Characterization of Torsional Vibration in a Mass Balancer System

Supervisor: Eng. José Dias Rodrigues
Author: António Miguel Figueiredo

July 2004
ABSTRACT

Studies of MBS behavior are a constant to achieve the optimization of the system. Vibration and Noise are, actually, one of the most important areas that concerns studies of comfort and silence because these are, more often, exigent requests of automobile buyers.

The use of spatial models is essential to analyze parameters as Natural Frequencies, Critical Speeds, Frequency Response, the importance of inertias, stiffness, and damping to improve the decreasing of torsion vibration.

When proceeding with the analysis of vibration characteristics, the first step is creating a model of the system considering inertias, springs and damping mechanisms. With that model, the creation of a mathematical model is crucial and the definition of differential equations is fundamental to proceed with the analysis of the system vibration behavior.

After proceeding with theory analysis, results must be understood and explained. Results obtained show that the MBS doesn’t reach any critical speed what is important to guarantee the torsion resistance under different operating regimes. Another important fact to check is the importance of tooth stiffness of the gears in the values of natural frequencies. If the tooth stiffness of the gears is increased, the last natural frequency will also increase reaching very high values when comparing the last natural frequency with the other five. As so, gears teeth are a very important factor that must be attended in the study of MBS parameters and should have more careful attention in the following project analysis.
ACKNOWLEDGEMENTS

This work was possible with the help of several people that the author would like to acknowledge:

First of all I would like to personally acknowledge Professor Jose Dias Rodrigues, my supervisor at FEUP (Faculdade de Engenharia Universidade do Porto), for his knowledge and several opinions to keep me in the right track to reach the goals of this project.

Professor Jorge Seabra for his co-operation in several projects subjects.

Eng. Ramiro, of CETRIB for his support in the 1st semester, helping me in some important parameters to develop the project.

Carlos Magalhães de Oliveira, Vice-Presidente do Conselho Pedagogico of FEUP, which guide me in several subjects of the course, and encourage me choosing this project.

Finally, and the most important, to all my friends, parents and sister that always support me giving me strength and encouragement to overcome the difficulties and reach the goals I was proposed to.
ACKNOWLEDGMENTS

Thank you for the dedication, support, and encouragement you have provided during the development of this project. Your contributions and guidance have been invaluable in ensuring the success of this endeavor.
# TABLE OF CONTENTS

1. **INTRODUCTION** ........................................................................................................... 9

2. **OBJECTIVES** ............................................................................................................... 11
   2.1 Project Goals........................................................................................................... 11
   2.2 Project Plan............................................................................................................ 11

3. **PROBLEM FORMULATION** ..................................................................................... 13
   3.1 Piston Operation Cycles........................................................................................ 14
   3.2 “Mass Balancing System” ..................................................................................... 18
      3.2.1 Historical Review........................................................................................... 18
      3.2.2 Operation Of MBS......................................................................................... 18
      3.2.3 Models of “Mass Balancing System” .............................................................. 20

4. **SPATIAL MODEL** ....................................................................................................... 25
   4.1 Lagrange Formalism.............................................................................................. 25
   4.2 Equations of the movement ................................................................................. 27

5. **MODAL MODEL** ......................................................................................................... 31
   5.1 Natural Frequencies and Mode Shapes .................................................................. 31
   5.2 Normalization of the modal vectors ..................................................................... 35
   5.3 Gearing Stiffness.................................................................................................... 36

6. **SYSTEM TIME RESPONSE** ..................................................................................... 37
   6.1 Modal Superposition ............................................................................................. 38
      6.1.1 Total System Response .................................................................................. 40
      6.1.2 Modal Contribution ....................................................................................... 41
   6.2 Transient Response ............................................................................................... 42

7. **FREQUENCY RESPONSE MODEL** ......................................................................... 45
   7.1 Bode Diagram – Receptance .................................................................................. 46
   7.2 Bode Diagram – Accelerance ............................................................................... 47
   7.3 FRF’s Superposition ............................................................................................. 48
   7.4 Modal Contribution – Receptance ....................................................................... 49

8. **CRITICAL SPEEDS** .................................................................................................. 51

9. **RESPONSE TO DIFFERENT EXCITATIONS** .......................................................... 53
   9.1 Impulse Train....................................................................................................... 53
      9.1.1 Time Response – 5000 rpm ........................................................................... 55
      9.1.2 Spectrum ....................................................................................................... 55
   9.2 Impulse Half-Sine Train ...................................................................................... 57
      9.2.1 Total Response ............................................................................................... 58
      9.2.2 Spectrum ....................................................................................................... 59
   9.3 Sine + Impulse Train (Backlash) ......................................................................... 60
      9.3.1 Time Response ............................................................................................... 61
      9.3.2 Spectrum ....................................................................................................... 61
   9.4 Acyclicism .............................................................................................................. 62
      9.4.1 Time Response ............................................................................................... 63
      9.4.2 Spectrum ....................................................................................................... 64

10. **CALCULUS SOFTWARE** ........................................................................................ 65

11. **CONCLUSIONS** ..................................................................................................... 71
    11.1 Final Conclusions ............................................................................................... 71
    11.2 Future Work ....................................................................................................... 71

12. **BIBLIOGRAPHY** ..................................................................................................... 73

António Figueiredo
ANNEXES ................................................................................................................. 75

ANNEX I - Tooth stiffness of the gears ................................................................. 77
ANNEX II – Time Response .................................................................................. 79
ANNEX III – Transient Response ......................................................................... 83
ANNEX IV – Frequency Response (Receptance) .................................................. 85
ANNEX V – Frequency Response (Acelerance) ..................................................... 87
ANNEX VI – Frequency Response (Modal Contribution-Receptance) ................. 89
ANNEX VII – Frequency Response (Modal Contribution -Acelerance) .............. 91
ANNEX VIII – Response to an impulse train ....................................................... 95
ANNEX IX – Response to a Sine Wave Impulse .................................................. 99
ANNEX X – Response to a Sine + Impulse Train (Backlash) .............................. 103
ANNEX XI – Acyclism (Total Response) .............................................................. 107
ABBREVIATIONS & ACRONYMS

MBS – Mass Balancer System
DOF – Degree of Freedom
TDC – Top Dead Center
BDC – Bottom Dead Center
K – Stiffness
G – Shear modulus
Ip – Polar moment of inertia
E – Young’s modulus
v – Poisson’s coefficient
J – Inertia
C – Damping coefficient
M(t) – Torque
ABBREVIATIONS & ACRONYMS

MICE - Mouse Breeding System
DOOR - Dealer of Record
TDC - Tool Data Center
ADC - Action Data Center
K - Stallions
E - Stalls
TR - Peter Trenholme
C - Cattleman Association
WO - Wayne Olds
1. INTRODUCTION

"Vibration is the variation with time of the magnitude of a quantity which is descriptive of the motion or position of a mechanical system, when the magnitude is alternately greater and smaller than some average value or reference." (Definition ISO 2041-1975)

Vibrations are the main cause of noise in automobile and mechanisms in general. Since long time ago, constructors are worried and looking forward to solve this problem, but luckily, the advance in technology made the automobiles more comfortable thanks to the great work developed by engineers to control the vibrations.

Vibrations and Noise that we had to tolerate when we were on a car were terrible and are now overcome. The comfort on driving is not only supported on a good suspension that filters the vibrations from the irregularities of the road. It's necessary, beyond that, no noise transmission from the engine, or road, to inside the vehicle.

This is a challenge that constructors and engineers have to face daily, using the last technology to fight the noise and vibrations in all of their car models.

Noise is not only a matter of comfort but also health. It's well known that noise is dangerous to the human ear and physical-psycho stability of the person.

There are different sources that contribute to noise but the most important are: mechanical components and aerodynamics. Gears, valves, bearings, breaks, and transmissions are some of the systems that enter in resonance with the engine noise and increase a lot the sound level in a mechanism.
Among the different solutions to decrease vibration in automobile engines, MBS is one of the most used systems to minimize these problems. Using twin shafts rotating at twice the crankshaft speed, and eccentric masses, this system is intended to counteract the second order harmonic forces present in the four-cylinder inline engine.

This work consists on the analysis of MBS torsion behavior and it was developed in two stages, both of them in "Vibrations Laboratory" of Mechanical Engineering Department in FEUP (Faculdade de Engenharia da Universidade do Porto).

The first stage consisted on a bibliographic research about engines vibration characteristics, different MBS constructions, methods for dynamic calculations and materials influence in the system behavior.

The second stage was the theory steps of vibration calculation in the components and graphics comprehension with the help of a graphical interface developed by the author, so that users could better understand what it's being analyzed.

Application of different loads at different speeds allows seeing how the system components behave and what's the consequence of impacts in each DOF (Degree of Freedom). Analysis suggest that noise levels are proportional with the impacts intensity if some acoustic parameters are ignored.
2. OBJECTIVES

2.1 Project Goals

The goals of this project are the creation of a model to analyze the torsion vibration of the "Mass Balancer System" (MBS) and characterization of the dynamic properties of the crankcase and the vibratory behavior of the MBS, as well as develop and implement in the Matlab environment some numerical tools to determine the dynamic characteristics of the system and his vibrational response both in time and frequency domains to different excitations.

2.2 Project Plan

Consequently, the direct intervenient and members of FEUP and Renault elaborated a project plan, to reach a realistic and profitable goal to the satisfaction of the needs intended. Of the planning makes part the analysis of the structure of the MBS, as well all their components and its contribution to the elimination of vibration and noise. The steps to reach the goals are:

1. Identification of the system components and their mechanical properties (weight, etc).
2. Modelization of the torsion vibration with base in the discrete system of inertia elements and elastic elements;
3. Formulation of the spatial model including the tooth stiffness of the gears;
4. Implementation of the spatial model in Matlab;
5. Determination of the Modal model in Matlab;
6. Determination of critical speeds (Diagram of Campbell);
7. Characterization and modelization of different excitement forces;
8. Determination of the Frequency Response Model in Matlab;
9. Carry out the study based on the mechanical simulation of the MBS under different operation regimes;
For this study a complete system was supplied by Renault (C.A.C.I.A) consisting of MBS, pinions, bearings and the body that supports them.

Because the project consists in the study of vibration, it was concluded that it would be important to develop a tool to allow the reconstitution of all the vibratory components of the MBS. The tool used was MatLab, a powerful software with plenty potentialities to the materialization of this system. Following that, the components were drawn to enable a better perception of the system in this case, and the modelization that allows to understand the propagation of the vibration in the engine and in the MBS.
3. PROBLEM FORMULATION

The MBS is a product to solve the existence of noise in the engine due to "acyclism". Looking at the great diversity of engines, in respect of the cylinder capacity, number and disposition of the cylinders, and motor type (gasoline or diesel), it is easily perceptible that all of them have different behaviors and each one of them will produce different loads, torques and power. However, this study is pointed to the MBS in a four-cylinder inline engine, which is the most widely used by automobiles manufacturers. Inside the engine it can be considered two different types of motion for the moving parts:

- Rotational motion which is the case of the crankshaft;

- Translation alternative motion, which is the case of the pistons, even so, there are parts that the motion isn't easy to define, as the connecting-rod, for an example.
3.1 Piston Operation Cycles

The operation of the pistons in a four-cylinder engine can be described:

- **Admission**: TDC $\rightarrow$ BDC: the mixture enters the cylinder. To increase the admitted mass, the admission valve opens a little before the start of this point and closes after it's finished.

- **Compression**: BDC $\rightarrow$ TDC: The mixture is compressed with the admission and expulsion valves closed. Before reaching TDC, the combustion starts and the pressure and temperature increase more quickly.

- **Power**: TDC $\rightarrow$ BDC: the gases (combustion products) create loads in the piston because of high temperatures and high pressure levels. The energy produced to the crankshaft it's about 5x the energy of compression. Before the piston reaches BDC, the expulsion valve opens, beginning the outlet of the cylinder gases.

- **Exhaustion**: BDC $\rightarrow$ TDC: The gases outlet the cylinder because of the different pressure between the inside cylinder and outlet conduct, and the movement of piston that expel them.

![Figure 1- Four-cylinder engine cycles](image)
The piston motion is a periodic movement with the frequency equal to the crankshaft frequency, and in the other hand the connecting rod moves at twice the frequency of the crankshaft, and it’s for this reason that MBS rotates at twice the speed of the crankshaft.

**Arrangement of the Cranks**

A - Front end for fitting pulley or vibration damper  
B - Web extension serves as balance weight  
C - Journal of crankshaft rotates in a main bearing  
D - Crankpin carries big-end of connecting rod linked to piston  
E - Flange to which flywheel is bolted  

*Figure 2 - Crankshaft*

It was found that the movement of the piston is not strictly sinusoidal, once a correction is added in the round trip movement of the piston caused by the inclination of the connecting rod that is changing each instant.

The teeth impact forces, shocks and backlash cause the MBS elements to vibrate and to generate rattling noises (low frequencies) and white noises (high frequency).

Gear teeth impacts as a source of pressure waves are related, among other things, with rattle noise levels. Some experimental results suggested that the intensity of the impacts corresponds with the noise level, if some extreme cases of acoustic resonances are ignored.

The major contributor to the MBS’s noise production is the mesh between its gears. This problem is enhanced by the fact that the two MBS shafts rotate at twice the speed of the crankshaft, producing something like between 3000 and 16000 individual contacts between teeth pairs per second in an engine speed ranging from 1000 to 5000 rpm.
The "backlash" is a phenomenon caused by changes in the torque applied to the gears which traduces itself when the meshing teeth are driven across the clearance distance between teeth (existent in all gear mesh contacts) and impact the tooth flank at the other end which cause noise.

The "acyclism" resultant from the engines operation was always a preoccupying factor because it is reflected in the noise level of the automobile. This factor is especially noticed when dealing with diesel engines whose "acyclism" is more abrupt. In a four-cylinder inline engine, "acyclism" is especially due to two types of forces.

The first type of force, also designated by "First Order Forces", or "Primary Forces", is a consequence of the inertia force generated by the vertical acceleration of the mass that constitutes the piston caused by the rotating crankpin motion along the line of stroke due to reciprocating motion of the piston assembly and are those forces of larger magnitude and whose contribution to the vibration is bigger than the others.

However, these forces are eliminated in group, because the engines studied are four-cylinder inline, a fact that allows the symmetry of the crankshaft and consequently an annulment of the forces produced by each one of the pistons, as we can see in the picture

\[ \sum_{i=1}^{4} F_i = 0 \quad \text{And} \quad \sum_{i=1}^{4} M_i = 0 \]

The sum of "First Order Forces" is zero, and the sum of moments are also null, due to counterbalancing action of piston 2 and 3 on the forces caused by pistons 1 and 4.
The second type of forces, known as "Second Order Forces", or "Secondary Forces" is the one that enable the resource of the MBS due the additional inertia force of the piston and of the connecting rod. These engine part (connecting rod) stays inclined twice, during one crankshaft rotation: one time during the up movement of the piston, and the second time during the down movement of the piston.

These forces are caused by the projected motion perpendicular to the line of stroke caused by the rotating motion of the connecting rod. As so, these loads can be understood, as due to the additional piston acceleration produced by the rotating crankpin increasing or decreasing the inclination of the connecting rod to the line of stroke.

\[
\sum_{i=1}^{4} F_i \neq 0 \quad \text{And} \quad \sum_{i=1}^{4} M_i = 0
\]

As it can be explained, the "Second Order Forces" increase and decrease magnitude at twice the frequency of the primary force, but their maximum values are only about \( 1/4 \) of the primary force. The primary forces are absorbed by the stiffness of the crankshaft, while the secondary forces are the cause of vibration and noise in the automobile. As so, MBS is intended to counteract the second order harmonic forces present in the engine.

Due to these types of efforts, the engines pass unpleasant vibration to the vehicles, and so the MBS function is to reduce these vibrations caused by engine cycles.
3.2 "Mass Balancer System"

3.2.1 Historical Review

Balancer shafts were invented by British Frederick Lanchester in early 20th Century. Mitsubishi put it into mass production in the 1976 Colt Celeste 2000, then Fiat group used it in its Lambda engine series, including the 1.6 liters Delta HF Turbo engine and Fiat Croma / Lancia Thema's 2 liter turbo engine. Meanwhile, Saab 9000 and Porsche 944 also introduced it into their powerful inline four-cylinder engine. All these carmakers obtained license from Mitsubishi.

Since the 80s, car engineers regard four-cylinder engines larger than 2 liters capacity had better to be equipped with a balancer system to damp the vibration. Although the strengthening of engine block, the use of hydraulic engine mount and lightweight pistons helped breaking such rule, the trend of pursuing refinement once again led to many engines larger than 2 liters to use balancer systems.

3.2.2 Operation Of MBS

The main idea of a MBS operation is easily comprehensible:

- The mounted gear in the crankshaft puts in action the gear of the MBS having this gear the reason of transmission of 1:2, allowing the system to turn moving at twice the speed of the crankshaft.

- This wheel is in the shaft together in two concentric gear wheels which allows the connection between the two shafts.

As so, the two shafts and respective eccentric masses will turn to double speed of the crankshaft but with opposite movements, a fact that is explained by the gearing between two wheels.
The engine "acyclism", referred to, previously, traduces itself in an increase of the noise, a fact that cannot pass unnoticed by the buyers of automobiles and that can discourage them from purchasing the vehicle. This phenomenon is more perceptible in diesel automobiles and engines with less number of cylinders. To allow noise reduce, several constructors fell back upon several techniques from the attempting of decrease noise in the engine adding the MBS that is expected to reduce the engine "acyclism" as much as possible.

It is worth pointing out that the MBS cannot balances the engine to 100%, because the system itself constitutes a noise source caused by the movement of gear wheels that constitute this system.

However, this system allows less engine "acyclism" and thus less uncomfortable noise.
(The MBS can reduce the noise by about 15 dB and the consumed torque of the engine does not cross the 5 %.)
3.2.3 Models of “Mass Balancer System”

The architecture is the main characteristic that distinguishes the several constructions of “Mass Balancer System”. Among the several architectures, there are two most commonly used: the "Lanchester" (most used) and the "Cassette".

The first (Lanchester) is designed for gasoline engines, where the rotation regimes verified are higher than in others engines. The objective is to create a resistant torque to the torque that causes the natural vibration of the engine. In this method, the MBS are located on each side of the engine but disposed in different horizontal plans.

![Figure 6 – Lanchester MBS](image)

Inside of this architecture, the set in motion form is the characteristic that distinguishes them:

- Set in motion of the two MBS through the jagged belt in one of the engine sides;

- Set in motion of the two MBS through the jagged belt on both sides, which allows a larger flexibility in the joint between the MBS and the engine.

- One of the MBS is put in motion by current and the other goes by the gearing with the pinion of the oil pump;

- One of the MBS is set in motion by an independent belt and the other goes by the gearing with the pinion of the oil pump;

- Set in motion of the two MBS just through the pinions, in that one of the system gears with the pinion mounted in the crankshaft and it also gears with the other MBS.

NOTE: (The first two are those that the manufacturers frequently appeal for).
The other architecture type "cassette" has been developed, and has been used by several automobile constructors. The "cassette" is usually fixed to the engine block, meeting inside the oil crankcase.

The disposition of the MBS in this architecture allows the set in motion in several ways:

- Put in motion of the primary shaft for a pinion mounted in the crankshaft, with twice the number of teeth as the pinion of the primary shaft being the cassette fixed to the engine block in the crankcase of the oil (the more common way).
- Set in motion of the primary shaft by current;
- Set in motion of the secondary shaft accomplished by the gearing of the pinions mounted in each one of the MBS, still being able to the same pinion of the primary shaft that receives the movement of the crankshaft to be the same that gears with the pinion of the secondary shaft (triplet system)

The support body can be:

- in "monoblock", whose main disadvantage is the difficulty of assembling the eccentric masses in the system;
- in two parts being the main disadvantage, the fact of existing difficulties in the machining and respective assembly and disassemble.

- "Open", being necessary the resource to a anti-emulsion plate to allow a better drainage of the oil;
- "Closed" (most used), that has as advantage the fact of being more rigid and of not hindering the drainage of the oil in the crankcase;
The MBS is, usually, composed by melted iron or steel obtained by casting or forging. On the other hand, the pinions are made of steel alloy (steel alloy chrome-manganese and chrome-molybdenum).

It is of extreme importance that the manufacturing process of the pinions is quite rigorous, and could rely upon the "shaving" or a good granding, allowing a good finish surface of the pinion, avoiding the possible gear noise. The noise of the MBS caused by the gearing will be less according to the quality of the teeth.

A vibration system is constituted by components to support potential energy (elastic elements), components to support kinetic energy (inertias) and components to dissipate energy (damping systems). As so, it is achieved a first spatial model considering all these facts, and obtained the next model with 5 DOF's.

![Spatial Model of MBS (5 DOF's)](image)

*Figure 11 - Spatial Model of MBS (5 DOF's)*
The following steps are the determination of material properties and values to the spatial model components.

Determination of the stiffness is based on the following expression: \( k = \frac{G \cdot I_p}{\ell} \), and considering: \( G = \frac{E}{2(1 + \nu)} \) the following properties of the material are obtained:

<table>
<thead>
<tr>
<th>Material Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>E (Pa)</td>
<td>2.10E+11</td>
</tr>
<tr>
<td>( \nu )</td>
<td>0.3</td>
</tr>
<tr>
<td>G (Pa)</td>
<td>8.08E+10</td>
</tr>
</tbody>
</table>

*Table 1 – Material Properties*

<table>
<thead>
<tr>
<th>Stiffness</th>
<th>Length [ m ]</th>
<th>K [ N/m ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Shaft</td>
<td>L1</td>
<td>0.0505</td>
</tr>
<tr>
<td></td>
<td>L2</td>
<td>0.0868</td>
</tr>
<tr>
<td>Secondary Shaft</td>
<td>L3</td>
<td>0.0522</td>
</tr>
<tr>
<td></td>
<td>L4</td>
<td>0.0881</td>
</tr>
</tbody>
</table>

*Table 2 – Shaft Stiffness*

<table>
<thead>
<tr>
<th>PINION</th>
<th>Primitive Diameter (mm)</th>
<th>Diameter Hole (mm)</th>
<th>External Diameter (mm)</th>
<th>Diameter Base (mm)</th>
<th>No. of Teeth</th>
<th>Width of the Tooth (mm)</th>
<th>Module</th>
<th>Weights (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entrance</td>
<td>82,509</td>
<td>21,95</td>
<td>84,42</td>
<td>76,729</td>
<td>49</td>
<td>12</td>
<td>1.55</td>
<td>0.378</td>
</tr>
<tr>
<td>Main</td>
<td>61,103</td>
<td>21,55</td>
<td>63,16</td>
<td>56,976</td>
<td>41</td>
<td>12</td>
<td>1.4</td>
<td>0.241</td>
</tr>
<tr>
<td>Secondary</td>
<td>61,103</td>
<td>21,55</td>
<td>63,16</td>
<td>56,976</td>
<td>41</td>
<td>12</td>
<td>1.4</td>
<td>0.295</td>
</tr>
</tbody>
</table>

*Table 3 – Gears Properties*
4. SPATIAL MODEL

4.1 Lagrange Formalism

The movement equations of a vibratory system can be obtained in a simple way in terms of the generalized coordinates by the use of Lagrange equations that are written as:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_j} \right) - \frac{\partial L}{\partial q_j} = Q_j$$

(1)

Where $L = T - V$ and so we have:

$$\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{q}_j} \right) - \frac{\partial T}{\partial q_j} + \frac{\partial V}{\partial q_j} = Q_j$$

(2)

The elastic potential energy is expressed as:

$$V_i = \frac{1}{2} f_i x_i$$

(3)

The kinetic energy associated to the mass $m_i$ is given by:

$$T_i = \frac{1}{2} m_i \dot{x}_i^2$$

(4)

If generalized coordinates $q_i$ $j$=1,...,n were used instead of the physical displacements $x_i$, the expressions (3) and (4) were obtained substituting $x$ by $q$.

When the generalized coordinates $q_i$ change $\delta q_i$, the energy is given by $\delta W_j$.

As so, the loads $Q_j$ corresponding to the generalized coordinate $q_j$:

$$Q_j = \frac{\delta W_j}{\delta q_j}$$

(5)

where $\delta W_j$ is the work when the generalized coordinates $q_j$ have some variation $\delta q_j$.

The calculus is made to a system with any number of DOF's, and it has been done to two spatial models: the first, with 5 DOF's and the last with 6 DOF's. It will be presented the calculus to the last spatial model considering 6 DOF's.
This model is different of the first one, because it has been included the tooth stiffness of the gears which increase the number of DOF’s. It will be presented the calculations to 6 DOF’s because it’s these the model that is more close to the real system, but the calculations are the same for both models.

DOF (Degree of Freedom) can be defined has being an independent coordinate necessary to achieve the component position in the system in all instants of time. It is considered the system as a discrete system with finite number of DOF’s (six) because is the enough to characterize the MBS model in a simple way with no massive calculations.

