Implement vibration test control system, in MATLAB for National Instrument equipment
**Implemention of vibration test control, in Matlab for National Instruments equipment**

**Abstract (in English)**

Finding non-linearity is a common application of modal testing but in this way, there is a need to control the input signal. Nowadays, commercial data acquisition software is not enough flexible in controlling the applied signals, whereas MATLAB as general software which supports National Instrument makes it possible to run modal test and control input signal via closed-loop controlling. In this work, using MATLAB commands, a modal test is run with a stepped-sine excitation and the input is controlled to achieve desired pure sinusoidal excitation which commonly is used in finding the non-linearity.

**Key Words**

MATLAB, Data Acquisition Toolbox, Modal test, National Instrument, closed loop controlling
Abstract

Finding non-linearity is a common application of modal testing but in this way, there is a need to control the input signal. Nowadays, commercial data acquisition software is not enough flexible in controlling the applied signals, whereas MATLAB as a general software which supports National Instrument makes it possible to run modal test and control input signal via closed-loop controlling. In this work, using MATLAB commands, a modal test is run with a stepped-sine excitation and the input is controlled to achieve desired pure sinusoidal excitation which commonly is used in finding the non-linearity.
Acknowledgement

We heartily appreciate our supervisor, Andreas Linderholt, whose encouragement, guidance and support from the initial to the final level enabled us to develop an understanding of the subject.

Lastly, we offer our regards and blessings to all of those who supported us in any respect during the completion of the project.

Fatemeh Fakhrarian

Ali Nafisi
Table of contents

ABSTRACT ........................................................................................................... III

ACKNOWLEDGEMENT ....................................................................................... IV

TABLE OF CONTENTS .......................................................................................... V

ABRAVIATION ......................................................................................................... 1

1. INTRODUCTION ............................................................................................... 2

1.1 Background ..................................................................................................... 3
    1.1.1 Hardware ................................................................................................. 3
    1.1.2 Software ................................................................................................. 5

1.2 Purpose ........................................................................................................... 6

1.3 Aim .................................................................................................................. 7

1.4 Hypothesis and Limitations .......................................................................... 7

1.5 Reliability, validity and objectivity .................................................................. 7

2. THEORY .............................................................................................................. 8

2-1 Frequency response function ........................................................................ 8

2-2 excitation in modal test .................................................................................. 9

2-3 non-linearity ................................................................................................... 10

2-4 Leakage .......................................................................................................... 11

2-5 Windowing ...................................................................................................... 11

2-6 Data Acquisition Toolbox ............................................................................. 12

2-7 Control system ............................................................................................... 13

3. LITERATURE REVIEW ..................................................................................... 14

4. METHOD ............................................................................................................ 15

4.1 Preparing test system ..................................................................................... 15

4.2 MATLAB commands .................................................................................... 16
4.2.1 Session Architecture: .................................................................................. 16
4.2.2 Counter Channels ....................................................................................... 16
4.2.3 Applied and useful codes: .......................................................................... 17

4.3 Running test system ...................................................................................... 19

5 RESULTS ........................................................................................................ 21

6. ANALYSIS OF THE RESULTS ..................................................................... 26

7. CONCLUSIONS ............................................................................................ 27

8. REFERENCES .................................................................................................. 28

9. APPENDIX A .................................................................................................. 29
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRF</td>
<td>Frequency Response Function</td>
</tr>
<tr>
<td>DAQ</td>
<td>Data Acquisition</td>
</tr>
<tr>
<td>NI</td>
<td>National Instruments</td>
</tr>
<tr>
<td>MDOF</td>
<td>Multi Degree Of Freedom</td>
</tr>
<tr>
<td>SDOF</td>
<td>Single Degree Of Freedom</td>
</tr>
<tr>
<td>cDAQ</td>
<td>compact Data Acquisition</td>
</tr>
</tbody>
</table>
1. Introduction

From the early days, that structural vibration recognized, observations have helped to understand the underlying physics. Vibration measurement is divided into two corresponding aspects of test. First, measuring of responses and sometimes forces, during structure operation; this is known as in situ testing and second, testing structures under controlled condition test. A vibration test for which responses together with controlled forces are measured is known as ‘Modal testing’.

