega t}) \\ +\mathrm{i}^2 \Omega^2 j^2 (\mathrm{M}_{-2} e^{-\mathrm{i}2\Omega t} + \mathrm{M}_0 + \mathrm{M}_2 e^{\mathrm{i}2\Omega t}) \end{pmatrix} e^{\mathrm{i}j\Omega t} \tag{2.79}$$

Equation (2.76) can only be fulfilled if all submatrices with the same harmonics ($n\Omega t$) are in balance. Thus, equating terms with the same harmonic coefficients to zero yields bloc tridiagonal hyper-M, -K, and -G matrices, which have constant coefficients.

$$\hat{\mathrm{M}} = \begin{bmatrix} \cdots & \cdots & \cdots & & & & \\ \cdots & \mathrm{M}_0 & 0 & \mathrm{M}_2 & & & \\ \cdots & 0 & \mathrm{M}_0 & 0 & \mathrm{M}_2 & & \\ & \mathrm{M}_{-2} & 0 & \mathrm{M}_0 & 0 & \mathrm{M}_2 & \\ & & \mathrm{M}_{-2} & 0 & \mathrm{M}_0 & 0 & \cdots \\ & & & \mathrm{M}_{-2} & 0 & \mathrm{M}_0 & \cdots \\ & & & & \cdots & \cdots & \cdots \end{bmatrix}, \quad \psi_k = \begin{pmatrix} \cdots \\ \psi_{k,-2} \\ \psi_{k,-1} \\ \psi_{k,0} \\ \psi_{k,1} \\ \psi_{k,2} \\ \cdots \end{pmatrix}$$

$$\hat{\mathrm{G}} = \begin{bmatrix} \cdots & \cdots & \cdots & & & & \\ \cdots & \mathrm{G}_0 + \mathrm{i}4\Omega\mathrm{M}_0 & 0 & \mathrm{G}_2 & & & \\ \cdots & 0 & \mathrm{G}_0 + \mathrm{i}2\Omega\mathrm{M}_0 & 0 & \mathrm{G}_2 - \mathrm{i}2\Omega\mathrm{M}_2 & & \\ & \mathrm{G}_{-2} + \mathrm{i}4\Omega\mathrm{M}_{-2} & 0 & \mathrm{G}_0 & 0 & \mathrm{G}_2 - \mathrm{i}4\Omega\mathrm{M}_2 & \\ & & \mathrm{G}_{-2} + \mathrm{i}2\Omega\mathrm{M}_{-2} & 0 & \mathrm{G}_0 - \mathrm{i}2\Omega\mathrm{M}_0 & 0 & \cdots \\ & & & \mathrm{G}_{-2} & 0 & \mathrm{G}_0 - \mathrm{i}4\Omega\mathrm{M}_0 & \cdots \\ & & & & \cdots & \cdots & \cdots \end{bmatrix}$$

$$\hat{K} = \begin{bmatrix} \dots & \dots & \dots & & & & \\ \dots & K_0 + i2\Omega G_0 - 4\Omega M_0 & K_1 & K_2 & & & \\ \dots & K_{-1} & K_0 + i\Omega G_0 - \Omega M_0 & K_1 & K_2 - i\Omega G_2 + \Omega M_2 & & \\ & K_{-2} + i2\Omega G_{-2} - 4\Omega M_{-2} & K_{-1} & K_0 & K_1 & K_2 - i2\Omega G_2 + 4\Omega M_2 & \\ & & K_{-2} + i\Omega G_{-2} - \Omega M_{-2} & K_{-1} & K_0 - i\Omega G_0 + \Omega M_0 & K_1 & \dots \\ & & & K_{-2} & K_{-1} & K_0 - i2\Omega G_0 + 4\Omega M_0 & \dots \\ & & & & \dots & \dots & \dots \end{bmatrix}$$

Equations (2.80)

The term K_1 is included in $\hat{K}$ for generality to allow for the bearing properties of the stator being developed in a power series; $\hat{K}$ then no longer has a tridiagonal shape but a bandwidth of $5N$ or higher.

After deciding of how many harmonics are to be included in the solution, the formulation of the eigensystem is straightforward. Since $\hat{M}$ and $\hat{K}$ have constant coefficients and can be inverted directly, the formulation in state-space does not involve the development of any power series for a matrix. This might prove to be a significant advantage if the amount of asymmetry is not marginal and the calculations can be executed by taking significantly fewer higher harmonic terms into consideration.

2.5.5 Test Case

An illustrative analysis with a simple structure is presented here to show how the numerical results reflect the expected behaviour described in the sections above.

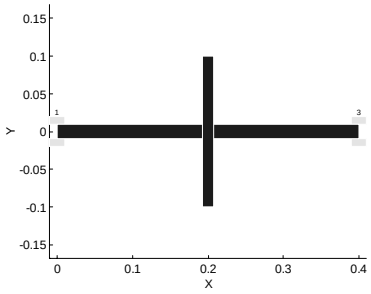


Figure 2.2 rotor with disc and bearings

The model is not axisymmetric but appears to be so in the Figure above because it is shown in the X-Y plane only. The finite element model consists of 2 shaft elements and a non-axisymmetric disc whose two diametrical moments of inertia differ by 20%, and the rotor speed was set to 12000 rpm

(200 Hz). The model has three nodes and therefore 12 degrees of freedom. Since the rotor is simply-supported at both ends, the 12 degrees of freedom are reduced to eight, N=8.

The harmonics -2, -1, 0, 1, 2 are included in the calculations. As described in section 2.5.3, the hyper-eigensystem in state-space formulation has the size of 80 × 80 ($2N \cdot 5$) containing 16 ($2N$) linearily independent eigenvalues and eigenvectors while all the others are redundant.

Analysing the numerical results:

Due to the formulation of the eigensystem in state-space, half of the $2N$ eigenvalues and eigenvectors are complex conjugates of the other half, so all further analysis is carried out with only half that number of eigenvalues and -vectors except in the following two Figures:

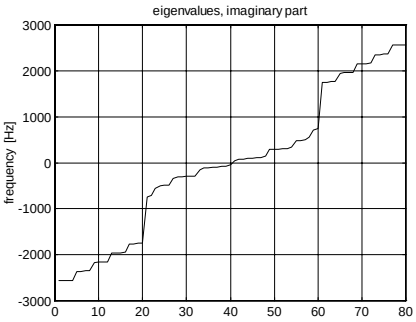


Figure 2.3 imaginary parts of eigenvalues

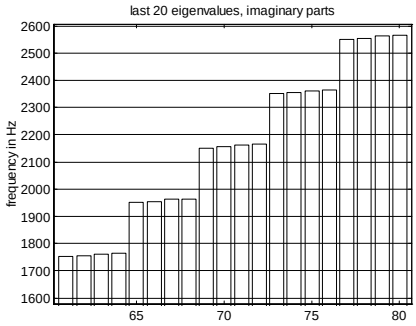


Figure 2.4 last 20 eigenvalues

These Figures show that the eigenvalues are grouped together on certain "plateaus", each consisting of four (or two, Figure 2.3) eigenvalues. In Figure 2.4 the last 20 eigenvalues are plotted. For every eigenvalue there are four others differing in magnitude by multiples of 200 Hz; they make up a family of five corresponding eigenvalues. This behaviour is explained by Equations (2.67) and (2.69) showing the redundancy of eigenvalues and eigenvectors. Consistent with equation (2.67), the last 20 eigenvalues contain four families, each containing five eigenvalues with five corresponding linearly-dependent eigenvectors, e.g. 61, 65, 69, 73, 77 are one such family shown in Figure 2.4. With a speed of rotation of Ω = 200 Hz, the gaps between these eigenvalues are k200 Hz, with k =0, 1, 2, ... However, the modeshapes belonging to the two or four frequencies lying close together on any "plateau" are completely independent.

The discussion above of the last 20 eigenvalues and vectors can be expanded over the whole range. Using the modal assurance criterion (MAC) [12] to find subvectors which are highly correlated with each other, one obtains the result shown in Figures 2.5 and 2.6. In these two Figures below the eigenvectors are placed in ascending order of their corresponding eigenvalues, i.e. eigenvectors with higher index numbers are modeshapes occurring at higher natural frequencies.

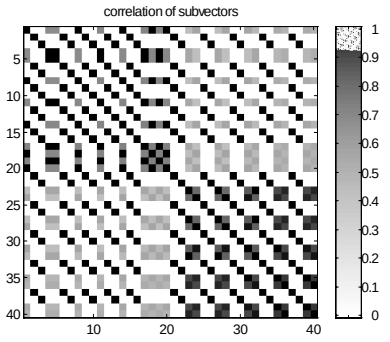


Figure 2.5 correlation of subvectors

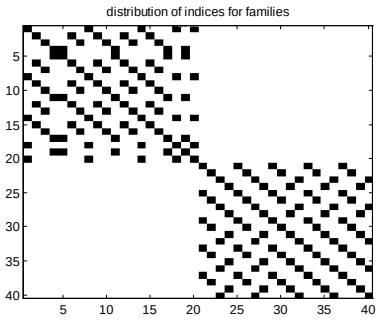
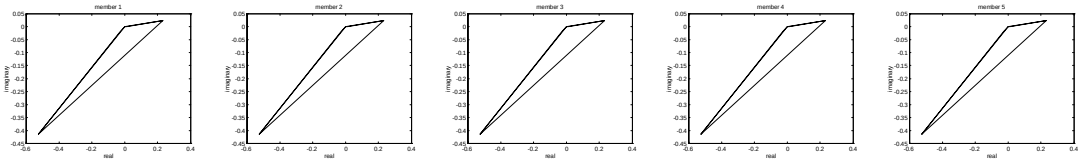


Figure 2.6 indices of families

Figure 2.5 was built by looking for the subvector with the highest norm in each eigenvector and normalising the subvector to unity. All dominant subvectors found this way were correlated with each other, and thus renders the correlation matrix with entries between zero and one. Having built the eigensystem including the harmonics -2, -1, 0, +1, +2, every family must consist of five linearly dependent eigenvectors (Equation (2.67)). For each eigenvector the highest five correlation coefficients are sought from the correlation matrix, and the result is shown in Figure 2.6. Every row (or column) contains a black spot for the index of the eigenvector itself and for each eigenvector in the family.

The Figures below shows an example of a whole eigenvector family with eigenvectors 21, 25, 29, 33, and 37. It can be seen that all five eigenvectors of the family have the same shape.



Figures 2.7 Argand plots of eigenvectors

Furthermore, there exists a high correlation in some cases between vectors of different families as well (with a huge difference in natural frequency, here around 1700 Hz) as can be seen from the structure of the correlation matrix. An example is given below:

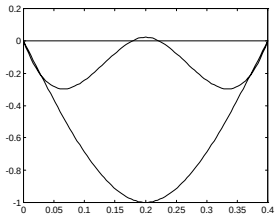


Figure 2.8 modes 2 and 29

The dotted curve in the Figure above represents mode 29, the continuous curve mode 2. In both cases the fundamental harmonic subvector was taken since it was the dominant one in both modes, i.e. $\psi_{2,0}$ and $\psi_{29,0}$. As can be seen in the correlation matrix, $\psi_{2,0}$ has a high correlation with $\psi_{29,0}$. In this example, the correlation coefficient is not equal to one as it is with modes within a family, i.e. $\psi_{2,-2}$, $\psi_{2,-1}$, $\psi_{2,0}$, $\psi_{2,1}$, $\psi_{2,2}$, but 0.95. In this case study the non-axisymmetric disc is mounted at the middle of the rotor, i.e. there are symmetric modes where the lack of symmetry becomes irrelevant since the angular deflection at the midpoint of the rotor is zero; the results in the X-Y and X-Z planes are identical; in these cases the whirl orbits for each harmonic are circles. All modes which have high correlation with other non-family modes are such symmetric modes, i.e. have a wavelength of $(2k+1)/2 \cdot$length of shaft where k = 0, 1, 2... . The existence of these symmetric modes also explains that some plateaus consist of four and others of two eigenvalues (or two and one, respectively, if only one part of the complex conjugate pairs is investigated) depending on whether Y-Z symmetry is occurring for this particular eigenfrequency.

2.6 The Structure of the Program and its graphical user interface

2.6.1 The Structure of the FE-Program

To make the way the program is organised transparent, the key functions and their interaction is outlined below. Furthermore, the code of the functions mentioned here is listed in Appendix D to allow for the possibility to check how one or the other problem is formulated.

The program consists of three main interfaces which allow the user to easily enter shaft elements, discs, boundary conditions and bearings. As the data is entered from these interfaces, the new information is passed on to the rotdata.m function, which assembles the geometry matrices, such as node vector, shaft matrix ..., into a format that can be read by the rotalee.m function. It assembles the global finite element matrices from the mass, stiffness, and gyroscopic matrices and the unbalance force for each single element computed in rotmass.m, rotstiff.m, rotgyro.m, and rotunb.m. For non-axisymmetric elements there are the functions rotasymm.m, rotasymk.m and rotasymg.m for the element wise calculation of mass, stiffness and gyroscopic matrices formulated in the stationary frame of reference.

The function rotadd.m adds symmetric elements to the equation of motion which were specified in a format not accounted for in the usual matrices, e.g. discs normally are specified via the interfaces with their geometric dimensions length, outer diameter, ..., but an alternative way is to give the mass, polar and diametrical moments of inertia.

The functions rotasp.m assembles the global matrices in the case when non-axisymmetric rotor elements are present and the equation of motion has periodic coefficients. The output is therefore split into matrices with $e^{\pm i k \Omega t}$ as coefficients, k positive and negative integers. rothill.m assembles these matrices into a finite hyper-system with time-constant-coefficients that can now be solved in an ordinary manner. rotsort.m uses the MAC to group the resulting eigenvectors and eigenvalues into families, where one eigenvector is a basis-eigenvector and the rest are redundant, as shown before. For some test cases and real applications where the stator is modelled as isotropic, the equation of motion is better expressed in a rotating frame of reference to avoid periodic coefficients. For this purpose the file rotcoor.m transforms the matrices build by rotasp.m ($M_0, M_2, G_0 \ldots$ in stationary frame of reference) into matrices for a rotating frame of reference and assembles the equations of motion in rotating co-ordinates.

2.6.2 The graphical user interfaces of the FE-program

The general principle is to enter all the dimensions and material properties of each shaft, disc or bearing element in the corresponding user-interface controls. When the entries for one element have been completed, the accept button is pressed and the new data, written as a vector, passed on to the rotdata function, where this vector is added, possibly after some sorting, e.g. in case of shaft elements where the nodes vector has to be updated as well, to the global geometry matrices. Jumping back and forth between the interfaces can be done by a simple button press, and the information in the geometry matrices is presented in the corresponding boxes. The simultaneous updating of the drawing of the model should help to recognise severe errors in the input, e.g. getting the length of an element wrong by an order of magnitude. For a more detailed description of how the interface is operating, see Appendix C. Just to give an impression of how they look like, the three main interface for entering shaft, disc, and bearing elements are shown below.