For this system in subject, and considering (3), (4) and (5):

\[
T = \frac{1}{2} \left( J_1 + J_7 \right) \dot{\theta}_1 + \frac{1}{2} J_2 \dot{\theta}_2 + \frac{1}{2} J_3 \theta_3 + \frac{1}{2} J_4 \dot{\theta}_4 + \frac{1}{2} J_5 \dot{\theta}_5 + \frac{1}{2} J_6 \dot{\theta}_6
\]

\[
V = \frac{1}{2} k_1 (\theta_2 - \theta_1)^2 + \frac{1}{2} k_2 (\theta_3 - \theta_2)^2 + \frac{1}{2} k_3 (\theta_4 - \theta_3)^2 + \frac{1}{2} k_4 (\theta_5 - \theta_4)^2 + \frac{1}{2} k_5 r^2 (\theta_6 - \theta_5)^2
\]

\[
\delta W = M_1 \delta \theta_1 + M_2 \delta \theta_2 + M_3 \delta \theta_3 + M_4 \delta \theta_4 + M_5 \delta \theta_5 + M_6 \delta \theta_6
\]
As the torque is applied only in the 1st DOF, it can be concluded that:
\[ M_2 = M_3 = M_4 = M_5 = M_6 = 0 \quad \Rightarrow \quad \delta W = M_1 \delta \theta_1 \]

Using the Lagrange equations (2) for each DOF:

\[
\begin{align*}
(J_1 + J_7) \ddot{\theta}_1 - k_1 (\theta_2 - \theta_1) - k_2 r^2 (\theta_6 - \theta_1) &= M_1 \\
J_2 \ddot{\theta}_2 + k_1 (\theta_2 - \theta_1) - k_2 (\theta_3 - \theta_2) &= 0 \\
J_3 \ddot{\theta}_3 + k_2 (\theta_3 - \theta_2) &= 0 \\
J_4 \ddot{\theta}_4 + k_3 (\theta_4 - \theta_3) &= 0 \\
J_5 \ddot{\theta}_5 - k_3 (\theta_4 - \theta_3) + k_4 (\theta_5 - \theta_6) &= 0 \\
J_6 \ddot{\theta}_6 - k_4 (\theta_5 - \theta_6) + k_5 r^2 (\theta_6 - \theta_1) &= 0
\end{align*}
\]  

(6)

### 4.2 Equations of the movement

The derived equations of motion can be rewritten in a matricial form as:

\[
[J]\{\ddot{\theta}(t)\} + [c]\{\dot{\theta}(t)\} + [k]\{\theta(t)\} = \{M(t)\}
\]

(7)

where \([J]\), \([c]\) and \([k]\) are, respectively, the mass, the damping and the stiffness matrices, while \(\{\theta(t)\}\), \(\{\dot{\theta}(t)\}\), \(\{\ddot{\theta}(t)\}\) and \(\{M(t)\}\) are, respectively, the displacement, velocity, acceleration and external force vectors of the system.
For this case, that has 6 DOF’s, and considering (6) and (7) we have the following inertia matrix:

\[
[J] = \begin{bmatrix}
J_1 + J_7 & 0 & 0 & 0 & 0 & 0 \\
0 & J_2 & 0 & 0 & 0 & 0 \\
0 & 0 & J_3 & 0 & 0 & 0 \\
0 & 0 & 0 & J_4 & 0 & 0 \\
0 & 0 & 0 & 0 & J_5 & 0 \\
0 & 0 & 0 & 0 & 0 & J_6
\end{bmatrix}
\]

Inertia matrix is a symmetric and principal diagonal matrix and the main characteristic of it, is:

\[m_{ij} = m_{ji}, \quad i \neq j \quad \implies \quad [J] = [J]^T\]

Considering (6) and (7), the stiffness matrix is given by:

\[
[k] = \begin{bmatrix}
(k_1 + k_5 \times r^2) & -k_1 & 0 & 0 & 0 & (-k_5 \times r^2) \\
-k_1 & (k_1 + k_2) & -k_2 & 0 & 0 & 0 \\
0 & -k_2 & k_3 & 0 & 0 & 0 \\
0 & 0 & 0 & k_3 & -k_3 & 0 \\
0 & 0 & 0 & -k_3 & (k_3 + k_4) & -k_4 \\
(-k_5 \times r^2) & 0 & 0 & 0 & -k_4 & (k_4 + k_5 \times r^2)
\end{bmatrix}
\]

In the analysis of the stiffness matrix it can be identified some characteristics of this matrix:

\[k_{ij} = k_{ji}, \quad i \neq j \quad \implies \quad [k] = [k]^T\]
Introducing the respective values in (6), the equation of motion (7) comes,

\[
\begin{pmatrix}
4.99E-4 & 0 & 0 & 0 & 0 & 0 \\
0 & 2.17E-4 & 0 & 0 & 0 & 0 \\
0 & 0 & 2.17E-4 & 0 & 0 & 0 \\
0 & 0 & 0 & 2.17E-4 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.28E-4 & 0 \\
\end{pmatrix}
\begin{pmatrix}
\theta_1 \\
\theta_2 \\
\theta_3 \\
\theta_4 \\
\theta_5 \\
\theta_6 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
2.971E06 & -36782.8 & 0 & 0 & 0 & -2.985E06 \\
-36782.8 & 58183 & -21400.2 & 0 & 0 & 0 \\
0 & -21400.2 & 21400.2 & 0 & 0 & 0 \\
0 & 0 & 0 & 35884.9 & -35884.9 & 0 \\
0 & 0 & 0 & -35884.9 & 56669.3 & 0 \\
-2.938E06 & 0 & 0 & 0 & -21084.4 & 2.956E06 \\
\end{pmatrix}
\begin{pmatrix}
\dot{q}_1 \\
\dot{q}_2 \\
\dot{q}_3 \\
\dot{q}_4 \\
\dot{q}_5 \\
\dot{q}_6 \\
\end{pmatrix}
\]

\[
M_1
\]
5. MODAL MODEL

5.1 Natural Frequencies and Mode Shapes

As can be seen in the bibliography, the natural vibration frequencies are obtained by solving the characteristic problem given by the following equation,

\[
\begin{bmatrix} K \end{bmatrix} - \omega^2 \begin{bmatrix} J \end{bmatrix} \{u\} = \{0\}
\]  

or by solving the generalized eigenproblem

\[
\begin{bmatrix} k \end{bmatrix} \{\phi\} = \omega^2 \begin{bmatrix} J \end{bmatrix} \{\phi\}
\]

which is more convenient for systems with several degrees of freedom and where the natural frequencies are given by the eigenvalues.

"Natural Frequency" can be defined as the oscillating frequency of the synchronous harmonic movement represented by the corresponding natural mode.

Thus, and solving the eigenvalue problem the following values are obtained for the natural frequencies:

<table>
<thead>
<tr>
<th>(\omega_n)</th>
<th>([\text{rad/s}])</th>
<th>([\text{Hz}])</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\omega_1)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(\omega_2)</td>
<td>6805,9</td>
<td>1083,2</td>
</tr>
<tr>
<td>(\omega_3)</td>
<td>10080,6</td>
<td>1604,4</td>
</tr>
<tr>
<td>(\omega_4)</td>
<td>18549,1</td>
<td>2952,2</td>
</tr>
<tr>
<td>(\omega_5)</td>
<td>19872,1</td>
<td>3162,7</td>
</tr>
<tr>
<td>(\omega_6)</td>
<td>170974,0</td>
<td>27211,4</td>
</tr>
</tbody>
</table>

*Table 4 – Natural Frequencies*

The natural forms of vibration (mode shapes) \(\{u\}_j\) \(j=1, \ldots, n\) are given by the eigenvectors.

"Mode shapes" are the spatial configuration of the system during the synchronous movement with the natural frequency \(\omega_j\).
Thus:

1\textsuperscript{st} Mode Shape

\[
\{u\}_1 = \begin{bmatrix} 25.8982 \\ 25.8982 \\ 25.8982 \\ 25.8982 \\ 25.8982 \\ 25.8982 \end{bmatrix} \Longleftrightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}
\]

\textit{Figure 13- 1\textsuperscript{st} Mode Shape}

The system has no connections to the exterior and so the system behaves as a rigid body. In this system, the movement is a combination of rigid body shapes and elastic shapes. The system has, at least, one natural frequency null and a degenerated mode shape with constant components as it can be seen in the figure 13. These system is known because of having a stiffness matrix \([k]\) that is singular (the elastic potential energy of deformation is a quadratic form semi-defined positive and the stiffness matrix is semi-defined positive)

2\textsuperscript{nd} Mode Shape

\[
\{u\}_2 = \begin{bmatrix} -4.420 \\ -20.550 \\ -38.713 \\ 41.745 \\ 29.966 \\ -4.183 \end{bmatrix}
\]

\textit{Figure 14- 2\textsuperscript{nd} Mode Shape}

The masses oscillate at frequency: \(\omega_{n2} = 6805.85\ \text{rad/s} = 1083.18\ \text{Hz}\). This mode shape has two "nodes": one between the 3\textsuperscript{rd} and the 4\textsuperscript{th} DOF and the other, between the 5\textsuperscript{th} and 6\textsuperscript{th} DOF. A node is a point where the torsion vibration is null. As so, the 3\textsuperscript{rd} and 4\textsuperscript{th} masses are in opposite phase and the 5\textsuperscript{th} and 6\textsuperscript{th} masses are also in opposite phase.
3rd Mode Shape

\[ \{u\}_3 = \begin{bmatrix}
-26.502 \\
-1.272 \\
43.392 \\
24.826 \\
9.459 \\
-26.360
\end{bmatrix} \]

![Figure 15 – 3rd Mode Shape](image)

The masses oscillate at frequency: \( \omega_{n3} = 10080.6 \text{ rad/s} = 1604.38 \text{ Hz} \).
The 3rd mode shape has two vibration nodes: one between the 2nd and 3rd DOF and the other between the 5th and 6th DOF. Thus, the 2nd and 3rd masses are in opposite phase, and the same happens between the 5th and 6th.

4th Mode Shape

\[ \{u\}_4 = \begin{bmatrix}
-11.117 \\
57.101 \\
-22.977 \\
14.962 \\
-16.398 \\
-11.322
\end{bmatrix} \]

![Figure 16 - 4th Mode Shape](image)

The masses oscillate at frequency: \( \omega_{n4} = 18549.1 \text{ rad/s} = 2952.18 \text{ Hz} \).
This mode shape has 4 nodes: one between the 1st and 2nd DOF, and the others, between the 2nd and the 3rd, between 3rd and 4th and between 4th and 5th DOF which makes these masses to be in opposite phase between those DOF’s.
**5th Mode Shape**

\[
\{u\}_5 = \begin{bmatrix}
-9.170 \\
17.239 \\
-5.746 \\
-36.874 \\
51.828 \\
-8.885
\end{bmatrix}
\]

![Figure 17- 5th Mode Shape](image)

The masses oscillate at frequency: \( \omega_{n5} = 19872.1 \text{ rad/s} = 3162.74 \text{ Hz} \).

The 5th mode shape has 3 nodes, as it can be seen in the figure. As verified before where exists a node that means the masses between those nodes are in opposite phase. In this case, the 1st and the 2nd masses are in opposite phase, as the 2nd and the 3rd. As so, it can be concluded that the 1st and the 3rd masses are in phase. The 5th and 6th masses are also in opposite phase.

**6th Mode Shape**

\[
\{u\}_6 = \begin{bmatrix}
-20.073 \\
0.112 \\
-0.0004 \\
0.0013 \\
-0.2510 \\
79.478
\end{bmatrix}
\]

![Figure 18- 6th Mode Shape](image)

In this mode shape there are five nodes. The figure is not very clear but the vector coordinates help to better identify the nodes.

The 1st and 2nd masses are in opposite phase and the same happens with the: 2nd and 3rd, the 3rd and 4th, the 4th and 5th, and the 5th and 6th.
5.2 Normalization of the modal vectors

The normalization for unitary modal masses consists on normalizing the modal vectors for the following condition to be verified:

$$\{\phi\}^T_j [m] \{\phi\}_j = 1 \quad j = 1, \ldots, n$$  \hspace{1cm} (10)

and the vectors \(\{\phi\}_j\) represent the modal vectors normalized for unitary modal masses.

With (7) and (9) we can obtain

$$\{\phi\}_j = \frac{1}{\sqrt{\{u\}_j^T [m] \{u\}_j}} \{u\}_j$$  \hspace{1cm} (11)

Thus:

\[
\{\phi\}_1 = \begin{bmatrix} 25.8982 \\ 25.8982 \\ 25.8982 \\ 25.8982 \end{bmatrix}, \quad \{\phi\}_2 = \begin{bmatrix} -4.420 \\ -20.550 \\ -38.713 \\ 41.745 \end{bmatrix}, \quad \{\phi\}_3 = \begin{bmatrix} -26.502 \\ -1.272 \\ 43.392 \\ 24.826 \end{bmatrix}
\]

\[
\{\phi\}_4 = \begin{bmatrix} -11.117 \\ 57.101 \\ -22.977 \\ 14.962 \end{bmatrix}, \quad \{\phi\}_5 = \begin{bmatrix} -9.170 \\ 17.239 \\ -5.746 \\ -36.874 \end{bmatrix}, \quad \{\phi\}_6 = \begin{bmatrix} -9.170 \\ 17.239 \\ -5.746 \\ -36.874 \end{bmatrix}
\]

The modal vectors can be grouped in a matrix where each column is one modal vector. This matrix is named: Modal Matrix.

Thus:

\[
\]
5.3 Gearing Stiffness

With the increasing of one DOF (5 to 6 DOF’s model) it is verified in the previous section that the first five natural frequencies have almost the same values, but the 6\textsuperscript{th}, and last one, is extremely high. If we think in the changes between the two spatial models it will be concluded that the only difference between them is the introducing of tooth stiffness of the gears in the model with 6 DOF’s.

Comparing the stiffness of the spatial model components and the tooth stiffness it is possible to check that the tooth stiffness is 1E4x higher than the others. As so, this fact traduces itself in higher values in stiffness matrix where it is being considered the tooth stiffness and as consequence it is achieved a high value to the last natural frequency (6\textsuperscript{th} Natural Frequency).

![Stiffness Versus Natural Frequency](image)

\textit{Figure 42- Stiffness versus Natural Frequency}

To have an idea how the spatial model would behave if tooth stiffness decreases, it is created a plot where it is possibly to identify that with the decreasing of tooth stiffness (to values near the other stiffness elements), the natural frequency also decreases to values more acceptable and near the other natural frequencies.

As this is a theory analyze and don’t have the real values of Renault MBS natural frequencies, we can only refer this as an important fact to have in mind and to check it out with practical tests.
6. SYSTEM TIME RESPONSE

The motion of the system is described by a system of differential equations in the generalized coordinates that has the form

\[
[J]\ddot{\vartheta}(t) + [c]\dot{\vartheta}(t) + [k]\vartheta(t) = M(t)
\]

with, in the present case, stiffness coupling.

In order to solve the system of differential equations to determine the response of the system to an external excitation, the modal analysis technique is used. Thus, using a coordinate transformation defined by the modal matrix \([\phi]\):

\[
\{\vartheta(t)\} = [\phi] \{\eta(t)\} = \left[ \{\phi_1\}, \ldots, \{\phi_n\} \right] \begin{bmatrix} \eta_1(t) \\ \vdots \\ \eta_n(t) \end{bmatrix}
\]

(12)

and premultiplying all terms by the transpose of the modal matrix, the differential equations in the generalized coordinates are projected in the modal basis, where they are independent and each one represents the equation of motion of one system with one degree of freedom (see bibliography for more details).

Thus, with the modal vector normalized for unitary modal masses, in the modal basis the equations of motion become,

\[
\ddot{\eta}_i(t) + 2\xi_i\omega_i \dot{\eta}_i(t) + \omega_i^2 \eta_i(t) = N_i(t) \quad i = 1, 2, \ldots, 6
\]

where \(\eta_i(t)\) represents the modal or natural coordinates and \(N_i(t)\) is the force projected in the modal basis.

These equations can easily be solved in the modal basis and after, the response of the system in the generalized coordinates can be achieved by using the transformation (12).
Considering that the solicitation $M(t)$ as an harmonic one, $M(t) = M_0 \cos(\omega t)$, the generalized forces in the modal basis are:

$$\{N(t)\} = [\phi]^T \{M(t)\}$$

For this type of excitation, the modal equations are written as:

$$\ddot{\eta}_i(t) + 2\xi_i\sigma_i \eta_i(t) + \sigma_i^2 \eta_i(t) = \chi_i \cos(\omega t)$$

and their solution is:

$$\eta_i(t) = \eta_i(\omega) \cdot \cos(\omega t - \phi_i)$$

with the amplitude $\eta_i(\omega)$ and the phase $\phi_i$ given by the following expressions:

$$\eta_i(\omega) = \frac{\chi_i}{\sigma_i^2 \sqrt{(1 - \beta_i^2)}^2 + (2\xi_i/\beta_i)^2}$$

and

$$\phi_i = \tan^{-1}\left(\frac{2\xi_i/\beta_i}{1 - \beta_i^2}\right)$$

With the response in the modal or natural coordinates, the response of the system can be expressed in the generalized coordinates as is explained in the next section.

### 6.1 Modal Superposition

According to the coordinate transformation (eq.12) and as the spatial model in use has 6 DOF’s, the response of the system is given by the modal superposition:

$$\begin{bmatrix}
\theta_1(t) \\
\theta_2(t) \\
\theta_3(t) \\
\theta_4(t) \\
\theta_5(t) \\
\theta_6(t)
\end{bmatrix} =
\begin{bmatrix}
\phi_{11} - \phi_{12} & \phi_{13} & \phi_{14} & \phi_{15} & \phi_{16} \\
\phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} & \phi_{25} & \phi_{26} \\
\phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} & \phi_{35} & \phi_{36} \\
\phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} & \phi_{45} & \phi_{46} \\
\phi_{51} & \phi_{52} & \phi_{53} & \phi_{54} & \phi_{55} & \phi_{56} \\
\phi_{61} & \phi_{62} & \phi_{63} & \phi_{64} & \phi_{65} & \phi_{66}
\end{bmatrix}
\times
\begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
= \begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
+ \begin{bmatrix}
\phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} & \phi_{15} & \phi_{16} \\
\phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} & \phi_{25} & \phi_{26} \\
\phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} & \phi_{35} & \phi_{36} \\
\phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} & \phi_{45} & \phi_{46} \\
\phi_{51} & \phi_{52} & \phi_{53} & \phi_{54} & \phi_{55} & \phi_{56} \\
\phi_{61} & \phi_{62} & \phi_{63} & \phi_{64} & \phi_{65} & \phi_{66}
\end{bmatrix}
\times
\begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
+ \begin{bmatrix}
\phi_{17} & \phi_{18} & \phi_{19} & \phi_{20} & \phi_{21} & \phi_{22} \\
\phi_{27} & \phi_{28} & \phi_{29} & \phi_{30} & \phi_{31} & \phi_{32} \\
\phi_{37} & \phi_{38} & \phi_{39} & \phi_{40} & \phi_{41} & \phi_{42} \\
\phi_{47} & \phi_{48} & \phi_{49} & \phi_{50} & \phi_{51} & \phi_{52} \\
\phi_{57} & \phi_{58} & \phi_{59} & \phi_{60} & \phi_{61} & \phi_{62} \\
\phi_{67} & \phi_{68} & \phi_{69} & \phi_{70} & \phi_{71} & \phi_{72}
\end{bmatrix}
\times
\begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
+ \begin{bmatrix}
\phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} & \phi_{15} & \phi_{16} \\
\phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} & \phi_{25} & \phi_{26} \\
\phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} & \phi_{35} & \phi_{36} \\
\phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} & \phi_{45} & \phi_{46} \\
\phi_{51} & \phi_{52} & \phi_{53} & \phi_{54} & \phi_{55} & \phi_{56} \\
\phi_{61} & \phi_{62} & \phi_{63} & \phi_{64} & \phi_{65} & \phi_{66}
\end{bmatrix}
\times
\begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
+ \begin{bmatrix}
\phi_{17} & \phi_{18} & \phi_{19} & \phi_{20} & \phi_{21} & \phi_{22} \\
\phi_{27} & \phi_{28} & \phi_{29} & \phi_{30} & \phi_{31} & \phi_{32} \\
\phi_{37} & \phi_{38} & \phi_{39} & \phi_{40} & \phi_{41} & \phi_{42} \\
\phi_{47} & \phi_{48} & \phi_{49} & \phi_{50} & \phi_{51} & \phi_{52} \\
\phi_{57} & \phi_{58} & \phi_{59} & \phi_{60} & \phi_{61} & \phi_{62} \\
\phi_{67} & \phi_{68} & \phi_{69} & \phi_{70} & \phi_{71} & \phi_{72}
\end{bmatrix}
\times
\begin{bmatrix}
\eta_1(t) \\
\eta_2(t) \\
\eta_3(t) \\
\eta_4(t) \\
\eta_5(t) \\
\eta_6(t)
\end{bmatrix}
$$

which denotes the contribution of each of the natural modes for the system response.
As was already said the system has a rigid body mode (the first one), which can be assimilated as degenerated mode, that doesn’t contribute to the vibrational behavior of the system response.

So, removing the first natural mode from the modal superposition, the system response can be expressed for the present system as:

\[
\begin{bmatrix}
\theta_1(t) \\
\theta_2(t) \\
\theta_3(t) \\
\theta_4(t) \\
\theta_5(t) \\
\theta_6(t)
\end{bmatrix} =
\begin{bmatrix}
\phi_{12} \\
\phi_{22} \\
\phi_{32} \\
\phi_{42} \\
\phi_{52} \\
\phi_{62}
\end{bmatrix} \eta_2(t) +
\begin{bmatrix}
\phi_{13} \\
\phi_{23} \\
\phi_{33} \\
\phi_{43} \\
\phi_{53} \\
\phi_{63}
\end{bmatrix} \eta_3(t) +
\begin{bmatrix}
\phi_{14} \\
\phi_{24} \\
\phi_{34} \\
\phi_{44} \\
\phi_{54} \\
\phi_{64}
\end{bmatrix} \eta_4(t) +
\begin{bmatrix}
\phi_{15} \\
\phi_{25} \\
\phi_{35} \\
\phi_{45} \\
\phi_{55} \\
\phi_{65}
\end{bmatrix} \eta_5(t) +
\begin{bmatrix}
\phi_{16} \\
\phi_{26} \\
\phi_{36} \\
\phi_{46} \\
\phi_{56} \\
\phi_{66}
\end{bmatrix} \eta_6(t)
\]

where the natural or modal coordinates \( \eta_i(t) \) represent the modal contribution for the total response.
6.1.1 Total System Response

![Graph showing system response over time](image)

Figure 19- Total Time System Response

This study is made considering as an excitement load, a torque that is 75% of the second natural frequency. $\omega_{exc} = 0.75 \times \omega_{n2}$

From the plot analysis it can be concluded that all DOF's will have a harmonic and periodic response with the frequency of MBS as it can be identified in the figure 19. The period of each response is the same as the period of the harmonic solicitation.
6.1.2 Modal Contribution

1\textsuperscript{st} DOF

![Graph showing modal contribution](image)

*Figure 19- Modal Contribution (1st DOF)*

In the plot analysis it's possible to identify the contribution of each natural mode to the system response in the DOF.

Analyzing the contribution of each natural mode to the system response at the 1\textsuperscript{st} DOF it's easily concluded that it's the third natural mode which contribution to the system response is higher than the others modes. The second natural mode also contributes to the system response but not so much significantly.

The 4\textsuperscript{th}, 5\textsuperscript{th} and 6\textsuperscript{th} natural modes contribution is widely insignificant and doesn't is suggested to analyze this modes in the 1\textsuperscript{st} DOF system response.
6.2 Transient Response

The transient response is the result of an impulsive load application. Impulsive load is a load of high magnitude that occurs during a small time interval $\Delta t$.

In this case it’s going to be made a study of the MBS response to a sine impulsive load, shown in the figure.

$$M(t) = \begin{cases} M_0 \sin(\omega t) & 0 \leq t \leq t_c \\ 0 & t > t_c \end{cases}$$

![Figure 20- Transient Response](image)

To a transient load, the characteristic period ($t_c$) is considered as the duration of the transient.

For this study, it was used: $t_c = 8E-4$, but it can be considered others values for the period of this transient.

From the books, the response in the instant $t$ is the sum of the responses of each one of the elementary impulses given by:

$$x(t) = \int_0^t f(\tau) h(t-\tau) d\tau = \frac{1}{m\sigma_d} \int_0^t f(\tau) e^{-\sigma_d(t-\tau)} \sin(\omega_d(t-\tau)) d\tau$$

Duhamel Integral.
The total response with initial conditions is given by:

\[ x(t) = e^{-\zeta \omega_d t} \left[ x_0 \cos(\omega_d t) + \frac{x_0 + \frac{1}{2} \omega_n x_0}{\omega_d} \sin(\omega_d t) \right] + \frac{1}{m \omega_d} \int_0^t f(\tau)e^{-\zeta \omega_n (t-\tau)} \sin(\omega_d (t-\tau)) d\tau \]

with \( \omega_d = \omega_n \sqrt{1 - \zeta^2} \), and \( x_0, x_0 \) represent the initial conditions of displacement and velocity.

For this load, considering the Duhamel Integral and the initial conditions null, it can be achieved the system response as:

- For \( 0 \leq t \leq t_c \):

  \[ \theta(t) = \frac{1}{J \omega_d} \int_0^t M(\tau) e^{-\zeta \omega_n (t-\tau)} \sin(\omega_d (t-\tau)) d\tau \]

  \[ \zeta = 0 \quad \Rightarrow \quad \theta(t) = \frac{M_0}{J \omega_n} \int_0^t \sin(\omega_n \tau) \sin(\omega_n (t-\tau)) d\tau \]

  \[ \theta(t) = \frac{M_0}{k} \left( \frac{1}{(\omega - \omega_n)^2} \left[ \sin(\omega t) - \frac{\omega}{\omega_n} \sin(\omega_n t) \right] \right) \]

  and

- For \( t > t_c \):

  \[ \theta(t) = \frac{1}{J \omega_d} \int_0^t f(\tau) e^{-\zeta \omega_n (t-\tau)} \sin(\omega_d (t-\tau)) d\tau \]

  \[ \zeta = 0 \quad \Rightarrow \quad \theta(t) = \frac{M_0}{J \omega_n} \int_0^t \sin(\omega_n \tau) \sin(\omega_n (t-\tau)) d\tau \]

  \[ \theta(t) = \frac{M_0}{k} \left( \frac{\omega_n}{\omega} \right) \left[ \sin(\omega_n t) + \sin(\omega_n (t - t_c)) \right] \left( \frac{\omega}{\omega_n} \right) \left[ 1 - \left( \frac{\omega}{\omega_n} \right)^2 \right] \]
Note that the above expressions for the transient response were established to use with the modal or natural coordinates.

Using Matlab, it can be easily obtained the response to the sine load for each DOF. The 1\textsuperscript{st} DOF is the chosen to explain the comprehension of the system behavior to an impulsive load.

\textbf{1\textsuperscript{st} DOF Response}

![Time Response to a Transient Load](image)

\textit{Figure 21- Time Response to a transient load (1\textsuperscript{st} DOF)}

In this plot it's identified two sections: the first is the left side of the plot and it can be easily understood to be correspondent to the time of the impulse application; the second is after the impulse load application and it shows that the system retakes the harmonic and periodic response stabilizing and achieving constant magnitude.
7. FREQUENCY RESPONSE MODEL

In this section it is going to be evaluated the system behavior in the frequency domain. Increasing and decreasing frequency, or velocity, is something that must be analyzed to understand how the system behaves in those speed conditions. As so, it will be shown two diagram models to better understand and analyze. They are:

- **Bode Diagram:** a diagram that relates frequency with magnitude and phase of system response. In this diagram it’s possible to identify natural frequencies, anti-resonances and what happens in magnitude ([dB]) and phase [°] in this conditions;

- **Modal Contribution:** this diagram allows to identify the contribution of each natural mode of vibration to the frequency response;

To proceed with the development of this diagrams it’s used appropriate calculations to frequency response that can be found in the bibliographic resources. It will be shown the diagrams of receptance and acelerance, in both diagrams (Bode and Modal Contribution). As the system have 6 DOF’s it will be chosen only one DOF (the first DOF because it’s where the load is applied) to explain the comprehension of this diagrams, and the diagrams for the other DOF’s are represented in ANNEX IV.
7.1 Bode Diagram – Receptance

![Figure 22- Bode Diagram (Receptance-1st DOF)](image)

The analysis of this plot helps to know that when the magnitude has a "pick point" (that means the magnitude takes a very high value), it has been reach a natural frequency of vibration. From the analyses of the plot it's possible to identify five natural frequencies (don't forget that MBS is a semi-defined system and so it has the 1st Natural frequency equal to zero).

As so it's concludes that the system has got five conditions of resonance given to the frequencies:

\[
\omega = \omega_{n2} = 1083.18 \text{ Hz}, \quad \omega = \omega_{n3} = 1604.38 \text{ Hz}, \quad \omega = \omega_{n4} = 2952.18 \text{ Hz}, \\
\omega = \omega_{n5} = 3162.74 \text{ Hz} \text{ and } \omega = \omega_{n6} = 27211.4 \text{ Hz}
\]
7.2 Bode Diagram – Acelerance

![Bode Diagram](image)

*Figure 23- Bode Diagram (Acelerance-1st DOF)*

Analyze of this plot is based in the same principles of what is said in the previous comment. The motion of the FRF (accelerance) is different from what is obtained to FRF receptance, but they still have something in common, and that is: Natural Frequencies. If superpose the two plots it’s possible to check that they have the “pick points” located at the same frequency (natural frequency).

