EASTERN MEDITERRANEAN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF MECHANICAL ENGINEERING

Course Code:	**MENG331**
Course Name:	**MECHANICAL VIBRATIONS**
Semester/ Year:	**Fall 2025-26**
Instructor:	**Prof. Dr. Qasim Zeeshan**
Assistant:	**Mr. Emmanuel Chukwueloka Onyibo**
	Mr. Osinachi Mbah
Assessment:	**Labs**
Submission Date:	**30 Dec 2025**

LAB PROJECT # 1: Modeling of a Discretized Lumped Parameter Spring Mass Damper System
LAB PROJECT # 2: Damped and Undamped Free vibration response
LAB PROJECT # 3: Frequency-Response Analysis of MIMO System
LAB PROJECT # 4: Mode shape analysis of cantilever beam with ANSYS
LAB PROJECT # 5: Modal Analysis of a Flexible Flying Wing Aircraft
LAB PROJECT # 6: Modal Analysis of a Simulated System and a Wind Turbine Blade
LAB PROJECT # 7: Balancing of Machines: Static and Dynamic Balancing
LAB PROJECT # 8: Vibration Analysis of Rotating Machinery

GROUP NO. : .....
STUDENT NO: ..... NAME, SURNAME: .....
STUDENT NO: ..... NAME, SURNAME: .....
STUDENT NO: ..... NAME, SURNAME: .....
STUDENT NO: ..... NAME, SURNAME: .....
STUDENT NO: ..... NAME, SURNAME: .....

SECTIONS:

No.	STUDENT OUTCOMES	COURSE LEARNING OUTCOMES	WEIGHT OF SECTION /100	MARKS OBTAINED	REMARKS
L-1	6	1, 2	2.5%		
L-2	6	5	2.5%		
L-3	6	6	2.5%		
L-4	6	6	2.5%		
L-5	6	6	2.5%		
L-6	6	6	2.5%		
L-7	6	8	2.5%		
L-8	6	8	2.5%		
TOTAL:(Out of 100)					

Course Learning Outcomes		Student Outcomes						
		1	2	3	4	5	6	7
1	Model and analyze of dynamic systems as discretized lumped parameters						●	
2	Perform Vibration analysis of Single DOF systems						●	
3	Understand Harmonically excited vibration							
4	Perform Vibration analysis of Multi DOF systems							
5	Perform Vibration analysis of damped systems						●	
6	Determine Natural Frequencies and Mode Shapes						●	
7	Understand Vibration control techniques							
8	Understand Balancing of machinery						●	

Instructions:

1. Late submissions will be penalized with 10 marks/ day.
2. Draw neat, labeled diagrams where necessary.
3. Write relevant equations and write your Assumptions where necessary.
4. Be clear and specific and include units. Give explanatory notes where necessary. Please carry out each step explicitly.
5. Please provide the references using APA Style of Referencing.
6. References should be from the Textbook or from an authentic source.
7. Please include the equations, diagrams, plots and figures in the report.
8. All figures and Tables must be numbered and captioned.
9. Please submit the pdf Soft copy of the report via the MOODLE LMS/ Microsoft Assignment along with the MATLAB codes, Simulink. AVI File. Code/ Simulation etc.
10. The file/ folder uploaded should be named as [MENG331-PROJECT-GROUP NO]. Files with any other name or format will be disregarded. Do not forget the cover page Files with any other name or format will be disregarded. Do not forget the cover page.

LAB PROJECT # 1:
Modeling of a Discretized Lumped Parameter Spring Mass Damper System

Model the Single Degree of Freedom Discretized Lumped Parameter System as Second Order differential Equation in MATLAB, Simulink and Simscape.

You may use ODE45 variable step solver to solve the Second Order differential Equation.

1. **Choose any arbitrary values of Mass, Stiffness, and Damping, and plot the Amplitude, Velocity and Acceleration vs Time.**
2. **Analyze the effect of change of Mass by plotting the results.**
3. **Analyze the effect of change of Stiffness by plotting the results.**
4. **Analyze the effect of change of Damping by plotting the results.**
5. **Match the excitation force frequency with natural frequency and plot the resonance graphs.**

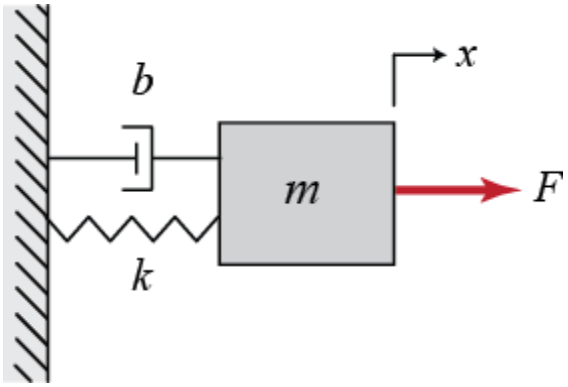
Mechanical Systems

Newton's laws of motion form the basis for analyzing mechanical systems. **Newton's second law**, Equation (1), states that the sum of the forces acting on a body equals the product of its mass and acceleration. **Newton's third law**, for our purposes, states that if two bodies are in contact, then they experience the same magnitude contact force, just acting in opposite directions.

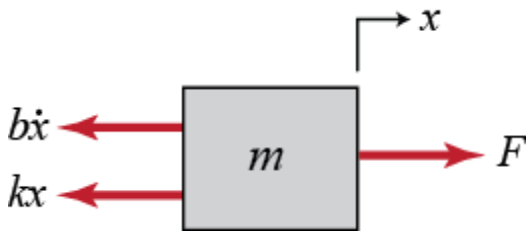
$$\Sigma \mathbf{F} = m\mathbf{a} = m\frac{d^2\mathbf{x}}{dt^2} \qquad (1)$$

When applying this equation, it is best to construct a **free-body diagram (FBD)** of the sysetm showing all of the applied forces.

Example: Mass-Spring-Damper System



The free-body diagram for this system is shown below. The spring force is proportional to the displacement of the mass, x, and the viscous damping force is proportional to the velocity of the mass, $v = \dot{x}$. Both forces oppose the motion of the mass and are, therefore, shown in the negative x-direction. Note also that $x = 0$corresponds to the position of the mass when the spring is unstretched.



Now we proceed by summing the forces and applying Newton's second law, Equation (1), in each direction. In this case, there are no forces acting in the y-direction; however, in the x-direction we have:

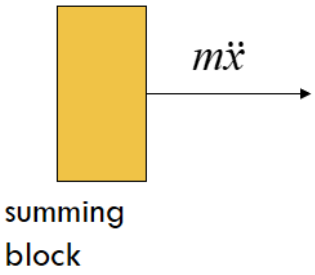
$$\Sigma F_x = F(t) - b\dot{x} - kx = m\ddot{x} \qquad (2)$$

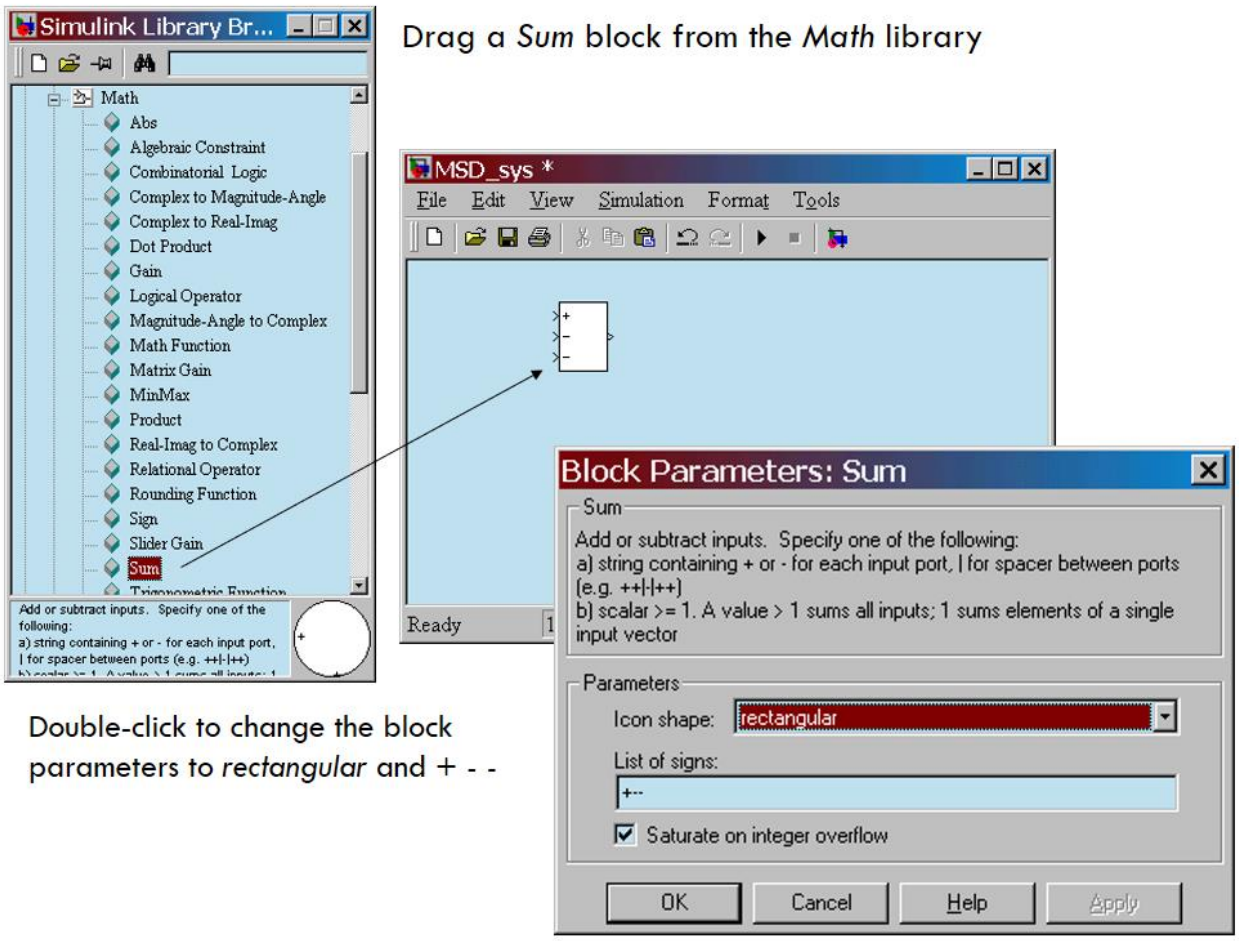
This equation, known as the **governing equation**, completely characterizes the dynamic state of the system. Later, we will see how to use this to calculate the response of the system to any external input, $F(t)$, as well as to analyze system properties such as stability and performance.

- First, solve for the term with highest-order derivative

$$m\ddot{x} = f(t) - c\dot{x} - kx$$

- Make the left-hand side of this equation the output of a summing block

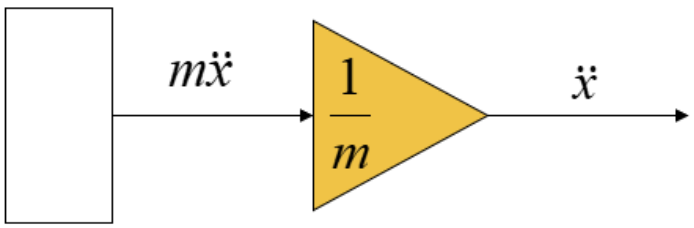




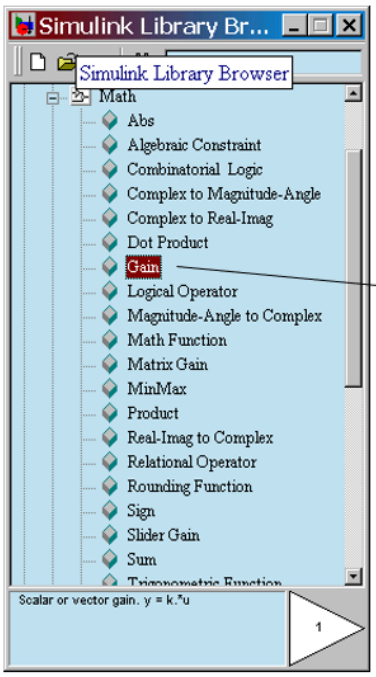
Drag a *Sum* block from the *Math* library

Double-click to change the block parameters to *rectangular* and + - -

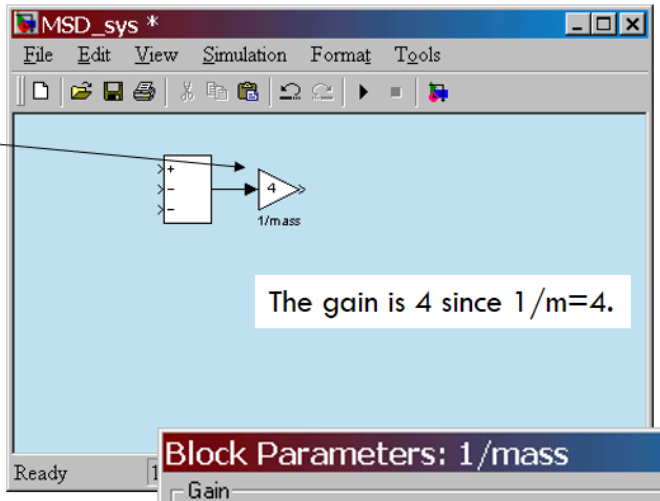
- Add a gain (multiplier) block to eliminate the coefficient and produce the highest-derivative alone



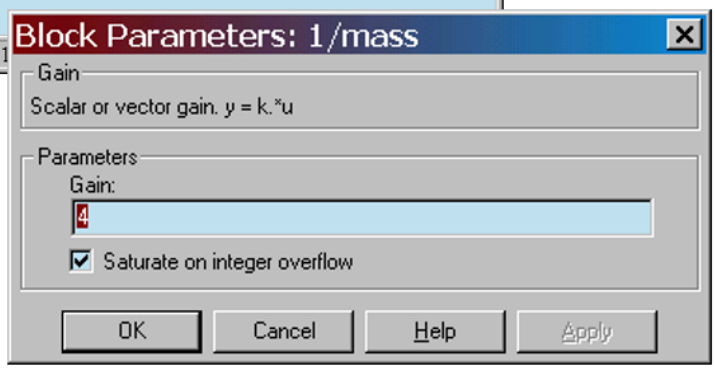
summing block



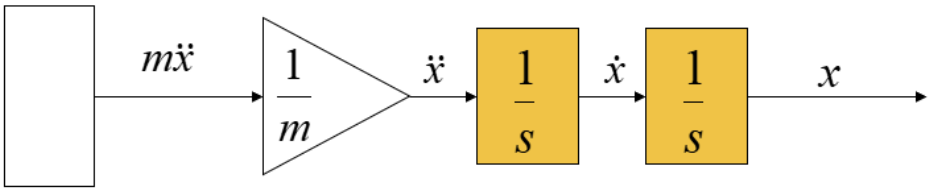
Drag a *Gain* block from the *Math* library



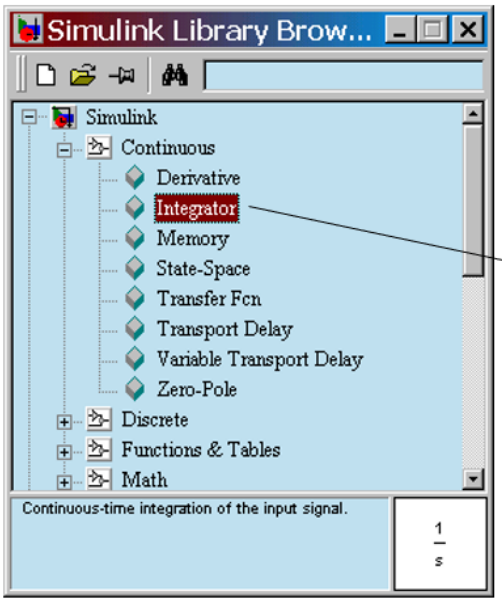
Double-click to change the block parameters.
Add a title.



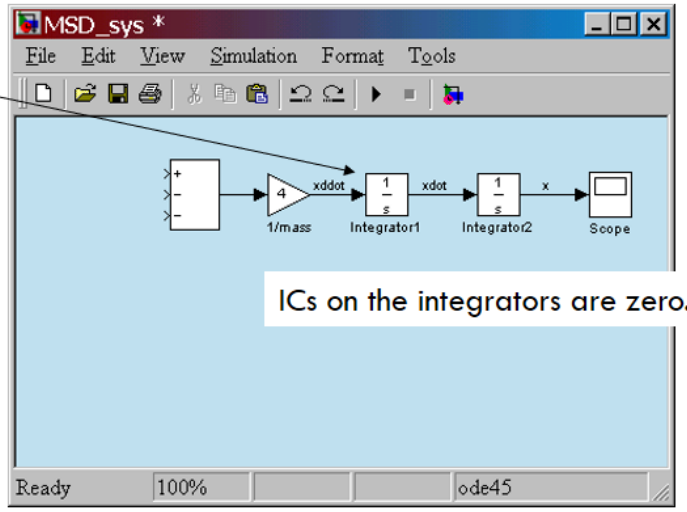
- Add integrators to obtain the desired output variable



summing block



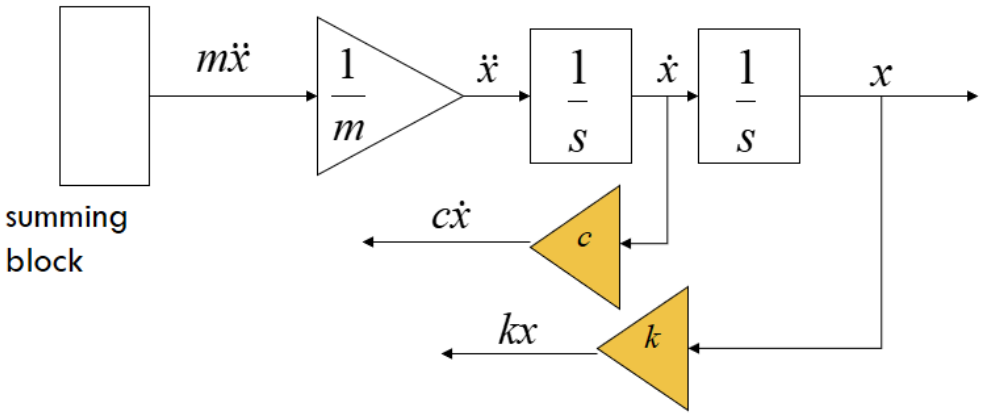
Drag *Integrator* blocks from the *Continuous* library

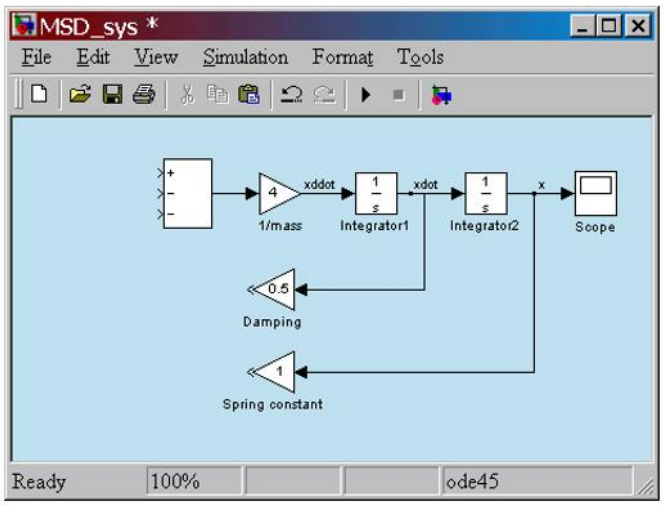


ICs on the integrators are zero.

Add a scope from the *Sinks* library.
Connect output ports to input ports.
Label the signals by double-clicking on the leader line.

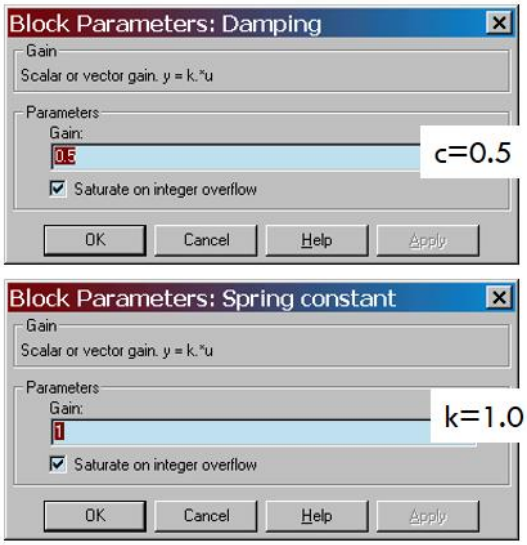
- Connect to the integrated signals with gain blocks to create the terms on the right-hand side of the EOM





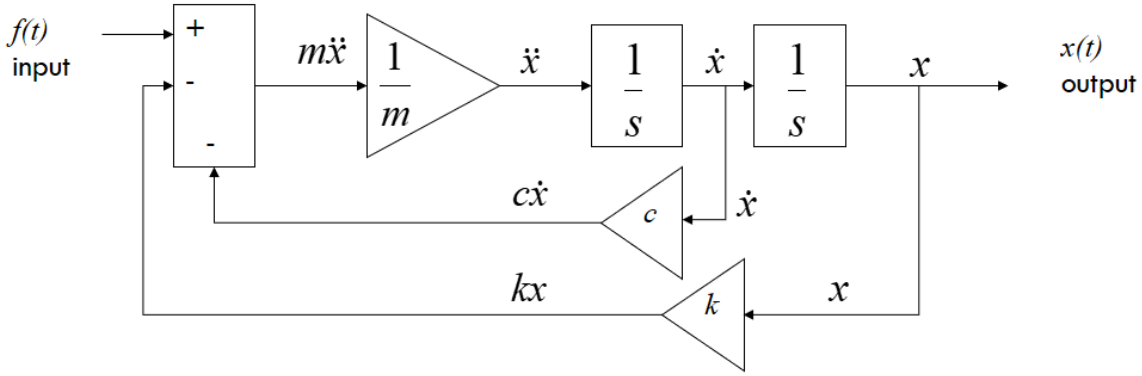
Drag new *Gain* blocks from the *Math* library

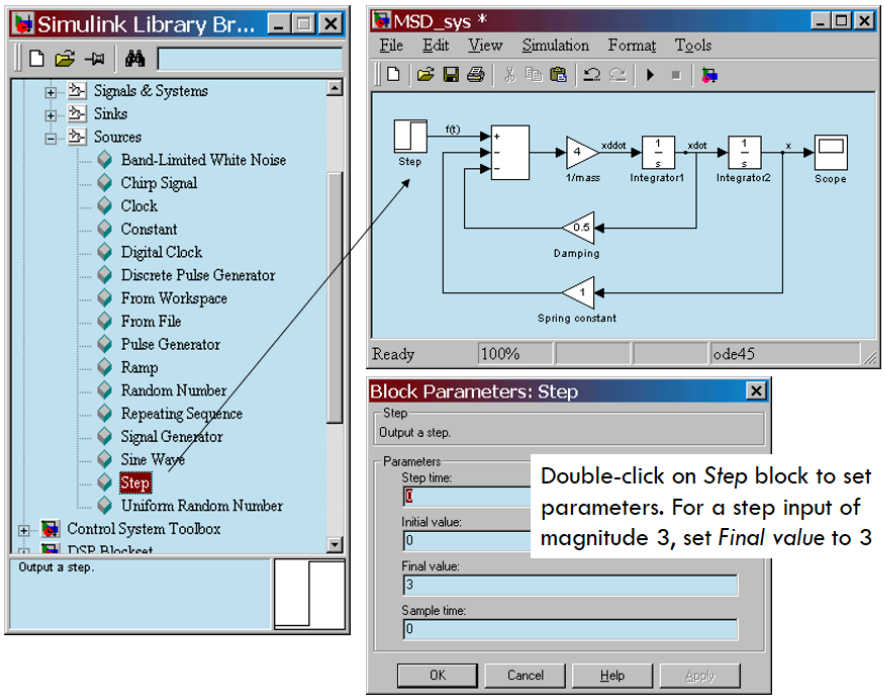
To flip the gain block, select it and choose *Flip Block* in the *Format* pull-down menu.