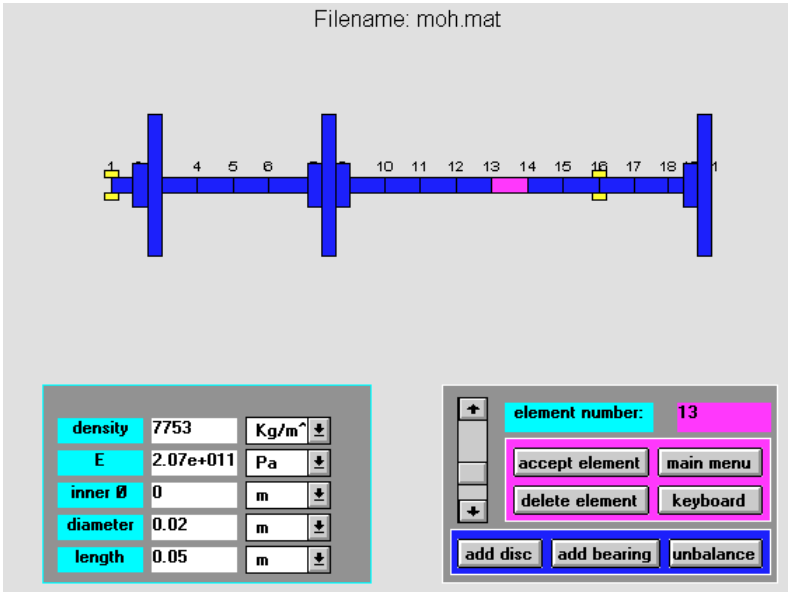


Figure 2.9 Main interface for shaft elements

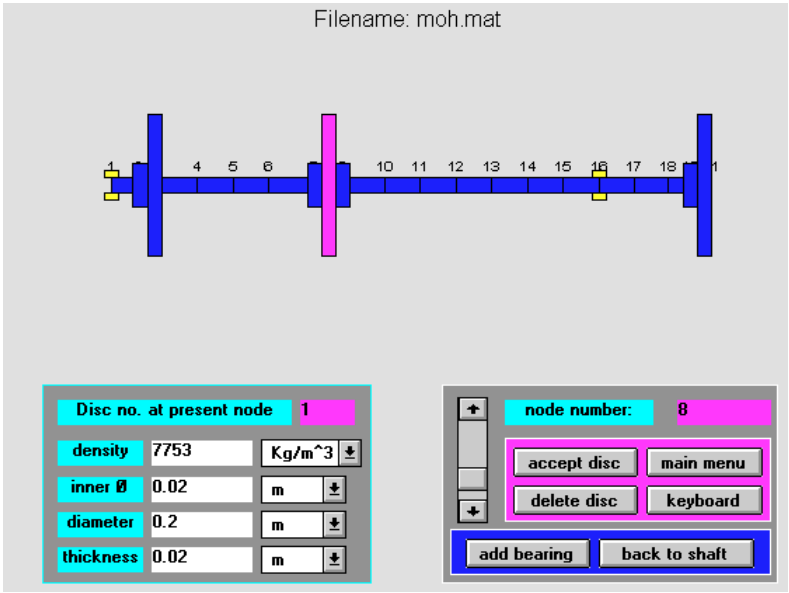


Figure 2.10 Interface for disc elements

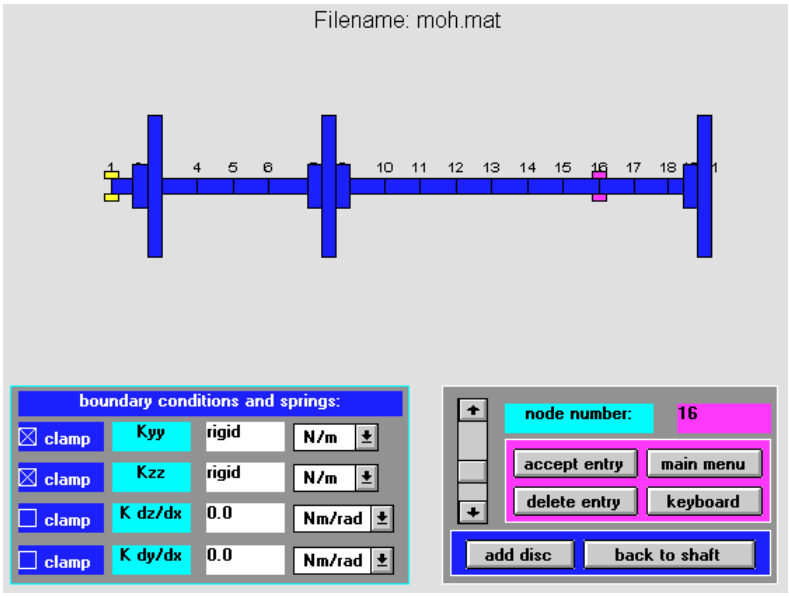


Figure 2.11 Interface for boundary conditions and bearings

3. Visualisation of Dynamic Properties

3.1 Display results of FE calculations

3.1.1 Display of Modeshapes

The solution of the eigenproblem yields modeshapes expressed in the generalised co-ordinates. These can be transformed into physical displacements in the Y and Z-directions via the shape functions described in section 2.2. However, the resulting eigenvectors are normally complex because of the complex values for the generalised co-ordinates usually obtained from solving the eigensystem, and hence a conversion into physically meaningful displacements has to be performed. Whirl orbits are forward or backward rotating ellipses, with forward whirl being defined as co-rotating in the same direction as the shaft, and backward whirl as counter-rotating.

Starting with the more common case of axisymmetric rotors, it is briefly described in the following paragraph how to extract the sense of the whirl direction and the modeshapes in general from the eigenvectors given as output from the FE program.

The deflection of the shaft centre of a rotor spinning at Ω follows an ellipse as a whirl orbit:

$$r(t) = Ae^{i\Omega t} + Be^{-i\Omega t} \tag{3.1}$$

which can be rewritten as

$$r(t) = (A+B)\cos\Omega t + \mathrm{i}(A-B)\sin(\Omega t)\,. \tag{3.2}$$

$A+B$ and $A-B$ are the semi-major and semi-minor axes of the ellipse. The direction of the elliptical whirl is determined by:

$$|A| > |B| \text{ for forward whirl and } |A| < |B| \text{ for backward whirl.} \tag{3.3}$$

The modeshapes given by the FE program for undamped or underdamped systems are complex conjugate pairs, and the whirl orbit is described by:

$$\begin{pmatrix} y(t) \\ z(t) \end{pmatrix} = \psi e^{\lambda t} + \overline{\psi} e^{\overline{\lambda} t} = 2\,\mathrm{Re}(\psi)\cos(\Omega t) - 2\,\mathrm{Im}(\psi)\sin(\Omega t) \tag{3.4}$$

Introducing the complex displacement u and rewriting (3.4),

$$u(t) = y(t) + \mathrm{i}\cdot z(t) = C\cos(\Omega t) + \mathrm{i}\cdot D\sin(\Omega t) \tag{3.5}$$

with $\frac{C}{2} = \mathrm{Re}(\psi_y) + \mathrm{i}\cdot \mathrm{Re}(\psi_z)$ and $\frac{D}{2} = \mathrm{i}\cdot \mathrm{Im}(\psi_y) - \mathrm{Im}(\psi_z)$,

where ψ_y and ψ_z are the y and z components of the modeshape, ψ.

Comparing Equations (3.2) and (3.5) gives the following relationships:

$$A = \frac{C+D}{2} \text{ and } B = \frac{C-D}{2} . \tag{3.6}$$

With Equation (3.3) the direction of the whirl can now be determined.

For systems with non-axisymmetric rotors, the whirl orbits are a superposition of phasors rotating at the speed of rotation and its higher harmonics. The results are the well-known Lissajous figures. With a whirl orbit of the form shown below,

$$u(t) = \sum_k A_k e^{\mathrm{i}\,k\Omega t} \tag{3.7}$$

the direction of the orbit is determined by the strongest component of all the A_k coefficients, where a negative k indicates backward whirl.

3.1.2 Display of the Campbell diagram

The natural frequencies of the system depend to a large extent on the speed of rotation of the rotor. In order to show this dependency, and as an important tool in understanding the vibration phenomena of rotating structures, the Campbell diagram plots the natural frequencies versus the speed of rotation. The range of rotation speed of interest is discretised with a specified number of points and the eigensystem solved for each of these points. This makes the computational effort required for a Campbell diagram over a high range of speeds and/or with a very high discretisation quite large when compared with other tasks such as computing the modeshapes at a particular speed of rotation. One recommendation for future improvement of this FE code is to introduce a modal or dynamic reduction technique as described in [13, 14]. In [15] Genta describes an iterative method based on a modal approach. It efficiently reduces the computation time, especially in the case where the splitting of damping and gyroscopic effects into proportional and non-proportional parts produces a non-proportional part that can be neglected.

A Campbell diagram can generally be plotted in two different ways. The first and most common is the plot with only positive frequencies and no distinction is made between forward and backward whirl. The second option is to analyse the direction of whirl as described in section 3.1.1 and to plot the frequencies that correspond to backward whirl as negative frequencies in the fourth quadrant in the Campbell diagram. This may often be of advantage, since forward or backward whirl have quite a different meaning for the material in terms of stresses and stress-cycles and, consequently, fatigue and internal damping, which in turn is an important instability parameter. There is the possibility, though, that whirl is not globally forward or backward, but may switch (possibly more than once) from one form to the other over the length of the rotor. This raises the question of how these mixed modes should be accounted for in a Campbell diagram with forward / backward whirl separation. The necessary calculation and analysis of the modeshapes, which is required for the separation in whirl direction only, also slows down the computation of the Campbell diagram. The Campbell diagram in this FE program features both, the more common type and forward / backward whirl separation. The algorithm for detecting the direction of the whirl is already incorporated in the code for calculating the modeshapes: it simply has to be added into the code for the computation of the natural frequencies for the Campbell diagram and the sign of the frequencies changed accordingly. Since this analysis slows down the computation of the Cambpell diagram, a selection has to be made beforehand whether forward/backward whirl separation is wanted. As the intersections of the natural frequencies with the first engine order (resonance due to unbalance), and sometimes the engine orders 0.5, 2, 3, ... (bearing and non-axisymmetric resonances) are of interest, these straight lines are included in this plot of the Campbell diagram.

3.1.3 Display of Unbalance Response

Unbalance is described in the literature as most common and strongest source of excitation for many rotating machines. The excitation due to unbalance is a synchronous force, so that wherever the first engine order line intersects with a natural frequency of the rotor system, large resonance amplitudes may be expected. The graph of unbalance response plots the amplitude of vibration due to unbalance over a specified speed of rotation. The amplitude of vibration is only the same in the Y and Z directions if both

rotor and stator are axisymmetric. The results shown in this program are therefore the maximum Y and Z components of the vibration.

3.1.4 Animation of whirl orbits

All different types of whirling can be visualised with the whirl function in the program. For easier programming, a complex notation for the vector q is used:

$$q = \begin{Bmatrix} y + \mathrm{i} \cdot z \\ -\beta + \mathrm{i} \cdot \gamma \end{Bmatrix} \tag{3.8}$$

Furthermore, there is a small advantage in using complex notation instead of a real co-ordinate approach. When the solution sought for whirling is of the following type, e.g. for unbalance,

$$q = q_0 \cdot e^{\mathrm{i}\lambda t} \tag{3.9}$$

the sign of the real part of the whirl frequency λ identifies the direction of the whirl motion (forward or backward) [16]. This is not the case for real co-ordinates, where the eigenvectors from the eigenproblem encountered in the study of free whirling need to be studied as well.

For $\lambda < \Omega$ (rotational speed) the whirling is supercritical, and subcritical for $\lambda > \Omega$. The vector q used in this function contains the deflection of the shaft-centre at a certain node at different times (usually over the period of one full whirl orbit). The vector q is plotted and the program shows an animation of the shaft moving along it's whirl orbit; the upper and lower halves of the shaft have different colours in order to visualise the rotation of the shaft and thus the direction of rotation (Figure 3.1).

Different parameters can be passed on to the function to drive the program, i.e. the number of frames per second in the animation, the total number of frames to compromise between waiting time and smoothness of the animation, and the number of cycles the frames are repeated when the "play again" button is pressed.

If no vector q (as calculated in the subroutines of the finite element program) is passed on to the function, the program can be used as a demonstration, which is based on the following case of a non-axisymmetric rotor with a symmetrical stator. As described in Chapter 2, the differential equation expressed in a stationary frame of reference has periodic time-varying

coefficients, while expressed in a rotating frame it has constant coefficients, only the solution needs to be transformed back into the stationary frame of reference. Referring to Equation (1.24) in section 2.2.2, with the only difference that the matrices are now assembled in complex notation, the equation of motion in rotating co-ordinates is of the following type:

$$M_{\mathrm{r}}\,\ddot{r} + G_{\mathrm{rot}}\,\dot{r} + K_{\mathrm{rot}}\,r = 0 \tag{3.10}$$

The solution for free whirling of the system is:

$$r = r_1 \cdot e^{\mathrm{i}\lambda' t} + r_2 \cdot e^{-\mathrm{i}\,\overline{\lambda'} t} \tag{3.11}$$

where r_1, r_2, λ' and $\overline{\lambda'}$ are
complex conjugate pairs, respectively.

λ' and $\overline{\lambda'}$ are the whirl frequencies in rotating co-ordinates, which can be transformed into whirl frequencies referring to the stationary frame of reference:

$$\lambda = \lambda' + \Omega, \qquad \overline{\lambda} = \overline{\lambda'} + \Omega \tag{3.12}$$

Remembering the relationship $r = q \cdot e^{\mathrm{i}\Omega t}$ (1.19) and substituting (3.12) into (3.11), the solution for free whirling expressed in stationary co-ordinates is:

$$q = r_1 \cdot e^{\mathrm{i}\lambda t} + r_2 \cdot e^{\mathrm{i}(2\Omega - \overline{\lambda})t} \tag{3.13}$$

An example of free whirling with r_2 = 0.2 , r_1 = 1 and λ / Ω = 0.2 is shown below:

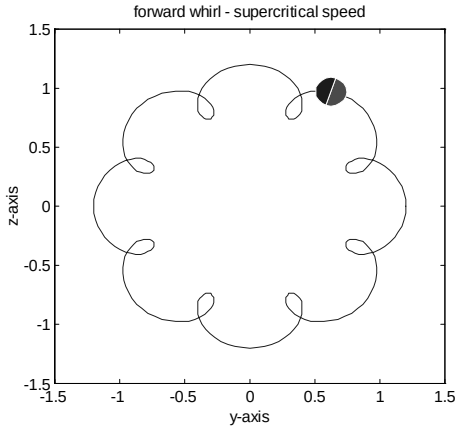


Figure 3.1

3.2 Visualisation of Travelling and Standing Wave Components

The main purpose of this visualisation tool is to animate the vibration of a rotating disc. In the beginning of the project, a need for such a tool was recognised. Without having a physical model or seeing an animation, it is difficult to purely imagine vibration patterns, especially if they consist of a superposition of more than one component. This tool allows the user to compose any vibration pattern consisting of a combination of individual components and to animate it. By this means of visualisation the perception of particular vibration patterns is improved and analysing vibration phenomena is thus supported.

As the rotation of the disc further complicates matters in terms of frequencies and perceived directions of travelling waves, animation in stationary as well as rotating frame of reference is offered.

The simulation of waves in the stationary and rotating frames of reference is more than an academic exercise. It is referred to the ROSTADYN project [2], in which rub between rotor and stator is investigated. The aim is to simulate loading of stationary turbine / compressor components in aircraft engines due to engine order excitation or similar excitations.

3.2.1 Description of vibration

Vibration described in terms of modes

A vibration of a disc can be written as summation over the contribution of all modes ψ_i.

$$x(t,\theta,r) = \sum \psi_i(\theta,r)\, p_i(t) \tag{3.14}$$

θ is the tangential co-ordinate. The radial co-ordinate r is of no interest here, so the deflections further dealt with are along a constant radius r_0 and are thus only a function of time t and angle φ: $x(t,\theta,r_0)$. The modal summation is valid for all structures with linear dynamic behaviour. The modeshapes of axisymmetric structures are harmonic functions in the circumferential direction with different nodal diameters (ND) or wave numbers n: $\psi_i = \gamma(r)\cos(n\varphi)$. The Figure below shows the first four modeshapes.

Figures 3.2 modeshapes with nodal diameters n = 1, 2, 3, 4.

For axisymmetric structures, these modes occur in pairs having identical frequencies and no preferred direction of propagation exists. For real (i.e. slightly mistuned) structures, these double eigenfrequencies split into close pairs.