Once more, at the natural frequency, the “pick point” is followed by a change of the phase, a characteristic that allows concluding that it has been reached a natural frequency of vibration.
7.3 FRF’s Superposition

Plotting the FRF’s of all DOF’s in the same plot, it’s possible to identify that they have something in common: **Natural Frequencies** values.

Independently of the DOF that is being analyzed, the natural frequency must be the same of the others DOF’s because natural frequency is a property of the system and independent of the DOF that is being studied.
7.4 Modal Contribution – Receptance

FRF’s are very useful to understand the behavior of the system in analyze in order to identify what really happens in any position when the frequency changes, or if you prefer the rotation speed changes.

As so, and to better understand where are located the frequencies or natural modes that most contribute to the FRF response, it is shown the figure above (figure 25). Looking at the picture it’s possible to see that when the system reaches a natural frequency, is the natural mode of that frequency that most contributes to the magnitude of the response.

That means for an example: when the system reaches a frequency equal to the 3\textsuperscript{rd} Natural frequency, it’s the 3\textsuperscript{rd} natural mode of vibration that contributes most to the frequency response in that point, being almost null the contribution of the others natural models.
8. CRITICAL SPEEDS

When a machine is built, constructors and engineers have to be extremely careful when they are projecting the system. As so, they must be sure that the system supports the loads without failure and with a good behavior.

In the case of MBS's, that is directly connected to the engine and rotating at twice the speed of it, that is one of main preoccupying factors engineers must face is that is making the system does not reach the critical speed or else the system will enter in resonance and could failure their components. Critical speed can be described as the speeds that are coincident with the natural frequency or the frequency from which the system reaches high magnitudes and the components can, or will, crash.

As it was expected, one of these project goals is to be sure that the MBS doesn’t reach any of the critical speeds under no conditions, in a normal four-cylinder engine. So, it will be plotted a linear increasing line of speed till the limits of the engine rotations (≈8000 rpm) and obtained the following figure:

![Figure 26- Critical Speeds (5 DOF's)](image)

António Figueiredo
Proceeding with the study of critical speeds in both of spatial models (5 and 6 DOF's), as it’s possible to see in the figures (Figure 26 and 27), in these conditions and with the critical speeds values calculated before, we can conclude that no critical speed is excited during the engine movement.

**NOTE:** the plot was done considering the speed of MBS that is twice of the engine frequency.
9. RESPONSE TO DIFFERENT EXCITATIONS

A periodic function is any function that repeats itself in time, that means, any function for which exists a time $T$ fixed, named "Period", in a way to verify $f(t) = f(t + nT)$ for all the $t$ values.

According to the theory developed by Fourier, all periodic functions with period $T$, can be represented by finite series.

Developing the study it will be applied different periodic functions. For all of them it has been used the Fourier series.

9.1 Impulse Train

![Impulse Train](image)

For this function, the Fourier series come as:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos \frac{n\pi t}{T} + \sum_{n=1}^{\infty} b_n \sin \frac{n\pi t}{T}$$

knowing that the frequency is: $\sigma = \frac{2\pi}{T}$

Figure 28- Impulse Train
Solving the determination of Fourier coefficients:

\[
a_n = \frac{2}{T} \int_{t_0}^{t_0 + \frac{T}{2}} A \cos \frac{n \pi t}{T} \, dt = \frac{2A}{n\pi} \left[ \sin \left( n\pi \frac{t_0}{T} \right) \right] \\
b_n = \frac{2}{T} \int_{t_0}^{t_0 + \frac{T}{2}} A \sin \frac{n \pi t}{T} \, dt = 0
\]

Thus,

\[
f(t) = \frac{A}{T} t_0 + \sum_{n=1}^{\infty} \frac{2A}{n \pi} \sin \left( n\pi \frac{t_0}{T} \right) \cos \frac{n \pi t}{T}
\]

Knowing that: \( \omega_{MIRS} = 2 \times \omega_{motor} \implies T = \frac{30}{n_{motor}} \)

In this case it has been considered: \( n_{motor} = 5000 \text{ rpm} \implies T = 0.006 \text{ sec} \)

and \( t_0 = \frac{T}{10} \)

Using the Modal Superposition calculation as it has been explained before, and with the help of Matlab, it is achieved the Response to this periodic impulse train to all the DOF's.

As it is explained before, it will only be explained the comprehension of the response to one DOF, continuing the analyses in the 1\text{st} DOF, because it's the DOF where the load is applied. (NOTE: the other DOF's are shown in the ANNEX VIII)
9.1.1 Time Response – 5000 rpm

![Time Response to impulse train at 5000 rpm (1st DOF)](image)

*Figure 29- Time Response to impulse train at 5000 rpm (1st DOF)*

The figure 29 gives the system response to an impulse train, and it's possible to identify his characteristic. When the impulse is applied the system response with a high magnitude value and when the impulse is over, the system tries to come back to his stability till the next impulse. The system response is, as the load applied, periodic but not harmonic. The period of the response is the same period of the impulse ($T = 0.006$ [sec]).

9.1.2 Spectrum

In the equipments project it is usual needed to know the maxim displacement of the system DOF's due to an excitant load. In some cases the project requests for a fixed limit to the maxim displacement of the system and so, that must be known. A way of identifying these is with the help of: Spectrum.

Response Spectrum is the representation of the maxim value of the system response in the DOF, function of natural frequency.
It will be now analyzed the spectrum response to the 1st DOF because it's where the load is located.

![Spectrum of impulse train and response (1st DOF)](image)

Each point in response Spectrum represents the maxim response of the system at this DOF with a given frequency. The first graphic corresponds to the excitement load, and the second graphic is the response spectrum in the 1st DOF to an impulse train. It can be concluded that when the frequency increases to values over 3500 Hz, the magnitude decreases and is considered null.
9.2 Impulse Half-Sine Train

Knowing that: $\omega_{MBS} = 2 \times \omega_{motor} \quad \Rightarrow \quad T = \frac{30}{n_{motor}}$

and $t_c = \frac{T}{11}$ (this value can be bigger or smaller depending the impulse duration wanted).

For this function, Fourier series can be expressed as:

$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos \frac{n\pi t}{T} + \sum_{n=1}^{\infty} b_n \sin \frac{n\pi t}{T}$$

knowing that the frequency is: $\sigma = \frac{2\pi}{T}$

The determination of Fourier coefficients is easy and so:

$$a_0 = \frac{2}{T} \int_{0}^{T} f(t) \, dt = \frac{2}{T} \left[ A \sin \left( \frac{\pi t}{t_c} \right) \right]_{0}^{T} = \frac{4A}{\pi T} t_c$$

$$a_n = \frac{2}{T} \int_{0}^{T} f(t) \cos (n\sigma t) \, dt = \frac{2}{T} \left[ A \sin (\sigma t) \cos (n\sigma t) \right]_{0}^{T} = -\frac{4T \times t_c}{\pi (T + 2n \times t_c)(-T + 2n \times t_c)} A \times \cos \left( \frac{\pi n t_c}{T} \right)^2$$

$$b_n = \frac{2}{T} \int_{0}^{T} f(t) \sin (n\sigma t) \, dt = \frac{2}{T} \left[ A \sin (\sigma t) \sin (n\sigma t) \right]_{0}^{T} = -\frac{4T \times t_c}{\pi (T + 2n \times t_c)(-T + 2n \times t_c)} A \times \sin \left( \frac{\pi n t_c}{T} \right) \times \cos \left( \frac{\pi n t_c}{T} \right)$$

António Figueiredo
Substituting coefficients in the Fourier series it is achieved the equation that gives the impulse. Using the Superposition Modal calculation as it was explained before, and with the help of Matlab, it is achieved the Response to this periodic impulse train to all the DOF's. To keep the consistence of the calculus, it will be considered the same engine speed as before: $n_{motor} = 5000 \, \text{rpm}$.

### 9.2.1 Total Response

![Graph: Total response to a half-sine impulse (1st DOF)](image)

*Figure 32- Total response to a half-sine impulse (1st DOF)*

The figure 32 shows the system response to a half-sine impulse train, and it's possible to conclude that it is periodic. When the impulse is applied the system responses with a high magnitude value and when the impulse is over, the system tries to come back to his stability till the next impulse. The period of the response is the same period of the impulse (As $n_{motor} = 5000 \, \text{rpm}$ $\implies T = 0.006 \, \text{[sec]}$)
9.2.2 Spectrum

![Graph showing Spectrum of a sine-half impulse (1st DOF)](image)

*Figure 33- Spectrum of a sine-half impulse (1st DOF)*

Each point in response Spectrum represents the maxim response of the system at this DOF with a given frequency. The first graphic corresponds to the excitement load, and the second graphic is the response spectrum in the 1st DOF to a half-sine wave impulse train. It is well noticed that the spectrum of excitation has the shape of a sine wave and while that sine wave exists, the system response has almost the same shape. The response spectrum takes the higher value when the excitant response reaches the higher magnitude.

It can be concluded that when the frequency increases to values over 3500 Hz the magnitude of spectrum response takes null values.
9.3 Sine + Impulse Train (Backlash)

![Backlash Impulse](image)

One problem that concerns the manufacturers of MBS, is the consequence of Backlash (described before). To traduce backlash as an impulse train, it can be considered the load as shown in the figure 34. Backlash can be traduced as a sine wave with an impulse before the maxim magnitude value.

Considering that \( n_{\text{motor}} = 5000 \text{ rpm} \) and \( \omega_{\text{ABS}} = 2 \times \omega_{\text{motor}} \), thus:

\[
T = \frac{30}{n_{\text{motor}}} = 0.006
\]

The magnitude of impulse considered is unitary.

It is obtained the graphics of system response in all DOF's, but it will only be shown and explained what happens in 1st DOF.
9.3.1 Time Response

![Graph showing time response to backlash (1st DOF)](image)

*Figure 35- Time Response to backlash (1st DOF)*

The figure 35 shows the system time response to backlash, and it’s possible to conclude that it is periodic. The period of the response is the same period of the impulse (As $n_{motor} = 5000 \text{ rpm } \rightarrow T = 0.006 [\text{sec}]$)

9.3.2 Spectrum

![Graph showing spectrum of backlash and response (1st DOF)](image)

*Figure 36- Spectrum of backlash and response (1st DOF)*
9.4 Acyclism

\[ f(t) = A \sin(\omega t) \quad 0 < t < T \]

One of the main problems of engines crankshaft is the "acyclism", a consequence of firing pulses. "Acyclism" can be traduced as an impulse train as it's shown at the picture above (figure 37), with the characteristic that the frequency is twice the engine speed, because firing pulses show up twice per engine revolution in a four-cylinder engine.

Thus:

\[ \omega_{acyclism} (motor) = 2 \times \omega_{motor} \]

But the MBS rotates at twice the engine speed and so:

\[ \omega_{MBS} = 2 \times \omega_{motor} \quad \rightarrow \quad \omega_{acyclism} (MBS) = 4 \times \omega_{motor} \]

Considering all these factors, it can be deduced the period this impulse train:

\[ T = \frac{15}{n_{motor}} \]

As so, it is studied the behavior of MBS in each DOF, but as before, it will be only shown the analyses of the 1st DOF while the others DOF's responses are in ANNEX XI. The analyses is made considering the engine working firstly at 1000 rpm and then at 5000 rpm.
9.4.1 Time Response

Due to what is said about acyclism, it seems important to study the MBS behavior under different operation regimes of engine, which means, under different speed. As so, it is obtained the two figures above (figure 38 and 39). As it can be concluded the magnitudes are not the same in both graphics. We can see that when the engine is working at 1000 rpm, the highest magnitude is given to higher values than when working at 5000 rpm. As so, it can be concluded that the acyclism is more noticed when working at low speed, and when increasing speed the acyclism doesn't reach so high values.

As a conclusion of this section, it can be said that the main problem of acyclism is when working at low speeds, a fact that must be carefully study.
9.4.2 Spectrum

Each point in response Spectrum represents the maxim response of the system at this DOF with a given frequency. The first graphic corresponds to the excitement load (acyclism), and the second graphic is the response spectrum in the 1st DOF to a half-sine wave impulse train, both to 1000 and 5000 rpm. It is well noticed that the spectrum of excitation has the same shape independently of the speed, but not in the same scale.

When the engine is working at 1000 rpm, the higher value of response is given to a frequency near the 1500 Hz and at the others frequencies of the spectrum, magnitude is null.

At 5000 rpm we also achieved a higher magnitude in response near the 1500 Hz but smaller than the verified when working at 1000 rpm.

These conclusions also helped to conclude that the main problem of acyclism is verified when working at low regimes of speed.
10. CALCULUS SOFTWARE

To improve the possibility to users have access to the results obtained in this project, the author developed an interface in MatLab, and that is going to be described with the help of the some images selected from the interface.

The main window of the program shows the project title, the university and author that develop this project and this interface. In this window, users can choose if they want to see the calculations results or if they prefer to see graphical analyses. Besides that, users have access to the project report where they can see the details and the descriptions of calculations steps and graphics analyzes.

![Menu Principal](image)

*Figure 43- Main Menu*

If users choose the "Calculations" icon, a new window will appear and give them the access to the main results obtained.

António Figueiredo
Natural Frequencies, inertia, stiffness and modal matrix are the choices and results that users could easily obtain. In these menus users will have access to full information about this, and if the concepts aren’t well known, they have the possibility to obtain a simple description of each concept meaning. If they want a more complex description it’s always possible to consult the report to better understand the concepts referred in this project.

![Figure 44- Calculus Menu](image)

**Figure 44- Calculus Menu**

![Figure 45- Calculus Menu (Natural Frequencies and Mode Shapes)](image)

**Figure 45- Calculus Menu (Natural Frequencies and Mode Shapes)**
If users prefer to see the graphical results a window will appear as the figure:

Here it's possible to choose and see all important graphics to vibration analyzes. Among them we can refer: response to an impulse load, frequency response and critical speeds. All the graphics are followed with the spatial model used and where users could easily identify the DOF that is being analyzed having the possibility to understand the system behavior under different regimes.

Figure 46- Graphic Menu

Figure 47- Spatial Model
As an example of it is being described, it has been chosen the “impulse menu” icon, which will open a new window:

Here users can choose to see the impulse wave whose response they want to analyze. Among them it’s the impulse waves that have been studied in theory and which results have been described in this report.

![Figure 48- Impulses Menu Option](image)

After choosing the impulse and going to the next window, users can see the spatial model in study and another window gives the possibility to choose the engine speed because this analyses is influenced by MBS speed, that has we know is twice the engine speed.

![Figure 49- Impulses Menu](image)
After choosing the option and DOF to study, users have access to all the graphics necessary to proceed with the analysis of MBS components behavior under different impulses waves and speed regimes.

![Figure 50- Impulses Menu Response](image)

This menu is prepared to give comments to the graphics obtained but this option is going to be a further development to be made.

Among other options, users can use "ZOOM" icon, if necessary, to better analyze the graphic and to have a better perception of values in each position of the plot.

António Figueiredo
As an example, the next menu can be analyzed:

In this menu (Modal Contribution FRF’s analyzes), users can identify the variation of magnitude response and each modal contribution with frequency, but the left side of the plot is difficult to analyze with the actual resolution. To increase resolution and improve the analysis, users are able to choose the “zoom” button and select the area of the plot to zoom. This option will allow a better identification of plots characteristics and points that are not perceptible at the normal resolution used by the plot commands.

These are some of the profits that the interface gives to users allowing them to have a better analyze of the system. This interface can be found in a CD-ROM existent in this report.
11. CONCLUSION

11.1 Final Conclusions

As final conclusions it is possible to say that the MBS doesn’t reach any torsion critical speed. In the other hand, it’s important to contrast the values obtained with practical tests executed by Renault with measurements and resistant tests.

Another important conclusion is the fact of the tooth stiffness of the gears to be extremely high and that increases the last natural frequency values to a higher level.

Also a numerical tool to determine the dynamical characteristics of the system as well as his response, both in the time and frequency domains for several types of excitation was developed and implemented in a user-friendly package.

After the analysis of all calculations it’s possible to conclude that the project goals were reached. The knowledge about systems vibration study and comprehension has been developed and it was very profit to the author.

11.2 Future Work

In the following, there are some tasks that can be developed, namely:

- Introduction of real values of engine characteristics (as power, torque, etc) which allows obtaining real magnitudes and values in this analyzes;

- Study of the tooth stiffness of the gears attempting to achieve the most real value as possible.
12. BIBLIOGRAPHY


THE BIBLIOGRAPHY


ANNEXES
ANNEX I - Tooth stiffness of the gears

Supposing the tooth as shown in the figure and the load $P$, resultant from the projection of the load produced by the gearing between the teeth of two gears we have the following procedure:

For the first stage of calculation we adopted an approximation of what happens in the real tooth. As so, we used a beam fixed at one side and free at the other side as we show in the figure:

$$h = h_{mod} = \frac{h_1 + h_2}{2}$$
$$L = a + b$$

Next step is the calculation of the deflection under the load according to the equation:

$$\frac{d^2 y}{dx^2} = \frac{Mf(x)}{EI}, \text{ where } y \text{ is the deflection.}$$

Knowing that the bending moment is given by the equation: $Mf(x) = -P(x-a)$ and proceeding to the calculations we achieved: $y(a) = -\frac{Pa^3}{3EI}$ (at the point where the load is applied)

As we know, $k = abs\left(\frac{P}{y}\right)$, we get the tooth stiffness of the gear as: $k = \frac{3EI}{a^3}$
A. Examine the tooth and note any cavities or fractures.

B. Take an X-ray to identify any internal damage.

C. Prepare the tooth for bonding by cleaning and etching.

D. Apply a bonding agent and cure with a light source.

E. Place the restoration material and cure.

F. Polish the restoration for a smooth finish.

G. Review the patient's bite and adjust if necessary.

H. Complete the appointment with a final check of the restoration.
ANNEX II – Time Response

1st DOF

2nd DOF

3rd DOF

4th DOF

5th DOF

6th DOF
Modal Contribution

2nd DOF

3rd DOF

4th DOF

- Resposta do Sistema
- 2º Modo Natural
- 3º Modo Natural
- 4º Modo Natural
- 5º Modo Natural
- 6º Modo Natural
ANNEX III – Transient Response

3\textsuperscript{rd} DOF

4\textsuperscript{th} DOF

5\textsuperscript{th} DOF

6\textsuperscript{th} DOF

António Figueiredo

83/108
ANNEX IV – Frequency Response (Receptance)

2nd DOF

3rd DOF

4th DOF
"Characterization of the Vibration and noise of a mass balance system of engine"

5th DOF

6th DOF
ANNEX V – Frequency Response (Acelerance)

2\textsuperscript{nd} DOF

3\textsuperscript{rd} DOF

4\textsuperscript{th} DOF

António Figueiredo
PROJECT FIM-DE-CURSO
"Characterization of the Vibration and noise of a mass balance system of engine"

5th DOF
FRFs (acelerancia) $\theta_{31}$

6th DOF
FRFs (acelerancia) $\theta_{61}$
ANNEX VI – Frequency Response (Modal Contribution-Receptance)
"Characterization of the Vibration and noise of a mass balance system of engine"

5th DOF

6th DOF
ANNEX VII – Frequency Response (Modal Contribution - Acelerance)

2nd DOF

3rd DOF

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

4th DOF

5th DOF
$6^{th}$ DOF

Response in Frequency/Modal Contribution (Acceleration) $\theta_{6}$

- $A_{61}$
- $c1^a$
- $c2^a$
- $c3^a$
- $c4^a$
- $c5^a$
- $c6^a$
ANNEX VIII – Response to an impulse train

Total response

2\textsuperscript{nd} DOF

3\textsuperscript{rd} DOF

4\textsuperscript{th} DOF

5\textsuperscript{th} DOF

6\textsuperscript{th} DOF

Response to an Impulse Train

Response to an Impulse Train

Response to an Impulse Train

Response to an Impulse Train

Response to an Impulse Train
Spectrum

2\textsuperscript{nd DOF}

Spectrum Response to an Impulse Train

3\textsuperscript{rd DOF}

Spectrum Response to an Impulse Train

4\textsuperscript{th DOF}

Spectrum Response to an Impulse Train
ANNEX IX – Response to a Sine Wave Impulse

Total response

2\textsuperscript{nd} DOF

Response to a Sin wave Impulse

3\textsuperscript{th} DOF

Response to a Sin wave Impulse
4th DOF
Response to a Sin wave Impulse

5th DOF
Response to a Sin wave Impulse

6th DOF
Response to a Sin wave Impulse
Spectrum

2\textsuperscript{nd} DOF

Spectrum Response to a Sin Wave Impulse

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig2ndDOF.png}
\end{figure}

3\textsuperscript{rd} DOF

Spectrum Response to a Sin Wave Impulse

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig3rdDOF.png}
\end{figure}

4\textsuperscript{th} DOF

Spectrum Response to a Sin Wave Impulse

\begin{figure}
\centering
\includegraphics[width=\textwidth]{fig4thDOF.png}
\end{figure}
ANNEX X – Response to a Sine + Impulse Train (Backlash)

**Total response**

\[ \text{2}^{\text{nd}} \text{ DOF} \]

Response to Sin+Impulse Train

\[ \text{3}^{\text{rd}} \text{ DOF} \]

Response to a Sin+Impulse Wave Train
4th DOF
Response to a Sin+Impulse Wave Train

5th DOF
Response to a Sin+Impulse Wave Train

6th DOF
Response to a Sin+Impulse Wave Train
Spectrum

2\textsuperscript{nd} DOF

Spectrum Response to a Sin+Impulse Wave Train

3\textsuperscript{rd} DOF

Spectrum Response to a Sin+Impulse Wave Train

4\textsuperscript{th} DOF

Spectrum Response to a Sin+Impulse Wave Train
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

**5th DOF**

**Spectrum Response to a Sin+Impulse Wave Train**

**6th DOF**

**Spectrum Response to a Sin+Impulse Wave Train**

Freq [Hz]
ANNEX XI – Acyclism (Total Response)

2\textsuperscript{nd} DOF

$\text{Response to the acyclism (1000 rpm)}$

$\text{Response to the acyclism (5000 rpm)}$

3\textsuperscript{rd} DOF

$\text{Response to the acyclism (1000 rpm)}$

$\text{Response to the acyclism (5000 rpm)}$

4\textsuperscript{th} DOF

$\text{Response to the acyclism (1000 rpm)}$

$\text{Response to the acyclism (5000 rpm)}$
"Characterization of the Vibration and noise of a mass balance system of engine"
ANNEX XII – MATLAB CODE

Characterization of Torsional Vibration in a Mass Balancer System

Supervisor:
Eng. José Dias Rodrigues

Author:
António Miguel Figueiredo

July 2004
ANNEX XII – Matlab Code

Dados

% DADOS DA ARVORE DE EQUILIBRAMENTO DE MOTOR

% warning off MATLAB:divideByZero;
% RODAS DENTADAS

% RODA 1
dprimt1=82.509/1000;
dfuro1=21.95/1000;
dext1=84.42/1000;
dbase1=76.729/1000;
nrdentes1=49;
largdentes1=12/1000;
mod1=1.55;
massarodal=0.378;
volum1=pi*((dprimt1/2)*2-(dfuro1/2)*2)*largdentes1;
massavolum1=massarodal/volum1;
Ip1=pi*((dprimt1/2)*4-(dfuro1/2)*4)/2;
J1=Ip1*massavolum1*largdentes1;
% J1=3.456E-4;

% RODA 6
dprimt6=61.103/1000;
dfuro6=21.55/1000;
dext6=63.16/1000;
dbase6=56.976/1000;
nrdentes6=41;
largdentes6=12/1000;
mod6=1.4;
massarodal6=0.241;
volum6=pi*((dprimt6/2)*2-(dfuro6/2)*2)*largdentes6;
massavolum6=massarodal6/volum6;
Ip6=pi*((dprimt6/2)*4-(dfuro6/2)*4)/2;
J6=Ip6*massavolum6*largdentes6;
% J6=1.278E-4;

% RODA 7
dprimt7=61.103/1000;
dfuro7=21.55/1000;
dext7=63.16/1000;
dbase7=56.976/1000;
nrdentes7=41;
largdentes7=12/1000;
mod7=1.4;
massaroda7=0.295;
volum7=pi*((dprimt7/2)*2-(dfuro7/2)*2)*largdentes7;
massavolum7=massaroda7/volum7;
Ip7=pi*((dprimt7/2)*4-(dfuro7/2)*4)/2;
J7=Ip7*massavolum7*largdentes7;
% J7=1.565E-4;
% MASSAS

% MASSA 1
valorlibra=0.454;
massavolum1=7.1/1E-3;
massavolum2=1.29/1E-3;
% SEM CONSIDERAR MASSA 2
massalibra=[0.852,0;0.853,0;0.85,0;0.854,0];

% massalibra=[0.852,0.067;0.853,0.068;0.85,0.067;0.854,0.068];
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

% SEM CONSIDERAR MASSA 2
%SEM CONSIDERAR MASSA 2
% diamextmassa=[0.059,0;0.0591,0;0.0591,0;0.0591,0];
% diamextmassa=[0.059,0.0587;0.0591,0.0587;0.0591,0.0581;0.0591,0.0581];
% diamextmassa=[0.023,0;0.0231,0;0.0231,0;0.0231,0];
% diamintmassa=[0.023,0.051;0.0231,0.051;0.0231,0.051;0.0231,0.051];
% SEM CONSIDERAR MASSA 2
% compmassa=[0.0521,0;0.0522,0;0.0522,0;0.0522,0];
% compmassa=[0.0521,0.0525;0.0522,0.052;0.0522,0.0515;0.0522,0.0515];
%SEM CONSIDERAR MASSA 2
espmassa=[0.018,0;0.018,0;0.018,0;0.018,0];
% espmassa=[0.018,0.00385;0.018,0.00385;0.018,0.00355;0.018,0.00355];

nrpesos=4;
for i=1:4
    for jc=1:2
        massagk(i,jc)=valorlibra*massalibra(i,jc);
        Ipma massa(i,jc)=pi*((diamextmassa(i,jc)/2)^3)-(diamintmassa(i,jc)/2)^3)/4;
        if jc==1
            volumassa(i,jc)=massagk(i,jc)/massavolum1;
            J(i,jc)=Ipma massa(i,jc)*massavolum1*compmassa(i,jc);
        end
        if jc==2
            volumassa(i,jc)=massagk(i,jc)/massavolum2;
            J(i,jc)=Ipma massa(i,jc)*massavolum2*compmassa(i,jc);
        end
    end
    Jtotal(i)=J(i,1)+J(i,2);
end
J2=Jtotal(1);
J3=Jtotal(2);
J4=Jtotal(3);
J5=Jtotal(4);
J2=Jtotal(1);
J3=Jtotal(2);
J4=Jtotal(3);
J5=Jtotal(4);
% J2=2.319E-4;
% J3=2.336E-4;
% J4=2.319E-4;
% J5=2.319E-4;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%VEIO PRINCIPAL
massaveiol=0.763;
diamveiol=0.022;
comprveiol=0.2754;
volumaveiol=pi*((diamveiol/2)^3)*comprveiol;
massavolumveiol=massaveiol/volumaveiol;
Ipveiol=pi*(diamveiol^3)/32;
Jveiol=Ipveiol*massavolumveiol*comprveiol;
% VEIO SECUNDARIO
massaveio2=0.763;
diamveio2=0.022;
comprveio2=0.2754;
volumaveio2=pi*((diamveio2/2)^3)*comprveio2;
massavolumveio2=massaveio2/volumveio2;
Ipveio2=pi*(diamveio2^3)/32;
Jveio2=Ipveio2*massavolumveio2*comprveio2;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