A modal model is defined as a set of natural frequencies, together with their corresponding vibrational mode shapes including their scale factors. In fact in modal testing the response of an excited structure is dominated by one eigenmode or a combination of some of them. [1]

There are some different purposes of modal testing; one is to measure vibration properties to compare the corresponding data to data stemming from finite element or theoretical models. Comparison between test and analytical data can be made within the time, frequency or modal domain. In the most cases, the Eigen modes are compared. In that case, accurate estimates of the natural frequencies, together with their corresponding mode shapes and modal masses are required. Thereby validated model can then be used for prediction of the structural behavior. [2]

For all of modal testing applications, accurate tests are so important; the accuracy affects the results and their corresponding usages for prediction of structural behavior, controlling of the forces and responses and comparison with theoretical methods etc. as far as it goes non-linearity as a phenomenon can be taken into consideration for this purpose which can is regarded as a property of the system as well.

MATLAB is useful to control modal tests. Various excitations and responses can be controlled by a closed loop. Additionally it is possible to improve the result by customizing the test data analysis procedure. As well as optimization for closed-loop control, is also easily made in MATLAB.
1.1 Background

Modal test devices are divided into two different aspects; hardware and software. Different companies develop complicated hardware and provide their related commercial software to make a useful and desired connection between hardware and systems. Figure (1-1) shows a typical modal test setup and its components.

![Modal test setup diagram](image)

**Fig. (1-1) modal testing component (shaker as exciter)**

1.1.1 Hardware

Vibration test hardware components are; [3]

- Source of excitation force (input).

The sources are divided into two major types; contacting and non-contacting. The contacting one remains attached to the structure during the test and excite it continuously. An electrodynamics shaker, together with a signal generator and an amplifier is an example of such a source. The other types are either out of contact with the structure or in contact with it for a short period which impact hammer is an example of this type. [2]

The excitation of a structure can be made in different ways. For a contacting type of exciter, the excitation is continuous and they can be classified as follows; Mechanical exciters, electromagnetic shakers and electrohydroulic shakers.
The mechanical exciter is based on a rotating out of balance mass. It can produce a specific force at different frequencies but, it suffers from low flexibility as well as being hard to control. [2]

Electromagnetic exciter converts an input signal into a magnetic force and then applies it on a coil core. In modal testing, this exciter is the most common one. Frequency and amplitude of an excitation can be controlled independently for this type which this property makes electromagnetic shakers more flexible in operation. [2]

- Force transducers and their associated signal conditioners

The piezoelectric type of transducer is commonly used and it is divided into three types; force gauges, accelerometers and impedance heads (the last one is a combination of a force transducer and an accelerometer. Piezoelectric material generates an electrical charge across its end faces when a mechanical stress is applied to it which can count as a principle for this type of transducers. For instance, a force transducer converts the applied force on the crystal (or a known fraction of it) to a corresponding charge. (Because of the relative stiffness in the crystal the transmitted force will be a fraction of the applied force.) [2]

- Output transducers such as accelerometers and their associated signal conditioners

Accelerometer transducers work by use of an auxiliary or seismic mass so that the force exerted on the crystal is the inertia force of the seismic mass. When the body vibrates, the internal mass applies a force to the crystal. According to Newtown’s law \((F = m.a)\) the output of the transducer is a fraction of the body acceleration and as a result, the structure acceleration to which the transducer is attached.

- Signal analyzer

Each analyzer is a form of voltmeter which can be a simple analogue or digital voltmeter or a computer. Nowadays computers are common analyzers via different kinds of software for data acquisition and analysis. All analyzers contain analog anti aliasing filters.
Aliasing occurs when different sampled signals are indistinguishable or signals are rapidly varied. An Anti-aliasing filter removes undesirable frequencies that lead to erroneous measurement. Figure (1-2) shows simply how two different frequencies of sinusoidal signals can produce the same sampled data. [4]

![Figure (1-2) aliasing in sinusoidal signals](image)

### 1.1.2 Software

Regardless of hardware type, information must be sent to the hardware and received from it. Also, configuration information such as sampling rate is sent to hardware. This information exchange is accomplished by cooperation of two kinds of software; [5]

1. **Driver software**

   Driver Software makes it possible to access and control the capabilities of the hardware. For instance driver software allows the data to be sent and gained from the board (NI cDAQ-9178 in this work) or to control the rate at which data are acquired.