Vibration described in terms of waves

Any combination of modeshapes is a modeshape, too, and one is free to choose:

$$\cos(n\theta) \pm \mathrm{i} \cdot \sin(n\theta) = e^{\pm \mathrm{i} n\theta}. \tag{3.15}$$

Modes are described in the form of $\sin(n\theta)$ or $\cos(n\theta)$, whereas travelling waves have the form $e^{\mathrm{i}n\theta}$ (forward direction) and $e^{-\mathrm{i}n\theta}$ (backward direction). Forward and backward travelling waves having the same amplitude, frequency and number of nodal diameters combine to form a standing wave (same as mode) as $e^{\mathrm{i}n\theta} + e^{-\mathrm{i}n\theta} = 2\cos(n\theta)$.

However, there exists an ambiguity in the literature concerning the term standing wave. The special case of a backward travelling wave having the same speed of rotation as the disc (but in the opposite direction) is sometimes referred to in the literature as a "standing wave" with respect to the stationary frame of reference, because the deflection of the disc in the stationary frame of reference appears static, although the material particles are actually oscillating; and this would be clear from an observation in the rotating frame of reference.

The effect of the choice of frame of reference is described with the Galilean transformation $\theta_{\mathrm{XYZ}} = \theta + \Omega t \Rightarrow e^{\mathrm{i}n\theta} = e^{\mathrm{i}n\theta_{\mathrm{XYZ}} - \mathrm{i}n\Omega t}$, where θ_{XYZ} is the circumferential angle in the stationary frame of reference. The directions of the travelling waves as observed in the stationary frame of reference are therefore shifted when compared with the propagation of the waves in the material.

3.2.2 2D Fourier transform for decomposition of vibration

A vibration pattern measured in an experiment can be decomposed into different spatial and frequential components of the signal [17]. In some cases the vibration patterns are known beforehand and under explicit control in the experiment, e.g. the experiments that will be conducted in the ROSTADYN project involve an experimental set-up where the excitation forces are tuned to excite specific vibration patterns [18, 17] to generate defined initial conditions. These vibration patterns must therefore be selected beforehand, a process where animation is bound to reduce the necessary number of experiments by supporting the experimenter in deciding upon a suitable vibration pattern.

Example of spatial separation:

A vibration signal is decomposed into its temporal and spatial components by a 2D Fourier transform, a short example of which is given here.

The number of sensors in this simulation is 8, and they are evenly distributed around the circumference of the disc. Referring to Shannon's sampling theorem, the highest number of nodal diameters that can be detected in this case without aliasing effects is 4. Due to forced excitation of the disc, let certain forward and backward travelling waves be excited:

$$x(t,\theta) = \underbrace{\cos(3t)\cos(2\theta)}_{\text{standing wave}} + \underbrace{3\cos(3t+2\theta)+\cos(3t-\theta)}_{\text{travelling waves}} + \underbrace{(0.3)}_{\text{DC}} \tag{3.16}$$

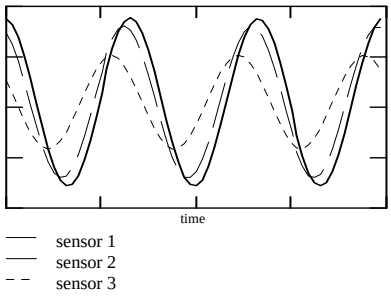


Figure 3.3 Signals from 3 sensors over time

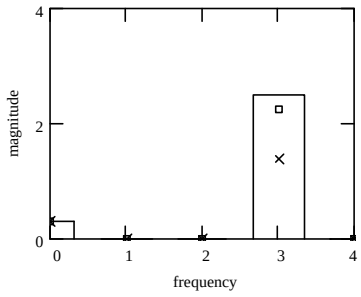


Figure 3.4 Frequency spectrum of the signal of the sensors

The signals, as three of the eight sensors see the vibration, are shown in Figure 3.3. the temporal Fourier transform of which reveals the frequency content of the vibration as shown in Figure 3.4. The response frequency is equal to the excitation frequency due to the linearity of the structure, so that

all components have the same frequency - the DC component would normally be filtered out. The whole frequency spectrum can be divided into parts, where each part holds the contents of the spectrum at one particular frequency. For further analysis, these parts are now treated individually. In this case, only the part at the frequency of excitation $\omega = 3$ rad / s contains any information. The result, the contribution of each sensor over the circumference to this part of the spectrum, is shown in Figure 3.5.

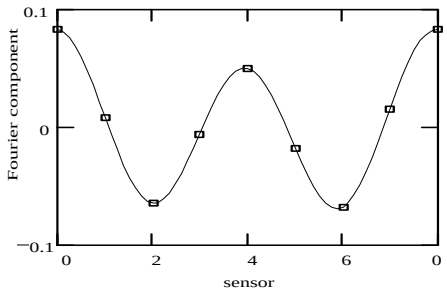


Figure 3.5 Spectral contribution of each sensor

Now a Fourier transform is performed over the spatial variable θ, yielding the decomposition of the vibration into travelling waves with forward / backward directions and different nodal diameters.

Since the structures under investigation are spatially periodic (axisymmetric in special cases), a Fourier transform can be applied properly without any windows since there is no leakage occurring.

The data shown in Figure 3.6 contain the spatial decomposition of the frequency spectrum at $\omega = 3$ rad / s shown in Figure 3.5.

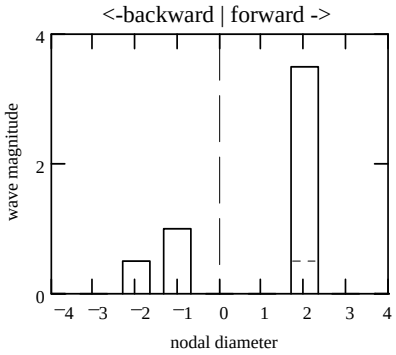


Figure 3.6 2D Fourier transform of the signals

$$x(t,\theta) = \underbrace{\cos(3t)\cos(2\theta)}_{\text{standing wave}} + \underbrace{3\cos(3t+2\theta)+\cos(3t-\theta)}_{\text{travelling waves}} + \underbrace{(0.3)}_{\text{DC}}$$

Equation (3.16) Synthesised vibration pattern

By comparison of Equation (3.16) and Figure 3.6 one can see that all information was conserved in the analysis. The sign of the nodal diameters *n* corresponds to forward / backward direction in the rotating frame. The vibration as represented in Figure 3.6 consists of a backward travelling wave with 2 ND of (here dimensionless) magnitude 0.5, a larger forward travelling wave with 2 ND and magnitude 3.5, and a backward travelling wave with 1 ND of magnitude 1. The *n* = -2 wave combines with 0.5 magnitude of the *n* = 2 wave to result in a 2 ND standing wave of magnitude 1, leaving a purely 2 ND forward travelling wave of magnitude 3.

3.2.3 The graphical interface for the vibration spectrum

Having decomposed the measured signal into its temporal and spatial components, the animation tool allows the user to visualise single vibration components or, what is more interesting, the superposition of any combination of these components, either as seen from a co-rotating observer or from a stationary position. The main interfaces of the animation tool are shown below (more detailed description in Appendix C):

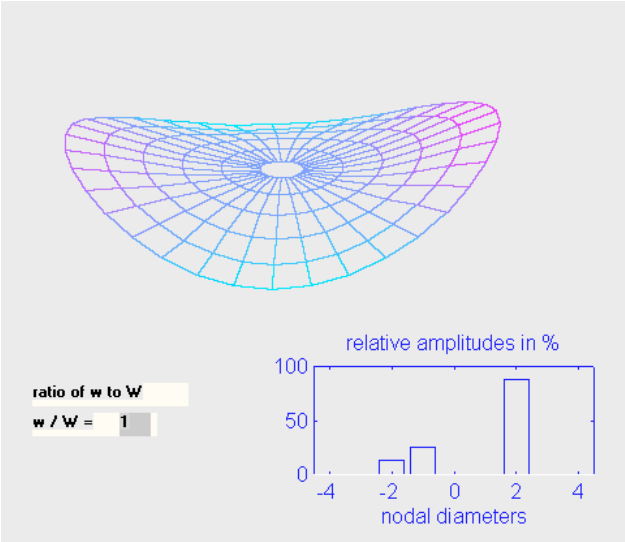


Figure 3.7 The animation window

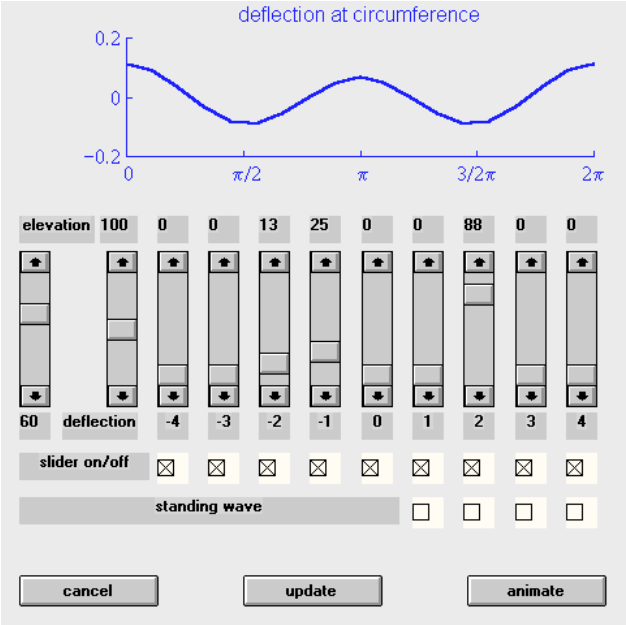


Figure 3.8 Interface for vibration input

In Figure 3.7 the window of the deflected disc is shown with a bar plot of the wave components present. Figure 3.8 shows the main interface for user input concerning the components of the vibration. Here the user can dial in a specific vibration pattern by giving the amplitude and direction of the travelling waves. Compare the deflection around the circumference shown in the graph in Figure 3.8 with the contribution of the sensors in Figure 3.5, and the corresponding wave patterns in Figure 3.6 and Figure 3.7.

There are sliders and editboxes to enter the amplitude of forward and backward travelling waves. The components can be switched on or off individually, and sliders can be synchronised for easy input of standing waves.

The absolute value of the speed of rotation Ω is irrelevant for the animation since it will not show a disc truly rotating at several 100s of rev / min on the screen. More important is the ratio r of excitation frequency to speed of rotation, $r = \omega_{ex} / \Omega$. The frequencies of the components with n nodal diameters can then be described by $\omega_n = \omega_{ex} + n\Omega = (r + n)\Omega$, expressed in the stationary frame of reference.

3.2.4 The interface for the animation parameters

The following animation parameters can be set: number of frames per second and overall repetitions for the playback speed and length, and number of frames per revolution and number of revolutions through which

the generation of the animation passes, which affects the smoothness of the animation. One needs to be aware that aliasing effects will occur if the number of frames per revolution N is not high enough to catch the waves over a certain frequency, so the program will give a warning if $N\Omega/2 < \omega_n$, which is equivalent to $N < 2|r+n|$, with n being the number of nodal diameters of the wave with the highest frequency present. Furthermore, if no temporal periodicity is ensured within the number of frames and rotations specified, the animation will jump between repetitions. This occurs if the sample period is not sufficient so that all active wave components can complete an integer number of cycles within this period. The program will then give a warning if the following value k is not an integer for all n:

$$k = rev \cdot \omega_n / \Omega = rev \cdot (r + n),$$

with *rev* being the number of disc revolutions in one cycle of the animation.

A further aid in the visualisation is the possibility to cut out a piece of material to enhance the perception of a rotating disc. The mesh shown on the disc often causes aliasing in the sense that one is tricked into believing the disc is rotating the wrong way around, so a piece cut out of a disc, meaning it has the same colour as the background, makes this quite clear.

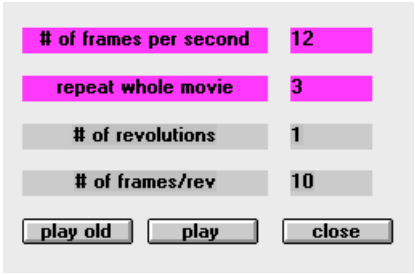


Figure 3.9 Animation parameters

4. Experimentally-Measured Data

Unfortunately, the electronic version of this chapter has been lost, and thus the pages 44 to 51 could not be included in this PDF file.

5. References

1. P. Schmiechen 1994 ROSTADYN internal report 5.01 Industrial review of the travelling wave speed coincidence. BRITE-EURAM 5463

2. A. Reister 1995 LISA user's guide. BRITE-EURAM project MARS BE5464

3. Y. D. Kim, C. W. Lee 1986 Journal of Sound and Vibration **111**(3), 441-456. Finite element analysis of rotor bearing sytems using a modal transformation matrix.

4. A.-C. Lee et al. 1992 Journal of Engineering for Industry 114, 465-475 Transient analysis of an asymmetric rotor-bearing system during acceleration

5. M. Meirovitch, 1980, Computational Methods in Structural Dynamics. New York: McGraw Hill

6. F. F. Ehrich (editor), 1992, Handbook of Rotordynamics. McGraw-Hill, New York.

7. R. Katz, C. W. Lee et al. 1988 Journal of Sound and Vibration **122**(1), 131-148. The dynamic response of a rotating shaft subject to a moving load.

8. C. W. Lee, 1993, Vibration Analysis of Rotors. Kluwer Academic Publishers, Dordrecht NL.

9. G. Ferraris and M. Lalanne, 1990, Rotordynamics Prediction in Engineering. John Whiley & Sons, Chichester.

10. Xu, R. Gasch, 1993: Kleiner Beitrag zur Behandlung linearer periodisch zeitvarianter Bewegungsgleichungen - Modale Entkopplung und Transformation in zeitinvariante Differentialgleichungen, Bericht aus dem Institut für Luft- und Raumfahrt der Technischen Universität Berlin, ILR-Mitteilung 278,.

11. Gasch, Knothe, 1989: Strukturdynamik II, Springer Verlag, Berlin.

12. R.J. Allemang and D. L. Brown 1982 1st IMAC, 110-116. A correlation coefficient for modal vector analysis.

13. K. E. Rauch and J. S. Kao 1980 Journal of Mechanical Desgin **102**(2), 360-368. Dynamic reduction in rotor dynamics by the finite element method.

14. R. Subbiah, R. B. Bhat and T. S. Sankar 1989 Journal of Mechanical Desgin **111**(4), 360-365. Dynamic response of rotors using modal reduction techniques.

15. G. Genta 1990 Journal of Sound and Vibration **155**(3) ,385-402. A fast modal technique for the computation of the Campbell diagram of multi-degree-of-freedom rotors.

16. G. Genta 1988 Journal of Sound and Vibration **124**(1) ,27-33. Whirling of unsymmetrical rotors: A finite element approach based on complex co-ordinates.

17. I. Bucher, D. J. Ewins, P. Schmiechen 1995 15th ASME Biennial Conference on Vibration and Noise, Boston, MA, USA. Multi-dimensional directional spectrograms and Campbell (Z-mod) diagrams for rotating machinery.

18. P. Schmiechen, D. J. Ewins, I. Bucher 1995 15th ASME Biennial Conference on Vibration and Noise, Boston, MA, USA. Exitation of arbitrary displacement / velocity conditions in rotationally periodic structures.