2/86 July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'

ModYoung=210E9;
coefpoisson=0.3;
ModG=ModYoung/(2*(1+coefpoisson));
comprimK1=5.05*10/1000;
comprimK2=8.68*10/1000;
comprimK3=5.22*10/1000;
comprimK4=8.81*10/1000;
k1=ModG*Ipveio1/comprimK1;
k2=ModG*Ipveio1/comprimK2;
k3=ModG*Ipveio2/comprimK3;
k4=ModG*Ipveio2/comprimK4;
% k1=3.6783E4;
% k2=2.14E4;
% k3=3.5584E4;
% k4=2.1084E4;

alturamedia=(3.1/1000+1.4/1000)/2;
bdente=dext7-dprimit7;
adente=dprimit7-dbase7;
compdente=dext7-dbase7;
MomInercia7=largd7*alturamedia^3/12;
% flecha=1/(ModYoung*MomInercia7))*(bdente^3/3+bdente*compdente^2-
bdente^2*compdente-compdente^3/3);
flecha=abs(-(adente^3)/(3*ModYoung*MomInercia7));
% flecha=abs((-1/3*ModYoung*MomInercia7)^3))
flecha=abs((1/(ModYoung*MomInercia7)) *((bdente^3-compdente^3)/3+bdente*compdente^2-
(bdente^2)*compdente));
k5=1/flecha;
% k5=1.02E4;
raio=dprimit7/2;

nrGL=6;
% Mt=M0*cos(w*t);
M0=100;

# 5 GRAUS DE LIBERDADE #

if nrGL==5
% MATRIZ DE MASSA
MM=[J1+J6+J7,0,0,0,0;J2,0,0,0,0,J3,0,0,0,0,J4,0,0,0,0,0;J5];

% MATRIZ DE RIGIDEZ
k=[k1+k3,-k1,0,-k3,0;-k1,k1+k2,-k2,0,0,0,-k2,k2,0,0,-k3,0,0,k3+k4,-k4,0,0,0,-k4,k4];
warning off MATLAB:nearlySingularMatrix;
F=[M0;0;0;0];
end

# 6 GRAUS DE LIBERDADE #

if nrGL==6
% MATRIZ DE MASSA
MM=[J1+J7,0,0,0,0,0;J2,0,0,0,0,0;J3,0,0,0,0,0,J4,0,0,0,0,0,0,0,0,0,0,0,J5];

% MATRIZ DE RIGIDEZ
k=[k1+k5*raio,-k1,0,0,-k5*raio;-k1,k1+k2,-k2,0,0,0,-k2,k2,0,0,0,0,0,0,k3,-k3,0,0,0,-k3,k3+k4,-k4,-k5*raio,0,0,0,-k4,k4+k5*raio];
warning off MATLAB:nearlySingularMatrix;
% Mt=M0*cos(w*t);
F=[M0;0;0;0];

António Figueiredo
3/86
"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
end
GL=size(MM,1);
%FRECUENCIAS E FORMAS NATURAIS
[uu,w]=eig(k,MM);
[w,n]=sort(sqrt(diag(w)));
% disp('Frequencias naturais [Hz]');
% fprintf('%6.3g \n',wn/2/pi);
for i=1:GL
   u(:,i)=uu(:,ip(i));
end
%NORMALIZAÇAO DOS VECTORES MODAIS PARA MASSAS MODAIS UNITARIAS
for i=1:GL
   fnor=transpose(u(:,i))*MM*u(:,i);
   fi(:,i)=(1/sqrt(fnor))*u(:,i); %MATRIZ MODAL
end
for i=2:1:GL
   fi2(:,i-1)=fi(:,i);
   wn2(i-1)=wn(i); %MATRIZ MODAL SEM O PRIMEIRO MODO NATURAL
end

Fourier Serie

%COEFICIENTES DA SERIE DE FOURIER
clear all;clc;
t=sym('t');
T=sym('T');
tc=sym('tc');
w=sym('w');
A=sym('A');
n=sym('n');
A0=2/T*int(A*sin(pi/T*t),t,0,T)
An=2/T*int(A*sin(pi/T*t)*cos(n*2*pi/T*x),t,0,T)
Bn=2/T*int(A*sin(pi/T*t)*sin(n*2*pi/T*t),t,0,T)

FRF Acelerancia

PROJECTO DE FIM DE CURSO
%CARACTERIZAÇÃO DA VIBRAÇÃO E RUIDO DE UMA ARVORE DE EQUILIBRAMGEM DE MOTOR
FRF ACELERANCIA

run('Dados');
warning('off');
% mc=0.05*10E-4*k;
mk=k*(1+j*0.0005);
%Resposta em frequencia
w=linspace(0,wn(GL)*1.2,2048);
teta=zeros(GL,length(w));
for i=1:length(w)
   teta(:,i)=(-w(i)^2*MM+mk)
   teta(:,i)=(-w(i)^2*MM+j*w(i)*mc+k);%F
end
tetamag=20*log10(abs(teta));
tetaf=angle(teta)*180/pi;
for i=1:GL
   tetamag(i,:)=20*log10(abs(-teta(i,:).*w.^2));
   tetafA(i,:)=angle(-teta(i,:).*w.^2)*180/pi;
end
for i=1:GL
   index=[num2str(i),',1');
   subplot(1,4,1,i);
   plot(w/2/pi,tetamagA(i,:));
````
'Characterization of the Vibration and noise of a mass balance system of engine'

set(gca, 'XTickLabel', '');
ylabel('Mag. [dB]');
title(['Resposta em Frequência (aceleração) \theta_{i\text{'},index,'}]');
subplot(3,1,3);
plot(w/2/pi,tetaA(i,:));
set(gca, 'YLim', [-180 180]);
set(gca, 'YTick', [-180 0 180]);
xlabel('f [Hz]'); ylabel('fase [°]');
backg= uicontrol ('style', 'pushbutton', 'units', 'normal', 'position', [0.91 0.5 0.090 0.095], ...
 'string', 'Menu', 'callback', 'close all,Graficos');
end

FRF Receptancia

run('Dados');
warning('off');
mc=0.05*10E-4*k;
mk=k*(1+j*0.0005);

%Resposta em frequencia
w=linspace(0,wn(GL)*1.2,2048);
teta=zeros(GL,length(w));
for i=1:length(w)
    teta(:,i)=(-w(i)^2.MM+mk)
    % teta(:,i)=(-w(i)^2.MM+J.*w(i)^4*mc+k)
end
tetamag=20*log10(abs(teta));
teta=angle(teta)*180/pi;
for i=1:GL
    tetamagA(i,:)=(20*log10(abs(-teta(i,:).*w.^2)))
    tetaf(i,:)=angle(-teta(i,:).*w.^2)*180/pi;
end
for i=1:GL
    index=([num2str(i),'1'])
    figure(i)
    subplot(1.4,1,1);
    plot(w/2/pi,tetamag(i,:));
    set(gca, 'XTickLabel', '');
ylabel('Mag. [dB]');
title(['Resposta em Frequência (receptancia) \theta_{i\text{'},index,'}]');
subplot(3,1,3);
plot(w/2/pi,teta(i,:));
set(gca, 'YLim', [-180 180]);
set(gca, 'YTick', [-180 0 180]);
xlabel('f [Hz]'); ylabel('fase [°]');
end

Menu Formas Naturais

function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @M_Formas_Naturais_OpeningFcn, ...
    'gui_OutputFcn', @M_Formas_Naturais_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin & isstr(varargin{1})

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% End initialization code - DO NOT EDIT

clc;

% --- Executes just before M_Fomas_Naturais is made visible.
function M_Fomas_Naturais_OpeningFcn(hObject, eventdata, handles, varargin)
    % This function has no input args, see OutputFcn.
    % hObject handle to figure
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % varargin command line arguments to M_Fomas_Naturais (see VARARGIN)
    % Choose default command line output for M_Fomas_Naturais
    handles.output = hObject;
    % Update handles structure
    guidata(hObject, handles);
    % UIWAIT makes M_Fomas_Naturais wait for user response (see UIRESUME)
    uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Fomas_Naturais_OutputFcn(hObject, eventdata, handles)
    % varargout cell array for returning output args (see VARARGOUT)
    % hObject handle to figure
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Get default command line output from handles structure
    varargout{1} = handles.output;

    scnsiz = get(0, 'ScreenSize');
    posi = [scnsiz(3)/12,6,scnsiz(3)/9,scnsiz(4)/20];
    set(hObject, 'Position',posi, 'Visible','on');

    set(hObject, 'Units', 'pixels');
    handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
    info = imfinfo('fundo.jpg'); % Determine the size of the image file
    position = get(hObject, 'Position');
    axes(handles.background);
    image(handles.banner);
    set(handles.background, ...
        'Visible', 'off', ...
        'Units', 'pixels', ...
        'Position', [0 0 15*info.Width 15*info.Height]);
    % --- Executes during object creation, after setting all properties.
    function popupmenu_CreateFcn(hObject, eventdata, handles)
    % hObject handle to popupmenu (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles empty - handles not created until after all CreateFcns called
    % Hint: popupmenu controls usually have a white background on Windows.
    % See ISPC and COMPUTER.
    run('Dados');
    if ispc
        set(hObject, 'BackgroundColor', 'white');
    else
        set(hObject, 'BackgroundColor', get(0, 'defaultUicontrolBackgroundColor'));
    end
    if nrGL==5
        set(hObject, 'String', {'1ª Forma Natural', '2ª Forma Natural', '3ª Forma Natural', '4ª Forma Natural', '5ª Forma Natural'});
    end
    if nrGL==6
        set(hObject, 'String', {'1ª Forma Natural', '2ª Forma Natural', '3ª Forma Natural', '4ª Forma Natural', '5ª Forma Natural', '6ª Forma Natural'});
    end
```

6/86 July 2004
% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
% contents{get(hObject,'Value')} returns selected item from popupmenu1

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

set(handles.text1,'Visible','off');
axes(handles.axes1);
cla;
run('Dados');
popupSel_index = get(handles.popupmenu1, 'Value');
switch popupSel_index
    case 1
        iGL=1;
        set(handles.text1, 'String','Forma Natural de Corpo Rígido em que todos os valores do vetor modal são iguais');
    case 2
        iGL=2;
        if nrNGL==5
            set(handles.text1, 'String','Forma Natural que apresenta 1 nóde vibração localizado entre o 3º e 4º GL');
        end
        if nrNGL==6
            set(handles.text1, 'String','Forma Natural que apresenta 2 nós naturais de vibração localizados entre o 3º e 4º GL e entre o 5º e o 6º GL');
        end
    case 3
        iGL=3;
        if nrNGL==5
            set(handles.text1, 'String','Forma Natural que apresenta 3 nós naturais de vibração localizados no 3º GL e na periferia do 4ºGL');
        end
        if nrNGL==6
            set(handles.text1, 'String','Forma Natural que apresenta 2 nós naturais de vibração');
        end
    case 4
        iGL=4;
        if nrNGL==5
            set(handles.text1, 'String','Forma Natural que apresenta 2 nós naturais de vibração localizados entre o 2º e 3º GL e entre o 4º e 5º GL');
        end
        if nrNGL==6
            set(handles.text1, 'String','Forma Natural que apresenta 4 nós naturais de vibração');
        end
    case 5
        iGL=5;
        if nrNGL==5
            set(handles.text1, 'String','Forma Natural que apresenta 4 nós naturais de vibração');
        end
        if nrNGL==6
            set(handles.text1, 'String','Forma Natural que apresenta 4 nós naturais de vibração');
        end
    case 6

António Figueiredo
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

iGL=6;
set(handles.text1, 'String', ' ....... ');
end
x=1:1:1GL;
elixo=zeros(GL,GL);
elixo(:,iGL)=0;
valormax=abs(max(fi(:,iGL)));
valormin=min(fi(:,iGL));
plot(x,fi(:,iGL),x,elixo(:,iGL),'k','LineWidth',1.5);hold on;grid off;
xlabel('Degree of Freedom');
set(gca,'xtick',1:1:GL)
title('Forma Natural');

set(handles.togglebutton1, 'Visible', 'on');
set(handles.pushbutton4, 'Visible', 'on');

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text1,'Visible', 'on'); % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text1,'Visible', 'off'); % toggle button is not pressed
end

M_G_Espectro

function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT

 gui_Singleton = 1;
 gui_State = struct('gui_Name',      mfilename,    ...
                     'gui_Singleton', gui_Singleton,...
                     'gui_OpeningFcn', @M_G_Espectro_OpeningFcn, ...
                     'gui_OutputFcn', @M_G_Espectro_OutputFcn, ...
                     'gui_LayoutFcn', [], ...        
                     'gui_Callback', []);

 if nargout && isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
 end

 if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
 else
    gui_mainfcn(gui_State, varargin{:});
 end
% End initialization code - DO NOT EDIT

clc;

% --- Executes just before M_G_Espectro is made visible.
function M_G_Espectro_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OpeningFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to M_G_Espectro (see VARARGIN)

% Choose default command line output for M_G_Espectro
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
% UIWAIT makes M_G_Espectro wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_G_Espectro_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
% set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner)
set(handles.background, ...
'Visible', 'off', ...
'Units', 'pixels', ...
'Position', [0 0 10*info.Width 10*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcsns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
set(hObject, 'String', {'1º Grau de Liberdade', '2º Grau de Liberdade', '3º Grau de Liberdade', '4º Grau de Liberdade'});

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu contents as cell array
% contents = get(hObject,'Value') returns selected item from popupmenu

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

popup_sel_index = get(handles.popupmenu, 'Value');
switch popup_sel_index
    case 1
        iGL=1;
wmotor=1000;
        wrot=2*wmotor;
        run('RespostaImpulsoPeriodico2');
        axes(handles.axes1);

António Figueiredo

9/86
"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
set(handles.axes1,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('1000 rpm');
wamotor=2000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes3);
set(handles.axes3,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('2000 rpm');
wamotor=3000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes4);
set(handles.axes4,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('3000 rpm');
wamotor=4000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('4000 rpm');
wamotor=5000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('5000 rpm');
case 2
iGL=2;
wamotor=1000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes1);
set(handles.axes1,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('1000 rpm');
wamotor=2000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes3);
set(handles.axes3,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('2000 rpm');
wamotor=3000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes4);
set(handles.axes4,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('3000 rpm');
wamotor=4000;
wr=2*wamotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Swh/2/pi,abs(Spr[iGL,:]));
xlabel('\omega\text{ rotacao [Hz]}');
title('4000 rpm');
```
wrmotor=5000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('5000 rpm');

case 3
iGL=3;
wmotor=1000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes1);
set(handles.axes1,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('1000 rpm');

wmotor=2000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes3);
set(handles.axes3,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('2000 rpm');

wmotor=3000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes4);
set(handles.axes4,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('3000 rpm');

wmotor=4000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('4000 rpm');

wmotor=5000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('5000 rpm');

case 4
iGL=4;
wmotor=1000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes1);
set(handles.axes1,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('1000 rpm');

wmotor=2000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes3);
set(handles.axes3,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('2000 rpm');

wmotor=3000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('3000 rpm');

wmotor=4000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('4000 rpm');

wmotor=5000;
wrot=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes8);
set(handles.axes8,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('\omega rotacao [Hz]');
title('5000 rpm');
"Characterization of the Vibration and noise of a mass balance system of engine"

run('RespostaImpulsoPeriodico2');
axes(handles.axes4);
set(handles.axes4,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('3000 rpm');
wmotor=4000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('4000 rpm');
wmotor=5000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('5000 rpm');

case 5
iGL=5;
wmotor=1000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes1);
set(handles.axes1,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('1000 rpm');
wmotor=2000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes3);
set(handles.axes3,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('2000 rpm');
wmotor=3000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes4);
set(handles.axes4,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('3000 rpm');
wmotor=4000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes5);
set(handles.axes5,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('4000 rpm');
wmotor=5000;
wr0t=2*wmotor;
run('RespostaImpulsoPeriodico2');
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)));
xlabel('
\omega rotacao [Hz]');
title('5000 rpm');
end
M_Resp_Freq_Bode_Acelerancia.m

function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', fullfilename, ... 
    'gui_Singleton', gui_Singleton, ... 
    'gui_OpeningFcn', @M_RespotaFreq_Bode_Acelerancia_OpeningFcn, ... 
    'gui_OutputFcn', @M_RespotaFreq_Bode_Acelerancia_OutputFcn, ... 
    'gui_LayoutFcn', [] , ... 
    'gui_Callback', []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin 
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before M_RespotaFreq_Bode_Acelerancia is made visible.
function M_RespotaFreq_Bode_Acelerancia_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to M_RespotaFreq_Bode_Acelerancia (see VARARGIN)

% Choose default command line output for M_RespotaFreq_Bode_Acelerancia
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_RespotaFreq_Bode_Acelerancia wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_RespotaFreq_Bode_Acelerancia_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
posi = [scnsize(3)/12,5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',posi, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner)
"Characterization of the Vibration and noise of a mass balance system of engine"

set(handles.background, ...
  'Visible', 'off', ...
  'Units', 'pixels', ...
  'Position', [0 0 15*info.Width 15*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
  set(hObject,'BackgroundColor','white');
else
  set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
  set(hObject,'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5','Sobreposição'});
end
if nrGL==6
  set(hObject,'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5','Grau de Liberdade 6','Sobreposição'});
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu contents as cell array
% contents{get(hObject,'Value')} returns selected item from popupmenu

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
cla;
run('Dados');
run('FFRDacelerancia');
sobre=0;
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
  case 1
    iGL=1;
    set(handles.text2,'String','............');
    set(handles.text3,'String','............');
case 2
    iGL=2;
    set(handles.text2,'String','............');
    set(handles.text3,'String','............');
case 3
    iGL=3;
    set(handles.text2,'String','............');
    set(handles.text3,'String','............');
case 4
    iGL=4;
'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
set(handles.txt2,'String','.............');
set(handles.txt3,'String','.............');

case 5
  iGL=5;
  set(handles.txt2,'String','.............');
  set(handles.txt3,'String','.............');

case 6
  if nrGL==5
    sobrep=1;
  end
  if nrGL==6
    iGL=6;
    set(handles.txt2,'String','.............');
    set(handles.txt3,'String','.............');
  end
end

if sobrep==1
  set(handles.axes3,'Visible','off');
  axes(handles.axes1);
  if nrGL==5
    plot(w/2/pi,tetamaga(1,:),w/2/pi,tetamaga(2,:),w/2/pi,tetamaga(3,:),w/2/pi,tetamaga(4,:),w/2/pi,tetamaga(5,:));
    legend('A_1_1', 'A_2_1', 'A_3_1', 'A_4_1', 'A_5_1','-l);
  end
  if nrGL==6
    plot(w/2/pi,tetamaga(1,:),w/2/pi,tetamaga(2,:),w/2/pi,tetamaga(3,:),w/2/pi,tetamaga(4,:),w/2/pi,tetamaga(5,:),w/2/pi,tetamaga(6,:));
    legend('1° DOF', '2° DOF', '3° DOF', '4° DOF', '5° DOF', '6° DOF','-l);
  end
end

if sobrep==1
  index=([num2str(iGL),',1']);
  axes(handles.axes1);
  plot(w/2/pi,tetamaga(iGL,:));
  set(gca, 'XTickLabel','');
  ylabel('Mag. [dB]');
  title(['[FRFs (acelerancia) \theta_{t_0}',',index,']']);
  axes(handles.axes3);
  plot(w/2/pi,tetaA(iGL,:));
  set(gca, 'YLim', [-180 180]);
  set(gca, 'YTick', [-180 0 180]);
  xlabel('f [Hz]');
  ylabel('fase [°]');
  set(handles.pushbutton3,'Visible','on');
  set(handles.pushbutton4,'Visible','on');
  set(handles.togglebutton1,'Visible','on');
  set(handles.togglebutton2,'Visible','on');
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject, 'Value') returns toggle state of togglebutton1

button_state = get(hObject, 'Value');
if button_state == get(hObject, 'Max')
  set(handles.axes3,'Visible','off');
  set(handles.txt3,'Visible','on') % toggle button is pressed
elseif button_state == get(hObject, 'Min')
  set(handles.txt3,'Visible','off'); % toggle button is not pressed
  set(handles.axes3,'Visible','on');
end

% --- Executes on button press in togglebutton2.
```

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

function togglebutton2_Callback(hObject, eventdata, handles)
  hObject handle to togglebutton2 (see GCBO)
  eventdata reserved - to be defined in a future version of MATLAB
  handles structure with handles and user data (see GUIDATA)

  % Hint: get(hObject, 'Value') returns toggle state of togglebutton2

  button_state = get(hObject, 'Value');
  if button_state == get(hObject, 'Max')
    set(handles.axes1, 'Visible', 'off');
    set(handles.text2, 'Visible', 'on')  % toggle button is pressed
  elseif button_state == get(hObject, 'Min')
    set(handles текст2, 'Visible', 'off');  % toggle button is not pressed
    set(handles.axes1, 'Visible', 'on');
  end

  % --- Executes on button press in pushbutton5.
  function pushbutton5_Callback(hObject, eventdata, handles)
  hObject handle to pushbutton5 (see GCBO)
  eventdata reserved - to be defined in a future version of MATLAB
  handles structure with handles and user data (see GUIDATA)

M_Resp_Freq_Bode_Receptancia.m

function varargout = respostatemp(varargin)
  % Begin initialization code - DO NOT EDIT
  gui_Singleton = 1;
  gui_State = struct('gui_Name',  
                      'M_Resp_Freq_Bode_Receptancia', ... 
                      'gui_OpeningFcn', @M_Resp_Freq_Bode_Receptancia_OpeningFcn, ... 
                      'gui_OutputFcn', @M_Resp_Freq_Bode_Receptancia_OutputFcn, ... 
                      'gui_LayoutFcn', [], ... 
                      'gui_Callback', []);

  if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
  end

  if nargin == 1
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
  else
    gui_mainfcn(gui_State, varargin{:});
  end
  % End initialization code - DO NOT EDIT

  % --- Executes just before M_Resp_Freq_Bode_Receptancia is made visible.
  function M_Resp_Freq_Bode_Receptancia_OpeningFcn(hObject, eventdata, handles, varargin)
  % This function has no output args, see OutputFcn.
  hObject handle to figure
  eventdata reserved - to be defined in a future version of MATLAB
  handles structure with handles and user data (see GUIDATA)
  varargin command line arguments to M_Resp_Freq_Bode_Receptancia (see VARARGIN)

  % Choose default command line output for M_Resp_Freq_Bode_Receptancia
  handles.output = hObject;

  % Update handles structure
  guidata(hObject, handles);

  % UIWAIT makes M_Resp_Freq_Bode_Receptancia wait for user response (see UIRESUME)
  uiswait(handles.figure1);

  % --- Outputs from this function are returned to the command line.
  function varargout = M_Resp_Freq_Bode_Receptancia_OutputFcn(hObject, eventdata, handles)
'Characterization of the Vibration and noise of a mass balance system of engine'

% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout(1) = handles.output;

scnsize = get(0,'ScreenSize');
pos1 = [scnsize(3)/12.5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',pos1, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundu.jpg'); % Read the image file banner.jpg
info = iminfo('fundu.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner);
set(handles.background, ...
'Visible', 'off', ...
'Units', 'pixels', ...
'Position', [0 0 15*info.Width 15*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nGlu==5
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
end
if nGlu==6
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
% contents{get(hObject,'Value')} returns selected item from popupmenu1

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

cla;
run('Dados');
run('FRDRecepcventa');
"Characterization of the Vibration and noise of a mass balance system of engine"

popup sel_index = get(handles.popupmenu1, 'Value');
switch popup sel_index
    case 1
        iGL=1;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
        case 2
        iGL=2;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
        case 3
        iGL=3;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
        case 4
        iGL=4;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
        case 5
        iGL=5;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
        case 6
        iGL=6;
        set(handles.text1,'String','..........');
        set(handles.text2,'String','..........');
end

index=[num2str(iGL),'1']);
axes(handles.axes1);
plot(w/2/pi,tetamag(iGL,:));
set(gca, 'XTickLabel', ' ');
ylabel('Mag. [dB]');
title([''Resposta em Frequencia (receptancia) \theta = ',index,'''
axes(handles.axes3);
plot(w/2/pi,tetaf(iGL,:));
set(gca, 'YLim', [-180 180]);
set(gca, 'YTick', [-180 0 180]);
xlabel('f [Hz]'); ylabel('fase [°]');
set(handles.pushbutton3, 'Visible', 'on');
set(handles.pushbutton4, 'Visible', 'on');
set(handles.togglebutton1, 'Visible', 'on');
set(handles.togglebutton2, 'Visible', 'on');

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject, 'Value') returns toggle state of togglebutton1

button_state = get(hObject, 'Value');
if button_state == get(hObject, 'Max')
    set(handles.axes3, 'Visible', 'off');
    set(handles.text1, 'Visible', 'off');
    set(handles.text2, 'Visible', 'on');
else
    set(handles.axes3, 'Visible', 'on');
    set(handles.text1, 'Visible', 'on');
    set(handles.text2, 'Visible', 'on');
end

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
M_Resp_Freq_Cont Modal_Acelerancia.m

function varargout = respostatemp(varargin)

% Begin initialization code - DO NOT EDIT

gui_Singleton = 1;

% gui_State = struct('gui_Name', mfilename, ...'
gui_Name', gui_Singleton, ...
'gui_OpeningFcn', @M_Resp_Freq_Cont Modal_Acelerancia_OpeningFcn, ...
'gui_OutputFcn', @M_Resp_Freq_Cont Modal_Acelerancia_OutputFcn, ...
'gui_LayoutFcn', [], ...'
gui_Callback', []);

if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargin}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% End initialization code - DO NOT EDIT

% --- Executes just before M_Resp_Freq_Cont Modal_Acelerancia is made visible.
function M_Resp_Freq_Cont Modal_Acelerancia_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to M_Resp_Freq_Cont Modal_Acelerancia (see VARARGIN)

% Choose default command line output for M_Resp_Freq_Cont Modal_Acelerancia
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Resp_Freq_Cont Modal_Acelerancia wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Resp_Freq_Cont Modal_Acelerancia_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT)
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
scnsiz e = get(0,'ScreenSize');
posl = [scnsiz e(3)/12.5,scnsiz e(3)/8.7,scnsiz e(4)/15];
set(hObject, 'Position', posl, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
% set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner);
set(handles.background, ...
'Visible', 'off', ...
'Units', 'pixels', ...
'Position', [0 0 15*info.Width 15*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenul_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenul (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
    set(hObject, 'String', {'1º Grau de Liberdade', '2º Grau de Liberdade', '3º Grau de Liberdade', '4º Grau de Liberdade', '5º Grau de Liberdade'});
else
    if nrGL==6
        set(hObject, 'String', {'1º Grau de Liberdade', '2º Grau de Liberdade', '3º Grau de Liberdade', '4º Grau de Liberdade', '5º Grau de Liberdade', '6º Grau de Liberdade'});
    end
end