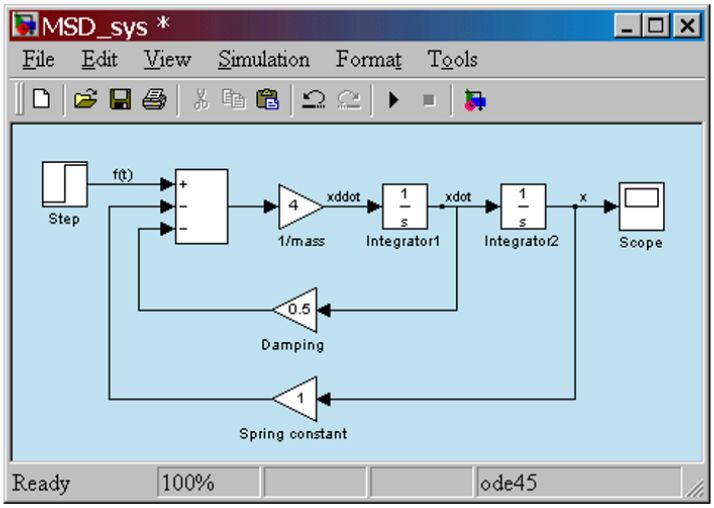


- Double-click on gain blocks to set parameters
- Connect from the gain block input backwards up to the branch point.
- Re-title the gain blocks.

- Bring all the signals and inputs to the summing block.
- Check signs on the summer.







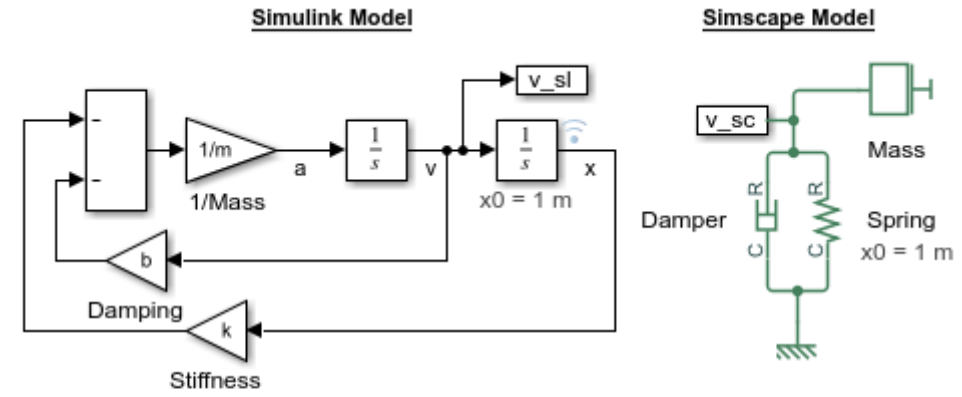
Mass-Spring-Damper in Simulink and Simscape

This example shows two models of a mass-spring-damper, one using Simulink® input/output blocks and one using Simscape™ physical networks.

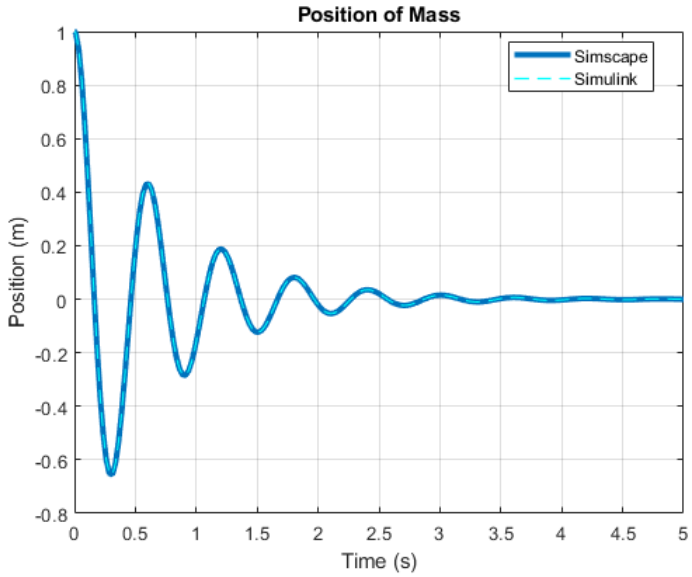
The Simulink model uses signal connections, which define how data flows from one block to another. The Simscape model uses physical connections, which permit a bidirectional flow of energy between components. Physical connections make it possible to add further stages to the mass-spring-damper simply by using copy and paste. Input/output connections require rederiving and reimplementing the equations.

The initial deflection for the spring is 1 meter. This is shown in the block annotations for the Spring and one of the Integrator blocks.

Model



Simulation Results from Simscape Logging




```
ssc_mass_spring_damper_sl
```

LAB PROJECT # 2:
Damped and Undamped Free vibration response

APPARATUS: TM16 Universal Vibration Apparatus

Picture shows the Frame and Cupboard (TM16a) and the forced vibration experiment

The TM16 series is a range of products that teach different aspects of vibrations and oscillations in mechanical systems. These include pendulums, mass-springs systems and shafts and beams. The TM16 series is a modular system, based around a Frame and Cupboard (TM16a).

Objective

To determine the fundamental concepts of free vibration:

- Damped
- Undamped

Apparatus

TM16 Range Universal Vibration Apparatus

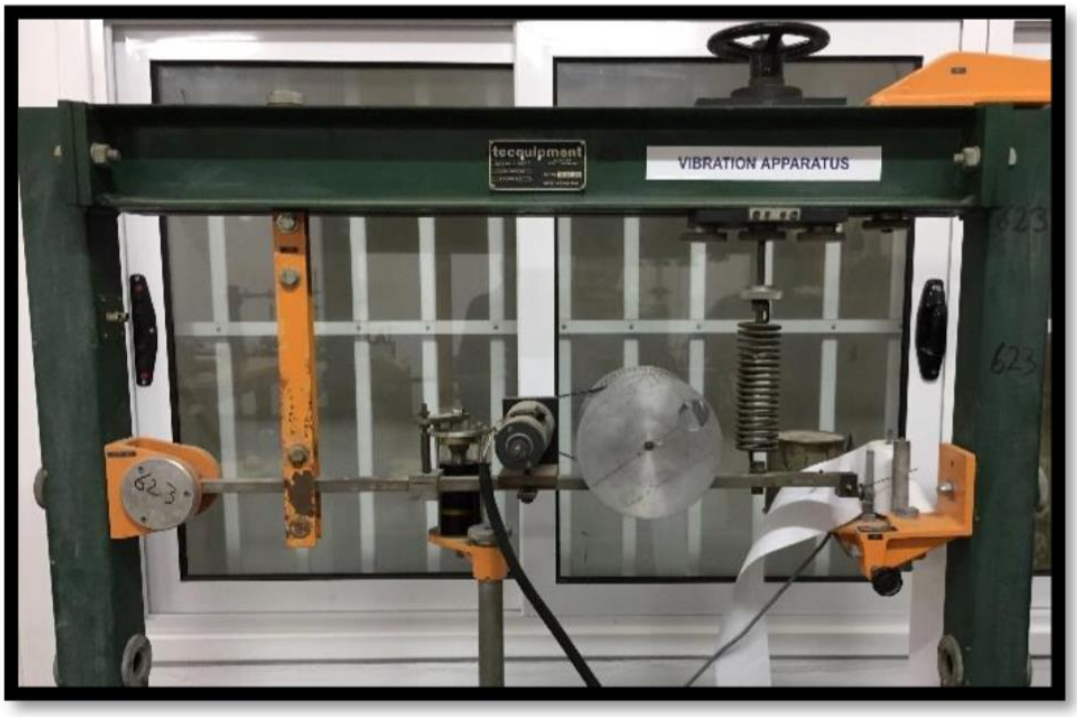


Figure 1: TM16 Universal Vibration Apparatus

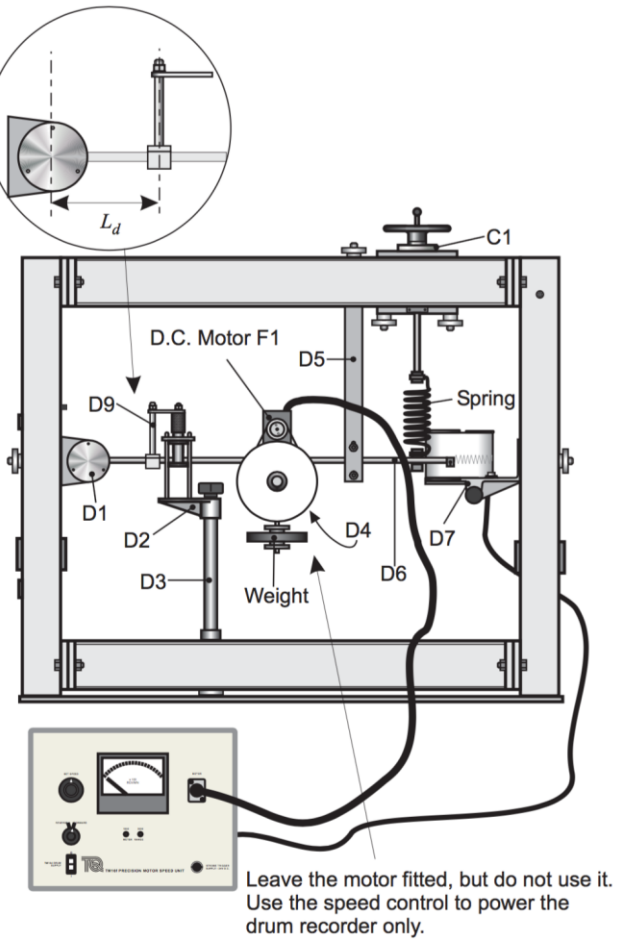
Introduction

A vibration is the movement of a physical quantity in relation to a reference location in a cyclically increasing and decreasing manner as a function of time. During vibrations, energy is dissipated and so a steady amplitude cannot be maintained without continuous replacement. Viscous damping in which force is proportional to velocity affords the simplest mathematical treatment. A convenient means of measuring the amount of damping present is to measure the rate of decay of oscillation. This is expressed by the term 'logarithmic decrement', which is defined as the natural logarithm of the ratio of successive amplitudes on the same side of the mean position.

There are two types of vibration: (1) free vibration and (2) forced vibration. Free Vibration: A system is left to vibrate on its own after an initial disturbance and no external force acts on the system. Forced Vibration: A system that is subjected to a repeating external force e.g., oscillation arises from motor.

Mechanical vibration classifications: (1) Damped Vibration: When any energy is lost or dissipated in friction or other resistance during oscillations, and (2) Undamped Vibration: When no energy is lost or dissipated in friction or other resistance during oscillations, finally (3) Resonance: It occurs when the frequency of the external force coincides with one of the natural frequencies of the system.

Parts needed form the equipment



Part	Number
Top loading assembly	C1
Coiled spring	
Mounting	D1
Dashpot	D2
Out of balance Discs	D4
Dashpot bracket	D3
Safety stop	D5
Bar and stylus	D6
Chart recorder	D7
Beam clamp	D9
Rectangular section bar	E6
DC Motor	F1
400 g Weights	

Figure 2: Setup of the experiment

- The exciter motor (F1) is not going to be used (but leave it fitted), free vibrations will be used.
- To drive the chart recorder (D7) you need the Speed Control unit.
- The system is set vibrating freely by pulling down on the free end of the beam a short distance (15 – 25 mm) and releasing.
- Use the chart recorder to obtain a trace of just three successive amplitudes on the same side of the mean position
- **FOR DAMPED:** Vary the damping by moving the dashpot (D2) and its clamps along the beam, and also by relative rotation of the two orifice plates in the dashpot to increase or decrease the effective area of the piston
- **FOR UNDAMPED:** Disengaged the damper and repeat above steps.

Procedure

- Connect the lead from the motor of recorder unit D7 to the auxiliary supply socket on the Speed Control unit.
- Switch on the speed control unit.
- Set the dashpot at distance L_d (the distance from the trunnion mounting to the center of the beam clamp D9)
- Bring the recording pen into contact with the paper to produce a trace of the decaying amplitude of vibration and thus produce a trace of the decaying applied amplitude on the chart recorder paper.
- Pull the beam down a short distance, under the point of attachment of the spring, and release.
- Start observing the system, its noise and from the chart recorder.
- Gradually start increasing the speed from the motor speed control unit and observe.

QUESTIONS:

Briefly explain/ discuss the following:

1. **What type of damping are present in the system?**
2. **Why is damping considered only in the neighborhood of resonance in most cases?**
3. **How do you find the response of a viscously damped system under rotating unbalance?**
4. **The equation of motion of a machine (rotating at frequency ω) of mass M, with an unbalanced mass m, at radius e, is given by**
5. **What happens to the response of an undamped system at resonance?**
6. **Is it possible to completely avoid resonance of a damped during start up and stopping of a rotating machinery**
7. **Is it possible to reduce the amplitude of resonance during start up and stopping of a rotating machinery**
8. **The damping force depends on the frequency of the applied force in the case of a. viscous-damping b. Coulomb damping c. hysteresis damping? And Why, explain**
9. **Is it possible to find the approximate value of the amplitude of a damped forced vibration without considering damping at all? If so, under what circumstances?**
10. **Is dry friction effective in limiting the reasonant amplitude?**

LAB PROJECT # 3:
Frequency-Response Analysis of MIMO System

AIM:

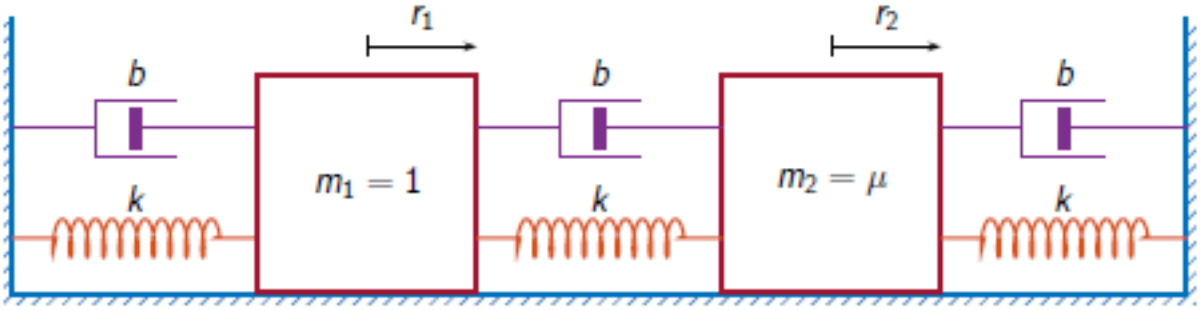
To perform frequency-response analysis based on the space-state model of a mass-spring-damper oscillator MIMO system.

INTRODUCTION

This example shows how to estimate the multi-input/multi-output (MIMO) transfer function for a two-body oscillator. This example starts by computing the system inputs and outputs in the time domain, emulating measurements. After applying the definition of the Z-transform to calculate the frequency-domain transfer function of the system, the example uses the functions tfestimate, modalfrf, and modalfitto estimate the transfer function from the simulated measurement data. The comparison between the outputs of these functions shows how you can retrieve the transfer function from system input/output data.

Two-Body Oscillating System

An ideal one-dimensional discrete-time oscillating system consists of two masses, m_1 and m_2, confined between two walls. The units are such that m_1=1 kg and m_2=μ kg. Each mass is attached to the nearest wall by a spring with elastic constant k (in N/m). An identical spring connects the two masses. Three dampers impede the motion of the masses by exerting on them forces proportional to speed, with damping constant b (in kg/s). Two sensors sample r_1 and r_2 (in m), the displacements of the masses, at a rate F_s=40 Hz.



Generate 24,000 time samples, equivalent to 10 minutes. Define the sampling interval Δt=1/F_s.

```
Fs = 40;
dt = 1/Fs;
N = 24000;
t = dt*(0:N-1);
```

The system can be described by the state-space model

$$x(k+1)=Ax(k)+Bu(k),$$

$$y(k)=Cx(k)+Du(k),$$

where $x=[r_1 v_1 r_2 v_2]^T$ is the state vector, r_i and v_i are respectively the location and the velocity of the ith mass, $u=[u_1 u_2]^T$ is the vector of input driving forces, and $y=[r_1 r_2]^T$ is the output vector. The state-space matrices are

$$A = \exp(A_c \Delta t), \quad B = A_c^{-1}(A - I)B_c, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

I is the 4×4 identity matrix, and the continuous-time state-space matrices are

$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2k & -2b & k & b \\ 0 & 0 & 0 & 1 \\ k/\mu & b/\mu & -2k/\mu & -2b/\mu \end{bmatrix}, \quad B_c = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1/\mu \end{bmatrix}.$$

Set k=400 N/m, b=0.1 kg/s, and μ=0.1.

```
k = 400;
b = 0.1;
mu = 0.1;

Ac = [0 1 0 0;-2*k -2*b k b;0 0 0 1;k/mu b/mu -2*k/mu -2*b/mu];
A = expm(Ac*dt);
Bc = [0 0;1 0;0 0;0 1/mu];
B = Ac\(A-eye(4))*Bc;
C = [1 0 0 0;0 0 1 0];
D = zeros(2);
```

A random input drives the masses for the first 390 seconds and then the masses are left to rest. Set the initial conditions of location and velocity to zero. Use the state-space model to compute the time evolution of the system. Plot the displacements of the masses as a function of time.

```
rng("default")
u = randn(2,N);
u(:,t>390) = 0;

y = [0;0];
x = [0;0;0;0];
for iter = 1:N
   y(:,iter) = C*x + D*u(:,iter);
   x = A*x + B*u(:,iter);
end
```

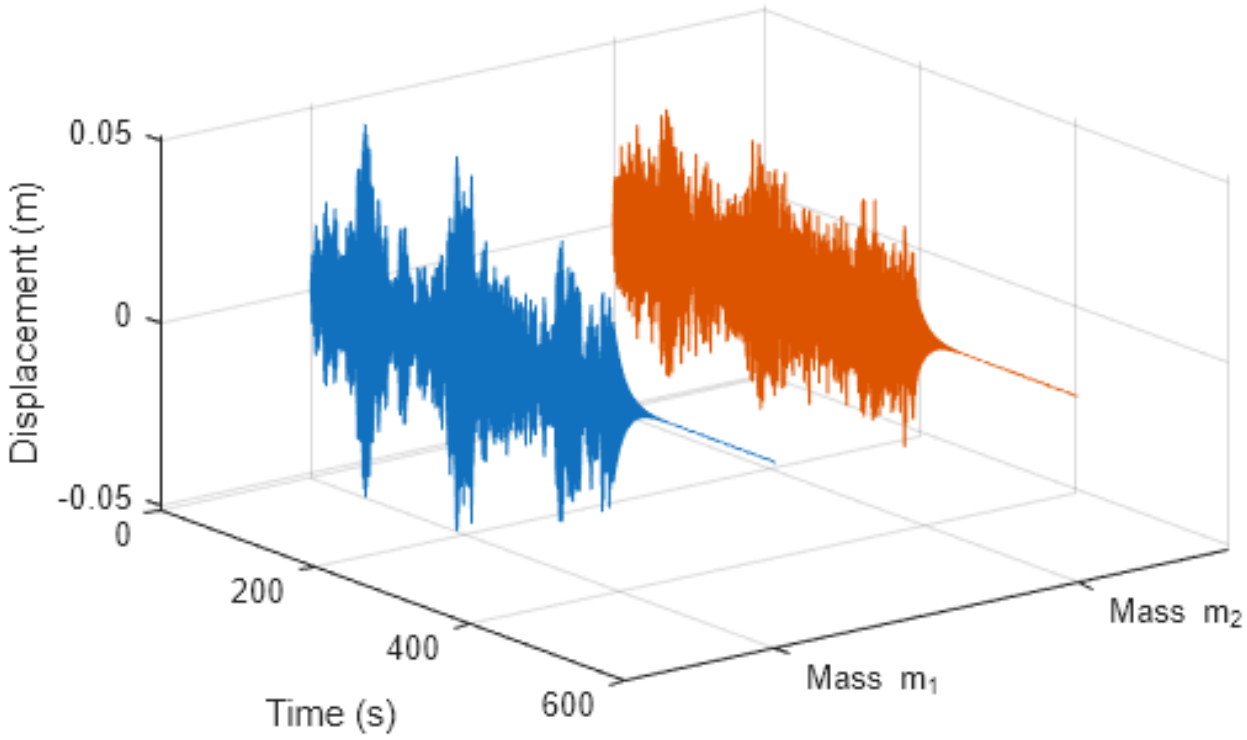
Set the system time, input, and response data to vertical vector format.

```
t = t';
u = u';
y = y';
```

Plot the system response over time for the two masses.

```
plot3([1 2].*ones(size(t)),t,y)
set(gca,Ydir="reverse")
```

```
xticks([1 2])
xticklabels("Mass m_"+[1 2])
ylabel("Time (s)")
zlabel("Displacement (m)")
xlim([0.5 2.5])
grid
```



Save the system data. The examples Frequency-Response Function of Two-Body Oscillator, Modal Analysis of Two-Body Oscillator, and Transfer Function Estimation of MIMO System use this data file.

```
save MIMOdata
```

Theoretical Frequency Response

The frequency-response function of a discrete-time system is the Fourier transform of its time-domain transfer function or, equivalently, the Z-transform of the transfer function evaluated at the unit circle.