Appendix A

The Derivation of the Element Matrices

A.1 The transformation of equations of motion between frames of references

The following lines are a transcrip of a MAPLE file written to validate the equations given in section 2.2.2. It shows how the equation of motion in a rotating frame of reference can be derived when the matrices for the equation of motion in the stationary frame of reference are known. Shown here is ony the transformation for matrices of non-axisymmetric parts (exactly the ones derived in the following section), the matrices for non-axisymmetric shaft elements are quite large and this demonstration would become rather messy.

```
lambda:=Omega*t: s:=sin(2*lambda): c:=cos(2*lambda):
omega:=evalm(Omega*array(1..4,1..4,[[0,0,-1,0],[0,0,0,-1],[1,0,0,0],[0,1,0,0]]));
M:=array(1..4,1..4,[[m,0,0,0],[0,J[m]+J[d]*c,0,J[d]*s],[0,0,m,0],[0,J[d]*s,0,J[m]-
J[d]*c]]);
G:=Omega*array(1..4,1..4,[[0,0,0,0],[0,-2*J[d]*s,0,J[p]+2*J[d]*c],[0,0,0,0],[0,-
J[p]+2*J[d]*c,0,2*J[d]*s]]);
```

The crossproduct written in matrix notation

$$\omega := \begin{bmatrix} 0 & 0 & -\Omega & 0 \\ 0 & 0 & 0 & -\Omega \\ \Omega & 0 & 0 & 0 \\ 0 & \Omega & 0 & 0 \end{bmatrix}$$

The mass and stiffness matrices for non-axisymmetric parts (as derived in the following section)

$$M := \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_m + J_d \cos(2\,\Omega\, t) & 0 & J_d \sin(2\,\Omega\, t) \\ 0 & 0 & m & 0 \\ 0 & J_d \sin(2\,\Omega\, t) & 0 & J_m - J_d \cos(2\,\Omega\, t) \end{bmatrix}$$

$$G := \Omega \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2\,J_d \sin(2\,\Omega\,t) & 0 & J_p + 2\,J_d \cos(2\,\Omega\,t) \\ 0 & 0 & 0 & 0 \\ 0 & -J_p + 2\,J_d \cos(2\,\Omega\,t) & 0 & 2\,J_d \sin(2\,\Omega\,t) \end{bmatrix}$$

The rotation matrix

```
s1:=sin(lambda):  c1:=cos(lambda):
T:=array(1..4,1..4,[[c1,0,s1,0],[0,c1,0,s1],[-s1,0,c1,0],[0,-s1,0,c1]]):
R:=linalg [transpose] (T);
```

$$R := \begin{bmatrix} \cos(\Omega\,t) & 0 & -\sin(\Omega\,t) & 0 \\ 0 & \cos(\Omega\,t) & 0 & -\sin(\Omega\,t) \\ \sin(\Omega\,t) & 0 & \cos(\Omega\,t) & 0 \\ 0 & \sin(\Omega\,t) & 0 & \cos(\Omega\,t) \end{bmatrix}$$

transforming the element matrices

```
J[m]:=1/2*(J[eta]+J[zeta]); J[d]:=1/2*(J[eta]-J[zeta]);  Grot:=array(1..4,1..4,[]):
Mr:=evalm(T&*M&*R): Gr:=evalm(T&*G&*R):
Krot:=evalm((Gr+Mr&*omega)&*omega):  Mrotg:=evalm(2*Mr&*omega):
for ii from 1 to 4 do for jj from 1 to 4 do
  Mr[ii,jj]:=combine(expand(evalm(Mr[ii,jj])),trig);
  Gr[ii,jj]:=combine(expand(evalm(Gr[ii,jj])),trig);
  Krot[ii,jj]:=combine(expand(evalm(Krot[ii,jj])),trig);
  Grot[ii,jj]:=combine(expand(evalm(Gr[ii,jj]+Mrotg[ii,jj])),trig);
od; od;
Mr=evalm(Mr);Gr=evalm(Gr);Grot=evalm(Grot);Krot=evalm(Krot);
```

$$J_m := \frac{1}{2} J_\eta + \frac{1}{2} J_\zeta$$

$$J_d := \frac{1}{2} J_\eta - \frac{1}{2} J_\zeta$$

$$Mr = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_\eta & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & J_\zeta \end{bmatrix}$$

$$Gr = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega\,J_\eta - \Omega\,J_\zeta + \Omega\,J_p \\ 0 & 0 & 0 & 0 \\ 0 & \Omega\,J_\eta - \Omega\,J_\zeta - \Omega\,J_p & 0 & 0 \end{bmatrix}$$

$$Grot = \begin{bmatrix} 0 & 0 & -2\,m\,\Omega & 0 \\ 0 & 0 & 0 & -\Omega\,J_{\eta} - \Omega\,J_{\zeta} + \Omega\,J_{p} \\ 2\,m\,\Omega & 0 & 0 & 0 \\ 0 & \Omega\,J_{\eta} + \Omega\,J_{\zeta} - \Omega\,J_{p} & 0 & 0 \end{bmatrix}$$

$$Krot = \begin{bmatrix} -m\,\Omega^2 & 0 & 0 & 0 \\ 0 & -\Omega^2\,J_{\zeta} + \Omega^2\,J_{p} & 0 & 0 \\ 0 & 0 & -m\,\Omega^2 & 0 \\ 0 & 0 & 0 & -\Omega^2\,J_{\eta} + \Omega^2\,J_{p} \end{bmatrix}$$

A.2 The derivation of non-axisymmetric element matrices

The following lines are the transcript of a MAPLE file deriving the element matrices form the formulation of the kinetic and potential energies. Intermediate results are given as well, if the output is not too long to be imported into this document.

a list of substitutions needed for differentiation:

```
change1:={alpha=Omega*t,v=v(t),w=w(t),vd=diff(v(t),t),wd=diff(w(t),t),b
d=diff(beta(t),t),gd=diff(gamma(t),t),beta=beta(t),gamma=gamma(t)};
change2:={w(t)='w',v(t)='v',diff(gamma(t),t)=gd,diff(beta(t),t)=bd,diff(ga
mma(t),t,t)=gdd,diff(beta(t),t,t)=bdd,gamma(t)='gamma',beta(t)='beta',diff
(w(t),t)=wd,diff(v(t),t)=vd,diff(w(t),t,t)=wdd,diff(v(t),t,t)=vdd}:
```

$$change1 := \left\{ \alpha = \Omega\,t,\ v = \mathrm{v}(t),\ w = \mathrm{w}(t),\ vd = \frac{\partial}{\partial t}\mathrm{v}(t),\ wd = \frac{\partial}{\partial t}\mathrm{w}(t),\ bd = \frac{\partial}{\partial t}\beta(t),\ gd = \frac{\partial}{\partial t}\gamma(t),\ \beta = \beta(t),\ \gamma = \gamma(t) \right\}$$

The rotational part of the kinetic energy easiest expressed in body fixed co-ordinates, a co-ordiante transformation to the generalised co-ordinates used in the FE program is therefore necessary. So first step is the derivation of the transformation matrix:

The basic rotation matrices are:
Around X axis

```
ca:=cos(alpha): sa:=sin(alpha):RX:=array(1..3,1..3,[[1,0,0],[0,ca,-
sa],[0,sa,ca]]);
```

$$RX := \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

Around Y axis

```
cb:=cos(beta): sb:=sin(beta):RY:=array(1..3,1..3,[[cb,0,sb],[0,1,0],[-sb,0,cb]]);
```

$$RY := \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix}$$

Around Z-axis, with angle γ as projection of g1 into the X-Z plane

```
g1:=solve(tan(bt)=tan(gamma)*cb,bt);
cg1:=cos(g1): sg1:=sin(g1): cg:=cos(gamma): sg:=sin(gamma):
RZ:=array(1..3,1..3,[[cg,-sg,0],[sg,cg,0],[0,0,1]]);
RZ1:=array(1..3,1..3,[[cg1,-sg1,0],[sg1,cg1,0],[0,0,1]]);
```

$$g1 := \arctan(\tan(\gamma)\cos(\beta))$$

$$RZ := \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$RZ1 := \begin{bmatrix} \dfrac{1}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & -\dfrac{\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & 0 \\ \dfrac{\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \dfrac{1}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The transformation matrices are thus:

```
TX:=linalg [transpose] (RX);
TY:=linalg [transpose] (RY);
TZ:=linalg [transpose] (RZ);
TZ1:=linalg [transpose] (RZ1);
```

$$TX := \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) \\ 0 & -\sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

$$TY := \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) \\ 0 & 1 & 0 \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix}$$

$$TZ := \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$TZ1 := \begin{bmatrix} \frac{1}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \frac{\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & 0 \\ -\frac{\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \frac{1}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Since the transformation matrix T=inverse(R) and the basic transformation matrices are orthogonal, T=transpose(R).

The total transformation matrix from body-fixed co-ordinates to rotating is therefore:

TT:=evalm(TX&*TY&*TZ1);

$$TT := \begin{bmatrix} \frac{\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \frac{\cos(\beta)^2\tan(\gamma)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & -\sin(\beta) \\ -\frac{-\sin(\alpha)\sin(\beta)+\cos(\alpha)\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \frac{\sin(\alpha)\sin(\beta)\tan(\gamma)\cos(\beta)+\cos(\alpha)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \sin(\alpha)\cos(\beta) \\ \frac{\cos(\alpha)\sin(\beta)+\sin(\alpha)\tan(\gamma)\cos(\beta)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \frac{\cos(\alpha)\sin(\beta)\tan(\gamma)\cos(\beta)-\sin(\alpha)}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} & \cos(\alpha)\cos(\beta) \end{bmatrix}$$

The speed of rotation around the XYZ-axes expressed in the XYZ-Framework is: (bd = d(beta)/dt gd = d(gamma)/dt)

w3:=array(1..3,1..1,[[0],[bd],[gd]]); w1:=array(1..3,1..1,[[Omega],[0],[0]]);

$$w3 := \begin{bmatrix} 0 \\ bd \\ gd \end{bmatrix}$$

$$w1 := \begin{bmatrix} \Omega \\ 0 \\ 0 \end{bmatrix}$$

The speed of rotation expressed in the body coordinates of the rotating, deflected shaft is calculated via the following transformation:

w4:=evalm(TT&*(w3) + w1);

$$w4 := \begin{bmatrix} \frac{\cos(\beta)^2\tan(\gamma)\,bd-\sin(\beta)\,gd\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}}+\Omega \\ \frac{bd\sin(\alpha)\sin(\beta)\tan(\gamma)\cos(\beta)+bd\cos(\alpha)+\sin(\alpha)\cos(\beta)\,gd\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} \\ \frac{bd\cos(\alpha)\sin(\beta)\tan(\gamma)\cos(\beta)-bd\sin(\alpha)+\cos(\alpha)\cos(\beta)\,gd\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}}{\sqrt{1+\tan(\gamma)^2\cos(\beta)^2}} \end{bmatrix}$$

readlib(mtaylor);

```
proc() ... end
```

The second order Taylor series approximation for the vector of rotation is

```
w4t:=array(1..3,1..1,[]):
for ii from 1 to 3 do
w4t[ii,1]:=subs([alpha=Omega*t],mtaylor(w4[ii,1],[gamma=0,beta=0],2)):
od:
w_taylor=evalm(w4t);
```

$$w\_taylor = \begin{bmatrix} \Omega + \gamma\, bd - \beta\, gd \\ bd\cos(\Omega\, t) + \sin(\Omega\, t)\, gd \\ -bd\sin(\Omega\, t) + \cos(\Omega\, t)\, gd \end{bmatrix}$$

the kinetic energy of a non-axisymmetric part is thus (rotational part expressed in rotating co-ordinates)

```
omega:=w4:
EK:=1/2*m*(vd^2+wd^2)+1/2*(J[p]*omega[1,1]^2+J[eta]*omega[2,1]^2+J
[zeta]*omega[3,1]^2):
```

approximating the energy expressions with their second order Taylor series:

```
EK1:=mtaylor(EK,[gamma=0,beta=0],2):
EK1:=subs([alpha=Omega*t],combine(EK1,trig)):
asym_part[Kinetic_Energy]:=subs(change1,EK1):
```

The kinetic energy (integration of eks over xi done later) for the non-axisymmetric shaft element is thus

```
omega:=w4:
eks:=L/2*rho*(A*(vd^2+wd^2)+I[p]*omega[1,1]^2+I[eta]*omega[2,1]^2+
I[zeta]*omega[3,1]^2):
```

and the corresponding taylor series expansion:

```
eks1:=mtaylor(eks,[beta=0,gamma=0],2):
eks1:=subs([alpha=Omega*t],combine(eks1,trig)):
asym_shaft[Kinetic_Energy]:=subs(change1,eks1):
```

for symmetric elements the equations for the kinetic energies are:

for the disc:

```
disc[Kinetic_energy]=subs([J[eta]=J[diam],J[zeta]=J[diam]],asym_part[Kin
etic_Energy]):
```

for the shaft element (integral over xi missing):

```
shaft[Kinetic_Energy]=subs([I[eta]=I[diam],I[zeta]=I[diam]],asym_shaft[K
inetic_Energy]):
```

Some severe trouble is encounterd when trying to express the MAPLE output from the last four paragraphs above in a word document. Therefore the energy xpressions are typed in again here:

The kinetic energy for a non-axisymmetric part is:

```
EK1:=1/2*((vd^2+wd^2)*m+J[mean]*(bd^2+gd^2)+J[Delta]*(bd^2-
gd^2)*cos(2*alpha)+2*J[Delta]*bd*gd*sin(2*alpha)+J[polar]*Omega^2+J[p
olar]*Omega*(bd*gamma-gd*beta)):
asym_part[Kinetic_Energy]:=subs(change1,EK1);
```

$$asym\_part_{Kinetic\_Energy} := \frac{1}{2}\left(\left(\frac{\partial}{\partial t}\mathrm{v}(t)\right)^2+\left(\frac{\partial}{\partial t}\mathrm{w}(t)\right)^2\right)m+\frac{1}{2}J_{mean}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2+\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)+\frac{1}{2}J_{\Delta}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2-\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)\cos(2\,\Omega\,t)+J_{\Delta}\left(\frac{\partial}{\partial t}\beta(t)\right)\left(\frac{\partial}{\partial t}\gamma(t)\right)\sin(2\,\Omega\,t)+\frac{1}{2}J_{polar}\,\Omega^2+\frac{1}{2}J_{polar}\,\Omega\left(\left(\frac{\partial}{\partial t}\beta(t)\right)\gamma(t)-\left(\frac{\partial}{\partial t}\gamma(t)\right)\beta(t)\right)$$

The kinetic energy for a non-axisymmetric shaft element is:

```
eks1:=1/2*((vd^2+wd^2)*m+I[mean]*(bd^2+gd^2)+I[Delta]*(bd^2-
gd^2)*cos(2*Omega*t)+2*I[Delta]*bd*gd*sin(2*Omega*t)+I[polar]*Omega
^2+I[polar]*Omega*(bd*gamma-gd*beta)):
asym_shaft[Kinetic_Energy]:=subs(change1,eks1):
```

$$asym\_part_{Kinetic\_Energy} := \frac{1}{2}\left(\left(\frac{\partial}{\partial t}\mathrm{v}(t)\right)^2+\left(\frac{\partial}{\partial t}\mathrm{w}(t)\right)^2\right)m+\frac{1}{2}J_{mean}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2+\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)+\frac{1}{2}J_{\Delta}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2-\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)\cos(2\,\Omega\,t)+J_{\Delta}\left(\frac{\partial}{\partial t}\beta(t)\right)\left(\frac{\partial}{\partial t}\gamma(t)\right)\sin(2\,\Omega\,t)+\frac{1}{2}J_{polar}\,\Omega^2+\frac{1}{2}J_{polar}\,\Omega\left(\left(\frac{\partial}{\partial t}\beta(t)\right)\gamma(t)-\left(\frac{\partial}{\partial t}\gamma(t)\right)\beta(t)\right)$$