2. **Application software**

   Application software provides a convenient front end to the driver software for generating events and managing stored data in a computer memory. For instance by using the MATLAB Data Acquisition Toolbox together with National Instrument hardware and NI-DAQ driver, the following relationship can be implied. [5]
LabView is one out of much DAQ software which made by National Instruments and it is being used in many vibrational tests in both industrial and academic fields. This software only connects to NI hardware.

Nevertheless, MATLAB is general software for a broad variety of experiments with widely desired flexibilities of the excitation in modal testing. To be able to control the measurement in an almost unlimited way, MATLAB is preferably used in this field.

Furthermore by MATLAB, comparing to the LabView, can make closed loop control easier. Closed-loop is important in generating desired excitation such as stepped sine, which is a pure sinusoidal excitation, with fixed amplitude.

Additionally, implementing user defined windowing (the most practical solution for leakage problem) would be far easier by using MATLAB. In the next chapter, leakage is explained more.

### 1.2 Purpose

The goal of this thesis is to develop a MATLAB code that controls a modal test from the beginning to the end. This code should make a bridge between National Instruments (NI) hardware, the test structure and the computer.

The Data Acquisition Toolbox has several commands to run the hardware including the commands for recognizing the hardware, introducing them to MATLAB, sending signals to output and acquiring the signals from the input. In chapter 2, there is more information regarding to these commands.
Additionally for accurate test, a closed-loop control is considered. Hence making code for measuring the applied forces and minimizing the difference between it and desired force constitutes the next part in this thesis.

### 1.3 Aim

Using Data Acquisition Toolbox and MATLAB commands, to control the hardware, acquire initial data and finally modify the output data.

In case of nonlinear systems, it is desirable to be able to control the force or acceleration amplitudes. To realize this purpose, a closed-loop control made by using MATLAB commands is required. In this step, some methods such as Newtonian method will be used to minimize the difference. i.e. the optimization of the input signal.

Overly there is a practical experiment to examine pursued theoretical issues to made codes.

### 1.4 Hypothesis and Limitations

The limitation of this thesis task can be classified in following different aspects;

- **Hardware**

  Gauges and measuring instruments have some sensitivity and accuracy and they will have some inevitable errors.

  Human mistake in preparing the test should also be taken into account. Wrong directions and inappropriate installation are some examples of these errors.

### 1.5 Reliability, validity and objectivity

Written MATLAB codes in this thesis are only used for Natural Instruments hardware.
2. Theory

2-1 Frequency response function

Generally, the equation of motion for systems can be expressed in the following way:

\[
[M]\ddot{x} + [C]\dot{x} + [K]x = \{f\} \quad (2-1)
\]

in which

[M]: mass matrix in MDOF (or mass value in SDOF)
[C]: damping matrix in MDOF (or damping value in SDOF)
[K]: stiffness matrix in MDOF (or stiffness value in SDOF)
{f}: force vector in MDOF (or force value in SDOF)

Within stepped-sine (defined in next section) testing, one excitation frequency at the time is used; \( f(t) = F_e e^{i\omega t} \). When steady state conditions are reached the force and accelerations are measured whereby the frequency response function, \( H_{jk} \), can be established:

\[
H_{jk}(\omega) = \frac{X}{F} = \sum_{r=1}^{N} \frac{(\phi_{jr}^T \phi_{kr})}{m_r(\omega^2_r - \Omega^2 + 2i\xi_r \omega_r \Omega)} \quad (2-2)
\]

In which

\( m_r \): Modal mass
\( \omega_r \): Natural frequency
\( \Omega \): Excitation frequency
\( \xi_r \): Relative damping factor
\( \phi_{j} \): Mode shape component of the degree of freedom j
\( \phi_k \): Mode shape component of the measured degree of freedom \( k \)

\( r \): mode number

\( N \): total number of modes included

Generally for an MDOF damped system, equation (2-2) is visualized in various formats. Some common plots are Nyquist plot and separated real and imaginary plot as functions of frequency. These three plots are projection of a three dimensional plot, see figure (2-2)

![3D FRF Plot](image)

Figure (2-1); three dimensional plot of an FRF for a sample of an MDOF system [1, page 133]

In the imaginary-frequency and the real-frequency projection peak points indicate resonances.