% --- Executes on selection change in popupmenul.
function popupmenul_Callback(hObject, eventdata, handles)
% hObject handle to popupmenul (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenul contents as cell array
% contents(get(hObject,'Value')) returns selected item from popupmenul

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

run('RespostaFreq2');
axes(handles.axes1);
cla;
popup_sel_index = get(handles.popupmenul, 'Value');
switch popup_sel_index
    case 1
        iGL=1;
        set(handles.text1,'String','............');
    case 2

20/86 July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
iGL=2;
    set(handles.text1,'String','.........');
end
```

```matlab
if nrGL==5
    plot(2/pi,alfas(1+iGL,:1),2/pi,alfasA(1+iGL,:2),2/pi,
        alfasB(1+iGL,:3),2/pi,alfasA(1+iGL,:4),2/pi,alfasA(1+iGL,:5));
    legend(['A_1',index,'']),'c1','c2','c3','c4','c5','c6','-1');
end
```

```matlab
if nrGL==6
    plot(2/pi,alfas(1+iGL,:1),2/pi,alfasA(1+iGL,:2),2/pi,
        alfasB(1+iGL,:3),2/pi,alfasA(1+iGL,:4),2/pi,alfasA(1+iGL,:5),2/pi,alfasA(1+iGL,:6));
    legend(['A_1',index,'']),'c1','c2','c3','c4','c5','c6','-1');
end
```

```matlab
title(['Resposta em Frequencia/Contribution Modal (Acelerencia) ','
\theta_1','index','']);
xlabel('f [Hz]'); ylabel('Mag. [dB]');
set(handles.pushbutton4,'Visible','on');
set(handles.pushbutton5,'Visible','on');
set(handles.togglebutton1,'Visible','on');
```

```matlab
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text1,'Visible','on');  % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text1,'Visible','off');  % toggle button is not pressed
end
```

```matlab
function varargout = respostatemp(varargin)
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
M_Resp_Freq_Cont_Modal_Receptancia.m
```

```matlab
function varargout = respostatemp(varargin)
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
```

```matlab
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1
```

```matlab
M_Resp_Freq_Cont_Modal_Receptancia.m
```

```matlab
function varargout = respostatemp(varargin)
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```

```matlab
% Begin initialization code - DO NOT EDIT
```
gui_mainfcn(gui_State, varargin{:});

end
% End initialization code - DO NOT EDIT

clc;

% --- Executes just before M_Resposta_Freq_Const_Modal_Recept is made visible.
function M_Resposta_Freq_Const_Modal_Recept_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
h_object = handle to figure
eventdata = reserved - to be defined in a future version of MATLAB
handles = structure with handles and user data (see GUIDATA)
varargin = command line arguments to M_Resposta_Freq_Const_Modal_Recept (see VARARGIN)

% Choose default command line output for M_Resposta_Freq_Const_Modal_Recept
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Resposta_Freq_Const_Modal_Recept wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Resposta_Freq_Const_Modal_Recept_OutputFcn(hObject, eventdata, handles)
% varargout = cell array for returning output args (see VARARGOUT)
h_object = handle to figure
eventdata = reserved - to be defined in a future version of MATLAB
handles = structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
pos1 = [scnsize(3)/12.5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',pos1, 'Visible','on');
refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner)
set(handles.background, ...
'Visible', 'off', ...
'Units', 'pixels', ...
'Position', [0 0 15*info.Width 15*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject = handle to popupmenu1 (see GCBO)
eventdata = reserved - to be defined in a future version of MATLAB
handles = empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject,'BackgroundColor','white');
else

22/86        July 2004
set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
    set(hObject,'String','[1º Grau de Liberdade', '2º Grau de Liberdade', '3º Grau de Liberdade', '4º Grau de Liberdade', '5º Grau de Liberdade']);
end
if nrGL==6
    set(hObject,'String','[1º Grau de Liberdade', '2º Grau de Liberdade', '3º Grau de Liberdade', '4º Grau de Liberdade', '5º Grau de Liberdade', '6º Grau de Liberdade']);
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
% contents(get(hObject,'Value')) returns selected item from popupmenu1

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

run('RespostaFreq2');
axes(handles.axes1);
clear;
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
    case 1
        iGL=1;
        set(handles.text1,'String','.......
    case 2
        iGL=2;
        set(handles.text1,'String','.......
    case 3
        iGL=3;
        set(handles.text1,'String','.......
    case 4
        iGL=4;
        set(handles.text1,'String','.......
    case 5
        iGL=5;
        set(handles.text1,'String','.......
    case 6
        iGL=6;
        set(handles.text1,'String','.......
end

index=str2num(iGL,'1');
if nrGL==5
    plot(w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:));
    legend(['R '_str2num(iGL,'1'),',',',c1°',',',',c2°',',',',c3°',',',',c4°',',',',c5°',',',',-1);
end
if nrGL==6
    plot(w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:),w/2/pi,alfasmm(iGL,:));
    legend(['R '_str2num(iGL,'1'),',',',c1°',',',',c2°',',',',c3°',',',',c4°',',',',c5°',',',',-1);
end
title(['Resposta em Frequencia/Contribuição Modal (Receptancia) 
\thetaeta_(',',index',',')']);

António Figueiredo
23/86
"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
xlabel('f [Hz]'); ylabel('Mag. [dB]');
set(handles.pushbutton3,'Visible','on');
set(handles.pushbutton4,'Visible','on');
set(handles.togglebutton1,'Visible','on');
% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
    % hObject handle to togglebutton1 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Hint: get(hObject, 'Value') returns toggle state of togglebutton1
    button_state = get(hObject, 'Value');
    if button_state == get(hObject, 'Max')
        set(handles.text1,'Visible','on'); % toggle button is pressed
    elseif button_state == get(hObject, 'Min')
        set(handles.text1,'Visible','off'); % toggle button is not pressed
    end

M_Respuesta_Impulsos.m

function varargout = M_Respuesta_Impulsos(varargin)
    % M_RESPUESTA_IMPULSOS M-file for M_Respuesta_Impulsos.fig
    % M_RESPUESTA_IMPULSOS, by itself, creates a new M_RESPUESTA_IMPULSOS or raises
    % the existing
    % singleton.*
    %
    % H = M_RESPUESTA_IMPULSOS returns the handle to a new M_RESPUESTA_IMPULSOS or
    % the handle to
    % the existing singleton*.
    %
    % M_RESPUESTA_IMPULSOS('CALLBACK', hObject, eventData, handles,...) calls the
    % local
    % function named CALLBACK in M_RESPUESTA_IMPULSOS.M with the given input
    % arguments.
    %
    % M_RESPUESTA_IMPULSOS('Property', 'Value', ...) creates a new
    % M_RESPUESTA_IMPULSOS or raises the
    % existing singleton*. Starting from the left, property value pairs are
    % applied to the GUI before M_Respuesta_Impulsos_OpeningFcn gets called.
    % An
    % unrecognized property name or invalid value makes property application
    % stop. All inputs are passed to M_Respuesta_Impulsos_OpeningFcn via varargin.
    %
    % *See GUIDE Options on GUIDE's Tools menu. Choose "GUI allows only one
    % instance to run (singleton)".
    %
    % See also: GUIDE, GUIDATA, GUIDATA
    %
    % Edit the above text to modify the response to help M_Respuesta_Impulsos
    %
    % Last Modified by GUIDE v2.5 25-May-2004 14:33:42
    %
    % Begin initialization code - DO NOT EDIT
    gui_Singleton = 1;
    gui_State = struct('gui_Name', mfilename, ...    
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @M_Respuesta_Impulsos_OpeningFcn, ...
    'gui_OutputFcn', @M_Respuesta_Impulsos_OutputFcn, ...
    'gui_LayoutFcn', [1] , ...  
    'gui_Callback', []);
    if nargin & isstr(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end
    if nargout
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end
```

July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'
"Characterization of the Vibration and noise of a mass balance system of engine"

velocidade=str2double(get(hObject,'String'));
if isnan(velocidade)
    set(hObject, 'String', 0);
    errordlg('Input must be a number', 'Error');
end

data = getappdata(gcaf, 'metricdata');
data.velocidade = velocidade;
setappdata(gcaf, 'metricdata', data);

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

data = getappdata(gcaf, 'metricdata');

set(handles.velocidade, 'Style', 'text', 'FontWeight', 'demi', 'FontSize', 12, 'BackgroundColor', [0.8 0.8 0.8]);
set(handles.pushbutton1, 'Visible', 'off');
set(handles.popupmenu1, 'Visible', 'on');
set(handles.pushbutton4, 'Visible', 'on');

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFncs called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject, 'BackgroundColor', 'white');
else
    set(hObject, 'BackgroundColor', get(0, 'defaultUicontrolBackgroundColor'));
end
set(hObject, 'String', {'Resposta Temporal', 'Contribuição dos Harmonicos', 'Representação da Solicitação', 'Espectro'});

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
% contents{get(hObject,'Value')} returns selected item from popupmenu1

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
    case 1
        set(handles.popupmenu1, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        set(handles.text5, 'String', 'Resposta Temporal', 'demi', 'FontSize', 10, 'Visible', 'on');
        TipoGraf=1;
data = getappdata(gcaf, 'metricdata');
data.TipoGraf = TipoGraf;
    end

26/86

July 2004
setappdata(gca, 'metricdata', data);
set(handles.popupmenu3, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');

case 2
set(handles.text5, 'String', 'Contribution to Harmonic', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=2;
data = getappdata(gca, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gca, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu3, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');

case 3
set(handles.text5, 'String', 'Representation of Solicitation', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=3;
data = getappdata(gca, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gca, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
wrot=2*X.data.velocidade;
run('RespostaImpulsoPeriodico');
axes(handles.axes1);
set(handles.axes1, 'Visible', 'on');
plot(t,T);
xlabel('t [seg]');
ylabel('f(t)');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.togglebutton1, 'Visible', 'on');
set(handles.text7, 'String', 'Representation of trem de impulsos');

case 4
set(handles.text5, 'String', 'Espectro', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=4;
data = getappdata(gca, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gca, 'metricdata', data);
set(handles.pushbutton9, 'Visible', 'on');
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu3, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');
end

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)

% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcons called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISFC and COMPUTER.
run('Dados');
if ispc
    set(hObject, 'BackgroundColor', 'white');
else
    set(hObject, 'BackgroundColor', get(0, 'default UIFigureBackgroundColor'));
end
if nGl==5
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
end
if nGl==6
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});

António Figueiredo
27/86
function popupmenu3_Callback(hObject, eventdata, handles) 
% hObject handle to popupmenu3 (see GCBO) 
% eventdata reserved - to be defined in a future version of MATLAB 
% handles structure with handles and user data (see GUIDATA) 

% Hints: contents = get(hObject,'String') returns popupmenu3 contents as cell array 
% contents(get(hObject,'Value')) returns selected item from popupmenu3 

function pushbutton5_Callback(hObject, eventdata, handles) 
% hObject handle to pushbutton5 (see GCBO) 
% eventdata reserved - to be defined in a future version of MATLAB 
% handles structure with handles and user data (see GUIDATA) 

data = getappdata(gca, 'metricdata'); 
wrot=2*data.velocidade; 
run('RespostaImpulsoPeriodico2'); 
popup_sel_index = get(handles.popupmenu3, 'Value'); 
set(handles.togglebutton4, 'Visible','on'); 
switch popup_sel_index 
    case 1 
        iGL=1; 
        set(handles.pushbutton7, 'Visible','on'); 
        set(handles.axes1, 'Visible','on'); 
        if data.TipoGraf==1 
            axes(handles.axes1); 
            plot(t,teta1); 
            xlabel('t (seg)'); 
            ylabel('\theta (\text{radians})'); 
            set(handles.pushbutton6, 'Visible', 'on'); 
            set(handles.text7, 'String','........'); 
        end 
    case 2 
        iGL=2; 
        if data.TipoGraf==1 
            axes(handles.axes1); 
            plot(t,teta2); 
            xlabel('t (seg)'); 
            set(handles.pushbutton6, 'Visible', 'on'); 
end
'Characterization of the Vibration and noise of a mass balance system of engine'

if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(2,:,k+1));
        xlabel('t [seg]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end

if data.TipoGraf==4
    set(handles.axes1,'Visible','off');
    axes(handles.axes6);
    set(handles.axes6,'Visible','on');
    stem(Swh/2/pi,Spf);  %Representação da magnitude do impulso em função da frequência
    axes(handles.axes8);
    set(handles.axes8,'Visible','on');
    stem(Swh/2/pi,abs(Spr(2,:)));
    xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end

case 3
    iGL=3;

    if data.TipoGraf==1
        axes(handles.axes1);
        plot(t,teta(3,:));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
    end

    if data.TipoGraf==2
        for k=1:ntsf
            axes(handles.axes1);
            plot(t,tetah(3,:,k+1));
            xlabel('t [seg]');
        end
        set(handles.pushbutton6, 'Visible', 'on');
    end

    if data.TipoGraf==4
        set(handles.axes1,'Visible','off');
        set(handles.axes6,'Visible','on');
        axes(handles.axes6);
        stem(Swh/2/pi,Spf);  %Representação da magnitude do impulso em função da frequência
        axes(handles.axes8);
        set(handles.axes8,'Visible','on');
        stem(Swh/2/pi,abs(Spr(3,:)));
        xlabel('Freq [Hz]');
        set(handles.pushbutton6, 'Visible', 'on');
    end

case 4
    iGL=4;

    if data.TipoGraf==1
        axes(handles.axes1);
        plot(t,teta(4,:));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
    end

    if data.TipoGraf==2
        for k=1:ntsf
            axes(handles.axes1);
            plot(t,tetah(4,:,k+1));
            xlabel('t [seg]');
        end
        set(handles.pushbutton6, 'Visible', 'on');
    end

    if data.TipoGraf==4
        set(handles.axes1,'Visible','off');
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
set(handles.axes6,'Visible','on');
axes(handles.axes6);
stem(SWh/2/pi,Spf);  %Representação da magnitude do impulso em função
da frequência
axes(handles.axes8);
set(handles.axes8,'Visible','on');
stem(SWh/2/pi,abs(Spr(4,:)));
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end

case 5
iGL=5;

if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(5,:));
    xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(5,:,k+1));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
    end
if data.TipoGraf==4
    set(handles.axes1,'Visible','off');
    axes(handles.axes6);
    set(handles.axes6,'Visible','on');
    stem(SWh/2/pi,Spf);  %Representação da magnitude do impulso em função
da frequência
    axes(handles.axes8);
    set(handles.axes8,'Visible','on');
    stem(SWh/2/pi,abs(Spr(5,:)));
    xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end

case 6
iGL=6;
end

if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(iGL,:));
    xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(iGL,:,k+1));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
    end
    axes(handles.axes6);
    cla;
    if data.TipoGraf==4
        set(handles.axes1,'Visible','off');
        if iGL==1
            axes(handles.axes6);
            set(handles.axes6,'Visible','on');
    ```
PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

stem(Sw/2/pi,Spf); %Representação da magnitude do impulso em função da

frequecia
end
axes(handles.axes8);
if iGL==1
  set(handles.axes6,'Visible','off');
end
set(handles.axes8,'Visible','on');
stem(Sw/2/pi,abs(Spr(iGL,:)))
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
  set(handles.togglebutton1,'Visible','off');
  set(handles.togglebutton2,'Visible','on');
  set(handles.togglebutton3,'Visible','on');
else
  set(handles.togglebutton1,'Visible','on');
end

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton7 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton9 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1
button_state = get(hObject, 'Value');
if button_state == get(hObject, 'Max')
  set(handles.text7,'Visible','on'); % toggle button is pressed
elseif button_state == get(hObject, 'Min')
  set(handles.text7,'Visible','off'); % toggle button is not pressed
end

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton2
button_state = get(hObject, 'Value');
if button_state == get(hObject, 'Max')
  set(handles.axes8,'Visible','on'); % toggle button is pressed
set(handles.axes8,'Visible','off'); % toggle button is not pressed
elseif button_state == get(hObject, 'Min')
  set(handles.axes8,'Visible','off'); % toggle button is not pressed
end

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

% Hint: get(hObject, 'Value') returns toggle state of togglebutton3

button_state = get(hObject, 'Value');
if button_state == get(hObject, 'Max')
    set(handles.text8, 'Visible', 'on'); % toggle button is pressed
    set(handles.axes6, 'Visible', 'off');
elseif button_state == get(hObject, 'Min')
    set(handles.text8, 'Visible', 'off'); % toggle button is not pressed
    set(handles.axes6, 'Visible', 'on');
end

M_Responta_Aciclisimo.m

function varargout = M_Responta_Impulsos_Aciclismo(varargin)

    M_Responta_Impulsos_Aciclismo.fig
    M_Responta_Impulsos_Aciclismo, by itself, creates a new
    singleton.
    H = M_Responta_Impulsos_Aciclismo returns the handle to a new
    singleton.
    M_Responta_Impulsos_Aciclismo or the handle to
    the existing singleton.
    M_Responta_Impulsos_Aciclismo('CALLBACK', hObject, eventdata, handles,...)
    calls the local
    function named CALLBACK in M_Responta_Impulsos_Aciclismo.m with the given
    input arguments.
    M_Responta_Impulsos_Aciclismo('Property', 'Value', ...) creates a new
    singleton*. Starting from the left, property value pairs are
    applied to the GUI before M_Responta_Impulsos_Aciclismo_OpeningFcn gets
    called. An
    unrecognized property name or invalid value makes property application
    stop. All inputs are passed to M_Responta_Impulsos_Aciclismo_OpeningFcn via
    varargin.
    *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
    instance to run (singleton)".

    See also: GUIDE, GUIDATA, GUIDATA

% Edit the above text to modify the response to help M_Responta_Impulsos_Aciclismo

% Last Modified by GUIDE v2.5 08-Jun-2004 15:20:22

% Begin initialization code - DO NOT EDIT

    gui_Singleton = 1;
    gui_State = struct('gui_Name', mfilename, ...
        'gui_Singleton', gui_Singleton, ...
        'gui_OpeningFcn', @M_Responta_Impulsos_Aciclismo_OpeningFcn, ...
        'gui_OutputFcn', @M_Responta_Impulsos_Aciclismo_OutputFcnn, ...
        'gui_LayoutFcn', [], ...
        'gui_Callback', []);
    if nargin & isstr(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargin
        varargin{1:nargin} = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end

% End initialization code - DO NOT EDIT

% --- Executes just before M_Responta_Impulsos_Aciclismo is made visible.
function M_Responta_Impulsos_Aciclismo_OpeningFcn(hObject, eventdata, handles, varargin)
'Characterization of the Vibration and noise of a mass balance system of engine'

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)
% varargin  command line arguments to M_Resposta_Impulsos_Acilismo (see VARARGIN)

% Choose default command line output for M_Resposta_Impulsos_Acilismo
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Resposta_Impulsos_Acilismo wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Resposta_Impulsos_Acilismo_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout(1) = handles.output;

 scnsz = get(0,'ScreenSize');
 pos   = [scnsz(3)/12.5,scnsz(3)/8.7,scnsz(4)/15];
 set(hObject, 'Position',pos, 'Visible','on');

refresh
set(hObject, 'Units','pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
 info = iminfo('fundo.jpg'); % Determine the size of the image file
 position = get(hObject, 'Position');
 axes(handles.background);
 image(handles.banner)
 set(handles.background,...
   'Visible','off',...
   'Units','pixels',...
   'Position', [0 0 10*info.Width 10*info.Height]);

% --- Executes during object creation, after setting all properties.
function velocidade_CreateFcn(hObject, eventdata, handles)
% hObject    handle to velocidade (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles   empty - handles not defined until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
  set(hObject,'BackgroundColor','white');
else
  set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

function velocidade_Callback(hObject, eventdata, handles)
% hObject    handle to velocidade (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of velocidade as text
% str2double(get(hObject,'String')) returns contents of velocidade as a double

velocidade=str2double(get(hObject,'String'));
if isnan(velocidade)
"Project FIM-DE-CURSO"
case 2
set(handles.text5, 'String', 'Contribuição do Harmonico', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=2;
data = getappdata(gcf, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcf, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu3, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');

case 3
set(handles.text5, 'String', 'Representação da Solicitação', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=3;
data = getappdata(gcf, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcf, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
wrot2*data.velocidade;
run('RespostaImpulsoPeriodicoAciclismo');
axes(handles.axes1);
set(handles.axes1, 'Visible', 'on');
plot(t, TIP);
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.togglebutton1, 'Visible', 'on');
set(handles.text7, 'String', 'Representação do trem de impulsos');

case 4
set(handles.text5, 'String', 'Espectro', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=4;
data = getappdata(gcf, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcf, 'metricdata', data);
set(handles.pushbutton9, 'Visible', 'on');
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu3, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'on');
end

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject, 'BackgroundColor', 'white');
else
    set(hObject, 'BackgroundColor', get(0, 'defaultUicontrolBackgroundColor'));
end
if nrGL==5
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
end
if nrGL==6
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
end
"Characterization of the Vibration and noise of a mass balance system of engine"

function popupmenu3_Callback(hObject, eventdata, handles)
  % hObject handle to popupmenu3 (see GCBO)
  % eventdata reserved - to be defined in a future version of MATLAB
  % handles structure with handles and user data (see GUIDATA)

  % Hints: contents = get(hObject,'String') returns popupmenu3 contents as cell array
  % contents{get(hObject,'Value')} returns selected item from popupmenu3

function pushbutton5_Callback(hObject, eventdata, handles)
  % hObject handle to pushbutton5 (see GCBO)
  % eventdata reserved - to be defined in a future version of MATLAB
  % handles structure with handles and user data (see GUIDATA)
  data = getappdata(gcf, 'metricdata');
  wrot = 2*data.velocidade;
  run(['RespostaImpulsoPeriodicoAlicilismo']);
  popup_sel_index = get(handles.popupmenu3, 'Value');
  switch popup_sel_index
    case 1
      iGL = 1;
      set(handles.pushbutton7, 'Visible', 'on');
      set(handles.axes1, 'Visible', 'on');
      if data.TipoGraf==1
        axes(handles.axes1);
        plot(t,teta(1,:));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
        set(handles.text7, 'String', '........');
      end
      if data.TipoGraf==2
        for k=1:ntsf
          axes(handles.axes1);
          plot(t,teta1(k,:));
          xlabel('t [seg]');
        end
        set(handles.pushbutton6, 'Visible', 'on');
      end
    case 2
      iGL = 2;
      if data.TipoGraf==1
        axes(handles.axes1);
        plot(t,teta(2,:));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
      end
      if data.TipoGraf==2
        for k=1:ntsf
          axes(handles.axes1);
          plot(t,teta2(k,:));
          xlabel('t [seg]');
        end
        set(handles.pushbutton6, 'Visible', 'on');
      end
    case 3
      iGL = 3;
      if data.TipoGraf==1
        axes(handles.axes1);
        plot(t,teta(3,:));
        xlabel('t [seg]');
        set(handles.pushbutton6, 'Visible', 'on');
      end
  end

36/86 July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tehta(3,:,k+1));
        xlabel('t [sec]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible', 'off');
    set(handles.axes6, 'Visible', 'on');
    axes(handles.axes6);
    stem(Swh/2/pi,Spf);  %Representaçaõ da magnitu de do impulso em funçao
da frequencia
    axes(handles.axes8);
    set(handles.axes8, 'Visible', 'on');
    stem(Swh/2/pi,abs(Spr(3,:)));
    xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end
case 4
iGL=4;
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,tehta(4,:));
    xlabel('t [sec]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tehta(4,:,k+1));
        xlabel('t [sec]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible', 'off');
    set(handles.axes6, 'Visible', 'on');
    axes(handles.axes6);
    stem(Swh/2/pi,Spf);  %Representação da magnitude do impulso em função
    da frequência
    axes(handles.axes8);
    set(handles.axes8, 'Visible', 'on');
    stem(Swh/2/pi,abs(Spr(4,:)));
    xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end
case 5
iGL=5;
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(5,:));
    xlabel('t [sec]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(5,:,k+1));
        xlabel('t [sec]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible', 'off');
end
```

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,Spf); %Representação da magnitude do impulso em função da frequência
axes(handles.axes8);
set(handles.axes8,'Visible','on');
stem(Swh/2/pi,abs(Spr(5,:)));
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end

case 6
iGL=6;
end

if data.TipoGraf==1
axes(handles.axes1);
plot(t,teta(iGL,:));
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
end

if data.TipoGraf==2
for k=1:ntsf
    axes(handles.axes1);
    plot(t,teta(iGL,:),k+1);
    xlabel('t [seg]');
end
set(handles.pushbutton6, 'Visible', 'on');
end
axes(handles.axes6);
cla;
if data.TipoGraf==4
    set(handles.axes1,'Visible','off');
if iGL==1
    axes(handles.axes6);
    set(handles.axes6,'Visible','on');
    stem(Swh/2/pi,Spf); %Representação da magnitude do impulso em função da frequência
end
axes(handles.axes8);
if iGL==1
    set(handles.axes6,'Visible','off');
end
set(handles.axes8,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.togglebutton1,'Visible','off');
set(handles.togglebutton2,'Visible','on');
set(handles.togglebutton3,'Visible','on');
else
    set(handles.togglebutton1,'Visible','on');
end

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton9 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
    % hObject handle to togglebutton1 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Hint: get(hObject,'Value') returns toggle state of togglebutton1
    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text7,'Visible','on'); % toggle button is pressed
    elseif button_state == get(hObject,'Min')
        set(handles.text7,'Visible','off'); % toggle button is not pressed
    end

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
    % hObject handle to togglebutton2 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Hint: get(hObject,'Value') returns toggle state of togglebutton2
    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text9,'Visible','on'); % toggle button is pressed
        set(handles.axes8,'Visible','off');
    elseif button_state == get(hObject,'Min')
        set(handles.text9,'Visible','off'); % toggle button is not pressed
        set(handles.axes8,'Visible','on');
    end

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
    % hObject handle to togglebutton3 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Hint: get(hObject,'Value') returns toggle state of togglebutton3
    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text8,'Visible','on'); % toggle button is pressed
        set(handles.axes6,'Visible','off');
    elseif button_state == get(hObject,'Min')
        set(handles.text8,'Visible','off'); % toggle button is not pressed
        set(handles.axes6,'Visible','on');
    end

% --- Executes on button press in togglebutton5.
function togglebutton5_Callback(hObject, eventdata, handles)
    % hObject handle to togglebutton5 (see GCBO)
    % eventdata reserved - to be defined in a future version of MATLAB
    % handles structure with handles and user data (see GUIDATA)
    % Hint: get(hObject,'Value') returns toggle state of togglebutton5

M_Resposta_Impulsos_Seno.m

function varargout = M_Resposta_Impulsos_Seno(varargin)

António Figueiredo

"Characterization of the Vibration and noise of a mass balance system of engine"

M_RESPOSTA_IMPULSOS_SENO M-file for M_Resposta_Idempulsos_Seno.fig
M_RESPOSTA_IMPULSOS_SENO, by itself, creates a new M_RESPPOSTA_IMPULSOS_SENO
or raises the existing
singleton*.