The potential energy for a non-axisymmetric shaft element in bending is:

```
asym_shaft[Potential_Energy]=L/2*Epsilon*(I[eta]*(diff(v(xi),xi,xi)/L^2)
^2+I[zeta]*(diff(w(xi),xi,xi)/L^2)^2); v:='v': w:='w':
```

$$asym\_shaft_{Potential\_Energy} = \frac{1}{2}L\,\mathrm{E}\left(\frac{I_{\eta}\left(\frac{\partial^2}{\partial\xi^2}\mathrm{v}(\xi)\right)^2}{L^4}+\frac{I_{\zeta}\left(\frac{\partial^2}{\partial\xi^2}\mathrm{w}(\xi)\right)^2}{L^4}\right)$$

for symmetric elements (i.e. disc) the equations for the kinetic energy is

```
disc[Kinetic_energy]=subs([J[mean]=J[diam],J[Delta]=0,bd=diff(beta(t),t)],
asym_part[Kinetic_Energy]);
```

$$disc_{Kinetic\_energy} = \frac{1}{2}\left(\left(\frac{\partial}{\partial t}\mathrm{v}(t)\right)^2+\left(\frac{\partial}{\partial t}\mathrm{w}(t)\right)^2\right)m+\frac{1}{2}J_{diam}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2+\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)+\frac{1}{2}J_{polar}\,\Omega^2+\frac{1}{2}J_{polar}\,\Omega\left(\left(\frac{\partial}{\partial t}\beta(t)\right)\gamma(t)-\left(\frac{\partial}{\partial t}\gamma(t)\right)\beta(t)\right)$$

for the shaft element (integral over xi missing):

```
shaft[Kinetic_Energy]=subs([I[mean]=I[diam],I[Delta]=0],asym_shaft[Kin
etic_Energy]);
```

$$shaft_{Kinetic\_Energy} = \frac{1}{2}L\,\rho\left(\left(\left(\frac{\partial}{\partial t}\mathrm{v}(t)\right)^2+\left(\frac{\partial}{\partial t}\mathrm{w}(t)\right)^2\right)A+I_{diam}\left(\left(\frac{\partial}{\partial t}\beta(t)\right)^2+\left(\frac{\partial}{\partial t}\gamma(t)\right)^2\right)+I_{polar}\,\Omega^2+I_{polar}\,\Omega\left(\left(\frac{\partial}{\partial t}\beta(t)\right)\gamma(t)-\left(\frac{\partial}{\partial t}\gamma(t)\right)\beta(t)\right)\right)$$

The matrices for the non-axisymmetric part attached to a node are derived in the following lines

```
qq:=array(1..4,1..1,[[v],[beta],[w],[gamma]]);qd:=array(1..4,1..1,[[vd],[bd],[
wd],[gd]]);qdd:=array(1..4,1..1,[[vdd],[bdd],[wdd],[gdd]]);
```

$$qq := \begin{bmatrix} v \\ \beta \\ w \\ \gamma \end{bmatrix}$$

$$qd := \begin{bmatrix} vd \\ bd \\ wd \\ gd \end{bmatrix}$$

$$qdd := \begin{bmatrix} vdd \\ bdd \\ wdd \\ gdd \end{bmatrix}$$

Using the approach from Meirovitch (works only for symmetric elements or the mass matrix if non-axisymmetric):

```
EK1:=subs([alpha=Omega*t],EK1):
Mp:=array(1..4,1..4,[]):
for ii from 1 to 4 do for jj from 1 to 4 do
  tmp:=diff(EK1 , qd[ii,1] , qd[jj,1]); Mp[ii,jj]:=tmp;
od; od;
M_asym_part=evalm(Mp);
```

$$M\_asym\_part = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & J_{mean} + J_{\Delta}\cos(2\,\Omega\,t) & 0 & J_{\Delta}\sin(2\,\Omega\,t) \\ 0 & 0 & m & 0 \\ 0 & J_{\Delta}\sin(2\,\Omega\,t) & 0 & J_{mean} - J_{\Delta}\cos(2\,\Omega\,t) \end{bmatrix}$$

deriving the gyroscopic matrix and mass, stiffness for comparison later on (by using Lagrange's formulation):

```
Gp:=array(1..4,1..4,[]): Kptest:=array(1..4,1..4,[]): Mptest:=array(1..4,1..4,[]):
for ii from 1 to 4 do for jj from 1 to 4 do
  tmp1:=-diff(EK1 , qq[ii,1]);
  tmp2:=diff(EK1, qd[ii,1]);   tmp2:=subs(change1,tmp2);
tmp3:=diff(tmp2,t):
  tmp3:=subs(change2,tmp3);
  Gp[ii,jj]:=diff(tmp1+tmp3,qd[jj,1]);
  Kptest[ii,jj]:=diff(tmp3+tmp1,qq[jj,1]);
```

```
  Mptest[ii,jj]:=diff(tmp3+tmp1,qdd[jj,1]);
od; od;
G_asym_part=evalm(Gp);
```

$$G\_asym\_part = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2\,J_{\Delta}\sin(2\,\Omega\,t)\,\Omega & 0 & J_{polar}\,\Omega + 2\,J_{\Delta}\cos(2\,\Omega\,t)\,\Omega \\ 0 & 0 & 0 & 0 \\ 0 & -J_{polar}\,\Omega + 2\,J_{\Delta}\cos(2\,\Omega\,t)\,\Omega & 0 & 2\,J_{\Delta}\sin(2\,\Omega\,t)\,\Omega \end{bmatrix}$$

comparing the results of Meirovitch and Lagrange for M, Kptest should be zero:

```
evalm(Mp-Mptest);evalm(Kptest);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

the shape functions used here for the shaft elements are:

```
N:=array(1..4,1..1,[[1-3*xi^2 + 2*xi^3],[L*(xi - 2*xi^2 + xi^3)] ,[3*xi^2 -
2*xi^3] ,[L*( xi^3-xi^2 ) ]]):
with (linalg):  qy:=array(1..4,[]):  qz:=array(1..4,[]):
   Warning: new definition for   norm
   Warning: new definition for   trace
```

$$N := \begin{bmatrix} 1 - 3\,\xi^2 + 2\,\xi^3 \\ L\left(\xi - 2\,\xi^2 + \xi^3\right) \\ 3\,\xi^2 - 2\,\xi^3 \\ L\left(\xi^3 - \xi^2\right) \end{bmatrix}$$

expressing the deflections via the shapefunctions

```
v:=N[1,1]*qy[1]+N[2,1]*qy[2]+N[3,1]*qy[3]+N[4,1]*qy[4]:
w:=N[1,1]*qz[1]+N[2,1]*qz[2]+N[3,1]*qz[3]+N[4,1]*qz[4]:
vd:=subs({qy[i]=qyd[i]}$i=1..4,v):  wd:=subs({qz[i]=qzd[i]}$i=1..4,w):
```

angular deflection expressed via shapefunctions:

```
unprotect(gamma):
gamma:=diff(v,xi)/L: gd:=diff(vd,xi)/L:
beta:=-diff(w,xi)/L:  bd:=-diff(wd,xi)/L:
eps1:=L/2*Epsilon*(I[eta]*(diff(v,xi,xi)/L^2)^2+I[zeta]*(diff(w,xi,xi)/L^2)
^2): eks1:=subs([I[mean]=1/2*(I[eta]+I[zeta]),I[Delta]=1/2*(I[eta]-
I[zeta])],eks1):
```

building arrays of co-ordinates to be able to differentiate (Lagrange)

```
q:=array(1..8,[(qy[i])$i=1..4,(qz[i])$i=1..4]):
qd:=array(1..8,[(qyd[i])$i=1..4,(qzd[i])$i=1..4]):
qdd:=array(1..8,[(qydd[i])$i=1..4,(qzdd[i])$i=1..4]):
```

one method of building the mass matrix, integrating the enery expression over ξ

```
M:=array(1..8,1..8):
for ii from 1 to 8 do for jj from 1 to 8 do
 tmp1:=diff(eks1 , qd[ii] , qd[jj]);    M[ii,jj]:= simplify( int( tmp1 , xi=0..1));
od; od;
```

building the gyroscopic matrix as well as the mass and stiffness matrices to check the formulation:

```
G:=array(1..8,1..8,[]): Ktest:=array(1..8,1..8,[]): Mtest:=array(1..8,1..8,[]):
for ii from 1 to 8 do for jj from 1 to 8 do
  tmp1:=-diff(eks1 , q[ii]); tmp2:=simplify(int(tmp1 , xi=0..1));
  tmp3:=diff(eks1, qd[ii]);
tmp3:=subs({qy[i]=qyt(t)[i]}$i=1..4,{qz[i]=qzt(t)[i]}$i=1..4,{qyd[i]=qydt(t)[i]
}$i=1..4,{qzd[i]=qzdt(t)[i]}$i=1..4,tmp3);
  tmp4:=diff(tmp3,t):
tmp4:=subs({diff(qyt(t)[i],t)=qyd[i]}$i=1..4,{diff(qzt(t)[i],t)=qzd[i]}$i=1..4,{
diff(qydt(t)[i],t)=qydd[i]}$i=1..4,{diff(qzdt(t)[i],t)=qzdd[i]}$i=1..4,{qyt(t)[i]
=qy[i]}$i=1..4,{qzt(t)[i]=qz[i]}$i=1..4,{qydt(t)[i]=qyd[i]}$i=1..4,{qzdt(t)[i]=q
zd[i]}$i=1..4,tmp4);
  tmp5:=simplify(int(tmp4, xi=0..1));
  G[ii,jj]:=diff(tmp2+tmp5,qd[jj]);
  Mtest[ii,jj]:=diff(tmp5,qdd[jj]);
```

```
  Ktest[ii,jj]:=diff(tmp2+tmp5,q[jj]);
od; od;
```

deriving the stiffness matrix:

```
K:=array(1..8,1..8,[]):
for ii from 1 to 8 do for jj from 1 to 8 do
   tmp1:=diff(eps1 , q[ii] , q[jj]);
  K[ii,jj]:= simplify(int( tmp1 , xi=0..1));
od; od;
```

showing that the two different ways to derive the mass and stiffness matrices yield the same results:

```
Mt:=array(1..8,1..8,[]): Kt:=array(1..8,1..8,[]): for ii from 1 to 8 do for jj from
1 to 8 do Mt[ii,jj]:=simplify(Mtest[ii,jj]-M[ii,jj]);
Kt[ii,jj]:=simplify(Ktest[ii,jj]*30*L / rho / Omega^2); od; od; evalm(Mt);
evalm(Kt);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

showing the matrices for symmetric elements (note the multiplications in the loops):

```
M1:=array(1..4,1..4,[]):  for ii from 1 to 4 do for jj from 1 to 4 do
M1[ii,jj]:=subs([I[eta]=A*d^2 / 16,I[zeta]=A*d^2 / 16],expand(evalm(M[ii,jj]
```

```
*3360*L/rho/A)));
od; od;
```

```
M_sym=evalm(M1);
```

$$M\_sym = \begin{bmatrix} 252\,d^2 + 1248\,L^2 & 176\,L^3 + 21\,L\,d^2 & 432\,L^2 - 252\,d^2 & -104\,L^3 + 21\,L\,d^2 \\ 176\,L^3 + 21\,L\,d^2 & 32\,L^4 + 28\,L^2\,d^2 & 104\,L^3 - 21\,L\,d^2 & -7\,L^2\,d^2 - 24\,L^4 \\ 432\,L^2 - 252\,d^2 & 104\,L^3 - 21\,L\,d^2 & 252\,d^2 + 1248\,L^2 & -176\,L^3 - 21\,L\,d^2 \\ -104\,L^3 + 21\,L\,d^2 & -7\,L^2\,d^2 - 24\,L^4 & -176\,L^3 - 21\,L\,d^2 & 32\,L^4 + 28\,L^2\,d^2 \end{bmatrix}$$

```
G1:=array(1..4,1..4,[]): for ii from 1 to 4 do for jj from 1 to 4 do
G1[ii,jj]:=subs([I[eta]=A*d^2/16,I[zeta]=A*d^2/16,I[polar]=A*d^2/8],exp
and(evalm(G[ii,jj+4]*240*L/rho/A/Omega/d^2)));
od; od;
```

```
G_sym=evalm(G1);
```

$$G\_sym = \begin{bmatrix} 36 & 3\,L & -36 & 3\,L \\ 3\,L & 4\,L^2 & -3\,L & -L^2 \\ -36 & -3\,L & 36 & -3\,L \\ 3\,L & -L^2 & -3\,L & 4\,L^2 \end{bmatrix}$$

```
K1:=array(1..4,1..4,[]): for ii from 1 to 4 do for jj from 1 to 4 do
K1[ii,jj]:=subs([I[eta]=I,I[zeta]=I],expand(evalm(K[ii,jj]*L^3/Epsilon/I)));
od; od;
```

```
K_sym=evalm(K1);
```

$$K\_sym = \begin{bmatrix} 12 & 6\,L & -12 & 6\,L \\ 6\,L & 4\,L^2 & -6\,L & 2\,L^2 \\ -12 & -6\,L & 12 & -6\,L \\ 6\,L & 2\,L^2 & -6\,L & 4\,L^2 \end{bmatrix}$$

```
i:='i': # used ascomlex unit in Matlab
```

extracting the matrices with sin(2*alpha) and cos(2*alpha) as coefficients and combining them to a complex matrix M2; M0 is matrix with constant coefficients

```
M0:=array(1..8,1..8,[]): M2c:=array(1..8,1..8,[]): M2s:=array(1..8,1..8,[]):
M2:=array(1..8,1..8,[]):
for ii from 1 to 8 do for jj from 1 to 8 do
aa:=subs([cos(2*Omega*t)=c2,sin(2*Omega*t)=s2],M[ii,jj]);
M2c[ii,jj]:=simplify(diff(aa,c2)); M2s[ii,jj]:=diff(aa,s2);
M2[ii,jj]:=simplify((M2c[ii,jj]-i*M2s[ii,jj])/2);
M0[ii,jj]:=expand(evalm(M[ii,jj]-M2c[ii,jj]*cos(2*Omega*t)-
M2s[ii,jj]*sin(2*Omega*t)));
od; od;
```

and the gyroscopic matrices

```
G0:=array(1..8,1..8,[]): G2c:=array(1..8,1..8,[]): G2s:=array(1..8,1..8,[]):
G2:=array(1..8,1..8,[]):
for ii from 1 to 8 do for jj from 1 to 8 do
aa:=subs([cos(2*Omega*t)=c2,sin(2*Omega*t)=s2],G[ii,jj]);
G2c[ii,jj]:=diff(aa,c2); G2s[ii,jj]:=diff(aa,s2);
G2[ii,jj]:=simplify((G2c[ii,jj]-i*G2s[ii,jj])/2);
G0[ii,jj]:=expand(evalm(G[ii,jj]-G2c[ii,jj]*cos(2*Omega*t)-
G2s[ii,jj]*sin(2*Omega*t)));
od; od;
```

and the stiffness matirces

```
K0:=array(1..8,1..8,[]): K2c:=array(1..8,1..8,[]): K2s:=array(1..8,1..8,[]):
K2:=array(1..8,1..8,[]):
for ii from 1 to 8 do for jj from 1 to 8 do
aa:=subs([cos(2*Omega*t)=c2,sin(2*Omega*t)=s2],K[ii,jj]);
K2c[ii,jj]:=simplify(diff(aa,c2)); K2s[ii,jj]:=diff(aa,s2);
K2[ii,jj]:=simplify((K2c[ii,jj]-i*K2s[ii,jj])/2);
K0[ii,jj]:=expand(evalm(K[ii,jj]-K2c[ii,jj]*cos(2*Omega*t)-
K2s[ii,jj]*sin(2*Omega*t)));
od; od;
```