**2-2 excitation in modal test**

A structure can be excited by different stimuli such as harmonic signals like stepped-sine excitation or periodic signal like multi-sine excitation, transient signal like impulse and stochastic signal like random. The most considered difference between them is spectral energy content and test duration.
A stepped-sine signal is a discrete sinusoid with fixed amplitude and frequency and it is called ‘‘standard excitation’’. [2]. In order to excite frequency range of interest the signal is stepped from one discrete value to another to provide the enough points in the FRF.

For response measurement in this type of excitation, it is necessary to wait for a while in order to steady- state condition would be attained, when a new frequency is applied. Otherwise, the transient part of the response would affect the data.

Using stepped-sine excitation the signals from the force transducers will be distorted especially closed to resonance frequencies. If a linear structure is being excited by a simple monochromatic signal such as a pure sine or cosine with frequency \( \Omega \) then the response will be monochromatic with the same frequency. Therefore, one useful application of stepped-sine excitation can be finding nonlinearities. [6, 7] By the exciting with different force levels, nonlinearities may be detected.

### 2-3 non-linearity

Probably all of practical engineering structures are nonlinear to some extent. The nonlinearity is the result of one or a combination of different factors like friction, boundary conditions which impose different stiffness constraints or material properties. There are some approaches to deal with the problems caused by nonlinearity in theory or practice; nevertheless it is crucial to find out whether the structure behavior is linear or nonlinear in advance to define validity of the linear equations boundaries.

In an electromagnetic exciter, the applied force on the structure is converted to the coil movement. Hence the applied force is approximately the difference between generated force and inertia force (the coil mass) therefore, it is smaller than them. Moreover, the applied force would be the reaction between the shaker and structure, thus it is better to measure the force applied to structure before its reaction with shaker (the contact surface on it). [2]

Since the nonlinear relation between the output voltage signal and the output force signal is unknown, there are some numerical methods to iterate and find the correct voltage signal. Closed-loop controlling measures the applied force before its application to the
structure and the output voltage will be improved so that the desired force is achieved. Some of iteration methods will be discussed in chapter four of this thesis.

2-4 Leakage

In the case of not properly treating of digital Fourier analysis, erroneous result may occur.[1] The reasons may be aliasing, leakage and windowing. In the section of analyzer, aliasing is briefly discussed.

Leakage refers to the finite length of the time history in periodic excitation and in the case when the cycle of the sine waves is not complete, the spectrum produced does not indicate the single excitation frequency. Therefore, energy has leaked into a number of spectral lines close to the true frequency and the spectrum will be spread over several lines.

Leakage is a problem in measurement dealing with FRF although that there are some ways to avoid it like changing the duration of sampling, capturing the exact number of cycles of signal or adding zeroes to the end of measured samples. [2]

2-5 Windowing

As it mentioned before, windowing is one of the most practical way for solving leakage problem. Winowing refers to some functions or profiles which impose on the time signal before performing Fourier Transform; and thereby improving the spectrum of the result. Figure (2-2) shows the effect of applying the window function.

Figure (2-2) Effect of window function on discrete Fourier transforms [2]
There are different types of windowing function (Figure (2-3)) that are applied to various excitations and in all of them, windowing function makes a rescaling to compensate of attenuation of the signals.[6]

![Figure (2-3) different types of windowing function. (a) Boxcar; (b) Hanning; (c) Cosine-taper; (d) Exponential [2]](image)

### 2-6 Data Acquisition Toolbox

Data Acquisition Toolbox can be used to customize the acquisitions, access the built-in features of hardware devices, and incorporate the analysis and visualization features of MATLAB and related toolboxes into design. Data can be analyzed, visualized and saved for post-processing. Data Acquisition Toolbox makes it possible to apply MATLAB as a single, integrated environment to support the entire data acquisition, data analysis, and application development process.

MATLAB Data Acquisition Toolbox supports M-Series, E-Series, Compact DAQ, USB, and many other types of data acquisition hardware from National Instruments.

Especially, the Data Acquisition Toolbox supports National Instruments data acquisition hardware that uses either the NI-DAQmx or traditional NI-DAQ driver software. MATLAB
supports NI-DAQmx version 8.7 or higher and Traditional NI-DAQ version 7.3 using this toolbox.

### 2-7 Control system

Regarding with a nonlinear system, it has to be able to make undistorted sine wave input in order to get a reliable result. Creating pure stepped-sine excitation is possible by different methods but they are based on development of control systems, i.e. numerical optimization methods to find the correct output voltage.