Compute the theoretical frequency-response functions. Use 2048 sampling points to calculate the discrete Fourier transform.

```
[b1,a1] = ss2tf(A,B,C,D,1);
[b2,a2] = ss2tf(A,B,C,D,2);


nfs = 2048;
fz = 0:Fs/nfs:Fs/2;
ztf(1,:,1) = freqz(b1(1,:),a1,fz,Fs);
ztf(1,:,2) = freqz(b1(2,:),a1,fz,Fs);
```

```
ztf(2,:,1) = freqz(b2(1,:),a2,fz,Fs);
ztf(2,:,2) = freqz(b2(2,:),a2,fz,Fs);
```

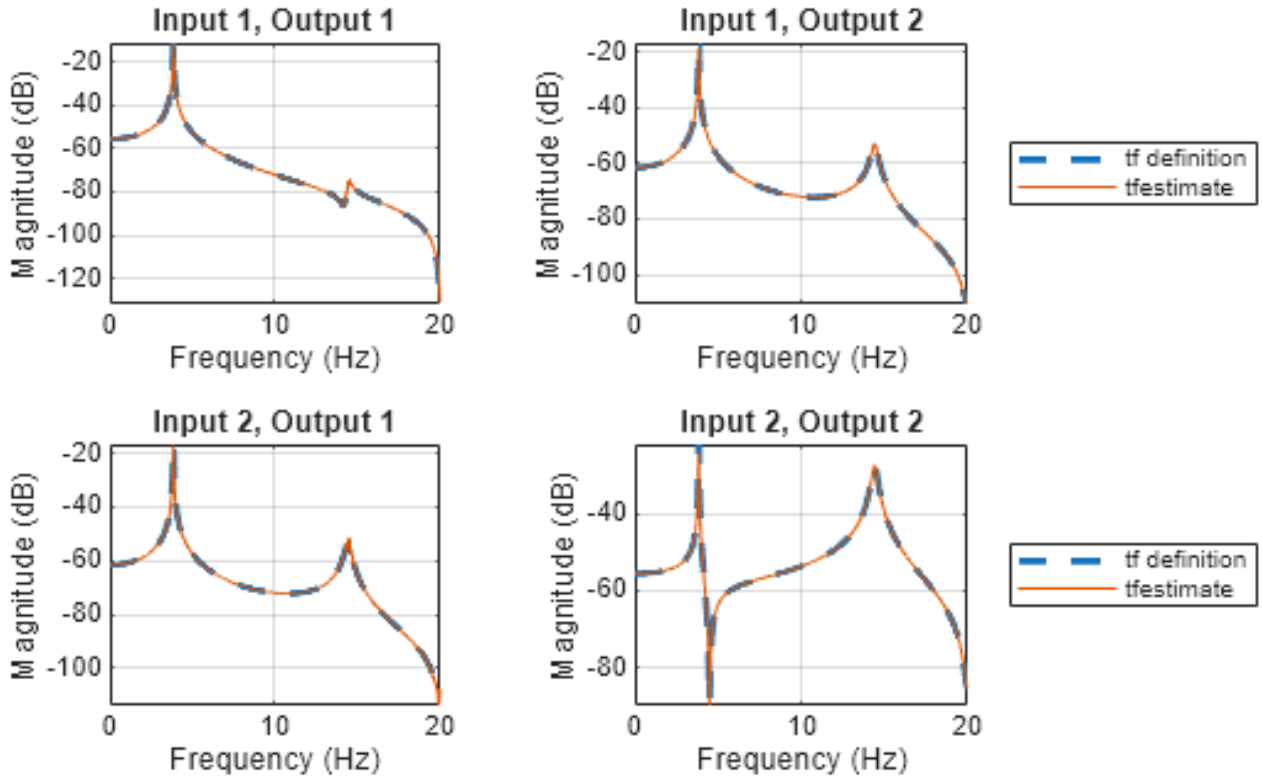
Transfer Function Estimate using tfestimate

Use the system data and the function tfestimate without output arguments to plot the estimate of the MIMO transfer functions. Select the "mimo" option to produce all four transfer functions. Use a periodic 5000-sample Hann window to divide the signals into segments. Specify 2500 samples of overlap between adjoining segments. Store the transfer function of the system as a function of frequency.

```
wind = hann(5000,"periodic");
nov = 2500;


[tXY,ft] = tfestimate(u,y,wind,nov,nfs,Fs,"mimo");
```

Verify that the estimate computed by tfestimate coincides with the definition.

```
plotComparison(fz,ztf,ft,tXY,"tfestimate")
```



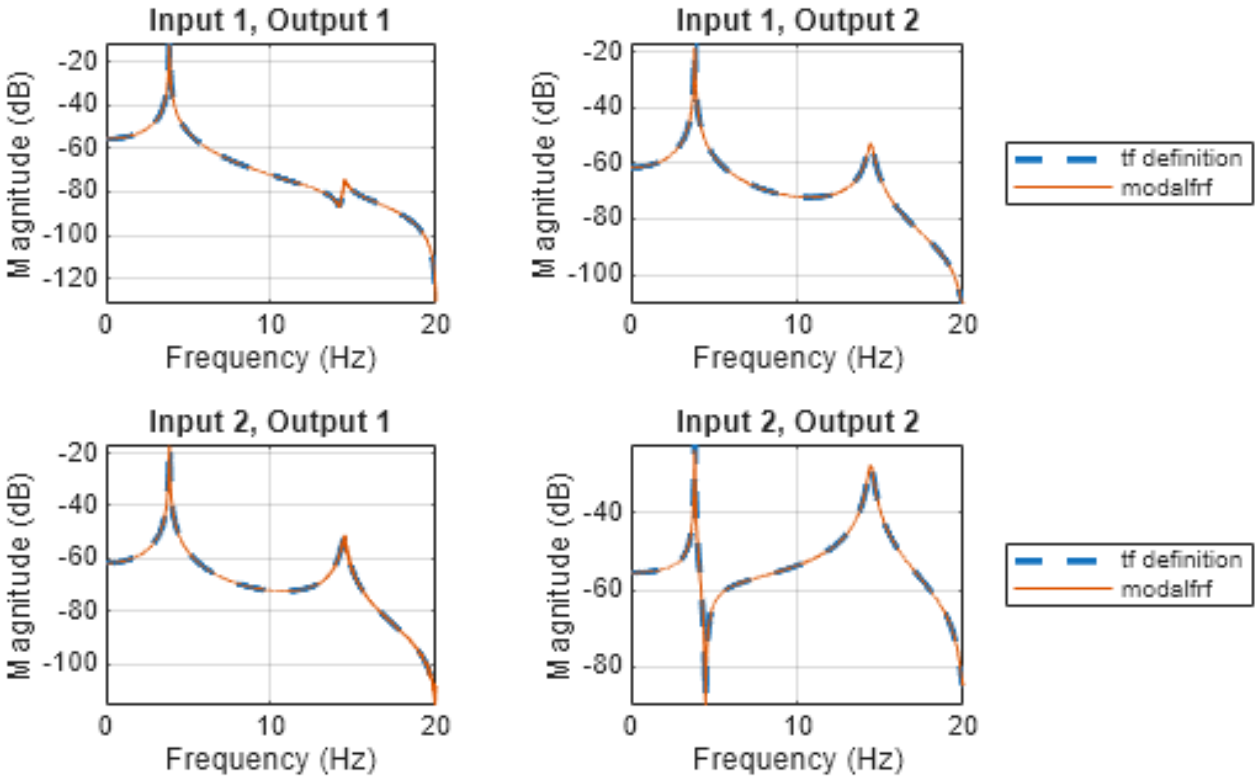
Transfer Function Estimate Using Modal Analysis

Estimate the modal frequency response function of the system using the function modalfrf. Specify the sensor data type as measured displacements.

```
[frf,f] = modalfrf(u,y,Fs,wind,nov,Sensor="dis");
```

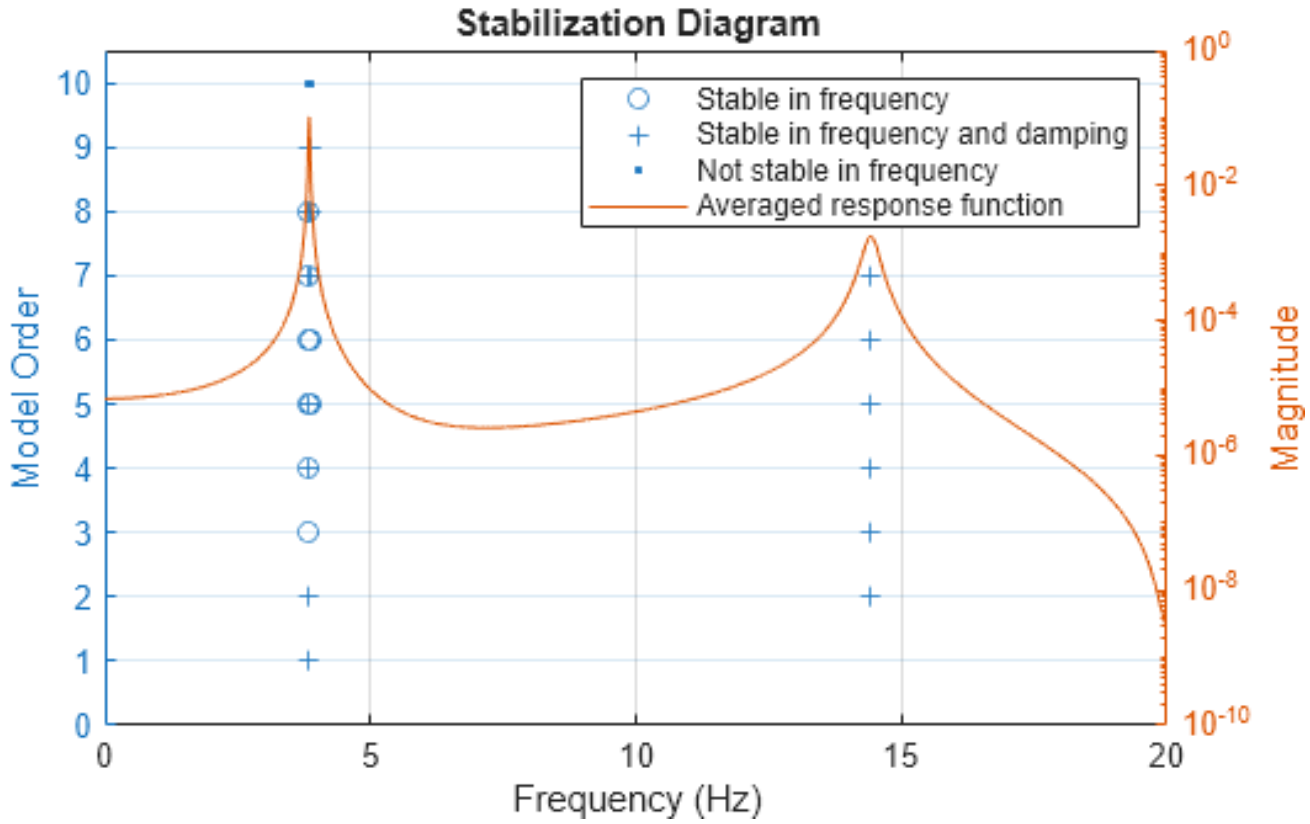
Plot the estimates and overlay the theoretical predictions.

```
plotComparison(fz,ztf,f,frf,"modalfrf")
```



Use modalsd with no output arguments to generate a stabilization diagram for modal analysis and estimate the minimum number of modes. Use a maximum of 10 modes and pick the least-squares rational function estimation method for the calculation.

```
figure
modalsd(frf,f,Fs,MaxModes=10,FitMethod="lsrf")
```



The stabilization diagram shows two "+" marks for 2, 4, 5 and higher model order, indicating a stable modal fit for both frequency and damping using at least two modes.

Estimate the natural frequencies, damping ratios, and mode shapes of the system. Use the function modalfit with two modes and pick the least-squares rational function estimation method for the calculation. Obtain the reconstructed transfer functions of the system in the frequency domain.

```
nModes = 2;
[fn,dr,ms,ofrf] = modalfit(frf,f,Fs,nModes,FitMethod="lsrf");
```

Compare the natural frequencies to the theoretical predictions for an undamped system.

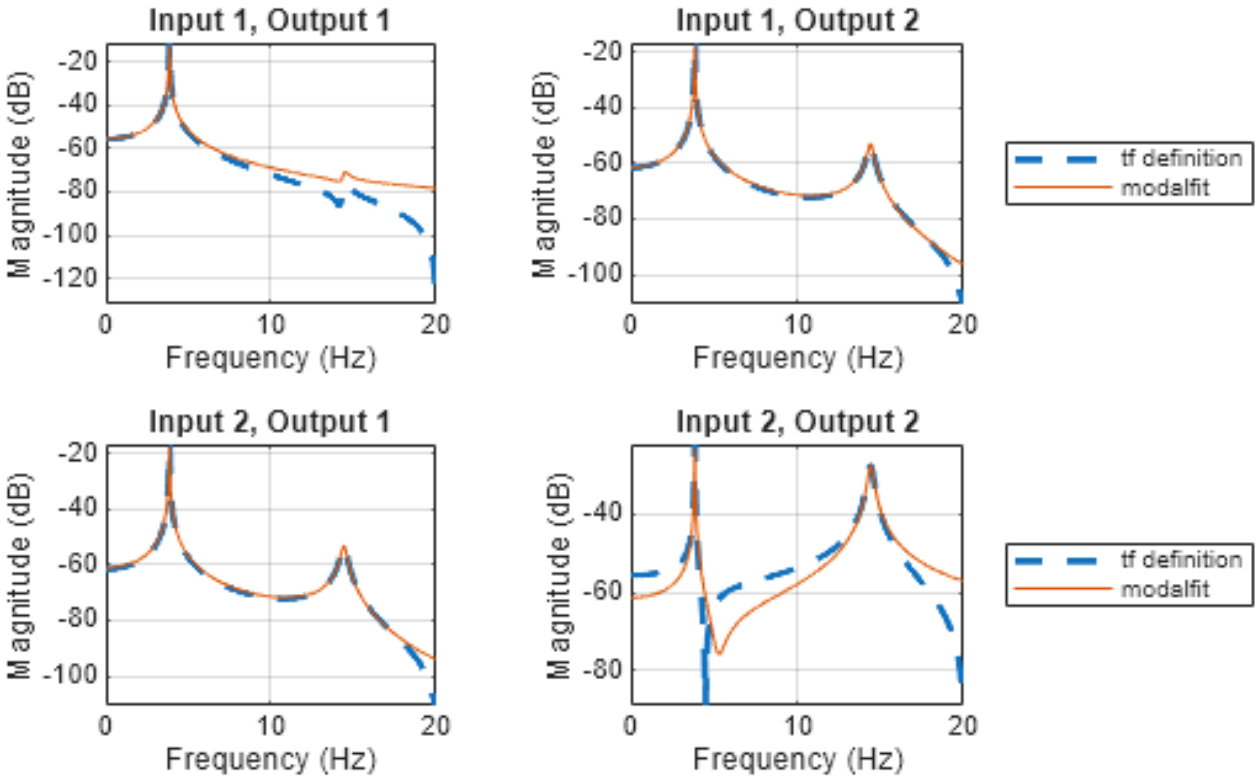
```
fn_theo = sqrt(eig([2*k -k;-k/mu 2*k/mu]))/(2*pi);
disp(table(fn,fn_theo,dr,RowNames= "Mode "+(1:nModes)', ...
   VariableNames=["fn" "fn_theo" "dr"]))
```

	fn	fn_theo	dr
Mode 1	3.8476	3.847	0.0035235
Mode 2	14.426	14.426	0.011308

The proximity in value of the damping ratio (dr) to zero indicates an underdamped system. The natural frequencies of the system (fn) approach the equivalent undamped natural frequency (fn_theo).

Compare the transfer-function definition with the recovered transfer function from the output of modalfit.

```
plotComparison(fz,ztf,f,ofrf,"modalfit")
```

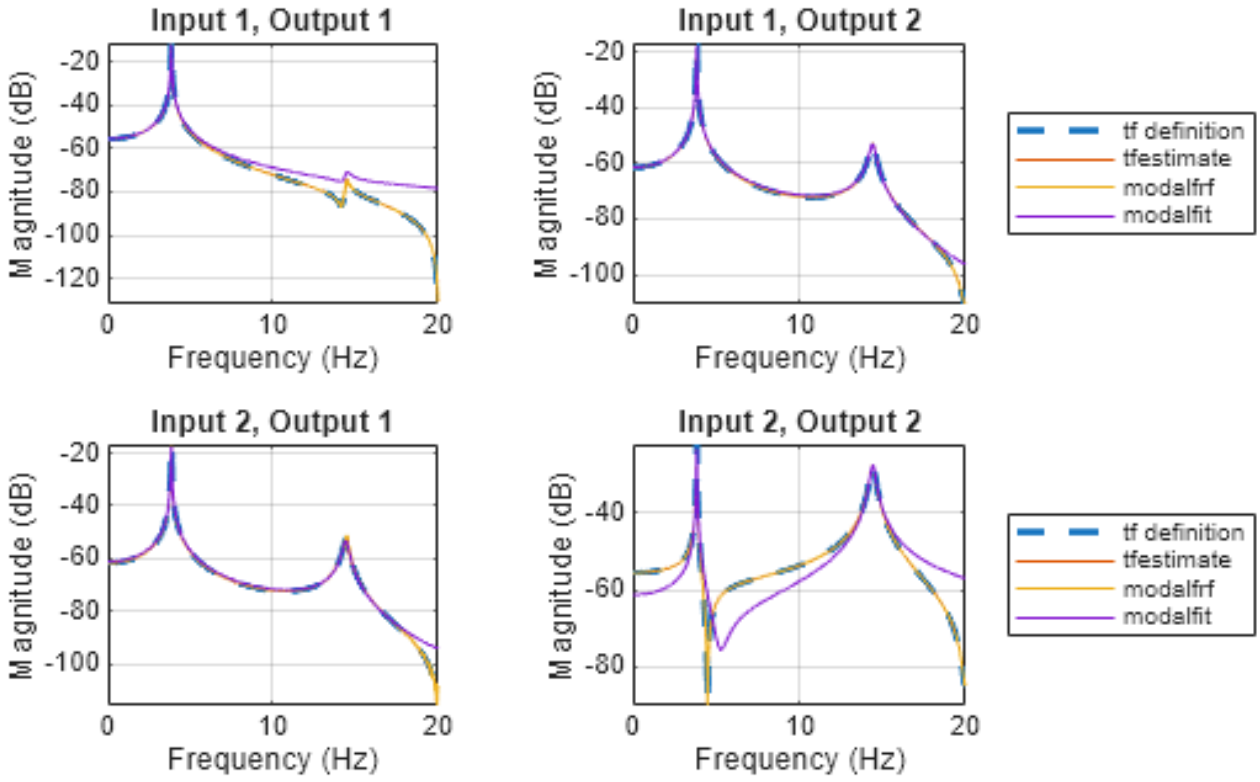


The natural frequencies agree between the recovered and theoretical frequency-response functions despite the differences outside the natural frequencies, correlating with the two modes calculated with modalfit.

Comparison

Compare all transfer function estimation methods discussed in this example.

```
plotComparison(fz,ztf,ft,tXY,"tfestimate",...
   f,frf,"modalfrf",f,ofrf,"modalfit")
```



This example uses the functions tfestimate, modalfrf, modalsd, and modalfit to perform frequency-response analysis based on the space-state model of a mass-spring-damper oscillator MIMO system. The functions tfestimate and modalfrfestimate and plot the frequency response of the system, given the information about the system input and output in the time domain. The function modalsd helps to identify the number of modes to use for modal fit. The modal parameters calculated using the function modalfit also help retrieve the frequency response of the system.

Helper Function

The function plotComparison plots and formats a comparison between theoretical and estimated transfer functions.

```
function plotComparison(varargin)
figure
tiledlayout flow
for jk = 1:2
   for kj = 1:2
      nexttile
      plot(varargin{1}, ...
         mag2db(abs(varargin{2}(jk,:,kj))),"--",LineWidth=2)
      hold on
      for it = 1:fix(nargin/3)
         plot(varargin{3*it},mag2db(abs(varargin{3*it+1}(:,jk,kj))))
      end
      hold off
```

```
        grid on
        axis tight
        title("Input "+jk+", Output "+kj)
        xlabel("Frequency (Hz)")
        ylabel("Magnitude (dB)")
    end
    legend("tf definition",varargin{5:3:end},Location="EastOutside")
end
end
```

WORK TO BE CARRIED OUT

You are to perform a frequency-response analysis on a 2 DOF system under external forces.