A.3 Assembly of the hyper matrices

Hills approach is used to transform the equations of motion into hyper matrices by balancing the terms containing the same harmonics. The column wise assembly of the hyper M, G, and K matrices is shown here. (K1 only used if bearings are approximated in a series)
NB: index cc stands for complex conjugate

```
M:=(M0+M2*exp(i*2*Omega*t)+M2cc*exp(-
i*2*Omega*t));G:=(G0+G2*exp(i*2*Omega*t)+G2cc*exp(-
i*2*Omega*t));K:=(K0+K1*exp(i*Omega*t)+K1cc*exp(-
i*Omega*t)+K2*exp(i*2*Omega*t)+K2cc*exp(-i*2*Omega*t));
```

$$M := M0 + M2\, e^{(2\, i\, \Omega\, t)} + M2cc\, e^{(-2\, i\, \Omega\, t)}$$

$$G := G0 + G2\, e^{(2\, i\, \Omega\, t)} + G2cc\, e^{(-2\, i\, \Omega\, t)}$$

$$K := K0 + K1\, e^{(i\, \Omega\, t)} + K1cc\, e^{(-i\, \Omega\, t)} + K2\, e^{(2\, i\, \Omega\, t)} + K2cc\, e^{(-2\, i\, \Omega\, t)}$$

component k of the solution vector (note: underscore stands for -k): (The primitive Math Office for MAPLE does not allow output in two rows, hence incredible small print, I am sorry.)

```
q:=qk(t)*exp(+i*k*Omega*t)+qk_(t)*exp(-i*k*Omega*t);
qd:=diff(q,t);
qdd:=diff(qd,t);
```

$$q := \mathrm{qk}(t)\, e^{(i\, k\, \Omega\, t)} + \mathrm{qk\_}(t)\, e^{(-i\, k\, \Omega\, t)}$$

$$qd := \left(\frac{\partial}{\partial t}\,\mathrm{qk}(t)\right) e^{(i\, k\, \Omega\, t)} + \mathrm{qk}(t)\, i\, k\, \Omega\, e^{(i\, k\, \Omega\, t)} + \left(\frac{\partial}{\partial t}\,\mathrm{qk\_}(t)\right) e^{(-i\, k\, \Omega\, t)} - \mathrm{qk\_}(t)\, i\, k\, \Omega\, e^{(-i\, k\, \Omega\, t)}$$

$$qdd := \left(\frac{\partial^2}{\partial t^2}\,\mathrm{qk}(t)\right) e^{(i\, k\, \Omega\, t)} + 2\left(\frac{\partial}{\partial t}\,\mathrm{qk}(t)\right) i\, k\, \Omega\, e^{(i\, k\, \Omega\, t)} + \mathrm{qk}(t)\, i^2\, k^2\, \Omega^2\, e^{(i\, k\, \Omega\, t)} + \left(\frac{\partial^2}{\partial t^2}\,\mathrm{qk\_}(t)\right) e^{(-i\, k\, \Omega\, t)} - 2\left(\frac{\partial}{\partial t}\,\mathrm{qk\_}(t)\right) i\, k\, \Omega\, e^{(-i\, k\, \Omega\, t)} + \mathrm{qk\_}(t)\, i^2\, k^2\, \Omega^2\, e^{(-i\, k\, \Omega\, t)}$$

the equation of motion for this particular component and substituting qk(t)=qk*exp(lambda*t):

```
e:=M*qdd+G*qd+K*q;
e:=subs([diff(qk(t),t,t)=qkdd*lambda^2,diff(qk_(t),t,t)=qk_dd*lambda^2,d
iff(qk(t),t)=qkd*lambda,diff(qk_(t),t)=qk_d*lambda,qk(t)='qk',qk_(t)='qk_'
],e):
```

In the hyper matrices, the rows make up the equations for the balanced frequencies. The structure of the assembly of Hill's hyper-M, G, K matrices

columnwise (i.e. the contribution of a vector qk throughout all the balanced equations) can be seen here :

```
f:=diff(e,qkdd);
```

$$f := \left(M0 + M2\, e^{(2\, i\, \Omega\, t)} + M2cc\, e^{(-2\, i\, \Omega\, t)} \right) \lambda^2\, e^{(i\, k\, \Omega\, t)}$$

this is the column of the gyroscopic matix for vector qk's contribution:

```
diff(e,qkd);
```

$$2 \left(M0 + M2\, e^{(2\, i\, \Omega\, t)} + M2cc\, e^{(-2\, i\, \Omega\, t)} \right) \lambda\, i\, k\, \Omega\, e^{(i\, k\, \Omega\, t)} + \left(G0 + G2\, e^{(2\, i\, \Omega\, t)} + G2cc\, e^{(-2\, i\, \Omega\, t)} \right) \lambda\, e^{(i\, k\, \Omega\, t)}$$

the stiffness matrix:

```
diff(e,qk);
```

$$\left(M0 + M2\, e^{(2\, i\, \Omega\, t)} + M2cc\, e^{(-2\, i\, \Omega\, t)} \right) i^2\, k^2\, \Omega^2\, e^{(i\, k\, \Omega\, t)} + \left(G0 + G2\, e^{(2\, i\, \Omega\, t)} + G2cc\, e^{(-2\, i\, \Omega\, t)} \right) i\, k\, \Omega\, e^{(i\, k\, \Omega\, t)} + \left(K0 + K1\, e^{(i\, \Omega\, t)} + K1cc\, e^{(-i\, \Omega\, t)} + K2\, e^{(2\, i\, \Omega\, t)} + K2cc\, e^{(-2\, i\, \Omega\, t)} \right) e^{(i\, k\, \Omega\, t)}$$

Appendix B

Comparison of LISA and Measurement Results

B.1 Comparison of the measurements from the ROSTADYN rig

The measurements were taken at four discrete speeds of rotation and the natural frequencies for the first two bending modes were measured. The FW indicates forward whirl (upper branch in the Campbell diagram, hece higher frequencies) and BW backward whirl (lower branch in Campbell diagram).

speed of rotation [rpm]		first Mode FE results [Hz]	first Mode measured data [Hz]	second Mode FE results [Hz]	second Mode measured data [Hz]
0	"BW"	36.4129	36.51	128.6948	129.81
0.	"FW"		36.58		130.01
300	BW	35.7411	36	125.4269	127
300	FW	37.0903	37.4	132.0407	133.5
600	BW	35.0755	35.04	122.2392	123.52
600	FW	37.7727	37.90	135.4616	136.62
900	BW	34.4168	34.6	119.1337	120.7
900	FW	38.4594	38.8	138.9543	140.2
1200	BW	33.7656	33.68	116.1118	117.6
1200	FW	39.1497	39.37	142.5149	143.6
1800	BW	32.4879	32.35	110.3226	112
1800	FW	40.5382	40.86	149.8226	150.9
2100	BW	31.8623	31.8	107.5560	109.2
2100	FW	41.2352	41.7	153.5599	154.6
3000	BW	30.0442	30	99.7652	101.4
3000	FW	43.3285	43.85	165.0383	165.8
3600	BW	28.8845	28.56	94.9867	96.4
3600	FW	44.7197	45.29	172.8503	173.3

The maximum deviation for the first mode is 1.25%, and for the second mode 1.6%. The rig shows some nonlinear behaviour, which is of relevance here because the natural frequencies become a function of amplitude. The measurements with two digits after the decimal point are correct to about

0.02 Hz and refer to an amplitude of 50 m/s^2 at one end. The measurements with less digits after the decimal point are less accurate being from uncontrolled amplitude tests. These measurements were done by Mr. Antony Stanbridge for the ROSTADYN Task No. 1.3.

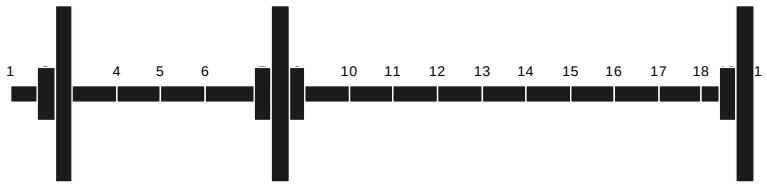


Figure B.1 The ROSTADYN FE-model, compared with measurements

B.2 Comparison of the results optained with LISA

The natural frequencies of the first five bending modes of the MARS rig calculated with the FE program are compared witht the results from LISA:

Mode No.	FE-results [Hz]	LISA results [Hz]
1	5.8574e+001	0.58539E+02
2	1.2350e+002	0.12318E+03
3	2.9440e+002	0.29207E+03
4	4.7943e+002	0.47115E+03
5	8.2434e+002	0.81037E+03
6	1.2082e+003	0.11745E+04
7	1.5827e+003	0.15324E+04
8	2.0988e+003	0.20857E+04
9	2.4664e+003	0.24093E+04
10	3.2621e+003	0.32203E+04

Figure B.1 The MARS FE model of the shaft, compared with LISA results

B.3 The element file of the model in LISA format as sent from Kaiserslautern

```
=====================================================================
# ======
# model of mars testrig
#
# part:  shaft
#
#
# file: shaft.mod
# date: 27.09.94
#
# ====== ==============================================================
#
matiso  1      1.96E+8     .3    7.850E-6
#
#        id        x      y       z
node     1        0.      0.      0.
node     2       15.      0.      0.
node     3       35.      0.      0.
node     4       43.27    0.      0.
node     5       94.      0.      0.
node     6      132.      0.      0.
#
#
# AMB A  VV
node     7      180.      0.      0.
node     8      205.      0.      0.
node     9      230.      0.      0.
# AMB A  ^^
#
node    10      280.      0.      0.
node    11      330.      0.      0.
node    12      380.      0.      0.
node    13      430.      0.      0.
node    14      480.      0.      0.
node    15      530.      0.      0.
#
#
```

```
# AMB B  VV
node    16      560.      0.      0.
node    17      585.      0.      0.
node    18      610.      0.      0.
# AMB B  ^^
#
node    19      660.      0.      0.
node    20      710.      0.      0.
node    21      760.      0.      0.
node    22      810.      0.      0.
node    23      860.      0.      0.
node    24      910.      0.      0.
node    25      960.      0.      0.
node    26     1021.      0.      0.
node    27     1046.      0.      0.
node    28     1088.      0.      0.
#
#
# mid of coupling  VV
node    29     1130.6     0.      0.
#
#
node    30     1135.      0.      0.
#
#
#
#
# streiche alle axialen und torsionsfreiheitsgrade
#
#        id   node  dof1  dof2  dof3  dof4  dof5  dof6
# fixed     1    1     1                 4
# fixed     1    2     1                 4
# fixed     1    3     1                 4
# fixed     1    4     1                 4
# fixed     1    5     1                 4
# fixed     1    6     1                 4
# fixed     1    7     1                 4
# fixed     1    8     1                 4
# fixed     1    9     1                 4
# fixed     1   10     1                 4
# fixed     1   11     1                 4
# fixed     1   12     1                 4
# fixed     1   13     1                 4
# fixed     1   14     1                 4
# fixed     1   15     1                 4
# fixed     1   16     1                 4
# fixed     1   17     1                 4
# fixed     1   18     1                 4
# fixed     1   19     1                 4
# fixed     1   20     1                 4
# fixed     1   21     1                 4
# fixed     1   22     1                 4
# fixed     1   23     1                 4
# fixed     1   24     1                 4
# fixed     1   25     1                 4
# fixed     1   26     1                 4
#

# flexible shaft
#
#       eid     pid     node1   node2     xvec    yvec    zvec
beam     1       1         1       2       0.      0.      1.
beam     2       2         2       3       0.      0.      1.
beam     3       3         3       4       0.      0.      1.
beam     4       3         4       5       0.      0.      1.
beam     5       4         5       6       0.      0.      1.
beam     6       5         6       7       0.      0.      1.
beam     7       6         7       8       0.      0.      1.
beam     8       6         8       9       0.      0.      1.
beam     9       5         9      10       0.      0.      1.
beam    10       5        10      11       0.      0.      1.
beam    11       5        11      12       0.      0.      1.
beam    12       5        12      13       0.      0.      1.
beam    13       5        13      14       0.      0.      1.
beam    14       5        14      15       0.      0.      1.
beam    15       5        15      16       0.      0.      1.
beam    16       6        16      17       0.      0.      1.
beam    17       6        17      18       0.      0.      1.
beam    18       5        18      19       0.      0.      1.
beam    19       5        19      20       0.      0.      1.
beam    20       5        20      21       0.      0.      1.
beam    21       5        21      22       0.      0.      1.
```

```
beam    22       5        22      23       0.      0.      1.
beam    23       5        23      24       0.      0.      1.
beam    24       5        24      25       0.      0.      1.
beam    25       5        25      26       0.      0.      1.
beam    26       4        26      27       0.      0.      1.
beam    27       7        27      28       0.      0.      1.
beam    28       8        28      29       0.      0.      1.
beam    29       8        29      30       0.      0.      1.

#
# properties of the beams
#
#  pbeam    pid   mid   rect  b     h
   pbeam     1     1    rect  16.   16.

#  pbeam    pid   mid   cyl   ri    ra
   pbeam     2     1    cyl   0.    11.6
   pbeam     3     1    cyl   0.    16.025
   pbeam     4     1    cyl   0.    17.5
   pbeam     5     1    cyl   0.    18.5
   pbeam     6     1    cyl   0.    19.0
   pbeam     7     1    cyl   0.    14.8
   pbeam     8     1    cyl   3.5   13.8


#=====          magnetic          bearing          bushes          ================
==================================
#
# (shaft part yielding stiffness)
#
beam    30       9         7       8       0.      0.      1.
beam    31       9         8       9       0.      0.      1.
beam    32       9        16      17       0.      0.      1.
beam    33       9        17      18       0.      0.      1.

#  pbeam    pid   mid   cyl   ri    ra
   pbeam     9     1    cyl   19.0  40.0

# (shaft part yielding mass & inertia)
#
# disk    eid   2   node   m        thetae         thetap       vx  vy  vz
disk      35    2     8    3.163    5.36E+3       7.19E+3       1.  0.  0.
disk      36    2    17    3.154    5.34E+3       7.163E+3      1.  0.  0.


# ====== rigid disk 20 kg ===================
============================================
#
# disk    eid   2   node   m        thetae         thetap       vx  vy  vz
disk      37    2     4    20.94    2.42E+5       4.94E+5       1.  0.  0.


# ====== coupling (rotor part) ==============
============================================
#
# disk    eid   2   node   m        thetae         thetap       vx  vy  vz
disk      38    2    29    1.022    1.2E+3        2.1E+5        1.  0.  0.


# ====== nut (coupling) =====================
============================================
#
# disk    eid   2   node   m        thetae         thetap       vx  vy  vz
disk      39    2    27    0.454    1.928E+2      2.755E+2      1.  0.  0.


# ====== nut (disk) =========================
========================================
#
# disk    eid   2   node   m        thetae         thetap       vx  vy  vz
disk      40    2     3    0.056    1.174E+1      1.448E+1      1.  0.  0.


# gstif22   26       1 1      0 0      1.E+1 -1.E+1 -1.E+1 1.E+1

quit
```

B.4 The Matlab functions used for reading LISA format

B.4.1 readlisa.m

```
function [nodeinf,beam,disk,pbeam,matiso]=readlisa
% [nodeinf,beam,disk,pbeam,matiso]=readlisa
%
% reads LISA files and converts the lines of the file
% into numeric matrices
%
% 19.12.94 GvG

if nargin==0,
   [File pt]=uigetfile('*.mod','Choose model file');
end
fid=fopen([pt File]);
if fid==-1
   return
end

nodeinf=[];beam=[];disk=[];pbeam=[];matiso=[];

while 1

   line = fgetl(fid);
   if ~isstr(line), break, end
      if length(line)>4
         ind=1;
         while isspace(line(ind)) & (ind<length(line)-1),
          ind=ind+1;
       end
     if ~strcmp(line(ind),'#'),
        if strcmp(line(ind:ind+3),'node')
           vals=['[' line(ind+4:length(line)) ']'];
           val=eval(vals); % assuming all in x direc.
           nodeinf=[nodeinf ; val(1:2)];
         elseif  strcmp(line(ind:ind+3),'beam')
           vals=['[' line(ind+4:length(line)) ']'];
           val=eval(vals); %assuming all in z direc.
           beam=[beam ; val(1:4)];
         elseif  strcmp(line(ind:ind+3),'disk')
           vals=['[' line(ind+4:length(line)) ']'];
           val=eval(vals);
           disk=[disk ; val];
         elseif  strcmp(line(ind:ind+4),'pbeam')
           cyl=999;rect=888;gen=777;
           vals=['[' line(ind+5:length(line)) ']'];
           val=eval(vals);
           pbeam=[pbeam ; val];
         elseif  strcmp(line(ind:ind+5),'matiso')
           vals=['[' line(ind+6:length(line)) ']'];
           val=eval(vals);
           matiso=[matiso ; val];

         end     % if nodeinf, beam, disk ...
       end       % if not comment line
   end           % if length(line)>4

end            % while fopen
fclose(fid);
```