Since the relationship between the input signal and the output signal is nonlinear and unknown some numerical methods are required to iterate and to find the correct voltage signal such as Newton’s method for minimization. [8]

Also there are two other iterative methods to solve the problem [9]

- A standard Newton-Raphson solver
- A quasi-Newton solver, Broyden’s method

Unlike the first method, the second one avoids the direct computation of the Jacobian matrix and only has linear convergence towards the correct solution. However, both of these methods need an initial close guess for convergence.

The Jacobian matrix is determined after selecting desired force amplitude and frequency and setting the voltage. Using Jacobian matrix and iteration the new voltage vector is calculated until the solution has converged. These steps are used for closed-loop control and in the next section the related MATLAB commands will be shown.
3. Literature review

Martin Magnevall, Andreas Josefsson and Kjell Ahlin [9] used mathworks in MATLAB to examine their algorithm of controlling the input systems. However, the hardware of testing is not mentioned in their articles. The other used references also worked more on results than MATLAB.

In this work, the focus is on the data acquisition toolbox in MATLAB v.2011a (R17) to make a bridge between the hardware and the test structure using National Instrument hardware.
4. Method

4.1 Preparing test system

The test system in this thesis consists of the following

1. Chassis (cDAQ-9178); as a chassis on which input or output devices are installed and moreover, the power of input module and output module is also supplies by the chassis.

2. Input module (NI 9234)

3. Output module (NI 9263)

4. Amplifier; since the voltage output from the NI hardware is not large enough to move the shaker an amplifier is used between shaker and output module.

5. The shaker and the force transducer; those are connected to each other and they apply the force signal on the test structure via a stringer. In the next part, the required MATLAB commands for running the system will be introduced. Figure (4-1) shows the test system installed in this thesis.

Fig (4-1) test system
4.2 MATLAB commands

Since the applied instruments in this thesis made by National Instrument Company are capable to be installed on a special chassis (cDAQ-9178), it is not possible to apply the ordinary Data Acquisition Toolbox codes. Therefore, Session-Based Interface related codes are used for programming.

4.2.1 Session Architecture:

The Session-based interface uses a session object that contains; [5]

1. Analog input channels and properties
   - Counter input channels and properties
   - Analog output channels and properties
   - Counter input channels and properties
   - Session properties

4.2.2 Counter Channels

Besides regular channels, counter ones are another kind of channels which can be introduced in the Session interface method. It has been tried to provide sufficient information and in a brief way in the following way [5]

   - Count events and pulses
   - Measure frequency
   - Generate pulses
   - Use hardware clock to measure pulses and frequency at specified intervals
4.2.3 Applied and useful codes:

1. Discovering hardware Devices; [5]

*(daq.getDevices)* to find out if the hardware has been installed and recognized by the computer. In order to have more information concerning the recognized instruments such as their rate range, supported channels, measurement type, counter properties etc, the row number of each device can be called by this code

2. Create a session;

*(s = daq.createSession('vendor'))* to create a session it is required to use vendor name into the session creation code which it is obvious between apostrophe, additionally created session must be labelled by an optional name which in the example is *s*. furthermore after creating the session MATLAB program automatically display some data concerning the created session like the rate of measurement per second, duration and type of measurement which can be defined as being continuous or not continuous [5]

3. Adding Channels to the Session;

There are two kinds of way to introduce channels based on the type of instrument (output or input) in the following way

- *daq.Session.addAnalogInputChannel*
- *daq.Session.addAnalogOutputChannel*

As it is clear for specifying an input channel firs command would be useful and for an output channel the second one

4. Change the rate; (s.Rate)

5. Start operation;

There are two ways to start generating and acquiring data with the aid of session-interface codes in the following ways;
• Foreground; *(s.startForeground());* this way of operation would block MATLAB until the operation completes. In an acquiring data case, this command would result in waiting for the entire acquisition process to complete then it would execute your next command. For a generating case it operates exactly the same.

• Background; *(s.startBackground());* if data acquisition or generation starts in this way there would an opportunity to continue working in the MATLAB Commands Window together with processing the data. In this method, it is required to use listener an event which will be explained afterwards.