H = M_RESPPOSTA_IMPULSOS_SENO returns the handle to a new
M_RESPPOSTA_IMPULSOS_SENO or the handle to
the existing singleton*.

M_RESPPOSTA_IMPULSOS_SENO('CALLBACK',(hObject, eventdata, handles,...) calls the
local
function named CALLBACK in M_RESPPOSTA_IMPULSOS_SENO.M with the given input
arguments.

M_RESPPOSTA_IMPULSOS_SENO('Property', 'Value', ...) creates a new
M_RESPPOSTA_IMPULSOS_SENO or raises the
existing singleton*. Starting from the left, property value pairs are
applied to the GUI before M_RESPPOSTA_IMPULSOS_SENO_OpeningFcn gets
called. An
unrecognized property name or invalid value makes property application
stop. All inputs are passed to M_Resposta_Idempulsos_Seno_OpeningFcn via
varargin.

*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
instance to run (singleton)".

See also: GUIDE, GUIDATA, GUIDATA

Edit the above text to modify the response to help M_Resposta_Idempulsos_Seno

Last Modified by GUIDE v2.5 08-Jun-2004 15:20:22

Begin initialization code - DO NOT EDIT

figure; close all;

if nargin & isstr(varargin{1})
    guiState = struct('gui_Name', mfilename, ...
        'gui_Singleton', guiSingleton, ...
        'gui_OpeningFcn', @M_Resposta_Idempulsos_Seno_OpeningFcn, ...
        'gui_OutputFcn', @M_Resposta_Idempulsos_Seno_OutputFcn, ...
        'gui_LayoutFcn', [], ...
        'gui_Callback', []);
end

if nargout
   argout{1:nargout} = gui_mainfcn(guiState, varargin{:});
else
    gui_mainfcn(guiState, varargin{:});
end

End initialization code - DO NOT EDIT

--- Executes just before M_Resposta_Idempulsos_Seno is made visible.
function M_Resposta_Idempulsos_Seno_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to M_Resposta_Idempulsos_Seno (see VARARGIN)

% Choose default command line output for M_Resposta_Idempulsos_Seno
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

UIWAIT makes M_Resposta_Idempulsos_Seno wait for user response (see UIRESUME)
uiwait(handles.figure1);
PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

% --- Outputs from this function are returned to the command line.
function varargout = M_Resposta_Impulsos_Seno_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
pos1 = [scnsize(3)/12.5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',pos1, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = iminfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
axes(handles.background);
image(handles.banner)
set(handles.background, ...
    'Visible', 'off', ...
    'Units', 'pixels', ...
    'Position', [0 0 10*info.Width 10*info.Height]);

% --- Executes during object creation, after setting all properties.
function velocidade_CreateFcn(hObject, eventdata, handles)
% hObject handle to velocidade (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcnns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

function velocidade_Callback(hObject, eventdata, handles)
% hObject handle to velocidade (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of velocidade as text
% str2double(get(hObject,'String')) returns contents of velocidade as a double

velocidade=str2double(get(hObject,'String'));
if isnan(velocidade)
    set(hObject, 'String', 0);
    errordlg('Input must be a number', 'Error');
end

data = getappdata(gcbf, 'metricdata');
data.velocidade = velocidade;
setappdata(gcbf, 'metricdata', data);

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

data = getappdata(gcbf, 'metricdata');
"Characterization of the Vibration and noise of a mass balance system of engine"

```matlab
set(handles.velocidade,'Style','text','FontSize',12,'BackgroundColor',[0.8 0.8 0.8]);
set(handles.pushbutton1,'Visible','off');
set(handles.popupmenu,'Visible','on');
set(handles.pushbutton4,'Visible','on');

% --- Executes during object creation, after setting all properties.
function popupmenu_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcn's called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
set(hObject, 'String', {'Resposta Temporal', 'Contribuição dos Harmonicos', 'Representação da Solicitação', 'Espectro'});

% --- Executes on selection change in popupmenu.
function popupmenu_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject, 'String') returns popupmenu contents as cell array
% contents(get(hObject, 'Value')) returns selected item from popupmenu

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

popup_sel_index = get(handles.popupmenu, 'Value');
switch popup_sel_index
    case 1
        set(handles.popupmenu, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        set(handles.text5, 'String', 'Resposta Temporal', 'FontSize',10,'Visible','on');
        TipoGraf=1;
        data = getappdata(gcgbf, 'metricdata');
data.TipoGraf = TipoGraf;
        setappdata(gcgbf, 'metricdata', data);  
        set(handles.popupmenu3, 'Visible', 'on');
        set(handles.pushbutton5, 'Visible', 'on');
    case 2
        set(handles.text5, 'String', 'Contribuição do Harmonico', 'FontSize',10,'Visible','on');
        TipoGraf=2;
        data = getappdata(gcgbf, 'metricdata');
data.TipoGraf = TipoGraf;
        setappdata(gcgbf, 'metricdata', data);  
        set(handles.popupmenu1, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        set(handles.popupmenu3, 'Visible', 'on');
        set(handles.pushbutton5, 'Visible', 'on');
    case 3
        set(handles.text5, 'String', 'Representação da Solicitação', 'FontSize',10,'Visible','on');
        TipoGraf=3;
```

42/86

July 2004
data = getappdata(gcf, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcf, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
wrot=2*data.velocidade;
run('RespostaImpulsoPeriodicoSeno2');
axes(handles.axes1);
set(handles.axes1, 'Visible', 'on');
plot(t,TIP);
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.togglebutton1, 'Visible', 'on');
set(handles.text7, 'String', 'Representação do trem de impulsos');
case 4
set(handles.text5, 'String', 'Espectro', 'FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
TipoGraf=4;
data = getappdata(gcf, 'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcf, 'metricdata', data);
set(handles.pushbutton9, 'Visible', 'on');
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton4, 'Visible', 'off');
set(handles.popupmenu3, 'Visible', 'on');
set(handles.popupmenu5, 'Visible', 'on');
end

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
    hObject handle to popupmenu3 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles empty - handles not created until after all CreateFcn's called

    % Hint: popupmenu controls usually have a white background on Windows.
    % See ISPC and COMPUTER.
    run('Dados');
    if ispc
        set(hObject, 'BackgroundColor', 'white');
    else
        set(hObject, 'BackgroundColor', get(0, 'defaultUicontrolBackgroundColor'));
    end
    if nrgl==5
        set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
    end
    if nrgl==6
        set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
end

% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
    hObject handle to popupmenu3 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles structure with handles and user data (see GUIDATA)

    % Hints: contents = get(hObject, 'String') returns popupmenu3 contents as cell array
    % contents(get(hObject, 'Value')) returns selected item from popupmenu3

% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
    hObject handle to pushbutton5 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles structure with handles and user data (see GUIDATA)
data = getappdata(gcf, 'metricdata');
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

wrt=2*data.velocidade;
run('RespostaImpulsoPeriodicoSeno2');
popup_sel_index = get(handles.popupmenu3, 'Value');
switch popup_sel_index

case 1
iGL=1;
set(handles.pushbutton7,'Visible','on');
set(handles.axes1,'Visible','on');
if data.TipoGraf==1
axes(handles.axes1);
plot(t,theta(1,:));
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.text7,'String','........');
end
if data.TipoGraf==2
for k=1:ntsf
axes(handles.axes1);
plot(t,theta(1,:),k+1);
xlabel('t [seg]');
end
set(handles.pushbutton6, 'Visible', 'on');
end

case 2
iGL=2;
if data.TipoGraf==1
axes(handles.axes1);
plot(t,theta(2,:));
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
for k=1:ntsf
axes(handles.axes1);
plot(t,theta(2,:),k+1);
xlabel('t [seg]');
end
set(handles.pushbutton6, 'Visible', 'on');
end

case 3
iGL=3;
if data.TipoGraf==1
axes(handles.axes1);
plot(t,theta(3,:));
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
for k=1:ntsf
axes(handles.axes1);
plot(t,theta(3,:),k+1);
xlabel('t [seg]');
end
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
set(handles.axes1,'Visible','off');
set(handles.axes6,'Visible','on');
axes(handles.axes6);
stem(Swh/2/pi,SpF);
end
end

44/86 July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
stem(Swh/2/pi, abs(Spr(3,:)));
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end

case 4
iGL=4;

if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(4,:));
xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(4,:,k+1));
xlabel('t [seg]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible','off');
    set(handles.axes6,'Visible','on');
    axes(handles.axes6);
    stem(Swh/2/pi,Spf);
%Representacao da magnitude do impulso em funcao da frequencia
    axes(handles.axes8);
    set(handles.axes8,'Visible','on');
    stem(Swh/2/pi,abs(Spr(4,:)))
xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end

case 5
iGL=5;

if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(5,:));
xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(5,:,k+1));
xlabel('t [seg]');
    end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1,'Visible','off');
    axes(handles.axes6);
    set(handles.axes6,'Visible','on');
    stem(Swh/2/pi,Spf);
%Representacao da magnitude do impulso em funcao da frequencia
    axes(handles.axes8);
    set(handles.axes8,'Visible','on');
    stem(Swh/2/pi,abs(Spr(5,:)))
xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end

case 6
iGL=6;
end
```

António Figueiredo
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(iGL,:));
    xlabel('[t [seg]]');
    set(handles.pushbutton6, 'Visible', 'on');
end

if data.TipoGraf==2
    for k=1:nts
        axes(handles.axes1);
        plot(t,tetah(iGL,:,k+1));
        xlabel('[t [seg]]');
        set(handles.pushbutton6, 'Visible', 'on');
    end
    axes(handles.axes6);
    cla;
    if data.TipoGraf==4
        set(handles.axes1, 'Visible', 'off');
        if iGL==1
            axes(handles.axes6);
            set(handles.axes6, 'Visible', 'on');
            stem(Sw/2/pi,Spf);
            xlabel('Frequecia');
            set(handles.pushbutton6, 'Visible', 'on');
        end
        axes(handles.axes8);
        if iGL==1
            set(handles.axes8, 'Visible', 'off');
        end
        set(handles.axes8, 'Visible', 'on');
        stem(Sw/2/pi,abs(Spr(iGL,:)));
        xlabel('Freq [Hz]');
        set(handles.pushbutton6, 'Visible', 'on');
    end
    if data.TipoGraf==4
        set(handles.togglebutton1, 'Visible', 'off');
        set(handles.togglebutton2, 'Visible', 'on');
        set(handles.togglebutton3, 'Visible', 'on');
    else
        set(handles.togglebutton1, 'Visible', 'on');
    end

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
    hObject handle to pushbutton7 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
    hObject handle to pushbutton9 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
    hObject handle to togglebutton1 (see GCBO)
    eventdata reserved - to be defined in a future version of MATLAB
    handles structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text7, 'Visible', 'on');  % toggle button is pressed

46/86  July 2004
"Characterization of the Vibration and noise of a mass balance system of engine"

elseif button_state == get(hObject,'Min')
    set(handles.text7,'Visible','off'); % toggle button is not pressed
end

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
    % hObject    handle to togglebutton2 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hint: get(hObject,'Value') returns toggle state of togglebutton2

    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text9,'Visible','on'); % toggle button is pressed
        set(handles.axes8,'Visible','off');
    elseif button_state == get(hObject,'Min')
        set(handles.text9,'Visible','off'); % toggle button is not pressed
        set(handles.axes8,'Visible','on');
    end

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
    % hObject    handle to togglebutton3 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hint: get(hObject,'Value') returns toggle state of togglebutton3

    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text8,'Visible','on'); % toggle button is pressed
        set(handles.axes6,'Visible','off');
    elseif button_state == get(hObject,'Min')
        set(handles.text8,'Visible','off'); % toggle button is not pressed
        set(handles.axes6,'Visible','on');
    end

% --- Executes on button press in togglebutton5.
function togglebutton5_Callback(hObject, eventdata, handles)
    % hObject    handle to togglebutton5 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hint: get(hObject,'Value') returns toggle state of togglebutton5

M_Resposta_impulsos_seno_e_impulso.m

function varargout = M_Resposta_Impulsos_Seno_e_Impulso(varargin)
    % M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO M-file for
    % M_Resposta_Impulsos_Seno_e_Impulso.fig
    % M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO, by itself, creates a new
    % M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO or raises the existing
    % singleton.
    %
    % H = M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO returns the handle to a new
    % M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO or the handle to
    % the existing singleton.
    %
    % M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO('CALLBACK', hObject, eventdata, handles, ...) calls the local
    % function named CALLBACK in M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO.m with the
    % given input arguments.

António Figueiredo 47/86
"Characterization of the Vibration and noise of a mass balance system of engine"

%% M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO('Property','Value',...) creates a new
%% M_RESPOSTA_IMPULSOS_SENO_E_IMPULSO or raises the
%% existing singleton*. Starting from the left, property value pairs are
%% applied to the GUI before M_Respuesta_Impulsos_Seno_e_Impulso_OpeningFunction
%% gets called. An
%% unrecognized property name or invalid value makes property application
%% stop. All inputs are passed to
%% M_Respuesta_Impulsos_Seno_e_Impulso_OpeningFcn via varargin.
%%
%% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
%% instance to run (singleton)".
%%
%% See also: GUIDE, GUIDATA, GUIDATA

%% Edit the above text to modify the response to help
M_Respuesta_Impulsos_Seno_e_Impulso

%% Last Modified by GUIDE v2.5 25-May-2004 14:33:42

%% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
'gui_Singleton', gui_Singleton, ...
'gui_OpeningFcn', @M_Respuesta_Impulsos_Seno_e_Impulso_OpeningFcn, ...
'gui_OutputFcn', @M_Respuesta_Impulsos_Seno_e_Impulso_OutputFcn, ...
'gui_LayoutFcn', [], ...
'gui_Callback', []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if narginout
    [varargout{1:narginout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
%% End initialization code - DO NOT EDIT

%% --- Executes just before M_Respuesta_Impulsos_Seno_e_Impulso is made visible.
function M_Respuesta_Impulsos_Seno_e_Impulso_OpeningFcn(hObject, eventdata, handles, varargin)
%% This function has no output args, see OutputFcn.
%% hObject handle to figure
%% eventdata reserved - to be defined in a future version of MATLAB
%% handles structure with handles and user data (see GUIDATA)
%% varargin command line arguments to M_Respuesta_Impulsos_Seno_e_Impulso (see
%% VARARGIN)

%% Choose default command line output for M_Respuesta_Impulsos_Seno_e_Impulso
handles.output = hObject;

%% Update handles structure
guidata(hObject, handles);

%% UIWAIT makes M_Respuesta_Impulsos_Seno_e_Impulso wait for user response (see
%% UIRESUME)
%% uihat(handles.figure1);

%% --- Outputs from this function are returned to the command line.
function varargout = M_Respuesta_Impulsos_Seno_e_Impulso_OutputFcn(hObject, eventdata, handles)
%% varargout cell array for returning output args (see VARARGOUT);
%% hObject handle to figure
%% eventdata reserved - to be defined in a future version of MATLAB
%% handles structure with handles and user data (see GUIDATA)

%% Get default command line output from handles structure
varargout{1} = handles.output;
scnssize = get(0,'ScreenSize');
posl = [scnssize(3)/12.5,scnssize(3)/8.7,scnssize(4)/15];
set(hObject,'Position',posl,'Visible','on');

refresh
set(hObject,'Units','pixels');
handles.banner = imread('fundo.jpg');  % Read the image file banner.jpg
info = imfinfo('fundo.jpg');  % Determine the size of the image file
position = get(hObject,'Position');
axes(handles.background);
image(handles.banner)
set(handles.background,...
'Visible','off',...
'Units','pixels',...
'Position',[0 0 10*info.Width 10*info.Height]);

%  --- Executes during object creation, after setting all properties.
function velocidade_CreateFcn(hObject, eventdata, handles)
  hObject           % hObject handle to velocidade (see GCBO)
  eventdata reserved - to be defined in a future version of MATLAB
  handles            % handles structure with handles and user data (see GUIDATA)

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
  set(hObject,'BackgroundColor','white');
else
  set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

function velocidade_Callback(hObject, eventdata, handles)
  hObject           % hObject handle to velocidade (see GCBO)
  eventdata reserved - to be defined in a future version of MATLAB
  handles            % handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of velocidade as text
% str2double(get(hObject,'String')) returns contents of velocidade as a double

velocidade=str2double(get(hObject,'String'));
if isnan(velocidade)
   set(hObject, 'String', 0);
   errordlg('Input must be a number', 'Error');
end

data = getappdata(gcbf, 'metricdata');
data.velocidade = velocidade;
setappdata(gcbf, 'metricdata', data);

%  --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
  hObject           % hObject handle to pushbutton1 (see GCBO)
  eventdata reserved - to be defined in a future version of MATLAB
  handles            % handles structure with handles and user data (see GUIDATA)

data = getappdata(gcbf, 'metricdata');

set(handles.velocidade,'Style','text','FontWeight','demi','FontSize',12,'BackgroundColor',[0.8 0.8 0.8]);
set(handles.pushbutton1,'Visible','off');
set(handles.popupmenu1, 'Visible', 'on');
set(handles.pushbutton4, 'Visible', 'on');

%  --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
  hObject           % hObject handle to popupmenu1 (see GCBO)
"Characterization of the Vibration and noise of a mass balance system of engine"

% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreatePcnns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
set(hObject,'String', {'Resposta Temporal', 'Contribuição dos Harmonicos', 'Representação da Solicitação', 'Espectro'});

% --- Executes on selection change in popupmenul.
function popupmenul_Callback(hObject, eventdata, handles)
% hObject handle to popupmenul (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenul contents as cell array
% contents(get(hObject,'Value')) returns selected item from popupmenul

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

popup_sel_index = get(handles.popupmenul, 'Value');
switch popup_sel_index
    case 1
        set(handles.popupmenul, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        set(handles.text5, 'String', 'Resposta Temporal', ' FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
        TipoGraf = 1;
        data = getappdata(gcf, 'metricdata');
        data.TipoGraf = TipoGraf;
        setappdata(gcf, 'metricdata', data);
        set(handles.popmenu3, 'Visible', 'on');
        set(handles.pushbutton5, 'Visible', 'on');
    case 2
        set(handles.text5, 'String', 'Contribuição do Harmonico', ' FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
        TipoGraf = 2;
        data = getappdata(gcf, 'metricdata');
        data.TipoGraf = TipoGraf;
        setappdata(gcf, 'metricdata', data);
        set(handles.popmenu1, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        set(handles.popmenu3, 'Visible', 'on');
        set(handles.pushbutton5, 'Visible', 'on');
    case 3
        set(handles.text5, 'String', 'Representação da Solicitação', ' FontWeight', 'demi', 'FontSize', 10, 'Visible', 'on');
        TipoGraf = 3;
        data = getappdata(gcf, 'metricdata');
        data.TipoGraf = TipoGraf;
        setappdata(gcf, 'metricdata', data);
        set(handles.popmenu1, 'Visible', 'off');
        set(handles.pushbutton4, 'Visible', 'off');
        wrot=2*data.velocidade;
        run('RespostaImpulsoPeriodicoSenoEImpulso2');
        axes(handles.axes1);
        set(handles.axes1, 'Visible', 'on');
        plot(t,TIP);
end
'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
set(handles.togglebutton1,'Visible','on');
set(handles.text7,'String','Representacao do trem de impulsos');
case 4
set(handles.text5,'String','Espectro','FontWeight',' demi','FontSize',10,'Visible','on');
TipoGraf=4;
data = getappdata(gcaf,'metricdata');
data.TipoGraf = TipoGraf;
setappdata(gcaf,'metricdata',data);
set(handles.pushbutton9,'Visible','on');
set(handles.popupmenu1,'Visible','off');
set(handles.pushbutton4,'Visible','off');
set(handles.popupmenu3,'Visible','on');
set(handles.pushbutton5,'Visible','on');
end

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
end
if nrGL==6
    set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
end

% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu3 (see GCBO)
% eventdata reserved to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu3 contents as cell array
% contents(get(hObject,'Value')) returns selected item from popupmenu3

% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton5 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
data = getappdata(gcaf,'metricdata');

wrot=2*data.velocidade;
run(['RespostaImpulsoPeriodicoSenoEImpulso2']);
popup_sel_index = get(handles.popupmenu3,'Value');
set(handles.togglebutton4,'Visible','on');
switch popup_sel_index
case 1
    IGL=1;
    set(handles.pushbutton7,'Visible','on');
```

António Figueiredo

51/86
"Characterization of the Vibration and noise of a mass balance system of engine"

```plaintext
set(handles.axes1,'Visible','on');
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(1,:));
    xlabel('t [seg]');
    set(handles.pushbutton6,'Visible','on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(1,:,k+1));
        xlabel('t [seg]');
    end
    set(handles.pushbutton6,'Visible','on');
end
case 2
iGL=2;
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(2,:));
    xlabel('t [seg]');
    set(handles.pushbutton6,'Visible','on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(2,:,k+1));
        xlabel('t [seg]');
    end
    set(handles.pushbutton6,'Visible','on');
end
case 3
iGL=3;
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t,teta(3,:));
    xlabel('t [seg]');
    set(handles.pushbutton6,'Visible','on');
end
if data.TipoGraf==2
    for k=1:ntsf
        axes(handles.axes1);
        plot(t,tetah(3,:,k+1));
        xlabel('t [seg]');
    end
    set(handles.pushbutton6,'Visible','on');
end
if data.TipoGraf==4
    set(handles.axes1,'Visible','off');
    set(handles.axes6,'Visible','on');
    axes(handles.axes6);
    stem(Swh/2/pi,Spf);  %Representação da magnitude do impulso em função da frequencia
    axes(handles.axes8);
    set(handles.axes8,'Visible','on');
    stem(Swh/2/pi,abs(Spr(3,:)));
    xlabel('Freq [Hz]');
    set(handles.pushbutton6,'Visible','on');
end
case 4
iGL=4;
if data.TipoGraf==1
    axes(handles.axes1);
end
```

52/86  
July 2004
plot(t, teta(4,:,));
xlabel('t [seg]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
for k=1:ntsf
    axes(handles.axes1);
    plot(t, teta(4,:,k+1));
xlabel('t [seg]');
end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible', 'off');
    set(handles.axes6, 'Visible', 'on');
    axes(handles.axes6);
    stem(Swh/2/pi, Spf); %Representação da magnitude do impulso em função da frequência
    axes(handles.axes8);
    set(handles.axes8, 'Visible', 'on');
    stem(Swh/2/pi, abs(Spr(4,:)))
xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end
case 5
iGL=5;
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t, teta(5,:));
xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
for k=1:ntsf
    axes(handles.axes1);
    plot(t, teta(5,:,k+1));
xlabel('t [seg]');
end
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
    set(handles.axes1, 'Visible', 'off');
    axes(handles.axes6);
    set(handles.axes6, 'Visible', 'on');
    stem(Swh/2/pi, Spf); %Representação da magnitude do impulso em função da frequência
    axes(handles.axes8);
    set(handles.axes8, 'Visible', 'on');
    stem(Swh/2/pi, abs(Spr(5,:)))
xlabel('Freq [Hz]');
    set(handles.pushbutton6, 'Visible', 'on');
end
case 6
iGL=6;
end
if data.TipoGraf==1
    axes(handles.axes1);
    plot(t, teta(iGL,:));
xlabel('t [seg]');
    set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==2
for k=1:ntsf

António Figueiredo
"Characterization of the Vibration and noise of a mass balance system of engine"

PROJECT FIM-DE-CURSO

axes(handles.axes1);
plot(t,tetah(iGL::k+1));
xlabel('t [seg]');
end
set(handles.pushbutton6, 'Visible', 'on');
end
axes(handles.axes6);
cla;
if data.TipoGraf==4
set(handles.axes1,'Visible','off');
if iGL==1
axes(handles.axes6);
set(handles.axes6,'Visible','on');
stem(Swh/2/pi,Spf); %Representação da magnitude do impulso em função da
frequencia
end
axes(handles.axes8);
if iGL==1
set(handles.axes6,'Visible','off');
end
set(handles.axes8,'Visible','on');
stem(Swh/2/pi,abs(Spr(iGL,:)));
xlabel('Freq [Hz]');
set(handles.pushbutton6, 'Visible', 'on');
end
if data.TipoGraf==4
set(handles.togglebutton1,'Visible','off');
set(handles.togglebutton2,'Visible','on');
set(handles.togglebutton3,'Visible','on');
else
set(handles.togglebutton1,'Visible','on');
end

%  --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%  --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%  --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text7,'Visible','on'); % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text7,'Visible','off'); % toggle button is not pressed
end

%  --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of togglebutton2

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text9,'Visible','on');  % toggle button is pressed
    set(handles.axes8,'Visible','off');
elseif button_state == get(hObject,'Min')
    set(handles.text9,'Visible','off');  % toggle button is not pressed
    set(handles.axes8,'Visible','on');
end

% --- Executes on button press in togglebutton3.
function togglebutton3_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton3

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text8,'Visible','on');  % toggle button is pressed
    set(handles.axes6,'Visible','off');
elseif button_state == get(hObject,'Min')
    set(handles.text8,'Visible','off');  % toggle button is not pressed
    set(handles.axes6,'Visible','on');
end

% M_Resposta_Temporal_Cont_MODAL.m
function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT
for i = 1:length(varargout)
    varargout{i} = int2str(i);
end

% End initialization code - DO NOT EDIT

clc;

% --- Executes just before M_Resposta_Temporal_Cont_MODAL is made visible.
function M_Resposta_Temporal_Cont_MODAL_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to M_Resposta_Temporal_Cont_MODAL (see VARARGIN)

% Choose default command line output for M_Resposta_Temporal_Cont_MODAL
handles.output = hObject;

% Update handles structure

António Figueiredo
% UIWAIT makes M_Resposta_Temporal_Cont Modal wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Resposta_Temporal_Cont_Modal_OutputFcn(hObject, eventdata, handles)
varargout = cell array for returning output args (see VARARGOUT);
 hObject = handle to figure
 eventdata = reserved - to be defined in a future version of MATLAB
 handles = structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
posl = [scnsize(3)/12.5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',posl,'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner);
set(handles.background, ...
'Visible','off', ...
'Units', 'pixels', ...
'Position', [0 0 10*info.Width 10*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu_CreateFcn(hObject, eventdata, handles)
 hObject = handle to popupmenu (see GCBO)
 eventdata = reserved - to be defined in a future version of MATLAB
 handles = empty - handles not created until after all CreateFcsns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
 set(hObject,'BackgroundColor','white');
else
 set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
 set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
end
if nrGL==6
 set(hObject, 'String', {'Grau de Liberdade 1', 'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
end