B.4.2 lisa2ib.m

```
function [nodes,shafts,shafts2,disks,disks2,unsym,name]=lisa2ib
% [nodes,shafts,shafts2,disks,disks2,unsym,FileName]=lisa2ib
%
% converts LISA's format of beam, disk ...matrices into
% shafts etc. suitable for the aleerot function
% example of format:
% shafts=[elem_no length diameter E dens]
%
% 19.12.94 GvG
```

```
% 5.2.95 minor changes GvG

if nargin<2,
   pt=[];
end
if nargin<1,
   FileName=[];
end
[nodeinf,beam,disk,pbeam,matiso]=readlisa;

% units at present not converted
% convert into m kg s system
% nodeinf(:,2)=nodeinf(:,2)/1000
%matiso(:,2)=matiso(:,2)*1000;
%matiso(:,4)=matiso(:,4)*1e9;
%pbeam(:,4:5)=pbeam(:,4:5)/1000;

% beam assumed to contain elements ordered:up=first,down=last
% shafts=[elem_no length diameter E dens inner_diameter]
% belem=[length diameter E dens inner_diameter node_no])
Nbeam=length(beam(:,1));
temp=[];shafts2=[];unsym=[];shafts3=[]; %asymmetric parts in future

for m=1:Nbeam
    belem=zeros(1,6);
    salida=0;n=0;
    belem(6)=beam(m,3);  % node number
    belem(1)=nodeinf(beam(m,4),2)-nodeinf(beam(m,3),2); % length
    while n<length(pbeam(:,1)) & salida==0
      n=n+1;
      if beam(m,2)==pbeam(n,1)
         if pbeam(n,3)==999
            belem(2)=2*pbeam(n,5);   % convert into meters
            belem(5)=2*pbeam(n,4);
            p=0;
            while p<length(matiso(:,1)) & salida==0
                p=p+1;
                if matiso(p,1)==pbeam(n,2)
                   belem(3)=matiso(p,2);
                   belem(4)=matiso(p,4);
                   salida=1;
                end
            end
         elseif pbeam(n,3)==888
            disp(['at node number ' int2str(belem(6)) ]);
            disp('element rectangular, assumes d=sqrt(pi*b*h/4)')
            belem(2)=sqrt(pi*pbeam(n,5)*pbeam(n,4)/4);
            belem(5)=0;shafts2=[shafts2;belem(6)];
            p=0;
            while p<length(matiso(:,1)) & salida==0
                p=p+1;
                if matiso(p,1)==pbeam(n,2)
                   belem(3)=matiso(p,2);
                   belem(4)=matiso(p,4);
                   salida=1;
                end
            end
         else,
             % shafts2=[node_no length A mass I_polar I_2 I_3]
             shafts3=[shafts3;belem(6)];  % not yet implemented
             salida=1;
         end
      end
    end   % while
    temp=[temp;belem(6),belem(1:5)];
end   % for
dl=[];   % nodes=elements+1 dl=[node_no length]
[nr m]=size(temp);
dl=[dl;temp(:,1:2)];
%   for n=1:2,   % options for future use such as asym.elements
%      whatshaft=['shaft' num2str(n)];
%      if exist(whatshaft),
%         eval('[nr m]=size(whatshaft);');
%   if m>4,eval(['dl=[dl;' whatshaft '(:,1:2)];']);end
%      end
%   end
ne=max(dl(:,1));  % number of shaft elements
[n s]=sort(dl(:,1));  % sort order of elements corresp. to node
shafts=temp(s,:);
nodes=nodeinf(1:ne+1,2);

% disks=[nod_no. diameter thickness density d_inner]
```

```
% disks2=[node_no mass I_polar I_equatorial]
disks=[];disks2=[];
for m=1:length(disk(:,1))
    if disk(m,2)==1,  % format disks above
       del(1)=disk(m,4);
       del(2)=2*disk(m,6);%/1000;
       del(3)=disk(m,7);%/1000;
       p=0;salida=0;
       while p<length(matiso(:,1)) & salida==0
          p=p+1;
          if matiso(p,1)==disk(m,3)
             del(4)=matiso(p,4);
             salida=1;
          end
       end
       del(5)=disk(m,5);%/1000;
       disks=[disks;del];
       clear del;
    else            % format disks2 above
       del(1)=disk(m,3);
       del(2)=disk(m,4);
       del(3)=disk(m,6);%*1e-6;
       del(4)=disk(m,5);%*1e-6;
       disks2=[disks2;del];
       clear del;
    end
end
[name pt]=uiputfile('*.mat','Save converted matrices ?');
if isstr(name)
   eval(['save ' pt name ' nodes shafts shafts2 shafts3 disks disks2 unsym name
']);
end
```

Appendix C

The Graphical User Interfaces

The graphical user interface allows easy construction of the model as well as easy manipulation of data entered in a previous session. The user-interface controls are a powerful Matlab tool to build graphical interfaces in the Matlab language that increase user comfort enormously and can avoid repeated input. When used sensibly the use of the software can become far more intuitive. These elements provide the possibility of building a screen full of push buttons, sliders, menus and boxes where the user can enter data via keyboard in the appropriate boxes appearing in the figure window or make selections per mouse click. The fact that these ui-controls are event-driven functions makes the interface highly flexible, e.g. discs and bearings, as well as any element properties, can be entered in a completely arbitrary order, even before the shaft has been constructed.

C.1 The Grahical User Interface of the FE Program

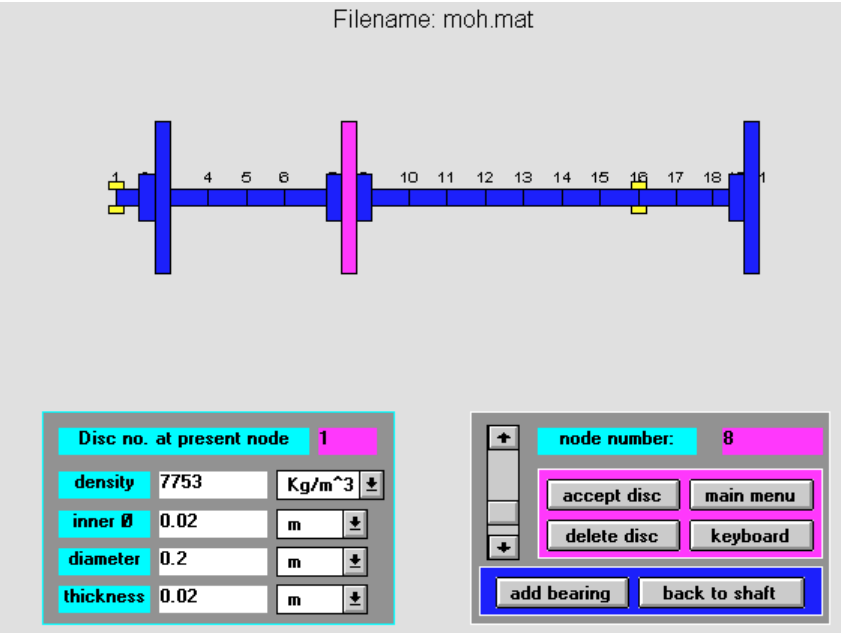


Figure C.2 Interface for disc elements

Shown here is the menu for defining the disc elements, which will be used to illustrate how all three interfaces work. The property boxes are inner diameter (non-zero in case the shaft goes through the disc), outer diameter, thickness and density. In case that there can be more than one disc attached

to a certain node, each disc has an index (shown above the property boxes) for identification.

It is possible to browse through the data of previously entered discs (by using the slider or changing the node number) while their properties are displayed in these property boxes. To make corrections or add a new disc, one simply changes the appropriate numbers in the edit-boxes and presses the Accept button. The colour of the node or element entry field changes from white to purple to acknowledge that the element is included in the matrices. Without pressing the Accept button, new data are not written into the matrices. Pressing the delete button throws the present element (if there is one selected, which is indicated by the color) out of the matrix.

When browsing through the nodes, the outer diameter of the shaft is the default value for the inner diameter of the disc. The data of the present disc is also kept when moving to the next node (unless there is a disc already present, in which case these properties are then shown) to allow for fast repetetive of the same discs at different nodes.

The model is drawn and upgraded in the figure window as the elements are entered or changed. Klicking the mouse button on any element in the plot of the model automatically switches to the right interface corresponding to that element (i.e. shaft, disc, or bearings) and displays the properties of this element in the interface. This makes the switching back and forth between interface and commiting changes to the model a whole lot faster.

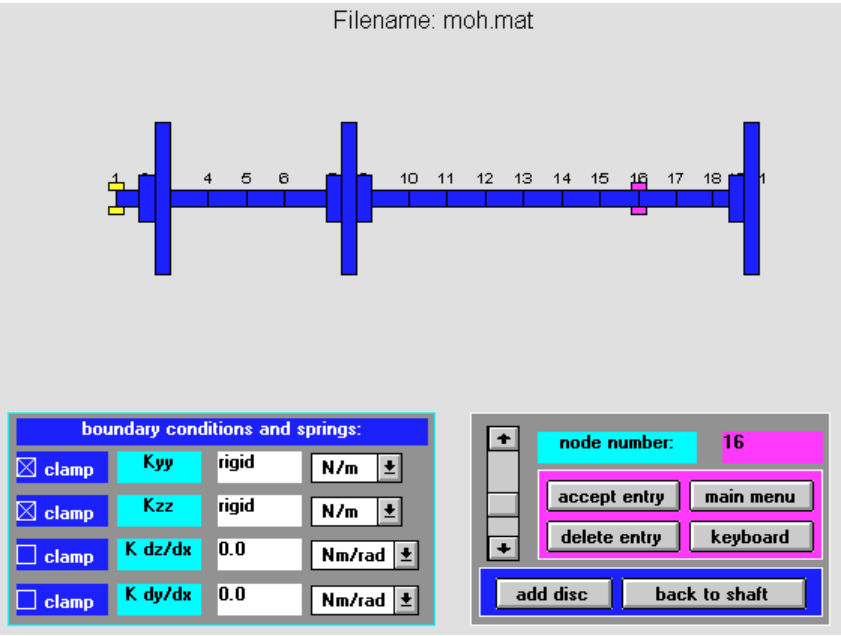


Figure C.3 Interface for boundary conditions and bearings

The add bearing button calls an interface for entering bearing properties and boundary conditions. Bearing properties and boundary conditions can be then entered for each direction of the general co-ordinates. Ticking the clamped box for a particular co-ordinate, e.g. q_1 at node 9, will result in cancelling the redundant row (no. 17=2·8+1, because there are 2 degrees of freedom per x and y-direction) in the global M, K, etc. matrices. The keyboard button halts the execution of the function and one can examine the present variables, commit changes manually or enter commands. The back to shaft button leads back to the shaft interface, where the elements of the rotor shaft can be entered and also has a menu bar (on top of the figure, not visible here) to allow for models to be cleared, saved and loaded, the amount of unbalance per element to be entered, and the program exited.

After finishing entering the model, one presses the main menu button in any of the interfaces, the windows are closed and an analysis menu appears. The available choices are: plotting the modeshapes and natural frequencies at a particular speed of rotation, plotting the Campbell diagram (including the first three engine orders), and plotting the unbalance response over a range of speeds. A special menu can be invoked in case non-axisymmetric elements are present.

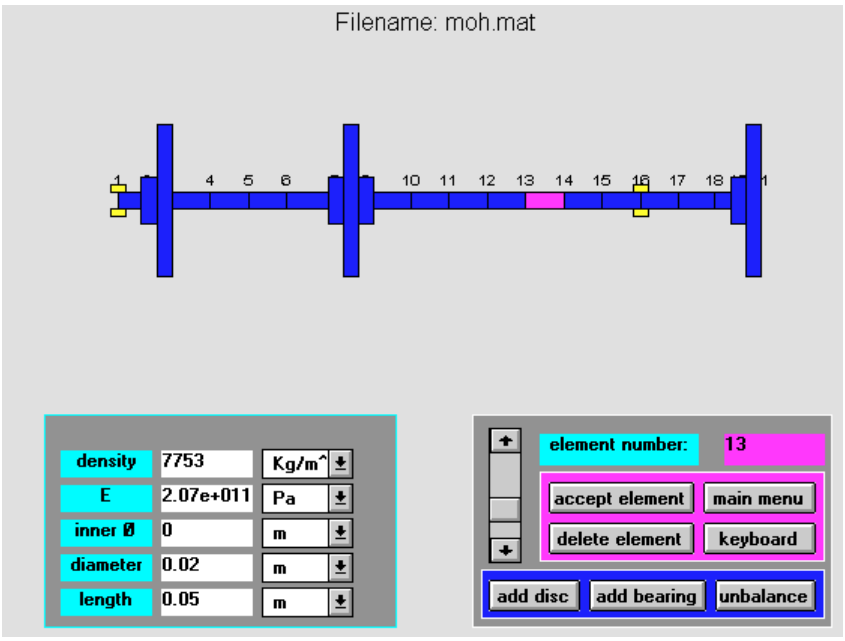


Figure C.4 Main interface for shaft elements

C.2 The Grahical User Interface of the Disc Animation Tool

The theoretical background is described in section 3.2. Here the layout of the interface is described.

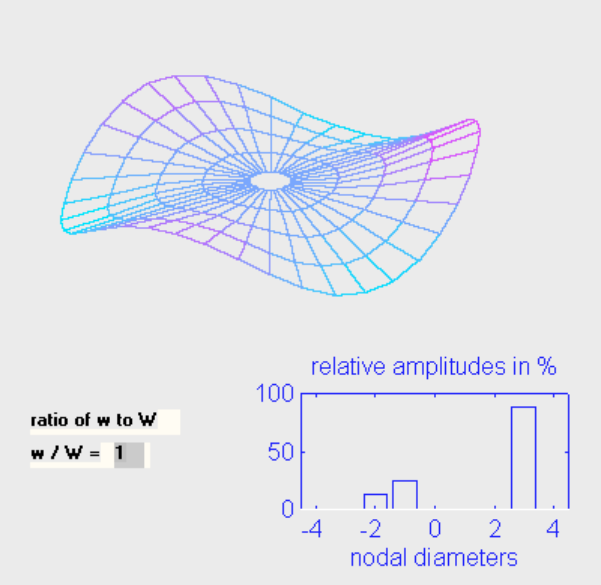


Figure C.5 Main animation window

The choice for the frame of reference for the animation, the size of the grid of the disc, the cutout feature (Figure C.8), mesh or surface plot, interpolation of colours, and colourmaps can be specified on the menubar which is located on top of the main animation window, but cannot be seen here. The colour can have two different meanings, depending on the type of vibration and viewing angle, sometimes one choice gives a much better understanding of the movement than the other. The two options are: colouring the z-deflection (shown here) or the body-fixed position on the disc.

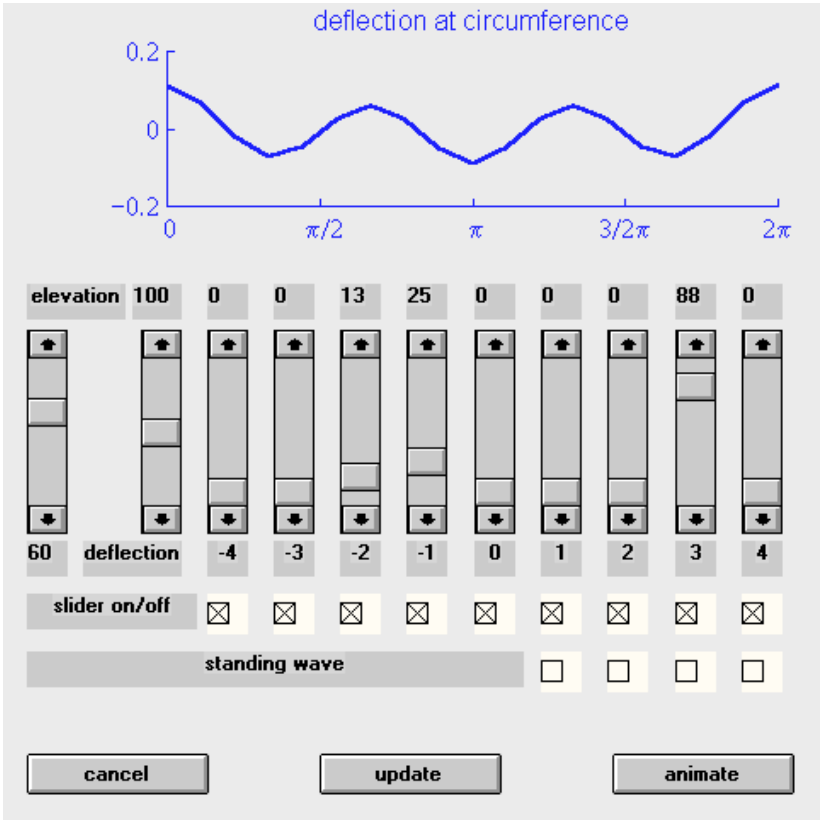


Figure C.6 Interface for specifying wave components

Under the sliders and editboxes for the magnitude of the travelling wave components are on/off and standing wave switches. They are there just for comfort to minimise repeated input. While experimenting with different combinations, it is not necessary to set the values for the not wanted components to zero, one can simply switch the contribution of these components to zero, the values in the sliders and editboxes do not change. As a standing wave is the superposition of a forward and backward travelling wave with equal number of nodal diameters, the interface has a 'standing wave switch' placed under the forward components. When this switch is on, forward and backward sliders are moved simultaneously, so a standing wave is added to the present setup. An adjustment of the absolute magnitude of deflection by changing the relative magnitude of every single component is not necessary, an additional control slider allows for this possibility.