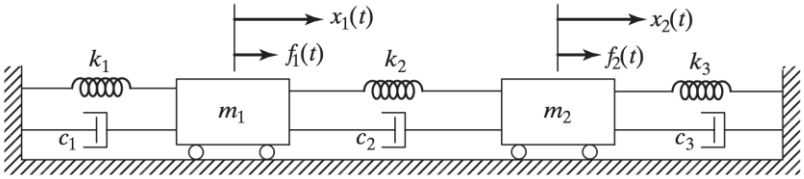


Figure 3: 2 DOF spring-mass-damper system

Assume the initial displacements and velocities of the two masses are specified as

$$x_1(t=0) = x_2(t=0) = 0$$

$$\dot{x_1}(t=0) = \dot{x_2}(t=0) = 0$$

The forces $F_1(t) = F_1 \cos \omega t\,, and\ F_2(t) = F_2 \cos \omega t\,, where\ F_1 = 2N, F_2 = 0.5N, \omega = 0.5 rad/s$

Set $k_1 = k_2 = k_3 = \frac{400N}{m}, c_1 = c_2 = c_3 = \frac{0.1kg}{s}, m_1 = 1kg, m_2 = 0.5kg$

The displacements of the masses are sampled at a rate $F_s = 40Hz$.

Generate 24,000 time samples, equivalent to 10 minutes. Define the sampling interval $\Delta t = 1/F_s$

```
Fs = 40;
dt = 1/Fs;
N = 24000;
t = dt*(0:N-1);
```

```
F_1=2;
F_2=0.5;

omega=0.5;

f1  = zeros(1,length(t));
f2  = zeros(1,length(t));
```

```
for i = 1:length(t)
   f1(1,i) = F_1 * cos(omega * t(i));
   f2(1,i) = F_2 * cos(omega * t(i));
end
```

Obtain the force vectors f_1 and f_2 from $F_1(t)$ and $F_2(t)$.

The system can be described by the state-space model

$$x(k+1) = Ax(k) + Bu(k),$$
$$y(k) = Cx(k) + Du(k),$$

where $x = [r_1 \;\; v_1 \;\; r_2 \;\; v_1]^T$ is the state vector, r_i and v_i are respectively the location and the velocity of the ith mass, $u = [\, f_1 \;\; f_2]^T$ is the vector of input driving forces, and $y = [r_1 \;\; r_2]^T$ is the output vector. The state-space matrices are

$$A = \exp(A_c \Delta t)\,, \qquad B = {A_c}^{-1}(A - I)B_c, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

I is the 4×4 identity matrix, and the continuous-time state-space matrices are

$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2k & -2c & k & c \\ 0 & 0 & 0 & 1 \\ k/m_2 & c/m_2 & -2k/m_2 & -2c/m_2 \end{bmatrix}, \qquad B_c = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix}$$

1. Derive the input x(k+1) and response y(k) vectors of the state-space model.
2. Plot the system response over time for the two masses.
3. Estimate the modal frequency response function of the system using the function ***modalfrf*** and plot it. Specify the sensor data type as measured displacements.
4. Estimate the natural frequencies, damping ratios, and mode shapes of the system using the function ***modalfit***. Use two modes and pick the least-squares rational function estimation method for the calculation.

LAB PROJECT # 4:
Mode shape analysis of cantilever beam with ANSYS

AIM:

To model and visualize the mode shapes of a cantilever beam.

TOOL REQUIRED:

ANSYS software

BRIEF INTRODUCTION

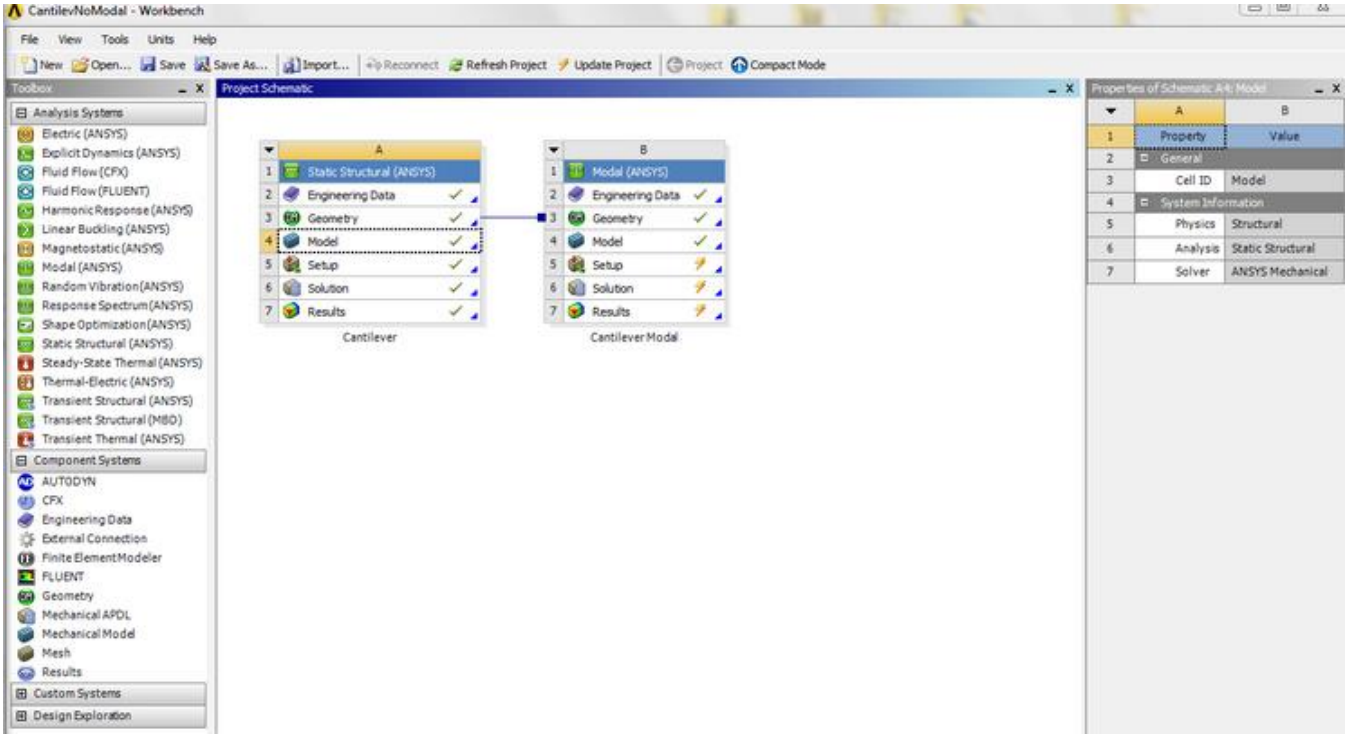
The modal analysis deals with the dynamics behavior of mechanical structures under the dynamics excitation. The modal analysis helps to reduce the noise emitted from the system to the environment. It helps to point out the reasons of vibrations that cause damage of the integrity of system components. Using it, we can improve the overall performance of the system in certain operating conditions. We know two basic methods of the modal analysis, namely the numerical modal analysis and the experimental modal analysis. The experimental modal analysis deals with measurement input data from which a mathematical model is derived. However, it has to take different levels of analysis, from which the model is constructed.

METHODOLOGY

Step 1:

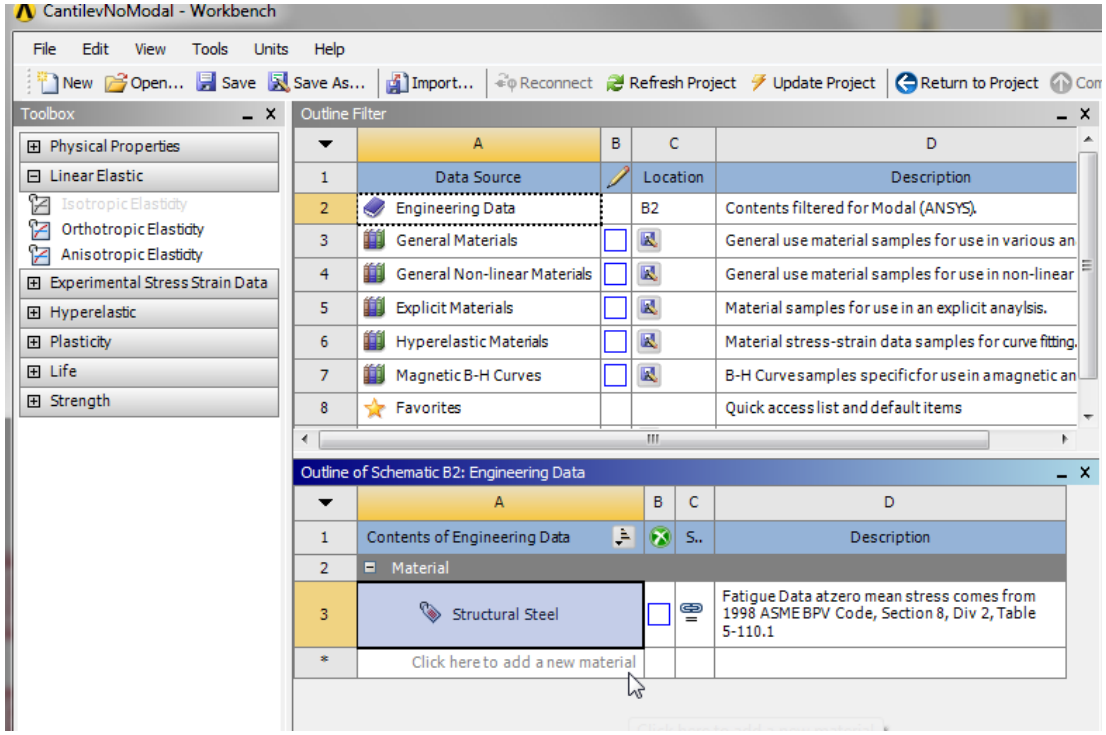
Modal (ANSYS) Project Selection

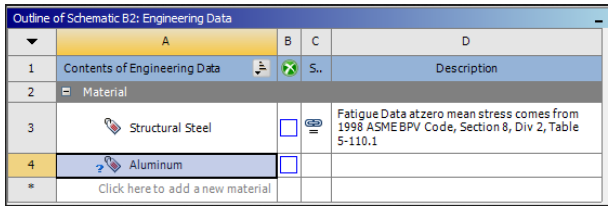
Left, click on ***Modal ANSYS***, and drag it to the right of the "Cantilever" project. You should then see a red box to the right of the "Cantilever" project that says "Create standalone system"



Double click on the “Engineering Data” a new window will open as shown below. Next, to add your own new material click on “click here to add new material” Name it whatever you want and press **enter**, eg “ALUMINIUM”

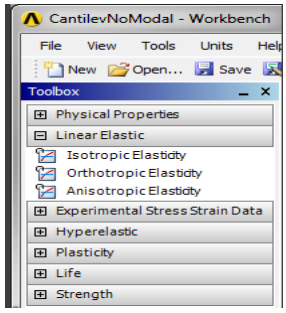
you should now have Aluminum listed as one of the materials in table called "Outline of Schematic B2: Engineering Data", as shown below.





Then, ***(expand) Linear Elastic***, as shown below.

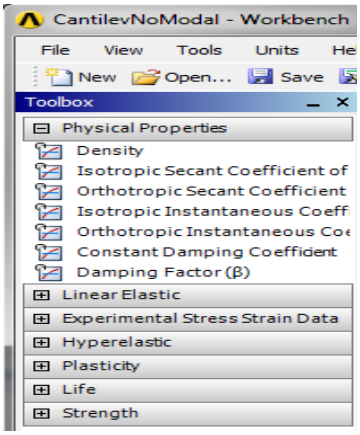
Now, ***(Double Click) Isotropic Elasticity***. Then set ***Young's Modulus*** to 70e9 Pa and set ***Poisson's Ratio*** to 0.35, as shown below



Next, ***(expand) Physical Properties***, as shown below

Properties of Outline Row 4: Aluminum

	A	B	C
1	Property	Value	Unit
2	Isotropic Elasticity		
3	Derive from	Young's Modulus and Poisson's Ratio	
4	Young's Modulus	7E+10	Pa
5	Poisson's Ratio	0.35	
6	Bulk Modulus	7.7778E+10	Pa
7	Shear Modulus	2.5926E+10	Pa



Now, ***(Double Click) Density***. Then, set ***Density*** value, as shown below

Properties of Outline Row 4: Aluminum

	A	B	C
1	Property	Value	Unit
2	Density	2700	kg m^-3
3	Isotropic Elasticity		
4	Derive from	Young's Modulus and Poisson's Ratio	
5	Young's Modulus	7E+10	Pa
6	Poisson's Ratio	0.35	
7	Bulk Modulus	7.7778E+10	Pa
8	Shear Modulus	2.5926E+10	Pa

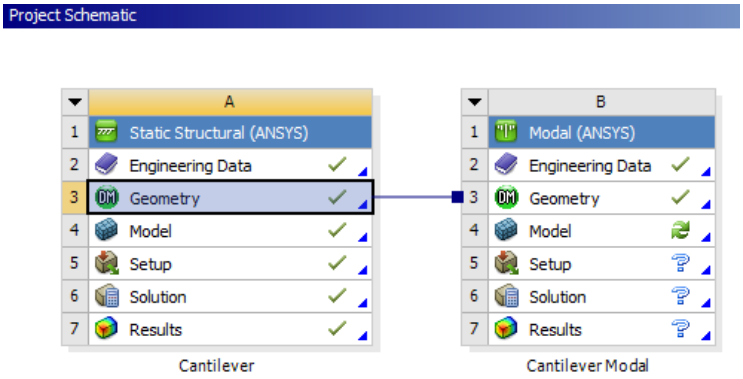
Now, the material properties for Aluminum have been specified. Lastly, ***(Click) Return To Project***,

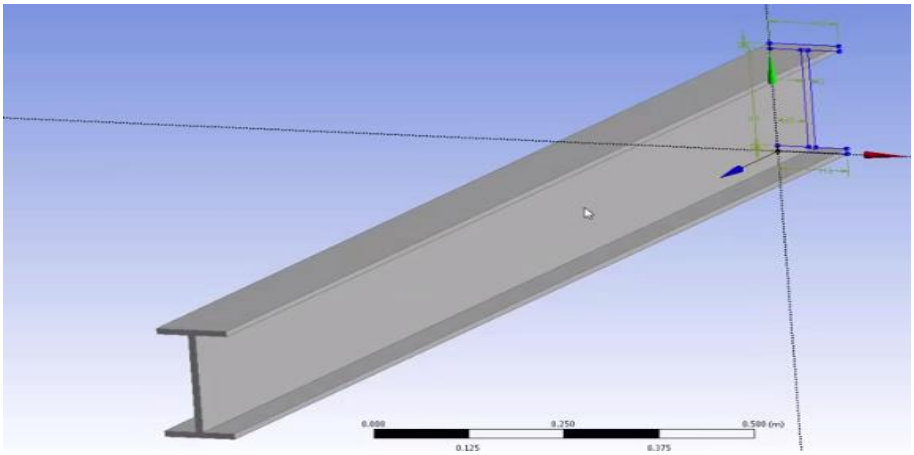
Save your project periodically, as you are working. ANSYS does not have an auto-save feature.

Step 2:

Geometry: Attach the Geometry

In order to attach the geometry, right click on the *"**Geometry**"* and either click on DM *"**New Design Modular Geometry**"* or import the geometry through *"**Import Geometry**"*





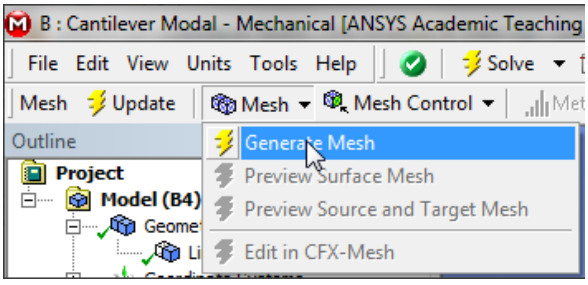
Step 3:

Generate Mesh

Go back to “project” right click or double click on “Model” and select “Edit”, click and a new window will open.

To generate default mesh:

First**, *(click) Mesh*** in the tree outline. Next, ***(click) Mesh > Generate Mesh*** as shown below

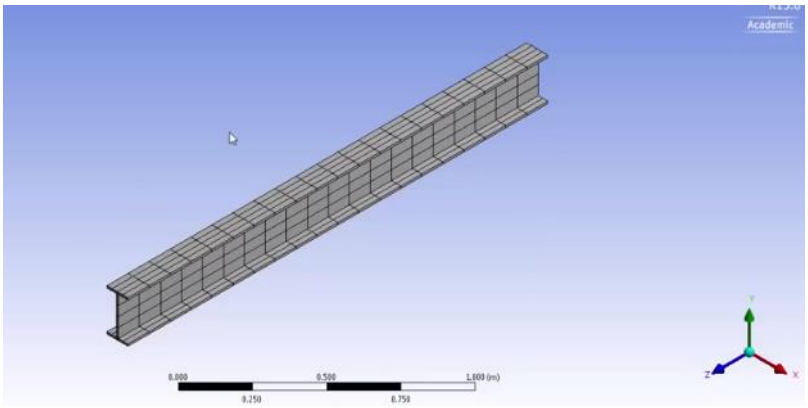


To generate Size mesh

In this section we will size the mesh, such that it has ten uniform elements. In order to size the mesh, first expand ***Sizing*** located within the *Details of "Mesh"* table. Next, set *Element Size* to 0.40 m, as shown below.

Details of "Mesh"	
Defaults	
Physics Preference	Mechanical
Relevance	0
Sizing	
Use Advanced Size Function	Off
Relevance Center	Coarse
Element Size	0.40 m
Initial Size Seed	Active Assembly
Smoothing	Medium
Transition	Fast
Span Angle Center	Coarse
Minimum Edge Length	4.0 m
Inflation	
Advanced	
Pinch	
Statistics	

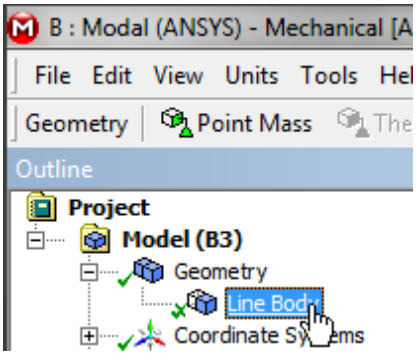
Now, ***(click) Mesh > Generate Mesh*** in order to generate the new mesh. You should obtain the mesh, that is shown in the following image.



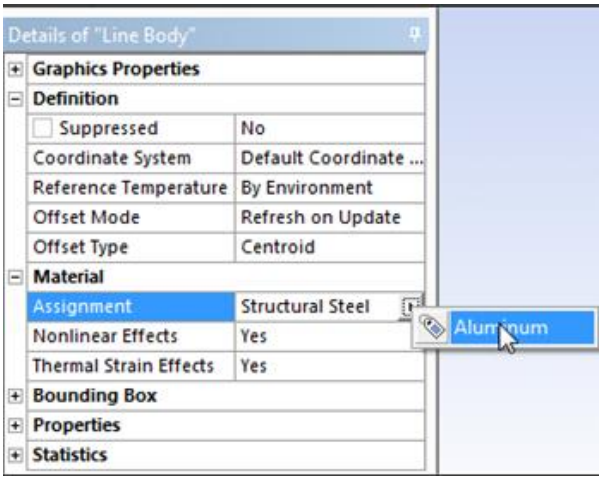
Step 4:

Material Assignment

At this point, we will tell ANSYS to assign the Aluminum material properties that we specified earlier to the geometry. First, ***(expand) Geometry*** then ***(click) Line Body***, as shown below.



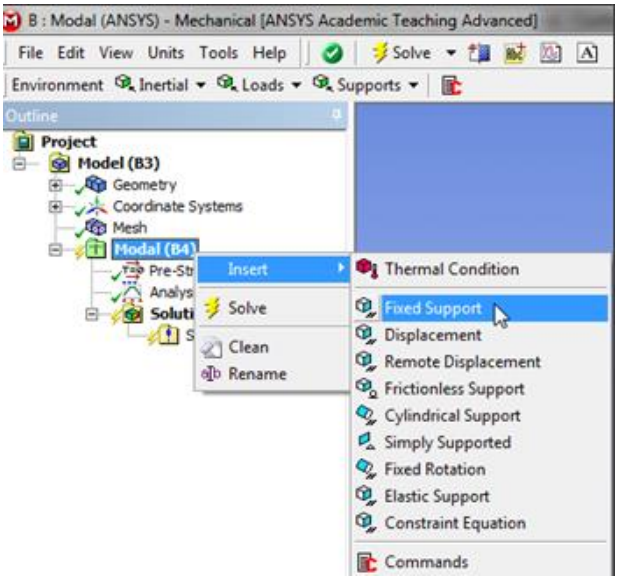
Then, ***(expand) Material*** in the "Details of Line Body" table and set ***Assignment*** to Aluminum, as shown below.