6. Addlistener and Event;

   *(lh = addlistener ('eventName', @callback)) as it can be seen the background acquisition and generation is totally dependent upon events and listeners. Since it would be above this thesis material to explain concerning the mechanism of the event and listener working in MATLAB, Data Acquisition related material will be explained as following.

7. For the event names;

   There are three options based on the operator or program requirements are used a) *'DataAvailable'* b) *'DataRequired'* c) *'ErrorOccured'*

   In the Callback section the desired function with the basis of application of the program is introduced or created, in addition to that it is possible to use anonymous function in the case that function file should be waived.

8. QueueOutputData:

   *(s.queueOutputData (data)*, with the aid of this command the data which is desired to generate by the instrument through different channels must be queued after each other. It might be important to know that in Foreground method of operation this command can determine the operating duration by queue more data after each other using loop or different matrices. Nevertheless if Background method is taken into consideration after determining
whether the process is continuous or not, there is no need to queue more data since the instrument automatically generates queued data continuously till operator stop it.

9. Continuity:

_IsContinuous_, it will define if the session operation will execute continuously until it stops by using the stop command. To set the session operation as a continuous process it must be set as true and if not it must be set as false.

10. Sensitivity of Accelerometer:

_s.channels.Sensitivity_, this command will change the default accelerometer sensitivity to any desired level of sensitivity

11. Examine a Session:

To test if a session is still operating or it has stopped following commands will be useful particularly when process is continuous and Background method is applying, because it cannot be determined from the MATLAB Command Window if it is still working.

- _Isdone(): to see if the operation of session has finished_
- _IsRunning(): to see if the operation of session is still executing_
- _Islogging (): Determine whether analog input object is logging data_

There is a m-file content at the appendix A which could show how the mentioned codes in a combination with MATLAB code has been used to execute and accomplish the reasonable results.

4.3 Running test system

Although the MATLAB m-file gives the best explanation of how the system concerned in this thesis operates; the procedure is hereby described.

By using the commands, mentioned in the previous section, firstly five sinusoidal voltage signals having frequencies 10, 20, 30, 40 and 50 Hz generated; each has a 5 second duration. Signals are made and sent by the output module (NI 9263). The generated voltage signal goes to amplifier to be increased and sent to a shaker.
The excitation force is measured by the force transducer. The transducer is connected between the stringer and the structure. The data are transmitted to the input module (NI9234).

After a delay, enough to reach a steady state condition, ten complete cycles of the force and the accelerations are captured. This is repeated for each excitation frequency.
5 Results

In this thesis, the vibration measurements have been made with five different frequencies sinusoidal excitations. Figure (5-1) shows the output sinusoidal excitation signals.

Fig (5-1) output sinusoidal excitation signals

A force transducer measures the forces exciting the structure. The results from the force transducer are shown in Figure (5-2) to Figure (5-6) each plot refers to the output voltage with its associated frequency (same color).
Fig (5-2) result from force transducer for 10 Hz frequency

Fig (5-3) result from force transducer for 20 Hz frequency

Fig (5-4) result from force transducer for 30 Hz frequency
The plate is excited by the generated force (sec 4.3), accelerometer measures the acceleration of the see Figure (5-7) to Figure (5-11).

This result besides the acceleration result is analysed in the next chapter.
Fig (5-7) Acceleration of plate with 10 Hz frequency excitation

Fig (5-8) Acceleration of plate with 20 Hz excitation

Fig (5-9) Acceleration of plate with 30 Hz excitation
Fig (5-10) Acceleration of plate with 40 Hz excitation

Fig (5-11) Acceleration of plate with 50 Hz excitation
6. Analysis of the results

As shown in chapter 5, the stepped-sine excitation is made by five different signals (figure (5-1)). Therefore, the shaker vibrates first at 10 Hz for 5 seconds, the excitation frequency steps up to 50 Hz.

Next, the force transducer gives the force applied to the system Fig(5-2 to 5-6). In this test a converter is used to link the amplifier to the shaker, therefore the results are not actually the force signals.

In the first acceleration plot (Fig (5-7)) some distortions in early cycles is appeared due to inappropriate plate supporting.

Finally, the FRF is calculated and plotted by use of MATLAB. Although the number of frequencies are is small, a clear resonance is depicted, see Figure (5-12).

Fig (5-12), FRF of test experiment
7. Conclusions

Many of the companies working in the data acquisition fields produce accurate instruments (hardware) and offer easy-to-use programs (software). Therefore, LabView as software has been embedded into National Instrument company package besides the hardware.