Further options for the animation are the viewing angle (elevation), also adjustable by slider or editbox.

Since the drawing of the model can take some time, especially when in surface plot mode with interpolating colours, any changes made in the

interface for wave components will not affect the plot of the disc in the main animation window unless the update button has been pressed. This was found to be necessary to increase the speed with which the user could try out different set-ups before deciding which one to animate. The preview in the interface for wave componets shows the deflection of the disc at the circumference, any changes made to the vibration pattern are immediately reflected here, as redrawing this is quite fast. It uses the same number of radial lines as the main disc plot, any shortcommings on this part, that is not enough radial lines for the present vibration, or it simply looks too rough, can therefore be seen immediately.

The cancel button resets all parameters to the values of the last update. When the animate button is pressed, the dialog for the animation parameters, shown below, appears.

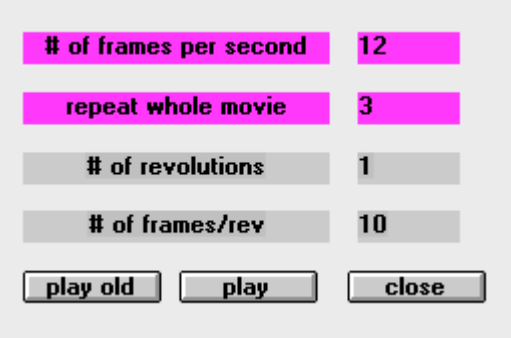


Figure C.7

The meaning of the animation parameters were described as well in the main text. Here it just should be pointed out that the upper two values are underlined with a different colour to indicate that these do not affect the generation of the movie frames. This means that after a movie has been generated, they can be changed without necessitating to start a new generation of movieframes. This allows the user to change the number of repetitions and the speed of the animation. As a new movie is generated, the old one can still be viewed when the play old button is pressed. This makes it possible to compare to movies, e.g. vibration in rotating and stationary frame of reference, without having to wait for the frames to be generated each time.

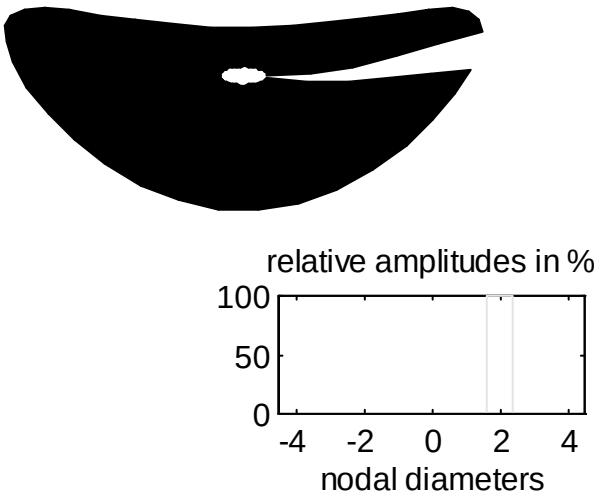


Figure C.8 surface plot with cutout

Appendix D

Some Code of the FE program and a short documentation

D.1 A short documentation of the FE program

The start

There are two main gateways into the program, depending on what is needed.

- **rotgui**
  To build a FE-model via the graphical user interface, type rotgui at the command line and a figure window with some buttons, sliders and editboxes will appear, they are pretty self explanatory.
- **rotmenu**
  If the geometry matrices are already existing, type rotmenu at the command line and use help button for further information.

The shortcuts

There are shortcuts to access functions directly, which is handy sometimes. If the mass, stiffness, ... matrices are present or if the results of further analysis are already saved as Matlab file, e.g. a Campbell diagram, one can do the calculations or retrieve the graphs by calling the corresponding function without having to go through all the menus again.

- **modeshap**
  starts < plot modeshape > function. Saved modeshapes can be loaded from here. The 'save modeshapes' option in this function only saves the modeshapes (global variables vv and dd for the eigenvalues) and not the M, K, G... or geometry matrices since this option is only directed towards retrieving the modeshapes without having to make the necessary calculations again.
- **campbell**
  starts < plot Campbell diagram > function. Saved Campbell diagrams can be loaded. Same as for the modeshap function, saved matrices include only those necessary to plot the Campbell diagram again (global F W, F is matrix of eigenvalues and W is speed of rotation vector) and do not include M, K, G
- **unbres**
  plots the unbalance response at a specified node over a range of speed of

rotaiton. Results are in global variables amplit_u and amplit_v for the response in Y and Z direction respectively.

- **rotasym**
  menu for non-axisymmetric analysis, i.e. equations of motion with time varying coefficients. The functions below all belong to this menu.
- **rothill**
  assembles the hyper-eigensystem using Hill's approach, the modeshapes are stored in vvv and the eigenvalues in ddd, both are global.
- **asymcam**
  plots the Campbell diagram by solving the hyper-eigensystem over a specified range of speed of rotation.
- **asymunb**
  plots the unbalance response by solving the hyper-eigensystem over a specified range of speed of rotation.
- **rotsort**
  sorts out the basis eigenvalues and vectors of the hyper-eigensystem using vvv and ddd as results from rothill to start with.
- **whirl**
  plots whirl orbit of FE-results or shows a demo if no input is given.

modshap, campbell and unbres prompt you what to do with the present data, if the output matrices such as modeshapes and frequencies already exist. You can plot them, or do new calculations (new speed of rotation or whatever) with the present M, K, G matrices. If you get tired of this dialog, type old=1; and the following default actions will be taken: modeshap, campbell, unbres check for existing M (mass), vv (modeshapes), F (frequencies, Campbell diagram), or amplit_u (unbalance response) matrices. If M alone is present, new calculations with the existing M, K, G matrices can be executed, the user is asked to specify new parameters, e.g. speed of rotation. If vv (or F, amplit_u) is present, they are shown. In order to invoke the load matrices prompt, type clear global M vv (or F, amplit_u) or use the option in the menu bar.

The format of the geometry matrices

In case alterations of the matrices are made during the program to take shortcuts or to specify things where an input routine has not yet been written (e.g. to have more than one element between the same two nodes, to input unbalance, or to delete a shaft element instead of starting from scratch again), it is necessary to know the format of the matrices. The matrices mentioned below (including the ones for

modeshapes, Campbell diagram) are all declared as global variables in all functions, so after a "highly unlikely" crash of the program the matrices are still accessible. Most files used by the program are script files so all the data is accesible after a crash and therefore providing a possibility to trick yourself out of the problem. When manually changing or entering any matrices via the keyboard buttons provided, check beforehand if the matrix wanted is present in the current workspace, e.g. size(xxx) and if the answer is [ ], type global xxx to get xxx into the current workspace, otherwise the input will be lost.

Notation:

- J denotes mass moments of inertia
- I denotes area moments of inertia
- X, Y, Z is the fixed frame of reference with X in shaft direction
- 1, 2, 3 are the rotating body co-ordinates with 1 in (deflected) shaft direction

The node vector:

nodes = [X(1) X(2) ... X(n)]

specifies the location on the X-axis where the index of the vector element is the node number.

The shaft element matrix:

shafts = [left_node_number outer_diameter length Young's_modulus density inner_diameter]

shafts2 or shafts3 pop up in the files for easy implementation of non-axisymmetric elements in the future. When assembling a model using the graphical interface, the elements are usually in order. However, this is not necessary, any element added to the end of the shaft matrix will be attached with its left side to the left_node_number specified in shafts(:,1).

It is also possible to have more than one element between two nodes. However, this has to done manually using the keyboard buttons provided in the interfaces to append all additional elements to the shafts matrix. Pressing the Accept button if there is an element already present (i.e. this lovely purple colour appears) does not add another element to the matrix but overwrites the present element at that node with the new parameters (e.g. length, diameter..) shown in the edit boxes. To add additional elements between the nodes would a nice recomendation for future work.

When deleting shaft elements the question arises whether elements such as discs or springs should be interpreted as fixed to the shaft, therefore move with the elements as some are kicked out, or fixed to their node number specified and not move with the shaft. When the corresponding row has been eliminated from the shafts matrix, node information stored in shafts(:,1) is updated as well as the nodes vector.

Note that the units of the parameters have to be consistent since there is no conversion routine implemented yet.

The discs matricies

disks = [node_number outer_diameter thickness density inner_diameter]

The discs are treated as perfectly rigid in this model. It is possible to attach more than one disc to a certain node. The number of discs present at one node is indicated by the disc index box in the interface. The order of discs within the disks matrix is arbitrary. Pressing the Accept button simply adds a disc with the given parameters to the system. If there is a disc already present (with the same index, i.e. again purple colour in the index box), pressing the Accept button results in overwriting the present data. The Delete disc button throws the disc out of the disks matrix.

It might be useful sometimes not to enter the geometric dimensions but rather mass and moments of inertia. In this case the format for disks2 can be used:

disks2 matrix

disks2 = [node-number mass J1 J2]

with J1 being moment of inertia in shaft direction (polar) and J2 consequently the moment of inertia in a direction perpendicular to the shaft (equatorial).

The bearings

springs = [node_number direction stiffness]

with the possible directions

1 for y-direction
2 for angular stiffness in the X-Y plane
3 for z-direction
4 for angular stiffness in the X-Z plane

The numbers for the directions coincide with the indices of the generalised co-ordinates.

The dampers and damping

dampers = [node_# direction c]

direction is the same as for springs, c the damping coefficient with the units force·time / length. There is no input routine yet, has to be done manually.

Proportional damping can easily be included, the matrix D (same size as M ...) just has to be created, e.g. D=alpha·M+beta·K;

The boundary conditions

BC = [node-number direction]

with the same values for direction as for springs. The information in the BC matrix will be used to eliminate obsolete rows from overall mass, stiffness, damping, and gyroscopic matrices as well as from the unbalance force vector, if present. Here again is the matrix not sorted with respect to node numbers.

The unbalance vector

unbalance(node_# , :) = [Y_direction Z_direction]

where Y_direction, Z_direction are columns specifying the amount of eccentricity in Y and Z direction respectively. The row indices of the columns coincide with the node number for which the eccentricity is entered.

The model for unbalance used here assumes a linear distribution of unbalance in an element. This distribution is specified by the amount of eccentricity given for the left and right nodes of the element. There is a mini-input routine in the grahical user interface which prompts you for input and does the rest (declaring variables as global and so on) automatically. However, many times the unbalance is only known in the form of a lumped unbalance mass and its excentricity. For these cases there is a different format avaiable:

unbal2(node_# , :) = [mass*Y_excentricity mass*Z_excentricity]

The matricies for non-axisymmetric elements are split into two types:

A non-axisymmetric part attatched to a node:

unsym = [node_# mass J1 J2 J3 ϕ]

ϕ is an angle (in radians) of body 2-axis to an arbitrary reference (the same one, of course, for all the non-symmetric elements) for the general case that the principal axes of the elements are not all in the same two planes. J1 is the moment of inertia in shaft direction and J2, J3 are the moments of inertia in the remaining principal axes of the non-axisymmetric element (without changes to the code, 2 and 3 are assumed to be perpendicular to each other).

Non-axisymmetric shaft elements:

shafts2 = [node_# length area I1 I2 I3 density Young's_modulus ϕ 2]

I1, I2, I3 are area moments of inertia, note the difference to unsym with J1, J2, J3.

area is the cross of the element. The last entry is the flag for the format.

or for rectangular elements:

shafts2 = [node_# length height width density Young's_modulus ϕ 1]

height is in 2-direction of the body co-ordinates and width in 3-direction.

If the non-axisymmetric elements in shafts2 are all of the same type, it is not necessary to specify the format. If there is no entry for angle ϕ , all angles are assumed to be zero.

elements in **disks2** and **unsym** do not appear in the drawing of the model due to lack of geometry information

Self documented help:

- all functions listed under shortcuts have self documented help, rotmenu has a more general overview
- help rotdata and rotadd to see the format of the symmetric element matrices
- help rotasp for the format of the non-axisymmetic element matrices
- help rotasymm, rotasymg, rotasymk for FE-matrices of one asymmetric element
- help rotmass, rotgyro, rotstiff for FE-matrices of one symmetric element
- help whirl to see a demo or calculated whirl orbit

Appendix E

Nomenclature

Symbols

A	cross-sectional area state space matrix forward whirl radius
B	backward whirl radius
d	diameter of shaft element or disc
d_0	diameter of gyration = $\sqrt{I / A}$
d_n	$\frac{n\pi d_0}{l}$
D	damping matrix
E	Young's modulus
e_y , e_z	eccentricity in Y and Z direction
f, F	force, force vector
G	gyroscopic matrix
i	$\sqrt{-1}$
I	area moments of inertia
Im	imaginary part
J	mass moments of inertia
K	stiffness matrix
l	length of shaft
L	length of rotor shaft element
m	mass of element or disc
M	mass matrix
N	dimension of the matrices M, K, G
n	harmonics, in terms like $n\Omega$ nodal diameter, wave number
Q	generalised forces
q_i	generalised co-ordinates, stationary frame of reference
r	co-ordinate vector, rotating frame of reference
R	rotation matrix
Re	real part
T	kinetic energy time period
T	co-ordinate transformation matrix = R^T

t	time
u	displacement, complex displacement = y +i·z rotation vector of two frames of references
U	cross-product operationally expressed in matrix form
v, w	deflection in Y and Z direction
y, z	deflections in Y,Z direction, stationary frame of reference
α	angle of rotation between XYZ and X'Y'Z' = Ωt
$-\beta$, γ	angular deflection or rotation in Y and Z directions
λ	whirl frequency, stationary (XYZ) frame of reference
λ'	whirl frequency, rotating (X'Y'Z') frame of reference
θ	angle in tangential (azimuthal) direction in disc
ρ	density
Ψ_i	shape functions
ψ	modeshapes
ξ	dimesionless parameter = x/L
ω	angular frequency
Ω	angular speed of rotation of the shaft

Subscripts

-2, -1, 0, 1, 2	matrices with $e^{-i2\Omega t}$, ... as coefficients
b	backward whirl direction
d	difference, moments of inertia
e	element matrix
ex	excitation frequency
f	forward whirl direction
i	1, .., 4 or 1, .. , 8 , usually for shapefunctions and generalised co-ordiantes
j	$-\infty,\ldots,\infty$
l	left node of element
n	nodal diameter, wavenumber general index
r	right node of element matrix in rotating frame of reference
rot	rotational energy, matrices in equation of motion in rotating co-ordinates
X, Y, Z	X, Y, Z directions
XYZ	stationary frame of reference
X'Y'Z'	rotating frame of reference, X'=X

ζ, η, ξ	body fixes co-ordinates

Superscripts

$', '', ''', ''''$	rotating frames of references
	$\frac{d}{dx}, \frac{d^2}{dx^2}, \ldots$
s	shaft
d	disc or bearings