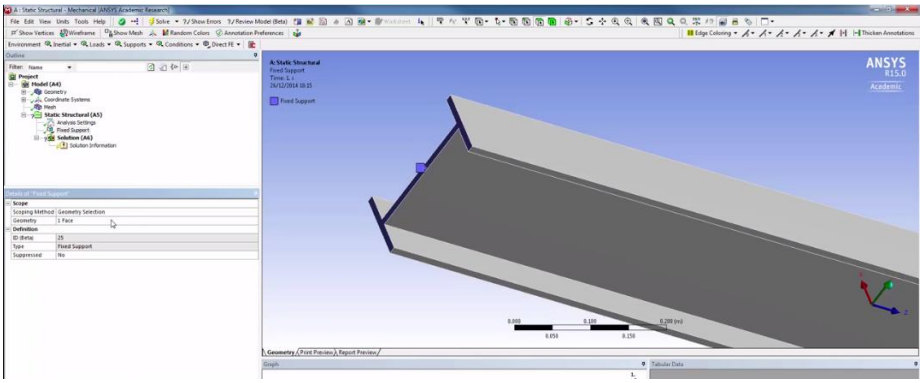


Step 5:

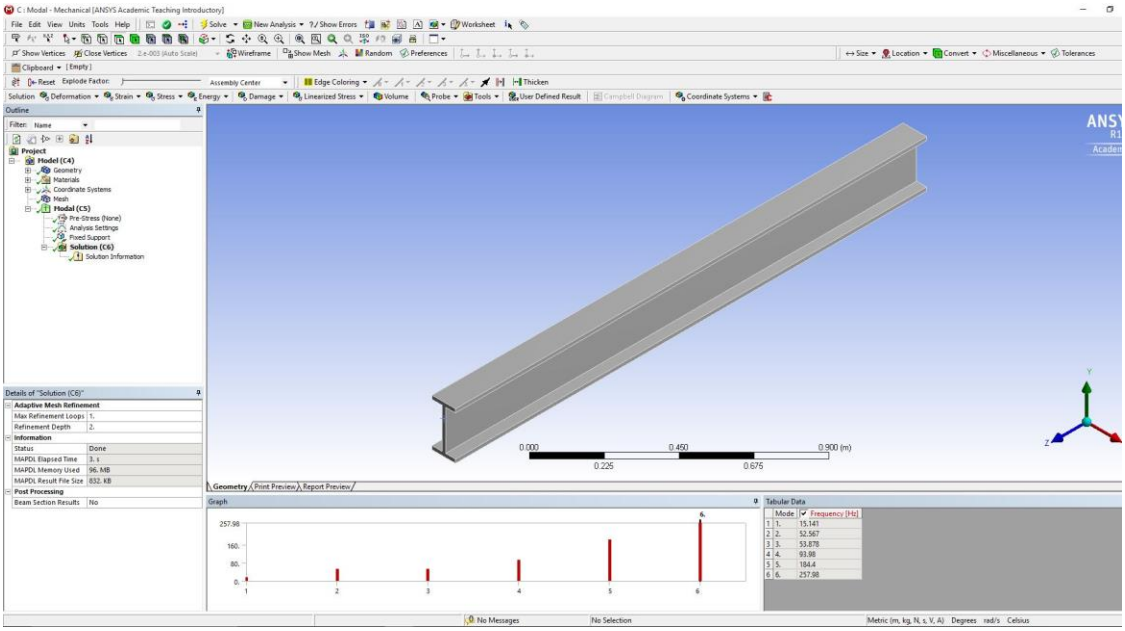
Assign Fixed Support

First, ***(right click) Modal > Insert > Fixed Support***, as shown below.

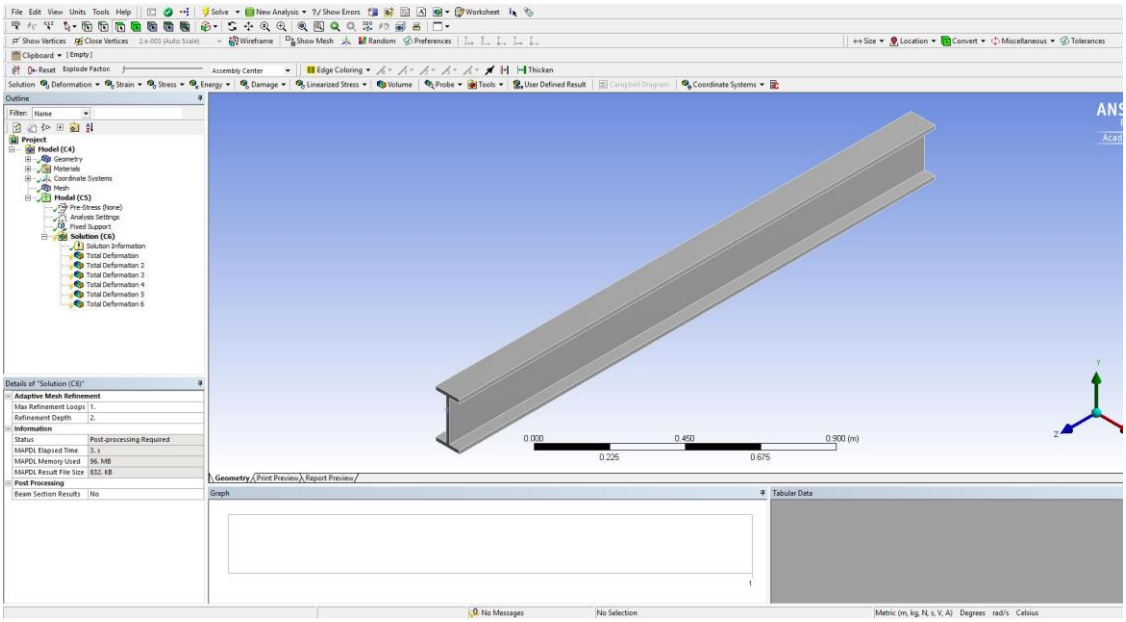




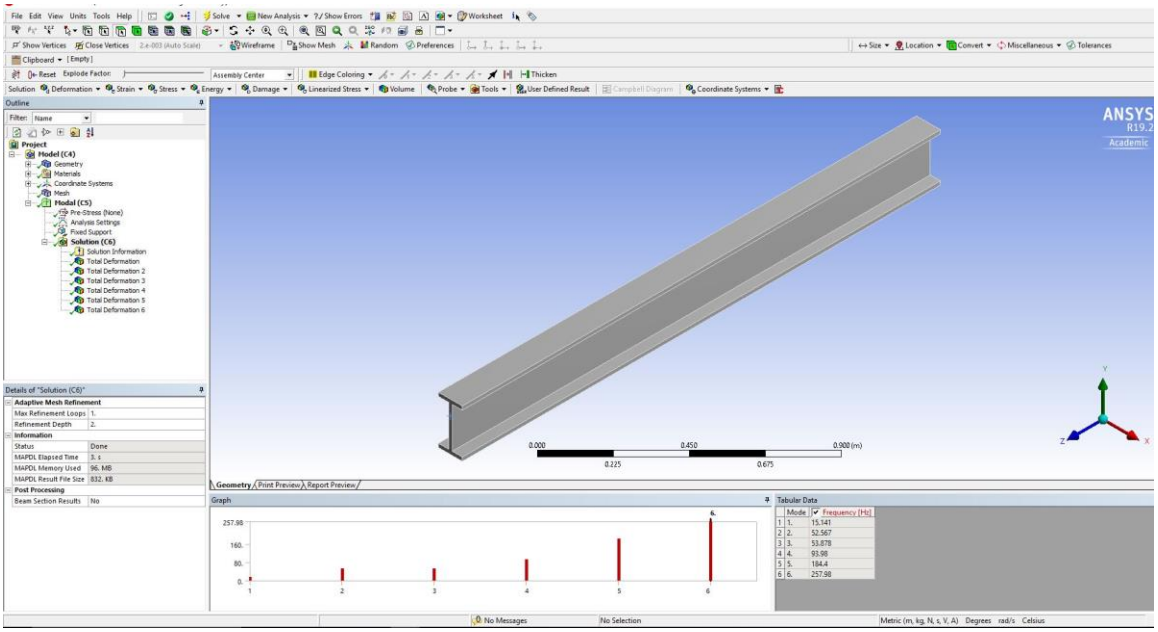
Next, click on the "**solve**" button and get the results as shown below.



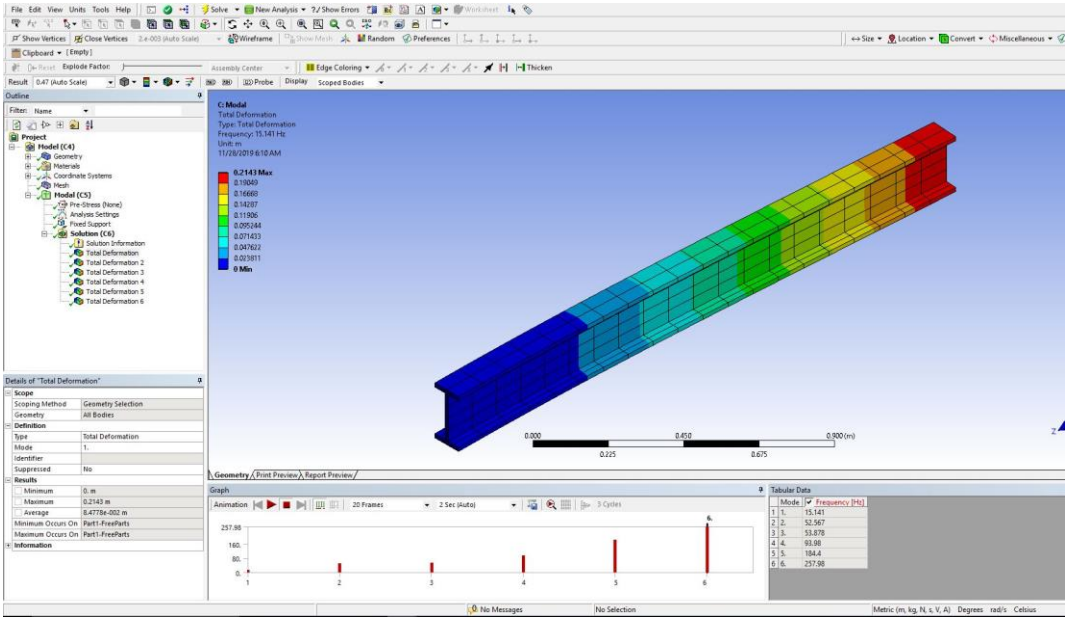
To see the model right click on the “graph” below the “geometry” window and select all. Right click again on the shaded blue region on the graph and click on the “**Create Mode Shape Results**”, “**total deformation**” 1 to 6 results will appear under the “**solution (C6)” as shown below.**



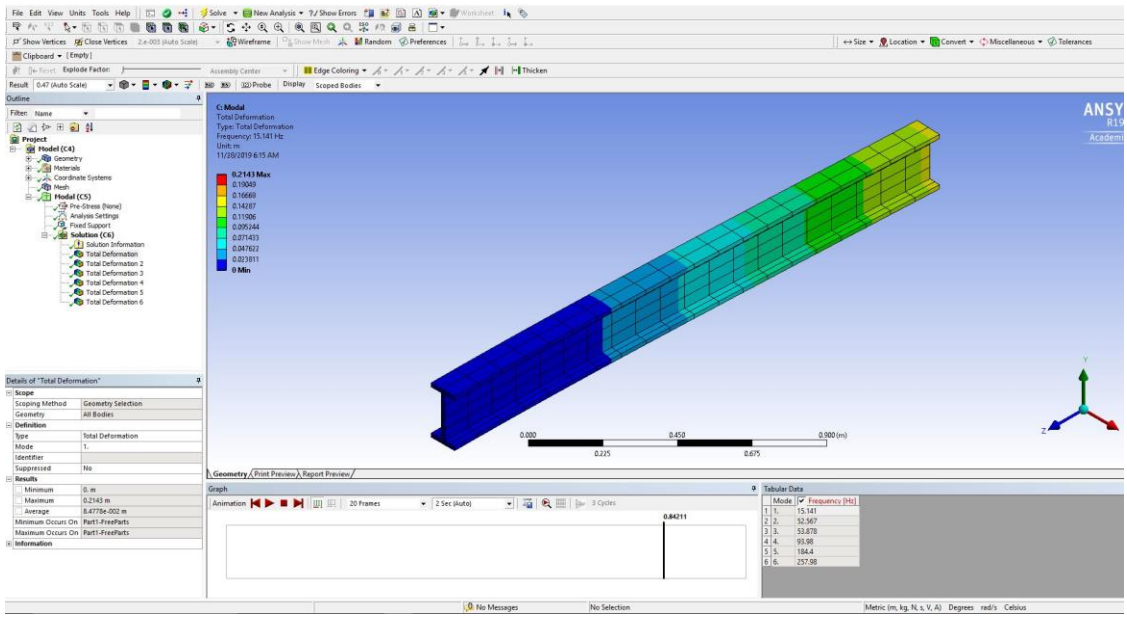
Then, click solve to obtain the mode shapes “Total Deformation” results as shown below. Here, we will tell ANSYS to find the deformation for the first six modes.



Select “total deformation 1” to visualize the mode shape result 1, select the rest for mode shape 2, 3, 4,5 and 6.



In order to visualize and run the simulation result click the “play” as shown below. Then, we will be able to see the shapes of the six modes. Additionally, we will be able to watch nice animations of the six modes.


C: Modal
Total Deformation
Type: Total Deformation
Frequency: 15.141 Hz
Unit: m
11/28/2019 6:15 AM
0.2143 Max
0.19049
0.16668
0.14287
0.11906
0.095244
0.071433
0.047622
0.023811
0 Min
0.000
0.450
0.900 (m)
0.225
0.675
ANSY
R19
Academi
Geometry
Print Preview
Report Preview
Graph
Tabular Data
Mode
Frequency [Hz]
1 1. 15.141
2 2. 52.567
3 3. 53.878
4 4. 93.98
5 5. 184.4
6 6. 257.98
No Messages
No Selection
Metric (m, kg, N, s, V, A) Degrees rad/s Celsius

WORK TO BE CARRIED OUT:

Based on your group select and use the given specifications below, and select the material properties of the corresponding material assigned from (Table 1: Material properties). All the dimensions from the given figures below are in mm.

	BEAM TYPE	*MATERIAL ASSIGNED*
Group 1	**I beam**	**Aluminum**
Group 2	**I beam**	**Steel**
Group 3	**I beam**	**Titanium**
Group 4	**T beam**	**Aluminum**
Group 5	**T beam**	**Steel**
Group 6	**T beam**	**Titanium**
Group 7	**O beam**	**Aluminum**
Group 8	**O beam**	**Steel**
Group 9	**O beam**	**Titanium**
Group 10	**L beam**	**Aluminum**
Group 11	**L beam**	**Steel**
Group 12	**L beam**	**Titanium**

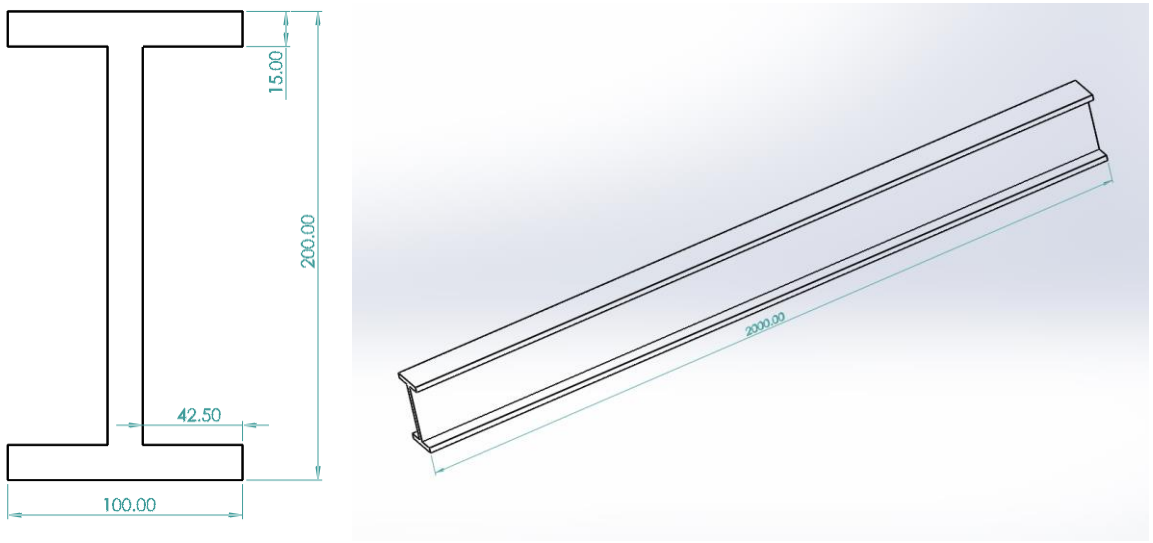


Figure 4: I Beam Dimensions

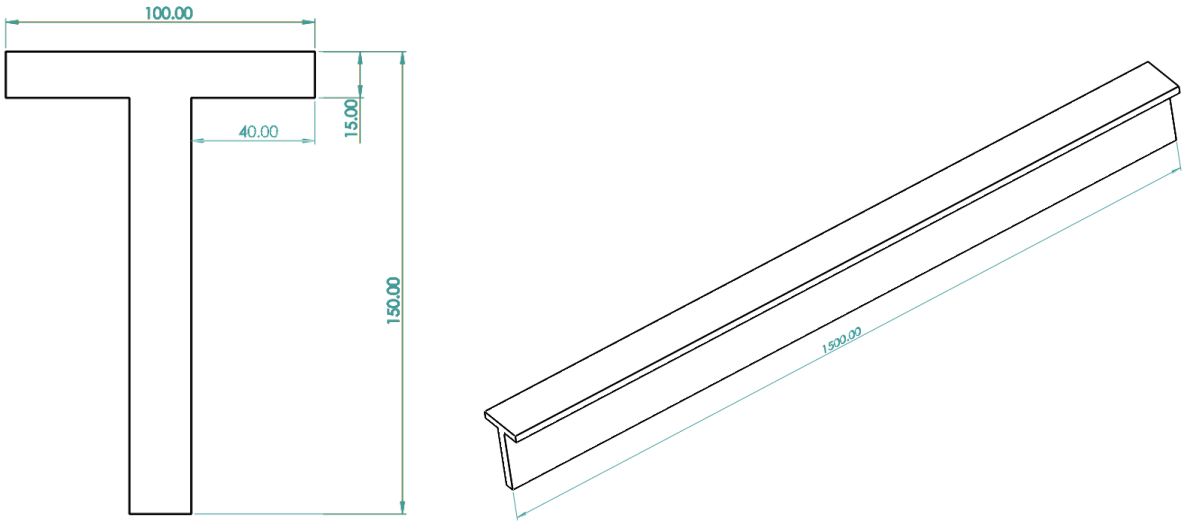


Figure 5: T Beam Dimensions

Figure 6: O Beam Dimensions

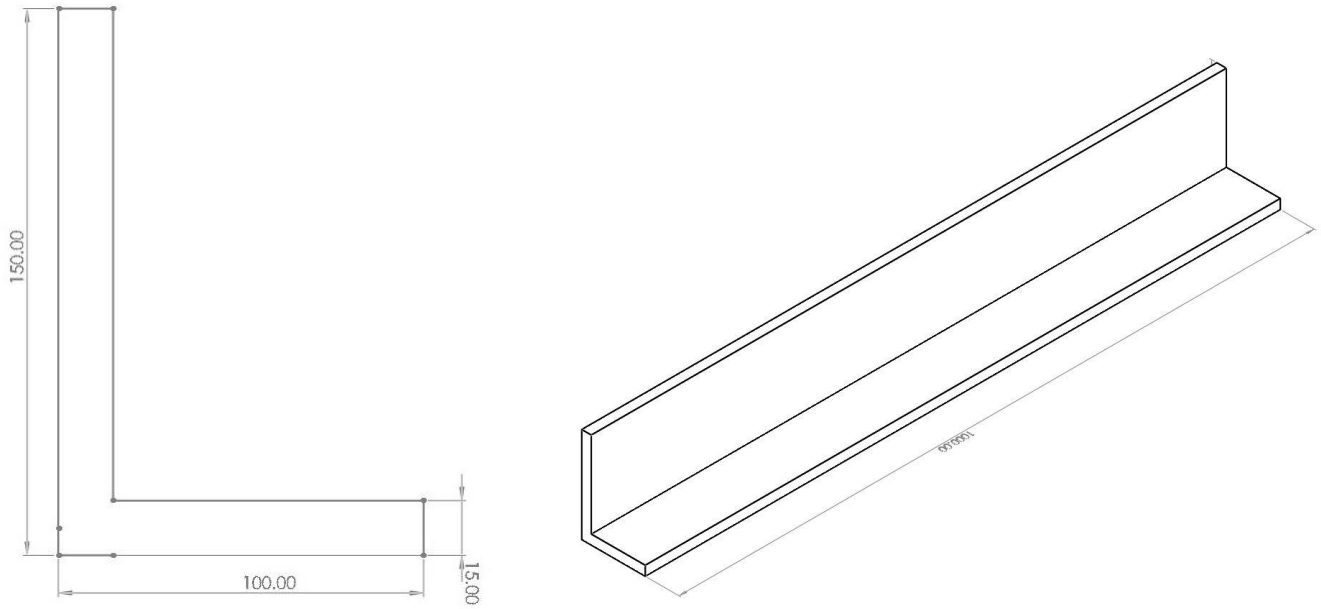


Figure 7: L Beam Dimensions

Table 2: Material properties

Material(s)	**Young s modulus (E)** [GPa]	**Poisson's ration (*v*)**	**Density (ρ)** [Kg/m^3]
Steel (Carbon steel)	**207**	**0.292**	**7850**
Titanium (Aluminum alloys)	**114**	**0.342**	**4430**
Aluminum (Aluminum alloys)	**71**	**0.334**	**2700**

1. **Prepare your report based on the given template, your report should include pictures of all the major steps and they must be labelled/named.**
    a. **Abstract**
    b. **Introduction**
    c. **A brief procedure that includes (refer back to METHODOLOGY):**
        i. **Necessary steps.**
        ii. **Attach fixed support at one end.**
        iii. **All the images/pics of the main steps.**
        iv. **First, carry out the simulation based on the default setup of "default details of mesh", then**
        v. **Carry out based on the given setups from table 2 as edited "details of mesh":**

Table 3: "details of mesh"

Relevance	**100**
Relevance center	**Fine**
Element size	**20 mm**

Smoothing	**High**

d. Results

i. **Under this section you are expected to show the solution of your simulation from the "edited details of mesh" only.**

ii. **Pics of <u>all the 6 mode shapes</u> analyzed from the edited "details of mesh" are expected and must be labelled.**

iii. **A table containing the mode shapes frequencies for both "default details of mesh" and "edited details of mesh".**

Table 4: ……

Mode shapes	**1**	**2**	**3**	**4**	**5**	**6**
Frequencies						

e. Discussion and Conclusion

i. **Discuss on the results obtained. Provide a comparison mode shapes table of the "default details of mesh" and the "edited details of mesh"**

Table 5: .....