% --- Executes on selection change in popupmenu.
function popupmenu_Callback(hObject, eventdata, handles)
 hObject = handle to popupmenu (see GCBO)
 eventdata = reserved - to be defined in a future version of MATLAB
 handles = structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu contents as cell array
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

axes(handles.axes1);
cla;
run('RespostaTemp');
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
  case 1
    Gliberd=1;
    set(handles.text1,'String','Grau de Liberdade 1','Visible','on');
  case 2
    Gliberd=2;
    set(handles.text1,'String','Grau de Liberdade 2','Visible','on');
  case 3
    Gliberd=3;
    set(handles.text1,'String','Grau de Liberdade 3','Visible','on');
  case 4
    Gliberd=4;
    set(handles.text1,'String','Grau de Liberdade 4','Visible','on');
  case 5
    Gliberd=5;
    set(handles.text1,'String','Grau de Liberdade 5','Visible','on');
  case 6
    Gliberd=6;
    set(handles.text1,'String','Grau de Liberdade 6','Visible','on');
end

data = getappdata(gcbf, 'metricdata');
data.Gliberd = Gliberd;
setappdata(gcbf, 'metricdata', data);
set(handles.popupmenu1, 'Visible', 'off');
set(handles.pushbutton1, 'Visible', 'off');
set(handles.popupmenu2, 'Visible', 'on');
set(handles.pushbutton3, 'Visible', 'on');

% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFncs called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.

run('Dados');
if ispc
  set(hObject,'BackgroundColor','white');
else
  set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
  set(hObject, 'String', {'Resposta Total', 'Contribuição 2\textsuperscript{a} modo natural', 'Contribuição 3\textsuperscript{a} modo natural', 'Contribuição 4\textsuperscript{a} modo natural', 'Contribuição 5\textsuperscript{a} modo natural'});
end
if nrGL==6
"Characterization of the Vibration and noise of a mass balance system of engine"

set(hObject, 'String', ['Resposta Total', 'Contribuição 2°modo natural', 'Contribuição 3°modo natural', 'Contribuição 4°modo natural', 'Contribuição 5°modo natural', 'Contribuição 6°modo natural']);
end

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu2 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject, 'String') returns popupmenu2 contents as cell array
% contents{get(hObject, 'Value')} returns selected item from popupmenu2

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
data = getappdata(gcbof, 'metricdata');
set(handles.pushbutton4, 'Visible', 'on');
set(handles.pushbutton5, 'Visible', 'off');
set(handles.pushbutton6, 'Visible', 'off');
set(handles.text2, 'Visible', 'off');
set(handles.text3, 'Visible', 'off');
set(handles.togglebutton1, 'Visible', 'off');
axes(handles.axes1);
cla;
run('Dados');
run('RespostaTemp');
popupSel_index = get(handles.popupmenu2, 'Value');
switch popupSel_index
    case 1
        if nrGL==5
            plot(t, teta(data.Giberd,:), t, tetaSM(data.Giberd,:,1), t, tetaSM(data.Giberd,:,2), t,
                 tetaSM(data.Giberd,:,3), t, tetaSM(data.Giberd,:,4));
            legend('Resposta do Sistema', '2°Modo Natural', '3°Modo Natural', '4°Modo Natural', '5°Modo Natural', '-1');
            if data.Giberd==1
                set(handles.text4, 'String', '...............');
            end
            if data.Giberd==2
                set(handles.text4, 'String', '...............');
            end
            if data.Giberd==3
                set(handles.text4, 'String', '...............');
            end
            if data.Giberd==4
                set(handles.text4, 'String', '...............');
            end
            if data.Giberd==5
                set(handles.text4, 'String', '...............');
            end
        end
if nrGL==6
    plot(t, teta(data.Giberd,:), t, tetaSM(data.Giberd,:,1), t, tetaSM(data.Giberd,:,2), t,
         tetaSM(data.Giberd,:,3), t, tetaSM(data.Giberd,:,4), t, tetaSM(data.Giberd,:,5));
    legend('Resposta do Sistema', '2°Modo Natural', '3°Modo Natural', '4°Modo Natural', '5°Modo Natural', '6°Modo Natural', '-1');
    if data.Giberd==1
        set(handles.text4, 'String', '...............');
    end
    if data.Giberd==2

PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
set(handles.text4,'String','......................');
end
if data.Glierd==3
    set(handles.text4,'String','......................');
end
if data.Glierd==4
    set(handles.text4,'String','......................');
end
if data.Glierd==5
    set(handles.text4,'String','......................');
end
if data.Glierd==6
    set(handles.text4,'String','......................');
end
set(handles.togglebutton1,'Visible','on');
end
case 2
    plot(t,teta(data.Glierd,:),t,tetasm(data.Glierd,:,1));
    legend('Resposta do Sistema','2ºModo Natural',-1);
case 3
    plot(t,teta(data.Glierd,:),t,tetasm(data.Glierd,:,2));
    legend('Resposta do Sistema','3ºModo Natural',-1);
case 4
    plot(t,teta(data.Glierd,:),t,tetasm(data.Glierd,:,3));
    legend('Resposta do Sistema','4ºModo Natural',-1);
case 5
    plot(t,teta(data.Glierd,:),t,tetasm(data.Glierd,:,4));
    legend('Resposta do Sistema','5ºModo Natural',-1);
case 6
    plot(t,teta(data.Glierd,:),t,tetasm(data.Glierd,:,5));
    legend('Resposta do Sistema','6ºModo Natural',-1);
end
xlabel('t [seg]');
ylabel('	heta(t)');
set(handles.pushbutton5,'Visible','on');
set(handles.pushbutton6,'Visible','on');
set(handles.text2,'Visible','on');
set(handles.text3,'Visible','on');

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
%(handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text4,'Visible','on')  % toggle button is pressed
elseif button_state == get(hObject,'Min')
```
function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT

gui_Singleton = 1;
gui_State = struct('gui_Name', 'mfilename', ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @M_Resposta_Temporal_Totem_OpeningFcn, ...
    'gui_OutputFcn', @M_Resposta_Temporal_Totem_OutputFcn, ...
    'gui_Layou inoc, [] , ...
    'gui_Callback', []);

if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

clc;

% --- Executes just before M_Resposta_Temporal_ContModal is made visible.
function M_Resposta_Temporal_ContModal_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to M_Resposta_Temporal_ContModal (see VARARGIN)

% Choose default command line output for M_Resposta_Temporal_ContModal
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Resposta_Temporal_ContModal wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Resposta_Temporal_ContModal_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
posl   = [scnsize(3)/12.5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',posl, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
'Characterization of the Vibration and noise of a mass balance system of engine'

PROJECT FIM-DE-CURSO

% set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
% axes(handles.background);
% image(handles.banner)
set(handles.background, ...
'Visible', 'off', ...
'Units', 'pixels',...
'Position', [0 0 10*info.Width 10*info.Height]);

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');
if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
if nrGL==5
    set(hObject, 'String', {'Resposta do Sistema','Resposta do 1ºGL','Resposta do 2ºGL','Resposta do 3ºGL','Resposta do 4ºGL','Resposta do 5ºGL'});
end
if nrGL==6
    set(hObject, 'String', {'Resposta do Sistema','Resposta do 1ºGL','Resposta do 2ºGL','Resposta do 3ºGL','Resposta do 4ºGL','Resposta do 5ºGL','Resposta do 6ºGL'});
end

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject handle to popupmenu1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns popupmenu1 contents as cell array
% contents(get(hObject,'Value')) returns selected item from popupmenu1

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

run('Dados');
run('RespostaTemp');
axes(handles.axes1);
cla;
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
  case 1
    if nrGL==5
      plot(t,teta(1,:),t,teta(2,:),t,teta(3,:),t,teta(4,:),t,teta(5,:));
      legend('Resposta GL 1','Resposta GL 2','Resposta GL 3','Resposta GL 4','Resposta GL 5','-1');
      set(handles.text3,'String','.............');
    end
    if nrGL==6
      plot(t,teta(1,:),t,teta(2,:),t,teta(3,:),t,teta(4,:),t,teta(5,:),t,teta(6,:));
      legend('Resposta GL 1','Resposta GL 2','Resposta GL 3','Resposta GL 4','Resposta GL 5','Resposta GL 6','-1');
      set(handles.text3,'String','.............');
    end
António Figueiredo

61/86
"Characterization of the Vibration and noise of a mass balance system of engine"

end
axis([0 max(t) 1.2*ymin+total 1.2*ymax+total]);

case 2
plot(t,teta(1,:));
axis([0 max(t) 1.2*ymin(1) 1.2*ymax(1)]);
set(handles.text3,'String','.............');

case 3
plot(t,teta(2,:));
axis([0 max(t) 1.2*ymin(2) 1.2*ymax(2)]);
set(handles.text3,'String','.............');

case 4
plot(t,teta(3,:));
axis([0 max(t) 1.2*ymin(3) 1.2*ymax(3)]);
set(handles.text3,'String','.............');

case 5
plot(t,teta(4,:));
axis([0 max(t) 1.2*ymin(4) 1.2*ymax(4)]);
set(handles.text3,'String','.............');

case 6
plot(t,teta(5,:));
axis([0 max(t) 1.2*ymin(5) 1.2*ymax(5)]);
set(handles.text3,'String','.............');

case 7
plot(t,teta(6,:));
axis([0 max(t) 1.2*ymin(6) 1.2*ymax(6)]);
set(handles.text3,'String','.............');

end
xlabel('t [seg]');
ylabel('\theta \theta (t)');
set(handles.pushbutton3, 'Visible','on');
set(handles.togglebutton1, 'Visible','on');
set(handles.pushbutton4, 'Visible','on');
set(handles.text1, 'Visible','on');
set(handles.text2, 'Visible','on');

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton3 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject handle to pushbutton4 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject handle to togglebutton1 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')

62/86  July 2004
PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

```matlab
set(handles.text3,'Visible','on') % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text3,'Visible','off'); % toggle button is not pressed
end

M_Responta_Transiente.m

function varargout = respostatemp(varargin)
% Begin initialization code - DO NOT EDIT

% gui_Singleton = 1;
% gui_State = struct('gui_Name', ...
%                       'gui_Singleton', gui_Singleton, ...
%                       'gui_OpeningFcn', @M_Responta_Transiente_OpeningFcn, ...
%                       'gui_OutputFcn', @M_Responta_Transiente_OutputFcn, ...
%                       'gui_LayoutFcn', [], ...
%                       'gui_Callback', []);

if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
else
    gui_State.gui_Callback = str2func(varargin{:});
end

% End initialization code - DO NOT EDIT

clc;
% --- Executes just before M_Responta_Transiente is made visible.
function M_Responta_Transiente_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to M_Responta_Transiente (see VARARGIN)

% Choose default command line output for M_Responta_Transiente
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Responta_Transiente wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = M_Responta_Transiente_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARargout);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
posl = [scnsize(3)/12,5,scnsize(3)/8.7,scnsize(4)/15];
set(hObject, 'Position',posl,'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = iminfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner)
set(handles.background, ...

António Figueiredo
63/86
"Characterization of the Vibration and noise of a mass balance system of engine"

% --- Executes during object creation, after setting all properties.
function popupmenu_CreateFcn(hObject, eventdata, handles)
    hObject       = popupmenu1;   % see GCBO
    eventdata     = reserved - to be defined in a future version of MATLAB
    handles       = empty - handles not created until after all CreateFcns called

    % Hint: popupmenu controls usually have a white background on Windows.
    % See ISPC and COMPUTER.
    run('Dados');
    if ispc
        set(hObject,'BackgroundColor','white');
    else
        set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
    end

    if nrGL==5
        set(hObject,'String', {'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5'});
    else
        set(hObject,'String', {'Grau de Liberdade 2', 'Grau de Liberdade 3', 'Grau de Liberdade 4', 'Grau de Liberdade 5', 'Grau de Liberdade 6'});
    end

% --- Executes on selection change in popupmenu.
function popupmenu_Callback(hObject, eventdata, handles)
    hObject       = popupmenu1;   % see GCBO
    eventdata     = reserved - to be defined in a future version of MATLAB
    handles       = structure with handles and user data (see GUIDATA)

    % Hints: contents = get(hObject,'String') returns popupmenu contents as cell array
    % contents(get(hObject,'Value')) returns selected item from popupmenu

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
    hObject       = pushbutton1;   % see GCBO
    eventdata     = reserved - to be defined in a future version of MATLAB
    handles       = structure with handles and user data (see GUIDATA)

    axes(handles.axes1);
    clc;
    cla;
    run('RespostaTransiente');
    popupSelIndex = get(handles.popupmenu1,'Value');
    switch popupSelIndex
        case 1
            plot(t,teta(1,:));grid on; hold on;
            set(handles.text1,'String', '.........................');
        case 2
            plot(t,teta(2,:));grid on; hold on;
            set(handles.text1,'String', '.........................');
        case 3
            plot(t,teta(3,:));grid on; hold on;
            set(handles.text1,'String', '.........................');
        case 4
            plot(t,teta(4,:));grid on; hold on;
            set(handles.text1,'String', '.........................');
        case 5
            plot(t,teta(5,:));grid on; hold on;
            set(handles.text1,'String', '.........................');
    end
    xlabel('t [seg]');
    ylabel('\theta (t)');
    title('Resposta no Tempo a um transiente');
PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

set(handles.pushbutton3,'Visible','on');
set(handles.togglebutton1,'Visible','on');

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
    % hObject    handle to togglebutton1 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hint: get(hObject,'Value') returns toggle state of togglebutton1

    button_state = get(hObject,'Value');
    if button_state == get(hObject,'Max')
        set(handles.text1,'Visible','on') % toggle button is pressed
    elseif button_state == get(hObject,'Min')
        set(handles.text1,'Visible','off'); % toggle button is not pressed
    end

M_Velocidades_Criticas.m

function varargout = M_Velocidades_Criticas(varargin)
% M_VELOCIDADES_CRITICAS M-file for M_Velocidades_Criticas.fig
% M_VELOCIDADES_CRITICAS, by itself, creates a new M_VELOCIDADES_CRITICAS or
% raises the existing singleton*.

% H = M_VELOCIDADES_CRITICAS returns the handle to a new
% M_VELOCIDADES_CRITICAS or the handle to
% the existing singleton*.

% M_VELOCIDADES_CRITICAS('CALLBACK',hObject,eventData,handles,...) calls the
% local
% function named CALLBACK in M_VELOCIDADES_CRITICAS.M with the given input
% arguments.

% M_VELOCIDADES_CRITICAS('Property','Value',...) creates a new
% M_VELOCIDADES_CRITICAS or raises the
% existing singleton*. Starting from the left, property value pairs are
% applied to the GUI before M_Velocidades_Criticas_OpeningFcn gets
% called. An
% unrecognized property name or invalid value makes property application
% stop. All inputs are passed to M_Velocidades_Criticas_OpeningFcn via
% varargin.

% See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)".

% See also: GUIDE, GUIDATA, GUIDATA

% Edit the above text to modify the response to help M_Velocidades_Criticas

% Last Modified by GUIDE v2.5 07-May-2004 12:16:45

% Begin initialization code - DO NOT EDIT

    gui_Singleton = 1;
    gui_State = struct('gui_Name', 'mfilename', ... 
        'gui_Singleton', gui_Singleton, ... 
        'gui_OpeningFcn', @M_Velocidades_Criticas_OpeningFcn, ... 
        'gui_OutputFcn', @M_Velocidades_Criticas_OutputFcn, ... 
        'gui_LayoutFcn', [], ... 
        'gui_Callback', []);

    if nargin & isstr(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargout
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_State.gui_Callback(gui_State(gui_Singleton~0),...
            gui_State(:, gui_Singleton~0));
    end
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

end
% End initialization code - DO NOT EDIT

% --- Executes just before M_Velocidades_Criticas is made visible.
function M_Velocidades_Criticas_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to M_Velocidades_Criticas (see VARARGIN)

% Choose default command line output for M_Velocidades_Criticas
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes M_Velocidades_Criticas wait for user response (see UIRESUME)
% uihat(handles.figure);

% --- Outputs from this function are returned to the command line.
function varargout = M_Velocidades_Criticas_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

scnsize = get(0,'ScreenSize');
posl = [scnsize(3)/12.6,scnsize(3)/9,scnsize(4)/20];
set(hObject, 'Position',posl, 'Visible','on');

set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
set(hObject, 'Position', [position(1:2) info.Width+100 info.Height+100]);
axes(handles.background);
image(handles.banner);
set(handles.background, ...
  'Visible', 'off', ... 
  'Units', 'pixels', ...
  'Position', [0 0 10*info.Width 10*info.Height]);

axes(handles.axes1);
run('Dados');

wrnpm=w*n*60/(2*pi);
line1=line([0 16000],[wrnpm(2) wrnpm(2)],'Color','b');
line2=line([0 16000],[wrnpm(3) wrnpm(3)],'Color','m');
line3=line([0 16000],[wrnpm(4) wrnpm(4)],'Color','g');
line4=line([0 16000],[wrnpm(5) wrnpm(5)],'Color','k');
if nrGL==6
  line5=line([0 16000],[wrnpm(6) wrnpm(6)],'Color','c');
end

line5=line([0 16000],[0 16000]);
set(line5, 'color', 'r', 'linewidth', 2);
xlabel('\Omega a r v o r e [rpm] ');
ylabel('Omega_n [rpm]');
title('Velocidades Criticas');
legend('2a vel critica','3a vel critica','4a vel critica','5a vel critica','Vel da Arvore','-1');
if nrGL==5
  legend('2a vel critica','3a vel critica','4a vel critica','5a vel critica','Vel da Arvore','-1');

66/86 July 2004
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

António Figueiredo

67/86
"Characterization of the Vibration and noise of a mass balance system of engine"

end

% End initialization code - DO NOT EDIT

% --- Executes just before Menu_Calculos_Freq_Naturais is made visible.
function Menu_Calculos_Freq_Naturais_OpeningFcn(hObject, eventdata, handles, varargin)

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin  command line arguments to Menu_Calculos_Freq_Naturais (see VARARGIN)

% Choose default command line output for Menu_Calculos_Freq_Naturais
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Menu_Calculos_Freq_Naturais wait for user response (see UIRESUME)
% uisave(handles.figure);

% --- Outputs from this function are returned to the command line.
function varargout = Menu_Calculos_Freq_Naturais_OutputFcn(hObject, eventdata, handles)

% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% scnsiz e = get(0,'ScreenSize');
pos1 = [scnsiz(3)/12.5,scnsiz(3)/9,scnsiz(4)/15];
set(hObject, 'Position',pos1,'Visible','on');

handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
axes(handles.background);
image(handles.banner);
set(handles.background,...
    'Visible', 'off',...
    'Units', 'pixels',...
    'Position', [0 0 10*info.Width 10*info.Height]);

set(hObject, 'Units', 'pixels');
handles.banner = imread('logo_demeji.jpg'); % Read the image file banner.jpg
info = imfinfo('logo_demeji.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
axes(handles.axes3);
image(handles.banner);
set(handles.axes3,...
    'Visible', 'off',...
    'Units', 'pixels');

% --- Executes during object creation, after setting all properties.
function popupmenu_CreateFcn(hObject, eventdata, handles)

% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
% See ISPC and COMPUTER.
run('Dados');

if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

if nGlu==5
    set(hObject, 'String', {'Frequencia Fundamental', '2ª Frequencia Natural', '3ª Frequencia Natural', '4ª Frequencia Natural', '5ª Frequencia Natural'});
end
if nGlu==6
    set(hObject, 'String', {'Frequencia Fundamental', '2ª Frequencia Natural', '3ª Frequencia Natural', '4ª Frequencia Natural', '5ª Frequencia Natural', '6ª Frequencia Natural'});
end

% --- Executes on selection change in popupmenul.
function popupmenul_Callback(hObject, eventdata, handles)
    % hObject    handle to popupmenul (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    %
    % Hints: contents = get(hObject,'String') returns popupmenul contents as cell array
    %       contents{get(hObject,'Value')} returns selected item from popupmenul

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
    % hObject    handle to pushbutton1 (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

clc;cla;
run('Dados');
popup_sel_index = get(handles.popupmenul, 'Value');
switch popup_sel_index
    case 1
        iGL=1;
    case 2
        iGL=2;
    case 3
        iGL=3;
    case 4
        iGL=4;
    case 5
        iGL=5;
    case 6
        iGL=6;
end
set(handles.text2, 'Visible', 'on', 'String', wn(iGL));
set(handles.text4, 'Visible', 'on', 'String', u(:,iGL));
set(handles.text6, 'Visible', 'on', 'String', wn(iGL)/2/pi);
set(handles.text8, 'Visible', 'on', 'String', (wn(iGL)/2/pi)*60);
set(handles.text1, 'Visible', 'on');
set(handles.text3, 'Visible', 'on');
set(handles.text5, 'Visible', 'on');
set(handles.text7, 'Visible', 'on');
set(handles.text9, 'Visible', 'on');
set(handles.text10, 'Visible', 'on');
set(handles.text11, 'String', 'Frequencia de oscilação do movimento natural de resposta a uma perturbação inicial');
set(handles.text12, 'String', 'Vector modal (solução do sistema homogeneo associado a cada uma das frequencias naturais)');
set(handles.togglebutton1, 'Visible', 'on');
set(handles.togglebutton2, 'Visible', 'on');
x=1:length(GL);
eyo=zeros(GL,GL);
"Characterization of the Vibration and noise of a mass balance system of engine"

eixo(:,1:GL)=0;
valormax=abs(max(fi(:,1GL)));
valormin=min(fi(:,1:GL));
axes(handles.axes5);
plot(x,fi(:,1:GL),x,eixo(:,1:GL),'k','LineWidth',1.5);hold on;grid off;
title('Forma Natural de Vibraçao');
xlabel('Degree of Freedom');
set(gca,'xtick',1:1:GL);

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton1
button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text10,'Visible','on');  % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text10,'Visible','off');  % toggle button is not pressed
end

% --- Executes on button press in togglebutton2.
function togglebutton2_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton2
button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text11,'Visible','on');  % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text11,'Visible','off');  % toggle button is not pressed
end

Menu_Calculos_matriz_Massa.m

function varargout = Menu_Calculos_Matriz_Massa(varargin)
% MENU_CALCULOS_MATRIZ_MASSA M-file for Menu_Calculos_Matriz_Massa.fig
% MENU_CALCULOS_MATRIZ_MASSA, by itself, creates a new
% MENU_CALCULOS_MATRIZ_MASSA or raises the existing
% singleton*.
% H = MENU_CALCULOS_MATRIZ_MASSA returns the handle to a new
% MENU_CALCULOS_MATRIZ_MASSA or the handle to
% the existing singleton*.
% MENU_CALCULOS_MATRIZ_MASSA('CALLBACK',hObject,eventData,handles,...) calls
% the local function named CALLBACK in MENU_CALCULOS_MATRIZ_MASSA.M with the given input
% arguments.
% MENU_CALCULOS_MATRIZ_MASSA('Property','Value',...) creates a new
% MENU_CALCULOS_MATRIZ_MASSA or raises the
% existing singleton*. Starting from the left, property value pairs are
% applied to the GUI before Menu_Calculos_Matriz_Massa_OpeningFcn gets
% called. An
% unrecognized property name or invalid value makes property application
% stop. All inputs are passed to Menu_Calculos_Matriz_Massa_OpeningFcn via
% varargin.
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES
"Characterization of the Vibration and noise of a mass balance system of engine"

% Edit the above text to modify the response to help Menu_Calculos_Matriz_Massa

% Last Modified by GUIDE v2.5 12-May-2004 18:44:49

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', ...filename, ...'
gui_Singleton', gui_Singleton, ...'
gui_OpeningFcn', @Menu_Calculos_Matriz_Massa_OpeningFcn, ...'
gui_OutputFcn', @Menu_Calculos_Matriz_Massa_OutputFcn, ...'
gui_LayoutFcn', [], ...'
gui_Callback', []);
if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    varargin{1:nargout} = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Menu_Calculos_Matriz_Massa is made visible.
function Menu_Calculos_Matriz_Massa_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to Menu_Calculos_Matriz_Massa (see VARARGIN)

% Choose default command line output for Menu_Calculos_Matriz_Massa
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Menu_Calculos_Matriz_Massa wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargin = Menu_Calculos_Matriz_Massa_OutputFcn(hObject, eventdata, handles)
% varargin cell array for returning output args (see VARARGIN);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargin{1} = handles.output;

scnsize = get(0,'ScreenSize');
posl = [scnsize(3)/12.4,scnsize(4)/25,scnsize(3)/8.5,scnsize(4)/30];
set(hObject, 'Position',posl, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = iminfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
axes(handles.background);
image(handles.banner)
set(handles.background, ...
    'Visible', 'off', ...
    'Units', 'pixels',...
"Characterization of the Vibration and noise of a mass balance system of engine"

set(handles.text10,'String','Matriz que agrupa a inércia existente em cada um dos graus de liberdade');

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
  % hObject handle to pushbutton1 (see GCBO)
  % eventdata reserved - to be defined in a future version of MATLAB
  % handles structure with handles and user data (see GUIDATA)
  run('Dados');
  set(handles.text2, 'Visible', 'on','String', MM(:,1));
  set(handles.text3, 'Visible', 'on','String', MM(:,2));
  set(handles.text4, 'Visible', 'on','String', MM(:,3));
  set(handles.text5, 'Visible', 'on','String', MM(:,4));
  set(handles.text6, 'Visible', 'on','String', MM(:,5));
  if nrGL==5
    set(handles.frame1, 'Visible', 'on');
  end
  if nrGL==6
    set(handles.text9, 'Visible', 'on','String', MM(:,6));
    set(handles.frame2, 'Visible', 'on');
  end
  set(handles.text7, 'Visible', 'on');
  set(handles.text8, 'Visible', 'on');

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
  % hObject handle to pushbutton2 (see GCBO)
  % eventdata reserved - to be defined in a future version of MATLAB
  % handles structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
  % hObject handle to togglebutton1 (see GCBO)
  % eventdata reserved - to be defined in a future version of MATLAB
  % handles structure with handles and user data (see GUIDATA)
  % Hint: get(hObject,'Value') returns toggle state of togglebutton1
  button_state = get(hObject,'Value');
  if button_state == get(hObject,'Max')
    set(handles.text10,'Visible','on');  % toggle button is pressed
  elseif button_state == get(hObject,'Min')
    set(handles.text10,'Visible','off');   % toggle button is not pressed
  end

Menu_Calculos_Matriz_MODAL.m

function varargout = Menu_Calculos_Matriz_MODAL(varargin)
  % MENU_CALCULOS_MATRIZ_MODAL M-file for Menu_Calculos_Matriz_MODAL.fig
  % MENU_CALCULOS_MATRIZ_MODAL, by itself, creates a new
  % MENU_CALCULOS_MATRIZ_MODAL or raises the existing
  % singleton.
  %
  % H = MENU_CALCULOS_MATRIZ_MODAL returns the handle to a new
  % MENU_CALCULOS_MATRIZ_MODAL or the handle to
  % the existing singleton*.
  %
  % MENU_CALCULOS_MATRIZ_MODAL('CALLBACK',hObject,eventData,handles,...) calls
  % the local
  % function named CALLBACK in MENU_CALCULOS_MATRIZ_MODAL.M with the given input
  % arguments.
PROJECT FIM-DE-CURSO

'Characterization of the Vibration and noise of a mass balance system of engine'

---

MENU CALCULOS_MATRIZ_MODAL('Property', 'Value', ...) creates a new
MENU CALCULOS_MATRIZ_MODAL or raises the
existing singleton*. Starting from the left, property value pairs are
applied to the GUI before Menu_Calculos_Matriz Modal_OpeningFunction gets
called. An
unrecognized property name or invalid value makes property application
stop. All inputs are passed to Menu_Calculos_Matriz Modal_OpeningFcn via
varargin.