Nevertheless, due to the increasingly varied usage of data acquisition instruments in the current industries and academic researches, the specific software might not satisfy customers’ requirements entirely and cover all matters under various circumstances. In other words, these kinds of software might not flexible enough.

Hence, in this thesis the task was to prepare for closed-loop controlling with the aid of the more general but more flexible software MATLAB.
8. References

4- http://en.wikipedia.org/wiki/Aliasing
5- 2011. Data Acquisition Toolbox. 2011a ed
12-identifying and quantifying structural nonlinearities in engineering application from measured frequency response functions
9. Appendix A

% To clarify and close previous information, matrices and figures
clear all
close all
clc
% To Create a session
D=daq.createSession('ni')
% To choose whether operation is continuous
D.IsContinuous=false
% To Introduce and define input and output channels to the MATLAB with the basis of their connection
D.addAnalogInputChannel('cDAQ1Mod1',0,'Voltage')
D.addAnalogInputChannel('cDAQ1Mod1',1,'Accelerometer')
D.addAnalogOutputChannel('cDAQ1Mod3',0,'Voltage')
% To define the rate of measurement for each second
D.Rate=2048;
% To define the used accelerometer sensitivity
D.Channels(2).Sensitivity = 0.988

% To Specify the empty matrices which will make the main matrices. These matrices are just for helping us and they will not have any effects on the result
A_f=[];
A_a=[];
ym_f=[];
Yest=[];
Acq_new=[];
ym_a=[];

% Loop of Measuring
% i represents the numbers of different frequency with which shaker must impose on the structure
for i=1:5
    % fi is the frequency value for each loop
    fi=10*i;
    % j represents the duration for operating
    for j=1:5
        Output=2*sin(linspace(0,fi*2*pi,2048))';
        D.queueOutputData(Output)
    end

% Acq is the acquired data and Time is a matrices which shows the time of operation
[Acq,Time]=D.startForeground();

% Interpolation
a=Acq(3*2048:round((3*2048)+(10*2048/fi))-1,:)
b=Time(1:length(a));
bi=0:10/(500*fi):10/fi;
ai = interp1(b,a,bi)
ai(end,:)=[]
bi(end)=[]
% New matrix of data with numbers of point ordered by interpolation
Acq_new=[Acq_new,ai]
t=linspace(0,500/2048,500)'

% Estimated matrix
Yest=[Yest;cos((2*pi*fi).*t),sin((2*pi*fi).*t)]
r_f=[1,3,5,7,9,11,13,15,17,19]

% Measured matrix by Force Transducer
ym_f=[ym_f;Acq_new(:,r_f(i))]
r_a=[2,4,6,8,10,12,14,16,18,20]

% Measured matrix by Accelerometer
ym_a=[ym_a;Acq_new(:,r_a(i))]

% Initial calculation to gain the FRF matrix
a_f=Yest((i-1)*500+1:500*i,:),ym_f((i-1)*500+1:500*i,:)
A_f=[A_f,a_f]
a_a=Yest((i-1)*500+1:500*i,:),ym_a((i-1)*500+1:500*i,:)
A_a=[A_a,a_a]

% Figure of Output data
figure(1)
subplot(5,1,i); plot(Output,"r") , grid
on,xlabel('Rate/Second'),ylabel('Output Voltage'), title('Output Voltage per Second'),

% Figure of gained data by Force Transducer
figure(2)
subplot(5,1,i); plot(b,a(:,1)), grid
on,xlabel('Time'),ylabel('Force'), title('Transducer Data')

% Figure of data gained by Accelerometer
figure(3)
subplot(5,1,i); plot(b,a(:,2)), grid
on,xlabel('Time'),ylabel('Acceleration'), title('Accelerometer Data')
end

% Final calculation to have FRF matrix
% Calculation of amplitudes
F_A_a=(A_a).^2
A=sqrt(sum(F_A_a))
F_A_f=A_f.^2
F=sqrt(sum(F_A_f))

% FRF matrix
FRF=A ./ F

% Figure of FRF matrix
figure(4)
plot(FRF,'--s','MarkerSize',5,'MarkerEdgeColor','k','MarkerFaceColor','g'),
grid on ,title(' FRF')