"default details of mesh" frequencies						
"edited details of mesh" frequencies						

ii. **What Conclusions could you draw?**

iii. **Based on the results obtained from section "e-i", which one is better? And Why? What are the ways of improving the results?**

iv. **Why the choice of Element Type is important in any FEA software?**

v. **Which Element Type you chose and why? Please explain in detail.**

All the work carried out should be burn on CD/DVD and submitted together with hard copy of the report. (Hard Copy conditional to university opening, otherwise softcopy via Microsoft Teams Assignment)

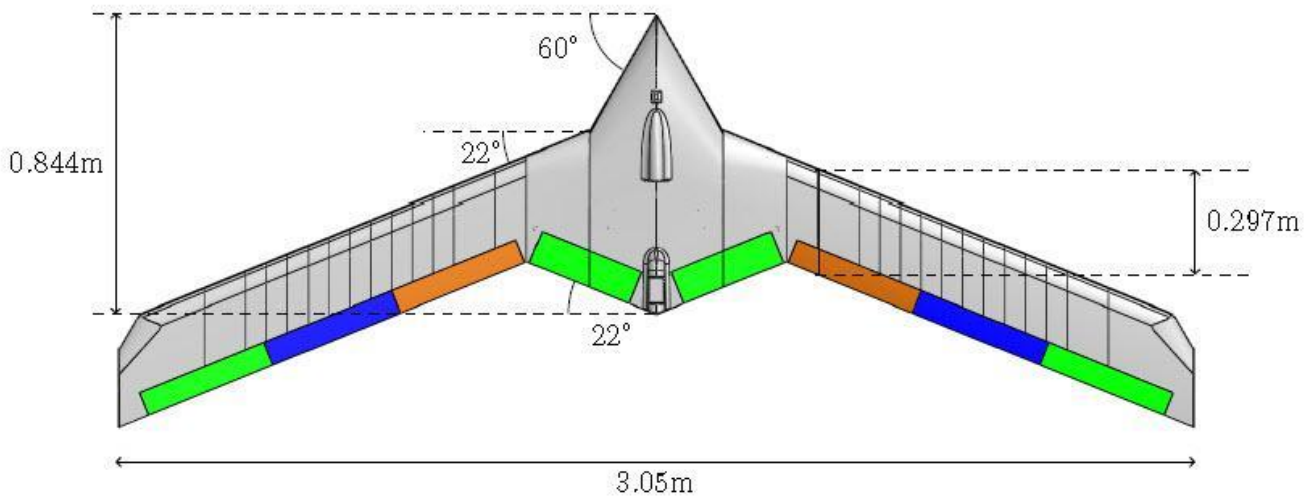
- **PDF report**
- **Figures of the main steps involved (see what is under METHODOLOGY)**
- **Figures of the 6 mode shapes**
- **Print preview and report preview files**
- **ANSYS Simulation, rendered avi video file of the simulation.**

LAB PROJECT # 5:
Modal Analysis of a Flexible Flying Wing Aircraft

This example shows computation of bending modes of a flexible wing aircraft. The vibration response of the wing is collected at multiple points along its span. The data is used to identify a dynamic model of the system. The modal parameters are extracted from the identified model. The modal parameter data is combined with the sensor position information to visualize the various bending modes of the wing. This example requires Signal Processing Toolbox™.

The Flexible Wing Aircraft

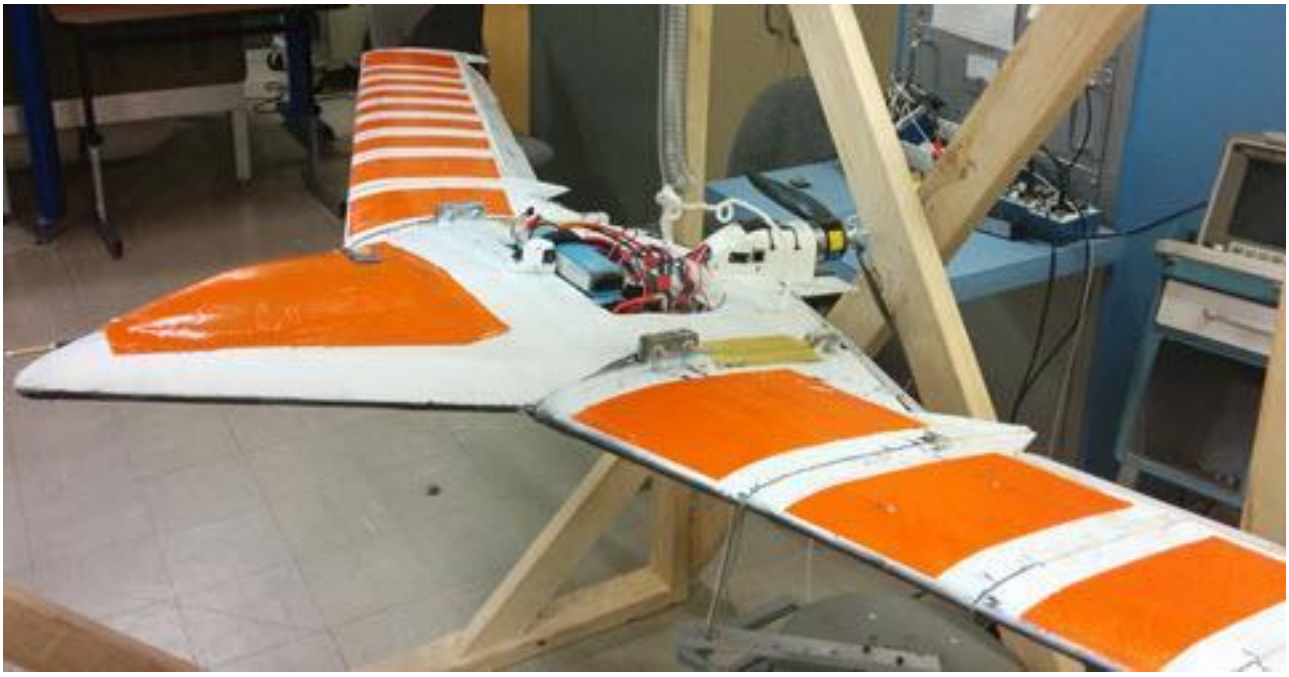
In this example, we study the data gathered from a small flexible flying wing aircraft built at the Uninhabited Aerial Vehicle Laboratories, University of Minnesota [1]. The geometry of the aircraft is shown below.

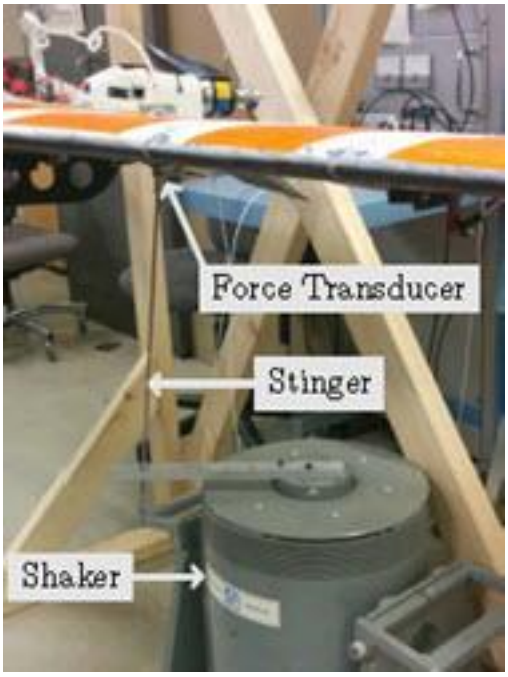


The aircraft wing can undergo large deformations under loading. The flexible mode frequencies are lower than those in common aircraft with more rigid wings. This flexible design reduces material costs, increases agility and the flight range of the aircraft. However, if not controlled, the flexible modes can lead to catastrophic aeroelastic instabilities (flutter). Designing effective control laws for suppressing these instabilities requires accurate determination of the wing's various bending modes.

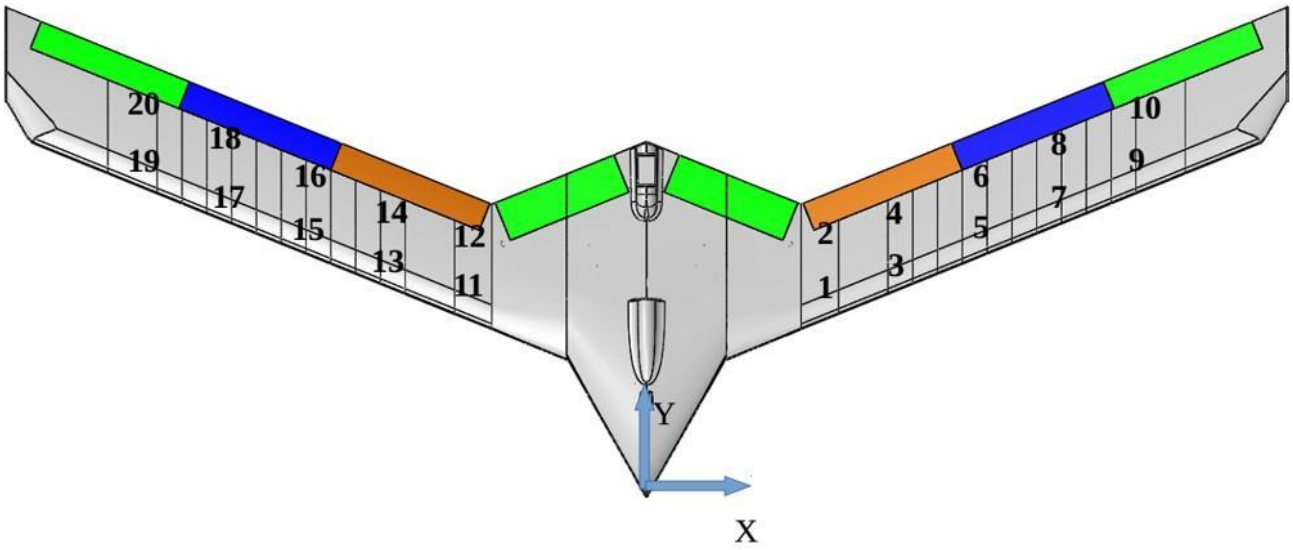
Experimental Setup

The objective of the experiment is to gather vibration response of the aircraft at various locations in response to an external excitation. The aircraft is suspended from a wooden frame using a single spring at its center of gravity. The spring is sufficiently flexible so that the natural frequency of the spring-mass oscillation does not interfere with the fundamental frequencies of the aircraft. An input force is applied via an Unholtz-Dickie Model 20 electrodynamic shaker near the center of the aircraft.





Twenty PCB-353B16 accelerometers are placed along the wing span to collect the vibration response as shown in the next figure.



The shaker input command is specified as a constant amplitude chirp input of the form $A\sin(\omega(t)t)$. The chirp frequency varies linearly with time, that is, $\omega(t)=c_0+c_1t$. The frequency range covered by the input signal is 3–35 Hz. The data is collected by two accelerometers (leading and trailing edge accelerometers at one x-location) at a time. Hence 10 experiments are conducted to collect all the 20 accelerometer responses. The accelerometer and force transducer measurements are all sampled at 2000 Hz.

Data Preparation

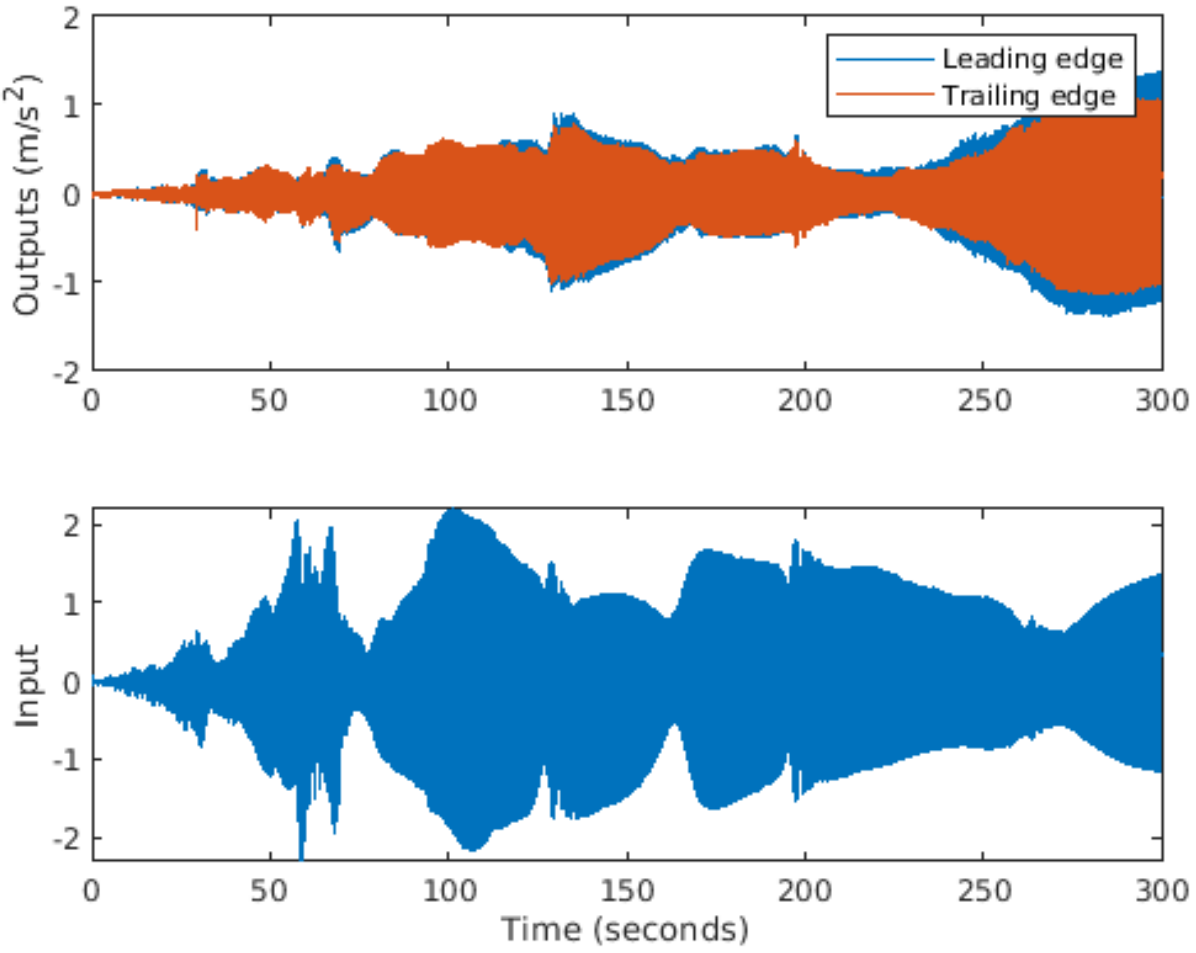
The data is represented by 10 sets of two-output/one-input signals, each containing 600K samples. The data is available at the MathWorks support files site. See the disclaimer. Download the data file and load the data into the MATLAB® workspace.

```
url = 'https://www.mathworks.com/supportfiles/ident/flexible-wing-GVT-data/FlexibleWingData.mat';
websave('FlexibleWingData.mat',url);
load FlexibleWingData.mat MeasuredData
```

The variable MeasuredData is a structure with fields Exp1, Exp2, ..., Exp10. Each field is a structure with fields y and u containing the two responses and the corresponding input force data. Plot the data for the first experiment.

```
fs = 2000;              % data sampling frequency
Ts = 1/fs;              % sample time
y = MeasuredData.Exp1.y;  % output data (2 columns, one for each accelerometer)
u = MeasuredData.Exp1.u;  % input force data
t = (0:length(u)-1)' * Ts;
figure
subplot(211)
plot(t,y)
ylabel('Outputs (m/s^2)')
legend('Leading edge','Trailing edge')
subplot(212)
plot(t,u)
```

```
ylabel('Input')
xlabel('Time (seconds)')
```



In order to prepare data for model identification, the data is packaged into iddata objects. The iddata objects are standard way of packaging time-domain data in System Identification Toolbox™. The input signal is treated as bandlimited.

```
ExpNames = fieldnames(MeasuredData);
Data = cell(1, 10);
for k = 1:10
  y =  MeasuredData.(ExpNames{k}).y;
  u =  MeasuredData.(ExpNames{k}).u;
  Data{k} = iddata(y, u, Ts, 'InterSample', 'bl');
end
```

Merge the dataset objects into one multi-experiment data object.

```
Data = merge(Data{:})
Data =
```

Time domain data set containing 10 experiments.

Experiment	Samples	Sample Time
Exp1	600001	0.0005
Exp2	600001	0.0005
Exp3	600001	0.0005
Exp4	600001	0.0005
Exp5	600001	0.0005
Exp6	600001	0.0005
Exp7	600001	0.0005
Exp8	600001	0.0005
Exp9	600001	0.0005
Exp10	600001	0.0005

Outputs	Unit (if specified)
y1	
y2	

Inputs	Unit (if specified)
u1	

Model Identification

We want to identify a dynamic model whose frequency response matches that of the actual aircraft as closely as possible. A dynamic model encapsulates a mathematical relationship between the inputs and outputs of the system as a differential or difference equation. Examples of dynamic models are transfer functions and state-space models. In System Identification Toolbox, dynamic models are encapsulated by objects such as idtf (for transfer functions), idpoly (for AR, ARMA models) and idss (for state-space models). Dynamic models can be created by running estimation commands such as tfest and ssest commands on measured data in either time-domain or frequency-domain.

For this example, we first convert the measured time-domain data into frequency response data by empirical transfer function estimation using the etfe command. The estimated FRF is then used to identify a state-space model of the aircraft's vibration response. It is possible to directly use time-domain data for dynamic model identification. However, FRF form of data allows compression of large datasets into fewer samples as well as more easily adjust estimation behavior to relevant frequency ranges. FRFs are encapsulated by idfrd objects.

Estimate a two-output/one-input frequency response function (FRF) for each data experiment. Use no windowing. Use 24,000 frequency points for response computation.

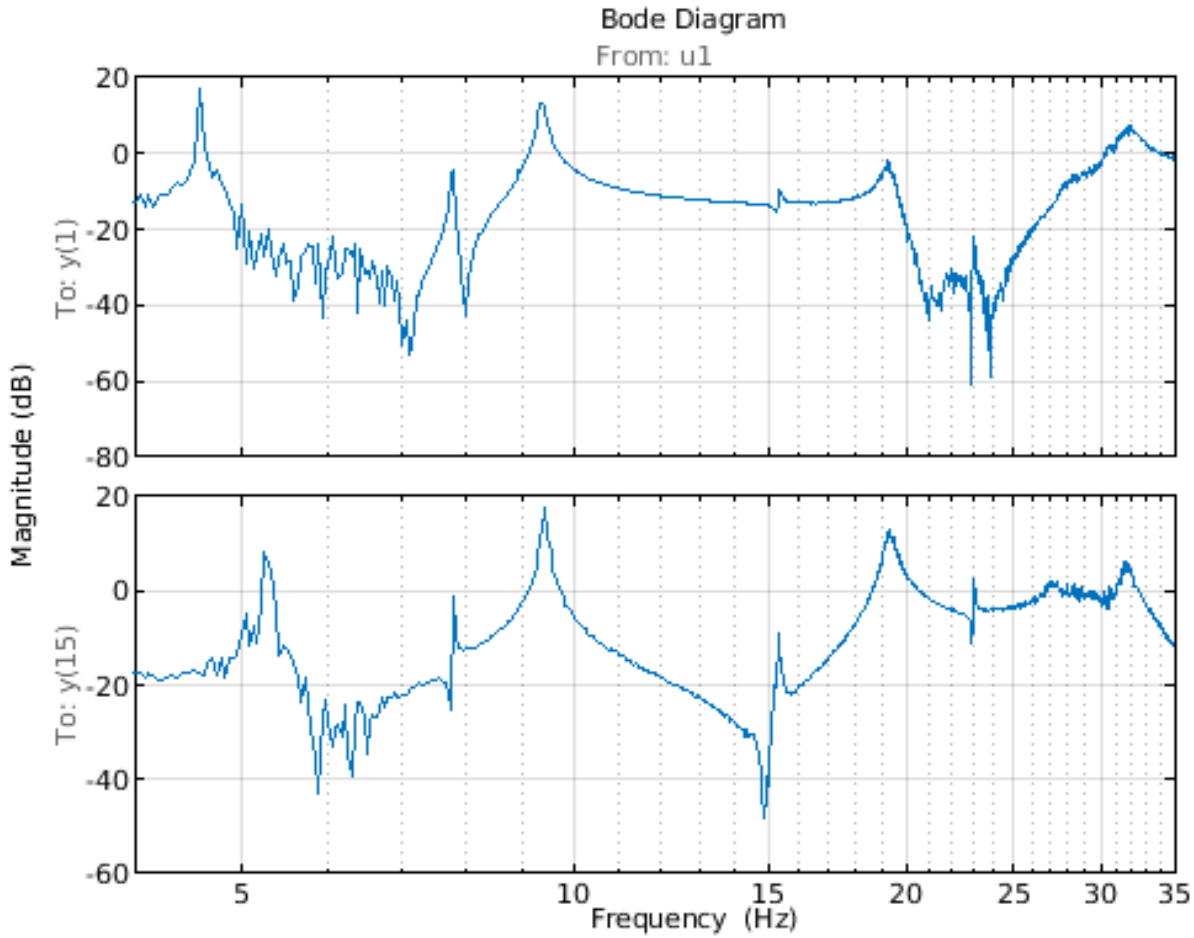
```
G = cell(1, 10);
N = 24000;
for k = 1:10
  % Convert time-domain data into a FRF using ETFE command
  Data_k = getexp(Data, k);
  G{k} = etfe(Data_k, [], N); % G{k} is an @idfrd object
end
```

Concatenate all FRFs into a single 20-output/one-input FRF.

```
G = cat(1, G{:});     % concatenate outputs of all estimated FRFs
G.OutputName = 'y';   % name outputs 'y1', 'y2', ..., 'y20'
G.InterSample = 'bl';
```

To get a feel for the estimated frequency response, plot the amplitude for outputs 1 and 15 (picked arbitrarily). Zoom into the frequency range of interest (4–35 Hz).

```
opt = bodeoptions;          % plot options
opt.FreqUnits = 'Hz';       % show frequency in Hz
opt.PhaseVisible = 'off';   % do not show phase
OutputNum = [1 15];         % pick outputs 1 and 15 for plotting
clf
bodeplot(G(OutputNum, :), opt)   % plot frequency response
xlim([4 35])
grid on
```



The FRF shows at least 9 resonant frequencies. For analysis we want to focus on 6-35 Hz frequency span where the most critical flexible bending modes of the aircraft lie. Hence reduce the FRF to this frequency region.

```
f = G.Frequency/2/pi;          % extract frequency vector in Hz (G stores frequency in rad/s)
```

```
Gs = fselect(G, f>6 & f<=32)    % "fselect" selects the FRF in the requested range (6.5 - 35 Hz)
```

```
Gs =
IDFRD model.
Contains Frequency Response Data for 20 output(s) and 1 input(s).
Response data is available at 624 frequency points, ranging from 37.96 rad/s to 201.1 rad/s.

Sample time: 0.0005 seconds
Output channels: 'y(1)', 'y(2)', 'y(3)', 'y(4)', 'y(5)', 'y(6)', 'y(7)', 'y(8)', 'y(9)', 'y(10)', 'y(11)', 'y(12)', 'y(13)', 'y(14)', 'y(15)', 'y(16)', 'y(17)', 'y(18)', 'y(19)', 'y(20)'
Input channels: 'u1'
Status:
Estimated model
```

Gs thus contains the frequency response measurements at the 20 measurement locations. Next, identify a state-space model to fit Gs. The subspace estimator n4sid provides a quick noniterative estimate. The state-space model structure is configured as follows:

1. We estimate an 24th-order continuous-time model. The order was found after some trials with various orders and checking the resulting fit of the model to the FRF.
2. The model contains a feedthrough term (D matrix is non-zero). This is because we are limiting our analysis to ≤ 35 Hz while the wing's bandwidth is significantly larger than that (response is significant at 35 Hz).
3. To speed up computation, we suppress computation of parameter covariance.
4. The FRF magnitude varies significantly across the frequency range. In order to ensure that the low amplitudes receive equal emphasis as the higher amplitudes, we apply a custom weighting that varies inversely as the square root of the response. Since there are 20 channels, the weighting vector is taken as the mean of the weights obtained from each (this works because the FRFs of the 20 channels are similar in shape).