*See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
instance to run (singleton)".

See also: GUIDE, GUIDATA, GUIDATA

Edit the above text to modify the response to help Menu_Calculos_Matriz Modal

Last Modified by GUIDE v2.5 12-May-2004 16:52:28

Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', 'gui_Singleton', ...
                    'gui_OpeningFcn', @Menu_Calculos_Matriz Modal_OpeningFcn, ...
                    'gui_OutputFcn', @Menu_Calculos_Matriz Modal_OpenOutputFcn, ...
                    'gui_Callback', []);
if nargin == 1 && isstr(varargin(1))
    gui_State.gui_Callback = str2func(varargin(1));
end

if nargin == 2
    varargin{1} = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

--- Executes just before Menu_Calculos_Matriz Modal is made visible.
function Menu_Calculos_Matriz Modal_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to Menu_Calculos_Matriz Modal (see VARARGIN)

% Choose default command line output for Menu_Calculos_Matriz Modal
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

%.UIWAIT makes Menu_Calculos_Matriz Modal wait for user response (see UIRESUME)
uiwait(handles.figure1);

--- Outputs from this function are returned to the command line.
function varargout = Menu_Calculos_Matriz Modal_OpenOutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;

António Figueiredo
scnsize = get(0,'ScreenSize');
posl = [scnsize(3)/12.4,scnsize(4)/25,scnsize(3)/8.5,scnsize(4)/30];

set(hObject, 'Position',posl, 'Visible','on');

refresh
set(hObject, 'Units', 'pixels');
handles.banner = imread('fundo.jpg'); % Read the image file banner.jpg
info = imfinfo('fundo.jpg'); % Determine the size of the image file
position = get(hObject, 'Position');
axes(handles.background);
image(handles.banner)
set(handles.background, ...
   'Visible', 'off', ...
   'Units', 'pixels', ...
   'Position', [0 0 10*info.Width 10*info.Height]);
set(handles.text8,'String','Matriz que agrupa os vectores modais ocupando cada um desses vectores uma coluna da matriz');

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
end

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
end

Menu_Calculos_Matriz_Rigidez.m

function varargout = Menu_Calculos_Matriz_Rigidez(varargin)

% MENU_CALCULOS_MATRIZ_RIGIDEZ M-file for Menu_Calculos_Matriz_Rigidez.fig
% MENU_CALCULOS_MATRIZ_RIGIDEZ, by itself, creates a new
% MENU_CALCULOS_MATRIZ_RIGIDEZ or raises the existing
% singleton.*
% 
% H = MENU_CALCULOS_MATRIZ_RIGIDEZ returns the handle to a new
% MENU_CALCULOS_MATRIZ_RIGIDEZ or the handle to
% the existing singleton.*
% 
% MENU_CALCULOS_MATRIZ_RIGIDEZ('CALLBACK',hObject,eventData,handles,...) calls
% the local
% function named CALLBACK in MENU_CALCULOS_MATRIZ_RIGIDEZ.M with the given
% input arguments.
%
"Characterization of the Vibration and noise of a mass balance system of engine"

% MENU_CALCULOS_MATRIZ_RIGIDEZ('Property', 'Value', ...) creates a new
% MENU_CALCULOS_MATRIZ_RIGIDEZ or raises the
% existing singleton. Starting from the left, property value pairs are
% applied to the GUI before Menu_Calcculos_Matriz_Rigidez_OpeningFcn gets
% called. An
% unrecognized property name or invalid value makes property application
% stop. All inputs are passed to Menu_Calcculos_Matriz_Rigidez_OpeningFcn via
% varargin.
% 
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)."
% 
% See also: GUIDE, GUIDATA, GUIHANDLES

% Last Edit the above text to modify the response to help Menu_Calcculos_Matriz_Rigidez

% Last Modified by GUIDE v2.5 13-May-2004 12:16:04

% Begin initialization code - DO NOT EDIT

gui_Singleton = 1;
gui_State = struct('gui_Name',    
    'gui_Singleton', 
    'gui_OpeningFcn', @Menu_Calcculos_Matriz_Rigidez_OpeningFcn, 
    'gui_OutputFcn', @Menu_Calcculos_Matriz_Rigidez_OutputFcn, 
    'gui_LayoutFcn', [], 
    'gui_Callback', []);

if nargin & isstr(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% End initialization code - DO NOT EDIT

% --- Executes just before Menu_Calcculos_Matriz_Rigidez is made visible.
% function Menu_Calcculos_Matriz_Rigidez_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% varargin command line arguments to Menu_Calcculos_Matriz_Rigidez (see VARARGIN)
% 
% Choose default command line output for Menu_Calcculos_Matriz_Rigidez
% handles.output = hObject;
% 
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Menu_Calcculos_Matriz_Rigidez wait for user response (see UIRESUME)
hwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
% function varargout = Menu_Calcculos_Matriz_Rigidez_OutputFcn(hObject, eventdata, handles)
% varargout cell array for returning output args (see VARARGOUT)
% hObject handle to figure
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
% 
% Get default command line output from handles structure
% varargout{1} = handles.output;

%Color: [0.9255 0.9137 0.8471]
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

scnsize = get(0,'ScreenSize');
pos1 = [scnsize(3)/12.4,scnsize(4)/25,scnsize(3)/8.5,scnsize(4)/30];

set(hObject, 'Position',pos1, 'Visible','on');

handles.banner = imread('fundo.jpg') % Read the image file banner.jpg
info = imfinfo('fundo.jpg') % Determine the size of the image file
position = get(hObject);
axes(handles.background);
image(handles.banner)
set(handles.background,...
   'Visible','off',...
   'Units', 'pixels',...
   'Position', [0 0 15*info.Width 15*info.Height]);

set(handles.text10, 'String','.................');

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% data       getappdata(0, 'metricdata');

run('Dados');
set(handles.text2, 'Visible','on','String', k(:,1));
set(handles.text3, 'Visible','on','String', k(:,2));
set(handles.text4, 'Visible','on','String', k(:,3));
set(handles.text5, 'Visible','on','String', k(:,4));
set(handles.text6, 'Visible','on','String', k(:,5));
set(handles.text7, 'Visible','on');
set(handles.text8, 'Visible','on');

if nrGL==5
    set(handles.frame1, 'Visible', 'on');
end
if nrGL==6
    set(handles.text9, 'Visible', 'on','String', k(:,6));
    set(handles.frame2, 'Visible', 'on');
end

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in togglebutton1.
function togglebutton1_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject, 'Value') returns toggle state of togglebutton1

button_state = get(hObject,'Value');
if button_state == get(hObject,'Max')
    set(handles.text10,'Visible','on') % toggle button is pressed
elseif button_state == get(hObject,'Min')
    set(handles.text10,'Visible','off'); % toggle button is not pressed
end

76/86
July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'
F=zeros(GL,1);  
A=1;               %Amplitude do impulso  
% wrot=10000;       % velocidade de rotação  
T=1/(wrot/60);     % Período  
t0=T/10;           % Duração do impulso  
ntsf=30;           
t=linspace(0,5*T,2048);  

% Resposta no tempo a uma solicitação harmônica  
tetah=zeros(GL,length(t),ntsf+1);  % resposta temporal para uma sequência de impulsos  
teta=zeros(GL,length(t));  
TIF=zeros(1,length(t));  

A0=2*A*t0/T;  
TIF=TIF+A0/2;  
Spf=zeros(1,ntsf+1);Spf(1)=A0/2;  % espectro de força  
Spr=zeros(GL,ntsf+1);  
Swh=zeros(1,ntsf+1);  
Ph0=A*t0/T;  
F(1)=Ph0;  
N0=transpose(fi2)*F;  

% RESPOSTA NO TEMPO  
eta=zeros(GL-1,length(t));  
for i=1:GL-1  
etai=[N0(i)/(wn2(i)^2)];  % Coordenadas generalizadas  
end  
tetasm=zeros(GL,length(t),GL-1);  
for i=1:GL-1  
tetasm(:,i)=fi2(:,i)*etai(:,1);  
  tetah(:,i)=tetah(:,i)+tetasm(:,i);  
end  
teta=teta+tetah(:,1);  
Spr(:,1)=transpose(max(transpose(tetah(:,1))));  % Espectro de resposta  

for k=1:ntsf  
  Fh=2*A/(k*Pi)*sin(k*Pi*t0/T);  % Força provocada pelo impulso  
  wh(k)=k^2*Pi/T;  % Frequência  
  Spf(k+1)=abs(Fh);  % Magnitude  
  Swh(k+1)=wh(k);  
  F(1)=Fh;  
  N=transpose(fi2)*F;  

  % RESPOSTA NO TEMPO  
eta=zeros(GL-1,length(t));  
  for i=1:GL-1  
    mz(i,i)=1/((wn2(i)^2+wh(k)^2)+j^2*csi(i)*wn2(i)*wh(k));  % Com amortecimento  
  end  
  amp=mz*N;  
  Spr(:,k+1)=fi2*amp;  

  % Sobreposição modal  
  for i=1:GL  
    tetah(i,:,k+1)=real(Spr(i,k+1)*exp(j*wh(k)*t));  
  end  
  teta=teta+tetah(:,i,k+1);  
end  
figure(5+i)  % Representação Resposta no grau de liberdade i  
plot(t,teta(i,:));  
end  

for i=1:GL  
  
78/86    

July 2004
'Characterization of the Vibration and noise of a mass balance system of engine'

% figure(5000+i);
% subplot(2,1,1);
% stem(Swh/2/pi,Spf);  %Representação da magnitude do impulso em função da frequencia
%
% subplot(2,1,2);
% stem(Swh/2/pi,abs(Spr(i,:))); %Representação da magnitude da resposta em função da frequencia
% end

RespostImpulsoPeriodicoAciclismo.m

%PROJETO DE FIM DE CURSO
%CARACTERIZAÇÃO DA VIBRAÇÃO E RUIDO DE UMA ARVORE DE EQUILIBRAGEM DE MOTOR run('Dados');

%Resposta do sistema(sobreposição modal)
% Nt=M0*cos(W*t);

csi=zeros(GL-1,1);%amortecimentos modais
% close all;

F=zeros(GL,1);  %Amplitude do impulso
A=0.75;

% wrot=1000;  %velocidade de rotação
wac=1/(wac/60);  %periodo
% T=1;
% wseno=pi/tc;

% t0=T/10;  %duração do impulso
ntsf=150;
t=transpose(linspace(0.5*T,1024));
An=zeros(ntsf,1); Bn=zeros(ntsf,1);

%Resposta no tempo a uma solicitação harmônica
teta= zeros(GL,length(t),ntsf+1); %resposta temporal para uma sequencia de impulsos
teta=zeros(GL,length(t));
TIP=zeros(length(t),1);

AO=4*A/pi;
TIP=1/2*A0;

for n=1:ntsf
    An(n)=-4/pi/(1+2*n)*A*cos(n*pi)^2;
    Bn(n)=-4/pi/(1+2*n)*A*sin(n*pi)*cos(n*pi);
end

Spf=zeros(1,ntsf+1); Spf(1)=AO/2; %espectro de força
Spr=zeros(GL,ntsf+1);
Swh=zeros(1,ntsf+1);

Fh0=AO/2;
F(1)=Fh0;
N0=transpose(fi2)*F;

%RESPOSTA NO TEMPO
teta= zeros(GL-1,length(t));
for i=1:GL-1
    teta(i,:)=N0(i)/(wn2(i)^2); %Coordenadas generalizadas
end

tetasm= zeros(GL,length(t),GL-1);
for i=1:GL-1

António Figueiredo
PROJECT FIM-DE-CURSO

"Characterization of the Vibration and noise of a mass balance system of engine"

\[ \text{tetasm}(i) = \text{fi2}(i) \times \text{eta}(i); \]
\[ \text{tetah}(i) = \text{tetah}(i) + \text{tetasm}(i); \]
end

\[ \text{teta} = \text{teta} + \text{tetah}(i); \]
\[ \text{Spr}(i) = \text{transpose}(\max(\text{transpose}(\text{tetah}(i))))); \]
\%espectro de resposta (VER)

for \( k = 1:\text{ntsf} \)
\[ \text{TIP} = \text{TIP} + \text{An}(k) \times \cos(k \times 2 \pi / T \times t) + \text{Bn}(k) \times \sin(k \times 2 \pi / T \times t); \]
\[ \text{wh} = k \times 2 \pi / T; \]
\%Frequencia
\[ \text{Spf}(k+1) = \text{abs}(\text{TIP}(k)); \]
\%Magnitude
\[ \text{Swh}(k+1) = \text{wh}; \]
\[ \text{F}(1) = \text{TIP}(k); \]
\[ \text{N} = \text{transpose}(\text{fi2}) \times \text{F}; \]

\% figure(30)
\% plot(t,TIP,'b'); hold on;
\%
\% RESPOSTA NO TEMPO
\% eta = zeros(GL-1,length(t));
for \( i = 1:\text{GL} \)
\[ \text{mz}(i,1) = 1/((\text{wn}(i)^2 - \text{wh}^2) + j^2 \times \text{csi}(i) \times \text{wn}(i) \times \text{wh}); \]
\% com amortecimento
end
\[ \text{amp} = \text{mz} \times \text{N}; \]
\[ \text{Spr}(i,k+1) = \text{fi2} \times \text{amp}; \]

\% Sobereposição modal
for \( i = 1:\text{GL} \)
\[ \text{tetah}(i,:) = \text{real}(\text{Spr}(i,k+1) \times \exp(j \times \text{wh} \times t)); \]
end
\[ \text{teta} = \text{teta} + \text{tetah}(i,:); \]
end

\% figure(10)
\% stem(abs(\text{An}));
\% figure(20)
\% stem(abs(\text{Bn}));
\% figure(40)
\% plot(t,TIP,'r','LineWidth',2)

%%%%%%%FUNCAO SENO + IMPULSO %%%%%%%%%%%%%%%%%%%%%%%%%%
\[ \text{Ab} = 2; \]
\[ \text{ftb} = \text{Ab} \times \sin(2 \pi / T \times \text{transpose}(t)); \]
\[ \text{ftt} = \text{transpose}(\text{TIP}) + \text{ftb}; \]
\% figure(80)
\% plot(t,ftt)

\% for \( i = 1:\text{GL} \)
\% figure(15+i);
\% plot(t,\text{teta}(i,:));
\% % Representação Resposta no grau de liberdade \text{i}
end
\% for \( i = 1:\text{GL} \)
\% figure(10+i);
\% subplot(2,1,1);
\% stem(\text{Swh}/2/\pi,\text{Spf});
\% % Representação da magnitude do impulso em função da frequência
\% subplot(2,1,2);
\% stem(\text{Swh}/2/\pi,\text{Spr}(i,:));
\% % Representação da magnitude da resposta em função da frequência
end
RespostalImpulsoPeriodicoSeno2.m

run('Dados');

% Resposta do sistema (sobreponto modal)
% Mt=M0*cos(w*t);

csii=zeros(GL-1,1); % amortecimentos modais
% close all;

F=zeros(GL,1);
A=0.75;
% wrot=6305.6; % velocidade de rotação

T=1/(wrot/60); % período
% T=1;
tc=1/11*T; % duração do impulso
wseño=pi/tc;

% t0=T/10; % duração do impulso
ntsf=100;
t=transpose(linspace(0,5*T,1024));
An=zeros(ntsf,1); Bn=zeros(ntsf,1);

% Resposta no tempo a uma solicitação harmonica
tetah=zeros(GL,length(t),ntsf+1); % resposta temporal para uma sequencia de impulsos
teta=zeros(GL,length(t));
TIP=zeros(length(t),1);

A0=4/T*A/pi/tc;
TIP=1/2*A0;

for n=1:ntsf
    An(n)=-4*T*tc/pi/(T+2*n*tc)/(-T+2*n*tc)*A*cos(pi/T*n*tc)^2;
    Bn(n)=-4*T*tc/pi/(T+2*n*tc)/(-T+2*n*tc)*A*sin(pi/T*n*tc)*cos(pi/T*n*tc);
end

Spf=zeros(1,ntsf+1); Spf(1)=A0/2; % espectro de força
Spr=zeros(GL,ntsf+1);
Shw=zeros(1,ntsf+1);

Ph0=A*tc/T;
F(1)=Ph0;
N0=transpose(fi2)*F;

% RESPOSTA NO TEMPO
eta=zeros(GL-1,length(t));
for i=1:GL-1
    eta(i,:)=(N0(i)/(wn2(i)^2)); % Coordenadas generalizadas
end

tetasm=zeros(GL,length(t),GL-1);
for i=1:GL-1
    tetasm(:,i)=fi2(:,i)*eta(i,:);
    tetah(:,i)=tetah(:,i)+tetasm(:,i);
end

teta=teta+tetah(:,1);
Spr(:,1)=transpose(max(transpose(tetah(:,1)))); % espectro de resposta (VER)

for k=1:ntsf
    TIP=TIP+An(k)*cos(k*2*pi/T*t)+Bn(k)*sin(k*2*pi/T*t);
end

António Figueiredo
wh=k^2*pi/T; %Frequencia
Sf(k+1)=abs(TIP(k)); %Magnitude
Swh(k+1)=wh;
F(1)=TIP(k);
N=transpose(fi2)*F;

figure(30)
plot(t,TIP,'b'); hold on;

%RESPOSTA NO TEMPO
eta=zeros(GL-1,length(t));
for i=1:GL-1
mx(i,i)=1/((wn2(i)^2-wh^2)+j*2*csi(i)*wn2(i)*wh); %com amortecimento
end
amp=mz*N;
Spr(:,k+1)=fi2*amp;
%Sobreposicao modal
for i=1:GL
 tetah(:,k+1)=real(Spr(:,k+1)*exp(j*wh*t));
end
teta=teta+tetah(:,k+1);

% figure(10)
% stem(abs(An))
% figure(20)
% stem(abs(Bn))
% figure(40)
% plot(t,TIP,'r','LineWidth',2)
% %%%%%%%%%%%%%%%%%%FUNCAO SENO + IMPULSO %%%%%%%%%%%%%%%%%%%%%%%
Ab=2;
ftb=Ab*sin(2*pi/T*transpose(t));

fft=transpose(TIP)+ftb;
% figure(80)
% plot(t,fft)
% for i=1:GL
% figure(15+i);
% plot(t,teta(i,:));   %Representacao Resposta no grau de liberdade
% for i=1:GL
% figure(10+i);
% subplot(2,1,1);
% stem(Swh/2/pi,Spf); %Representacao da magnitude do impulso em funcao da
% frequencia
% subplot(2,1,2);
% stem(Swh/2/pi,Spr(:,i)); %Representacao da magnitude da resposta em funcao
da frequencia
% end

RespostalImpulsoPeriodicoSenoElImpulso2.m
%PROJECTO DE FIM DE CURSO
%CATEGORIZACAO DA VIBRAO E RUIDO DE UMA ARVORE DE EQUILIBRAMENDE MOTOREn

run('Dados');

%Resposta do sistema(sobreposicao modal)
%M=M0*cos(w*t);

csi=zeros(GL-1,1);%amortecimentos modais
% close all;
F=zeros(GL,1); %Amplitude do impulso
A=0.75; %Amplitude do impulso
wrot=6305.6; %velocidade de rotação
T=1/(wrot/60); %periodo
T=1;
tc=1/11*T; %duração do impulso
weno=pi/tc;
% tc=T/10; %duração do impulso
ntsf=100;
t=transpose(linspace(0,5*T,1024));
An=zeros(ntsf,1); Bn=zeros(ntsf,1);

%Resposta no tempo a uma solicitação harmônica
theta=zeros(GL,length(t),ntsf+1); %resposta temporal para uma sequência de impulsos
teta=zeros(GL,length(t));
TIP=zeros(length(t),1);
Fh=zeros(length(t),1);

A0=4/T*A/pi/tc;
TIP=1/2*A0;
for n=1:ntsf
    An(n)=4*T*pi/(T+2*n*tc)/(T+2*n*tc)*A*cos(pi/T*n*tc).^2;
    Bn(n)=4*T*pi/(T+2*n*tc)/(T+2*n*tc)*A*sin(pi/T*n*tc)*cos(pi/T*n*tc);
end

Spf=zeros(1,ntsf+1);Spf(1)=A0/2; %espectro de força
Srp=zeros(GL,ntsf+1);
Swh=zeros(1,ntsf+1);

Pho=A*tc/T;
F(1)=Pho;
N0=transpose(fi2)*F;
%RESPOSTA NO TEMPO
teta=zeros(GL-1,length(t));
for i=1:GL-1
    teta(i,:)=N0(i)/(wn2(i).^2); %Coordenadas generalizadas
end
teta=teta+teta(:,1);
Spr(:,1)=transpose(max(transpose(teta(:,1)))); %espectro de resposta(VER)

for k=1:ntsf
    Fh=Fh+An(k)*cos(k*2*pi/T*t)+Bn(k)*sin(k*2*pi/T*t);
    wh=k*2*pi/T;
    Spf(k+1)=abs(Fh(k)); %Magnitude
    Swh(k+1)=wh;
    F(1)=Fh(k);
    N=transpose(fi2)*F;
end

figure(30)
plot(t,TIP,'b'); hold on;

%%%%%%%%%%%%%%%%FUNÇÃO SENO + IMPULSO %%%%%%%%%%%%%%%%%%%%%%%
Ab=2;
ftb=Ab*sin(2*pi/T*transpose(t));
TIP=transpose(Fh)+ftb;
figure(80)
plot(t,TIP)
for k=1:ntsf
wh=k^2*pi/T;
F(1)=TIP(k);
N=transpose(fi2)*F;

figure(30)
plot(t,TIP,'b'); hold on;

%RESPOSTA NO TEMPO
eta=zeros(GL-1,length(t));
for i=1:GL-1
mx(i,i)=1/((wn2(i)^2-wh^2)+j*2*csi(i)*wn2(i)*wh); %com amortecimento
end
amp=mx*N;
Spr(:,k+1)=fi2*amp;

%Sobreposicao modal
for i=1:GL
tetah(i,:,k+1)=real(Spr(i,k+1)*exp(j*wh*t));
end
teta=teta+tetah(:,,:,k+1);

figure(10)
stem(abs(An))
figure(20)
stem(abs(Bn))
figure(40)
plot(t,TIP,'r','LineWidth',2)
for i=1:GL
figure(15+i);
plot(t,teta(i,:)); %Representacao Resposta no grau de liberade 1
end
for i=1:GL
figure(10+i);
subplot(2,1,1);
stem(Swh/2/pi,Spf);
%Representacao da magnitude do impulso em funcao da frequencia
%subplot(2,1,2);
stem(Swh/2/pi,Spr(:,:)); %Representacao da magnitude da resposta em funcao da frequencia
end

RespostaTemp.m

%PROJECTO DE FIM DE CURSO
%CARACTERIZACAO DA VIBRACAO E RUIDO DE UMA ARVORE DE EQUILIBRAMG DE MOTOR
run('Dados');

%Resposta do sistema(sobreposicao modal)
%Mt=M0*cos(w*t);

eta0=zeros(GL,1);
etapt0=zeros(GL,1);
etapa0=fi2'*MM'*eta0;
etapt0=fi2'*MM'*etapt0;
csi=zeros(GL-1,1); %amortecimentos modais
N=transpose(fi2)*F;

%Resposta no tempo a uma solicitacao harmonica
% t=0:0.0001:3.5E-3;
t=linspace(0,3.5E-3,1024);
teta=zeros(GL,length(t));
wxcf=0.75*wn(2);

%RESPOSTA NO TEMPO
eta=zeros(GL-1,length(t));
for i=1:GL-1
    beta=wexcft/wn2(i);
    ampl(1)=N(i)/wn2(i)-2*(1/rt((1-beta)^2+(2*csi(i)*beta)^2));
    desf=atan2(2*csi(i)*beta,(1-beta^2));
    eta(i,:)=amplit*cos(wexcft*t-desf);
end

%Sobreposição modal
tetasm=zeros(GL,length(t),GL-1);
ymax=zeros(GL,1);
ymin=zeros(GL,1);
for i=1:GL-1
    tetasm(:,i)=fi2(:,i)*eta(i,:);
    eta=eta+tetasm(:,i);
    for j=1:GL
        ymax(j)=max(teta(j,:))
        ymin(j)=min(teta(j,:));
    end
end
ymax=max(teta(:,:));
ymin=min(teta(:,:));
yaxtotal=max(ymax);
yxntotal=min(ymin);

% figure(1)
% plot(t,eta(1,:),t,eta(2,:),t,eta(3,:),t,eta(4,:),t,eta(5,:));
% legend('Resposta GL 1','Resposta GL 2','Resposta GL 3','Resposta GL 4','Resposta GL 5');
% back=uitable('style','pushbutton','units','normal','position',[0.91 0.5 0.090 0.095],...
% 'string','Menu','callback','close all,Graficos');
% pause;
% for i=1:GL-1
%    plot(t,tetasm(1,:,i));hold on;
% end
% for i=1:GL
%    figure(i+1)
%    plot(t,eta(i,:),t,tetasm(i,:,1),t,tetasm(i,:,2),t,tetasm(i,:,3),t,tetasm(i,:,4);
%    ymax=max(teta(i,:));
%    ymin=min(teta(i,:));
%    xmax=max(t);
%    axis([0 xmax 1.2*ymin 1.2*ymax]);
%    legend('Resposta do Sistema','2°Modo Natural','3°Modo Natural','4°Modo Natural',...
%    '5°Modo Natural','-1');
%    xlabel('tempo [seg]');
%    ylabel('theta(t)');
%    title('Contribuição Modal para a Resposta do Sistema');
%    back=uitable('style','pushbutton','units','normal','position',[0.91 0.5 0.090 0.095],...
%    'string','Menu','callback','close all,Graficos');
%    pause;
% end

RespostaTransiente.m
%PROJECTO DE FIM DE CURSO
%CARACTERIZAÇÃO DA VIBRAÇÃO E RUIDO DE UMA ARVORE DE EQUILIBRAGEM DE MOTOR

run('Dados');

%Resposta Transiente
P0=1;
% t=0:0.0001:3.5E-3;
t=linspace(0,3.5E-3,1024);
tc=8E-4;

António Figueiredo
wi=0.75*wm(2);
theta=zeros(GL-1,length(t));
for i=2:GL
    wn=wm(i);
    ki(i-1)=MM(i,i)*(wn^2);
    ks=ki(i-1);
    for j=1:length(t)
        t(j)=t(j);
        if t(j) <= tc
            theta(i-1,j)=(F0/ks)*(1/(1-(wi/wni)^2))*(sin(wi*t(j)-
                (wi/wni)*sin(wi*t(j)))); % Solicitação seno
        else
            theta(i-1,j)=(F0/ks)*((wni/wi)/(1-(wi/wni)^2))*
                (sin(wi*t(j)+sin(wi*(t(j-
                tc)))); % Solicitação seno
        end
        theta(i-1,j)=(cos(wi*(t(j-tc)))-cos(wi*t(j)))*F0/(wni^2)); % Solicitação
end
end
% for i=1:GL-1
% figure(i)
% plot(t,theta(i,:));grid on; hold on;
% back=uicontrol('style','pushbutton','units','normal','position',[0.91 0.5
0.090 0.095],....
% 'string','Menu','callback','close all,Graficos');
% end