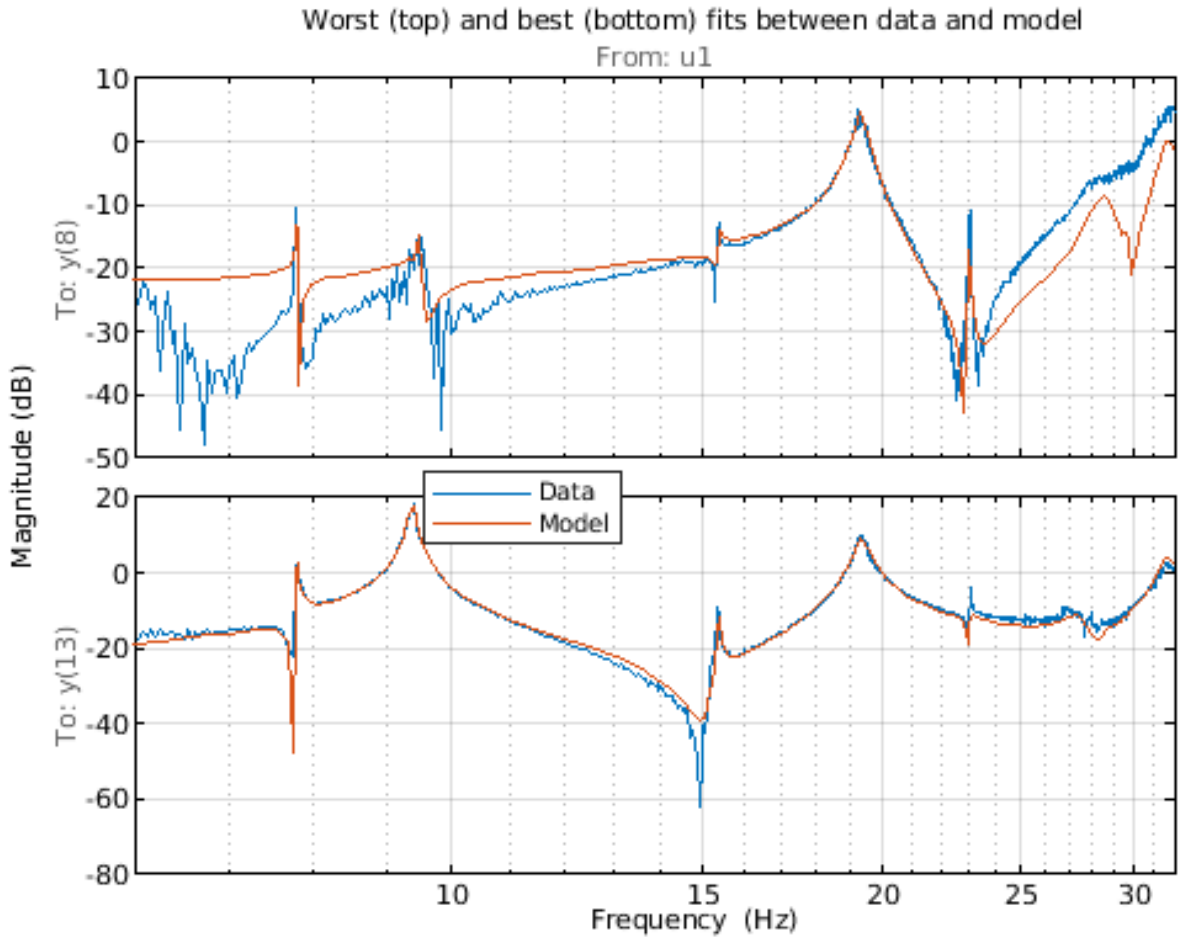
We set up the estimation options for n4sid using n4sidOptions.

```
FRF = squeeze(Gs.ResponseData);
Weighting = mean(1./sqrt(abs(FRF))).';
n4Opt = n4sidOptions('EstimateCovariance',false,...
  'WeightingFilter',Weighting,...
  'OutputWeight',eye(20));
sys1 = n4sid(Gs,24,'Ts',0,'Feedthrough',true,n4Opt);
Fit = sys1.Report.Fit.FitPercent'
```

```
Fit = 1×20

   54.2689   55.3738   84.0188   85.7017   81.9259   80.2293   55.7448   40.6773   58.6429   76.2997   83.5370   84.6555   85.9240   84.8780   76.8937   81.0399   74.8321   79.4069   65.0803   76.3328
```

The "Fit" numbers shows the percentage fit between the data (Gs) and model's (sys1) frequency response using a normalized root-mean-square-error (NRMSE) goodness-of-fit measure. The poorest and best fits are plotted next.

```
[~,Imin] = min(Fit);
[~,Imax] = max(Fit);
clf
bodeplot(Gs([Imin, Imax],:), sys1([Imin, Imax],:), opt);
xlim([6 32])
title('Worst (top) and best (bottom) fits between data and model')
grid on
legend('Data', 'Model')
```



The fits achieved with model sys1 can be improved even more by iterative nonlinear least-squares refinement of model's parameters. This can be achieved using the ssest command. We set up the estimation options for ssest using the ssestOptions command. This time the parameter covariance is also estimated.

```
ssOpt = ssestOptions('EstimateCovariance',true,...
  'WeightingFilter',n4Opt.WeightingFilter,...
  'SearchMethod','lm',...     % use Levenberg-Marquardt search method
  'Display','on',...
  'OutputWeight',eye(20));
sys2 = ssest(Gs, sys1, ssOpt);  % estimate state-space model (this takes several minutes)
```

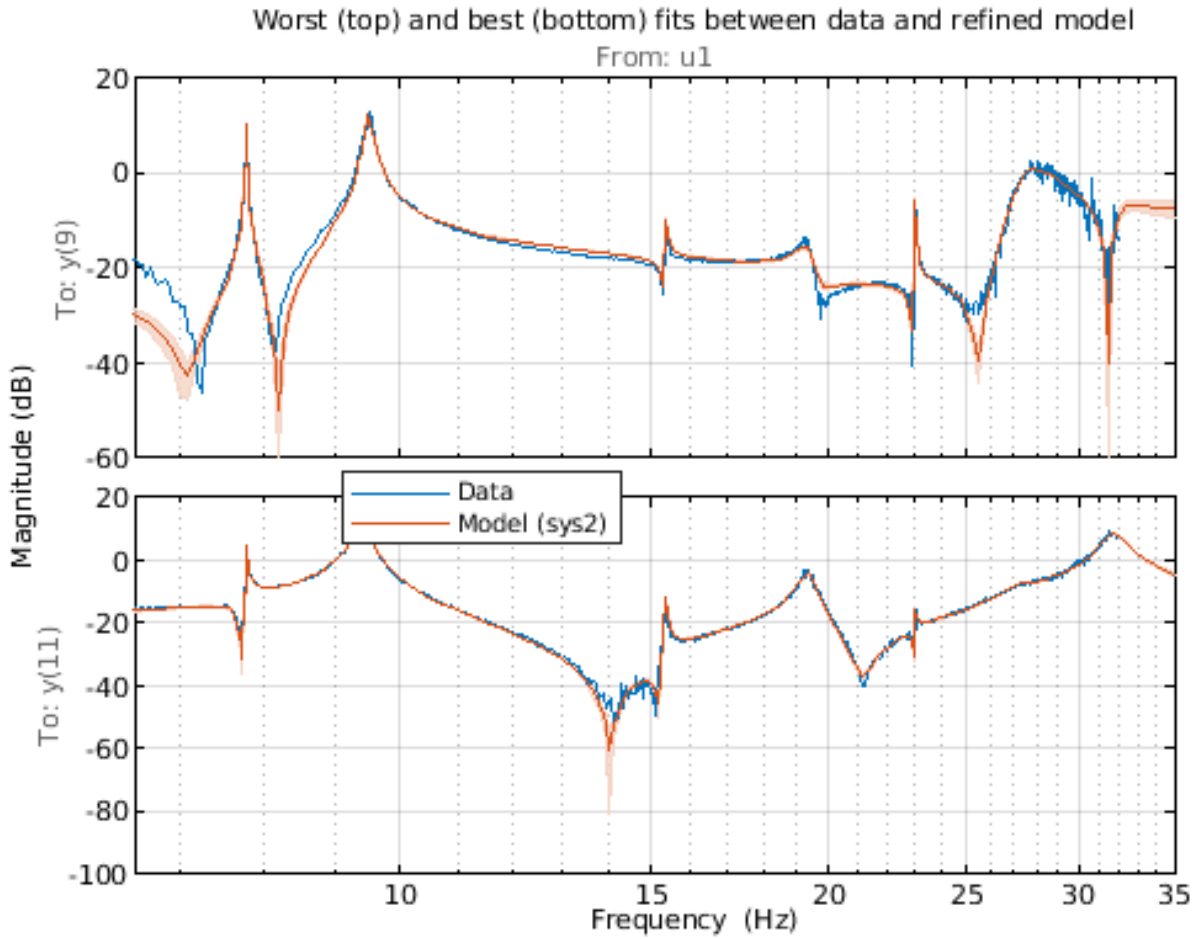
```
Fit = sys2.Report.Fit.FitPercent'
```

Fit = *1×20*

```
   88.8715   88.6587   88.9388   89.8630   87.8597   88.1235   79.7016   82.5930   74.2458   82.6593   90.2663
88.7739   89.9033   89.0571   88.8040   88.2251   86.9307   87.5250   83.2249   82.8676
```

As before we plot the worst and the best fits. We also visualize the 1-sd confidence region for the model's frequency response.

```
[~, Imin] = min(Fit);
[~, Imax] = max(Fit);
clf
h = bodeplot(Gs([Imin, Imax],:), sys2([Imin, Imax],:), opt);
showConfidence(h, 1)
xlim([6.5 35])
title('Worst (top) and best (bottom) fits between data and refined model')
grid on
legend('Data', 'Model (sys2)')
```



The refined state-space model sys2 fits the FRFs quite well in the 7–20 Hz region. The uncertainty bounds are tight around most resonant locations. We estimated a 24th-order model which means that there are at most 12 oscillatory modes in the system sys2. Use the modalfit command to fetch the natural frequencies in Hz for these modes.

```
f = Gs.Frequency/2/pi;     % data frequencies (Hz)
fn = modalfit(sys2, f, 12); % natural frequencies (Hz)
disp(fn)
```

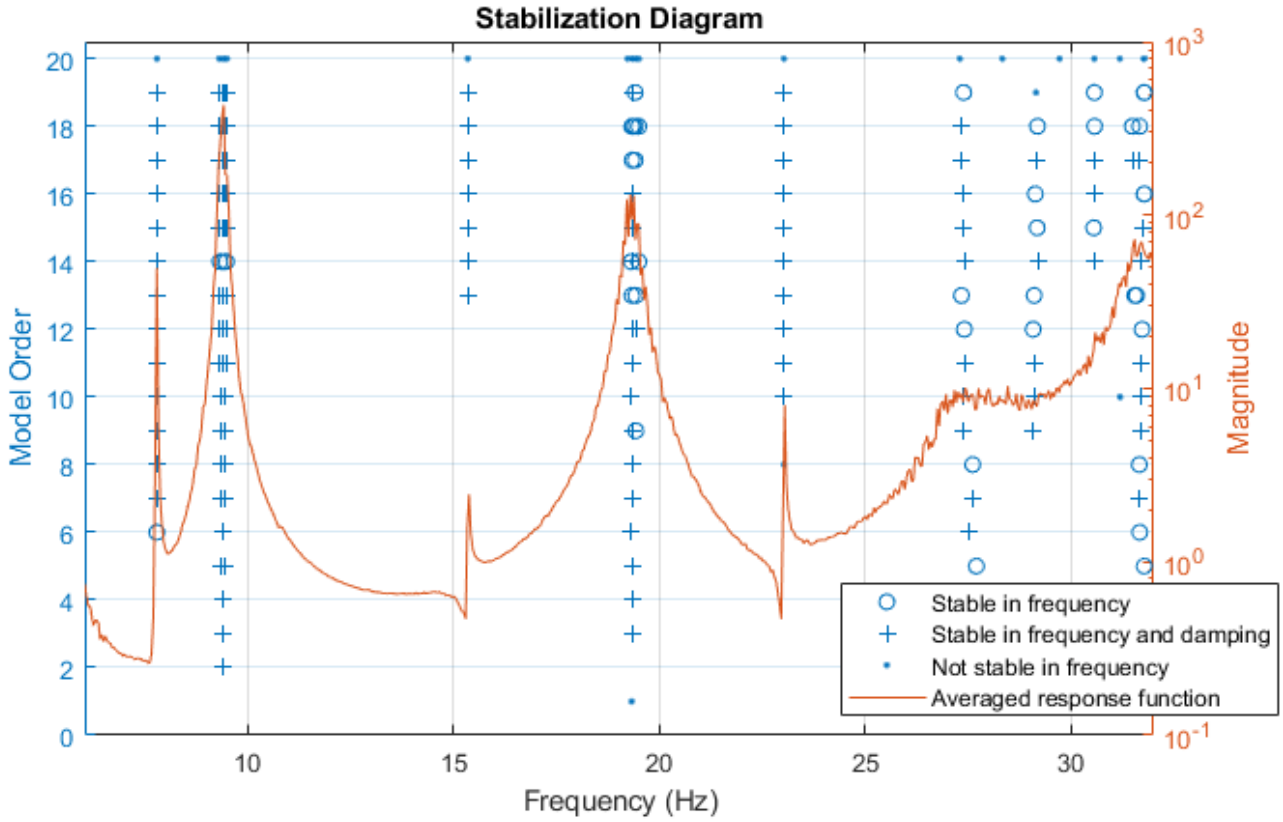
```
    7.7720
    7.7954
    9.3176
    9.4052
    9.4919
   15.3462
   19.3281
   23.0219
   27.4035
   28.6873
   31.7036
   64.2624
```

The values in fn suggest two frequencies very close to 7.8 Hz and three close to 9.4 Hz. An inspection of frequency responses near these frequencies reveals that the peaks location shift a little bit across outputs. These discrepancies may be removed by better control over the experiment process, performing direct time-domain identification with input bandwidth limited to narrow range centered at these frequencies, or fitting a single oscillatory mode to the frequency response around these frequencies. These alternatives are not explored in this example.

Modal Parameter Computation

We can now use the model sys2 to extract the modal parameters. An inspection of the FRFs indicates around 10 modal frequencies, roughly around the frequencies [5 7 10 15 17 23 27 30] Hz. We can make this assessment more concrete by using the modalsd command that checks the stability of modal parameters as the order of the underlying model is changed. This operation takes a long time. Hence the resulting plot is inserted directly as an image. Execute the code inside the comment block below to reproduce the figure.

```
FRF = permute(Gs.ResponseData,[3 1 2]);
f = Gs.Frequency/2/pi;
%{
 figure
 pf = modalsd(FRF,f,fs,'MaxModes',20,'FitMethod','lsrf');
%}
```



Inspection of the plot and pf values suggests a refined list of true natural frequencies:

```
Freq = [7.8 9.4 15.3 19.3 23.0 27.3 29.2 31.7];
```

We use the values in Freq vector as a guide for picking most dominant modes from the model sys2. This is done using the modalfit command.

```
[fn,dr,ms] = modalfit(sys2,f,length(Freq),'PhysFreq',Freq);
```

fn are the natural frequencies (in Hz), dr the corresponding damping coefficients and ms the normalized residuals as column vectors (one column for each natural frequency). In the process of extraction of these modal parameters, only the stable, under-damped poles of the model are used. The ms columns contain data for only the poles with positive imaginary part.

Mode Shape Visualization

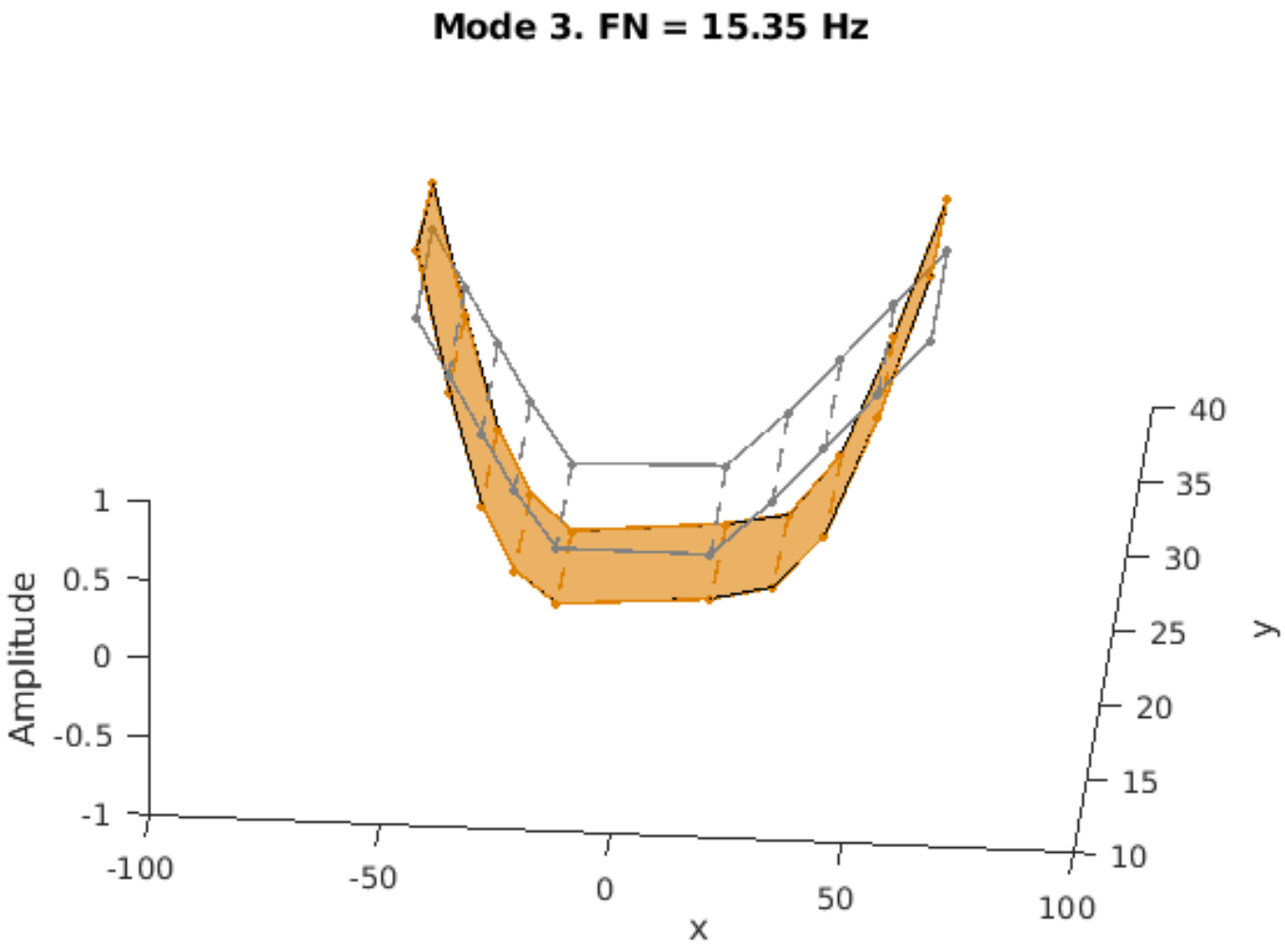
To visualize the various bending modes, we need the modal parameters extracted above. In addition we need information on the location of the measurement points. These positions (x-y values) is recorded for each accelerometer in the matrix AccePos:

```
AccelPos = [...   % see figure 2
  16.63 18.48;   % nearest right of center
  16.63 24.48;
  27.90 22.22;
  27.90 28.22;
  37.17 25.97;
  37.17 31.97;
  46.44 29.71;
```

```
46.44 35.71;
55.71 33.46;
55.71 39.46;  % farthest right
-16.63 18.48; % nearest left of center
-16.63 24.18;
-27.90 22.22;
-27.90 28.22;
-37.17 25.97;
-37.17 31.97;
-46.44 29.71;
-46.44 35.71;
-55.71 33.46;
-55.71 39.46]; % farthest left
```

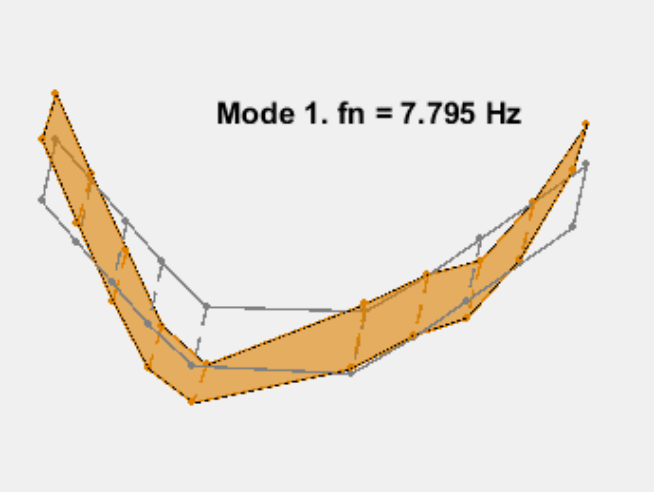
The mode shapes are contained in the matrix ms where each column corresponds to the shape for one frequency. Animate the mode by superimposing mode shape amplitudes over the sensor coordinates and varying the amplitudes at the mode's natural frequency. The animation shows the bending with no damping. As an example, consider the mode at 15.3 Hz.

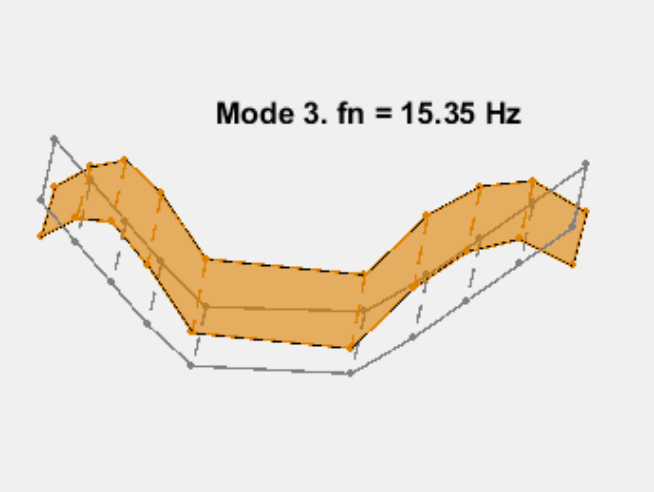
```
AnimateOneMode(3, fn, ms, AccelPos);
```

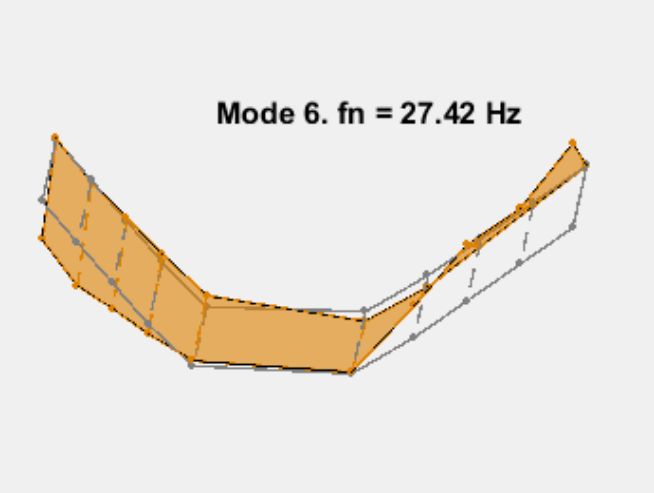


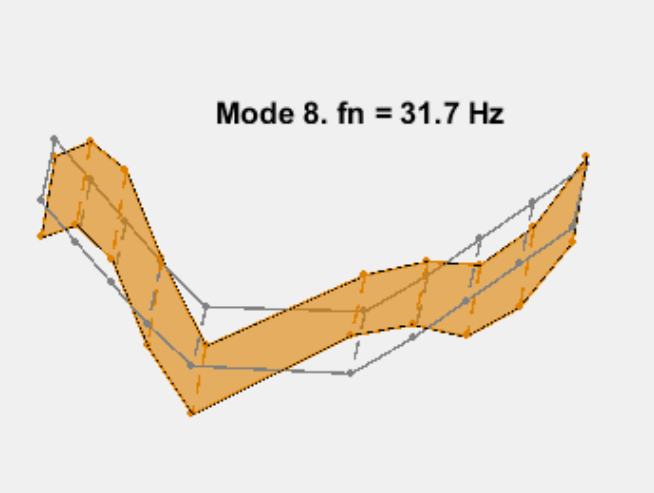
Conclusions

This example shows a parametric model identification based approach to modal parameter estimation. Using a state-space model of 24th order, 8 stable oscillatory modes in the frequency region 6–32 Hz were extracted. The modal information was combined with the knowledge of the accelerometer positions to visualize the various bending modes. Some of these modes are shown in the figures below.









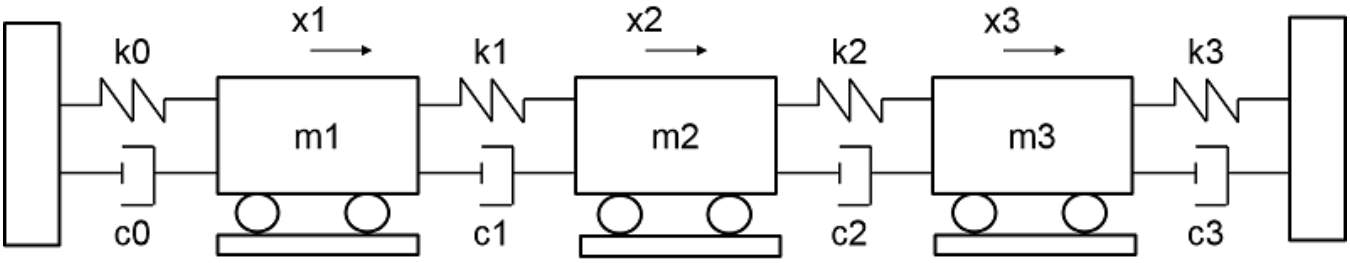
LAB PROJECT # 6:
Modal Analysis of a Simulated System and a Wind Turbine Blade

This example shows how to estimate frequency-response functions (FRFs) and modal parameters from experimental data. The first section describes a simulated experiment that excites a three-degree-of-freedom (3DOF) system with a sequence of hammer impacts and records the resulting displacement. Frequency-response functions, natural frequencies, damping ratios, and mode shape vectors are estimated for three modes of the structure. The second section estimates mode shape vectors from frequency-response function estimates from a wind turbine blade experiment. The turbine blade measurement configuration and resulting mode shapes are visualized. This example requires System Identification Toolbox (TM).

Natural Frequency and Damping for a Simulated Beam

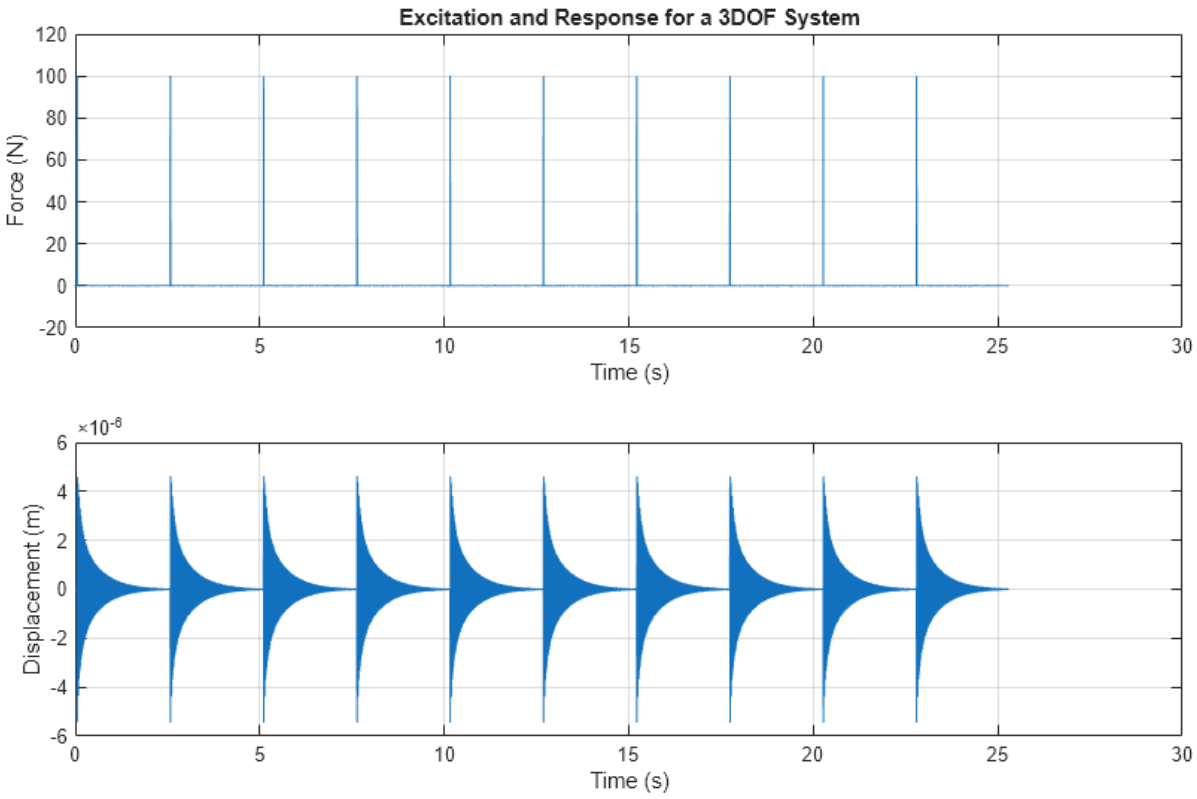
Single-Input/Single-Output Hammer Excitation

A series of hammer strikes excite a 3DOF system, and sensors record the resulting displacements. The system is proportionally damped, such that the damping matrix is a linear combination of the mass and stiffness matrices.



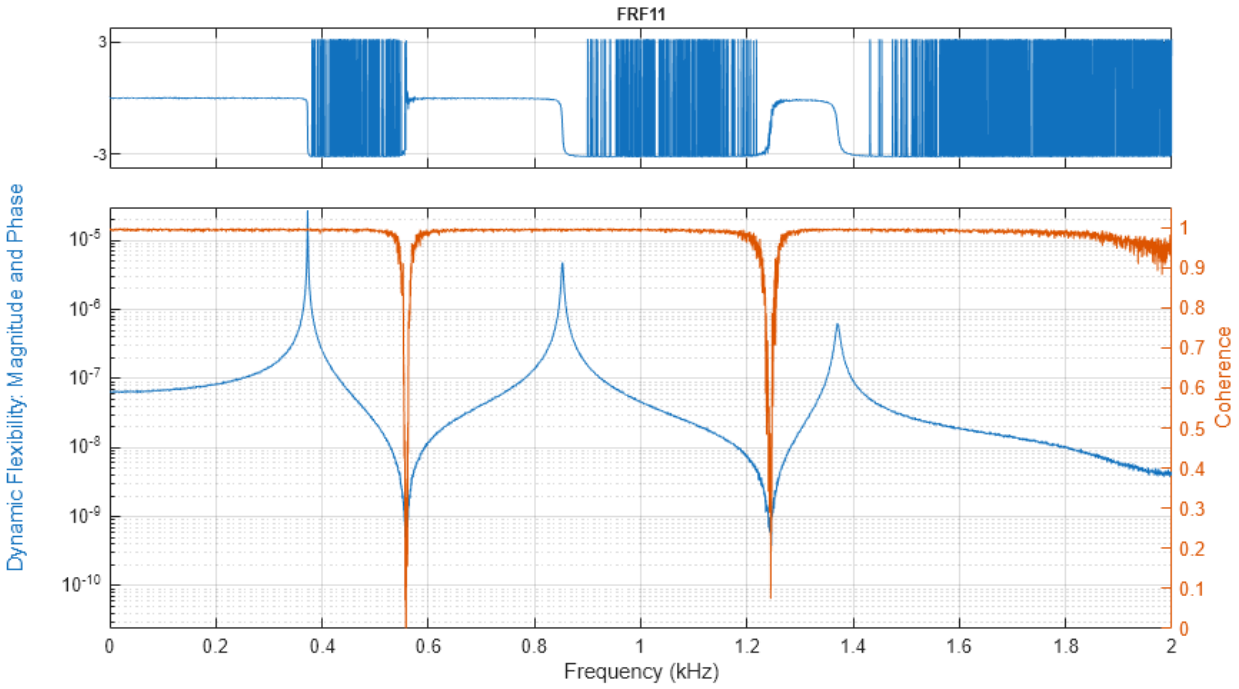
Import the data for two sets of measurements, including excitation signals, response signals, time signals, and ground truth frequency-response functions. The first set of response signals, `Y1`, measures the displacement of the first mass, and the second, `Y2`, measures the second mass. Each excitation signal consists of ten concatenated hammer impacts, and each response signal contains the corresponding displacement. The duration for each impact signal is 2.53 seconds. Additive noise is present in the excitation and response signals. Visualize the first excitation and response channel of the first measurement.

```
[t,fs,X1,X2,Y1,Y2,f0,H0] = helperImportModalData();

X0 = X1(:,1);

Y0 = Y1(:,1);

helperPlotModalAnalysisExample([t' X0 Y0]);
```



Compute and plot the FRF for the first excitation and response channels in terms of dynamic flexibility, which is a measure of displacement over force [1]. By default, the FRF is computed by averaging spectra of windowed segments. Since each hammer excitation decays substantially before the next excitation, a rectangular window can be used. Specify the sensor as displacement.

```
winLen = 2.5275*fs; % window length in samples

modalfrf(X0,Y0,fs,winLen,'Sensor','dis')
```



The FRF, estimated using the default `'H1'` estimator, contains three prominent peaks in the measured frequency band, corresponding to three flexible modes of vibration. The coherence is close to one near these peaks, and low in anti-resonance regions, where the signal-to-noise ratio of the response measurement is low. Coherence near to one indicates a high quality estimate. The `'H1'` estimate is optimal where noise exists only at the output measurement, whereas the `'H2'` estimator is optimal when there is additive noise only on the input [2]. Compute the `'H1'` and `'H2'` estimates for this FRF.

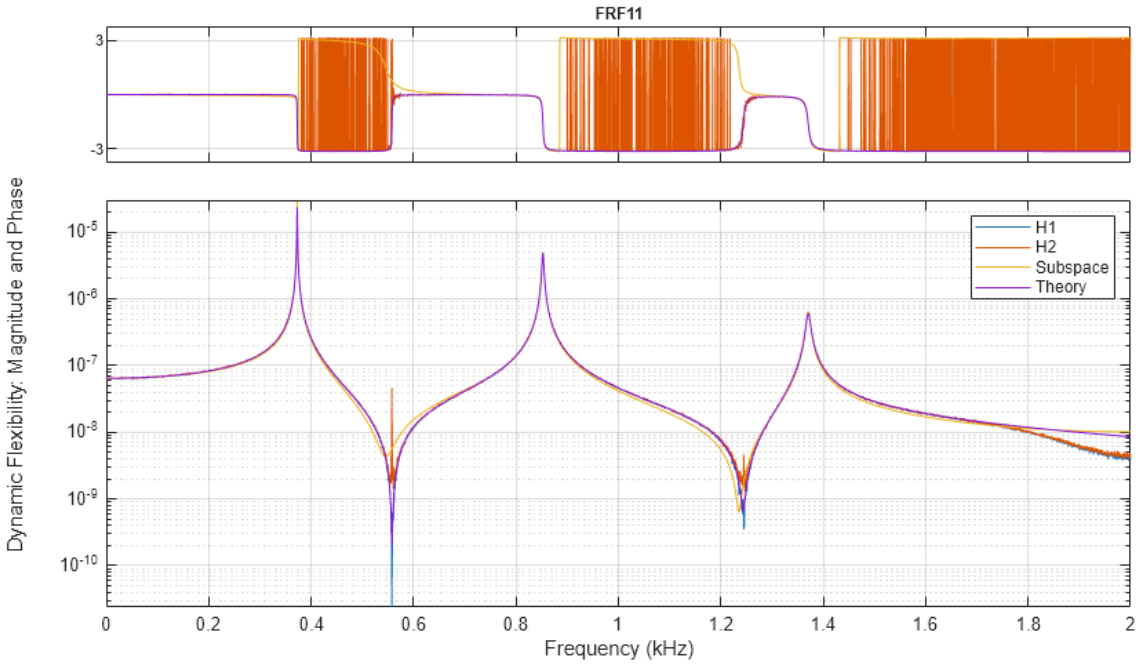
```
[FRF1,f1] = modalfrf(X0,Y0,fs,winLen,'Sensor','dis');  % Calculate FRF (H1)

[FRF2,f2] = modalfrf(X0,Y0,fs,winLen,'Sensor','dis','Estimator','H2');
```

When there is significant measurement noise or the excitation is poor, parametric methods can offer additional options for accurately extracting the FRF from the data. The 'subspace' method first fits a state-space model to the data [3] and then computes its frequency-response function. The order of the state-space model (equal to the number of poles) and presence or lack of feedthrough can be specified to configure the state-space estimation.

```
[FRF3,f3] =
modalfrf(X0,Y0,fs,winLen,'Sensor','dis','Estimator','subspace','Feedthrough',true);
```

Here FRF3 is estimated by fitting a state-space model containing a feedthrough term and of the optimal order in the range 1:10. Compare the estimated FRFs using `'H1'`, `'H2'` and `'subspace'` methods to the theoretical FRF.

```
helperPlotModalAnalysisExample(f1,FRF1,f2,FRF2,f3,FRF3,f0,H0);
```



The estimators perform comparably near response peaks, while the `'H2'` estimator overestimates the response at the antiresonances. The coherence is not affected by the choice of the estimator.

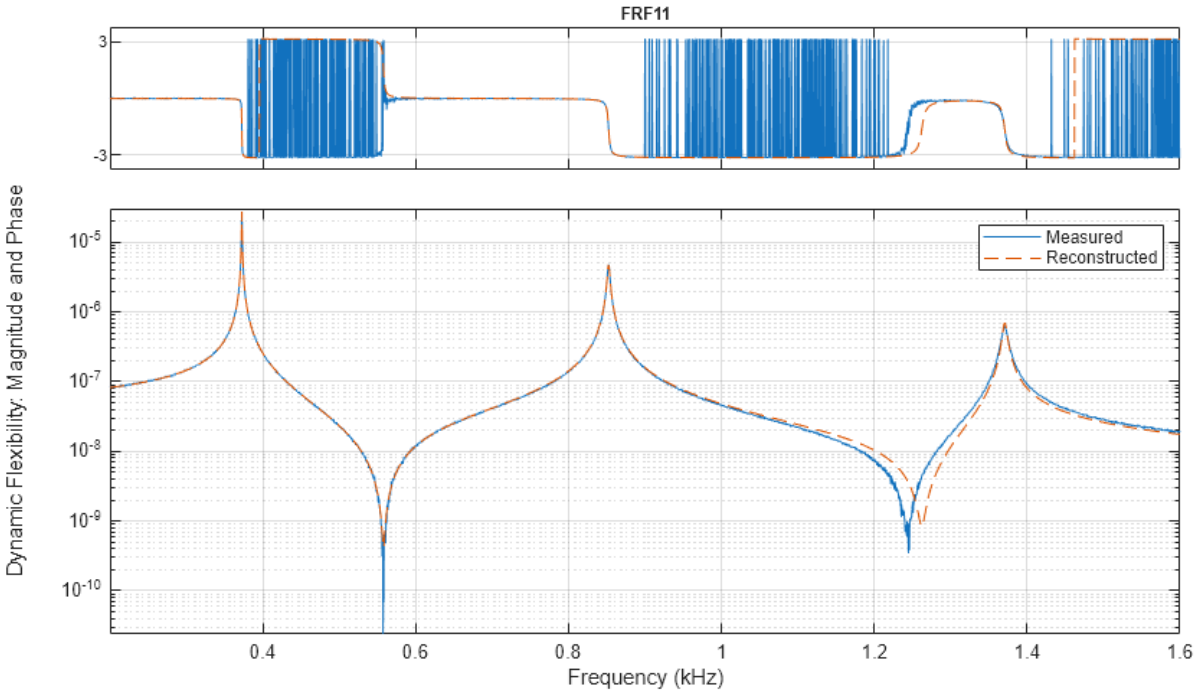
Next, estimate the natural frequency of each mode using the peak-picking algorithm. The peak-picking algorithm is a simple and fast procedure for identifying peaks in the FRF. It is a local method, since each estimate is generated from a single frequency-response function. It is also a single-degree-of-freedom (SDOF) method, since the peak for each mode is considered independently. As a result, a set of modal parameters is generated for each FRF. Based on the previous plot, specify a frequency range from 200 to 1600 Hz, which contains the three peaks.

```
fn = modalfit(FRF1,f1,fs,3,'FitMethod','PP','FreqRange',[200 1600])
```

```
fn =


   1.0e+03 *


    0.3727

    0.8525

    1.3707
```

The natural frequencies are approximately 373, 853, and 1371 Hz. Plot a reconstructed FRF and compare it to the measured data using `modalfit`. The FRF is reconstructed using the modal parameters estimated from the frequency-response function matrix, `FRF1`. Call `modalfit` again with no output arguments to produce a plot containing the reconstructed FRF.

```
modalfit(FRF1,f1,fs,3,'FitMethod','PP','FreqRange',[200 1600])
```

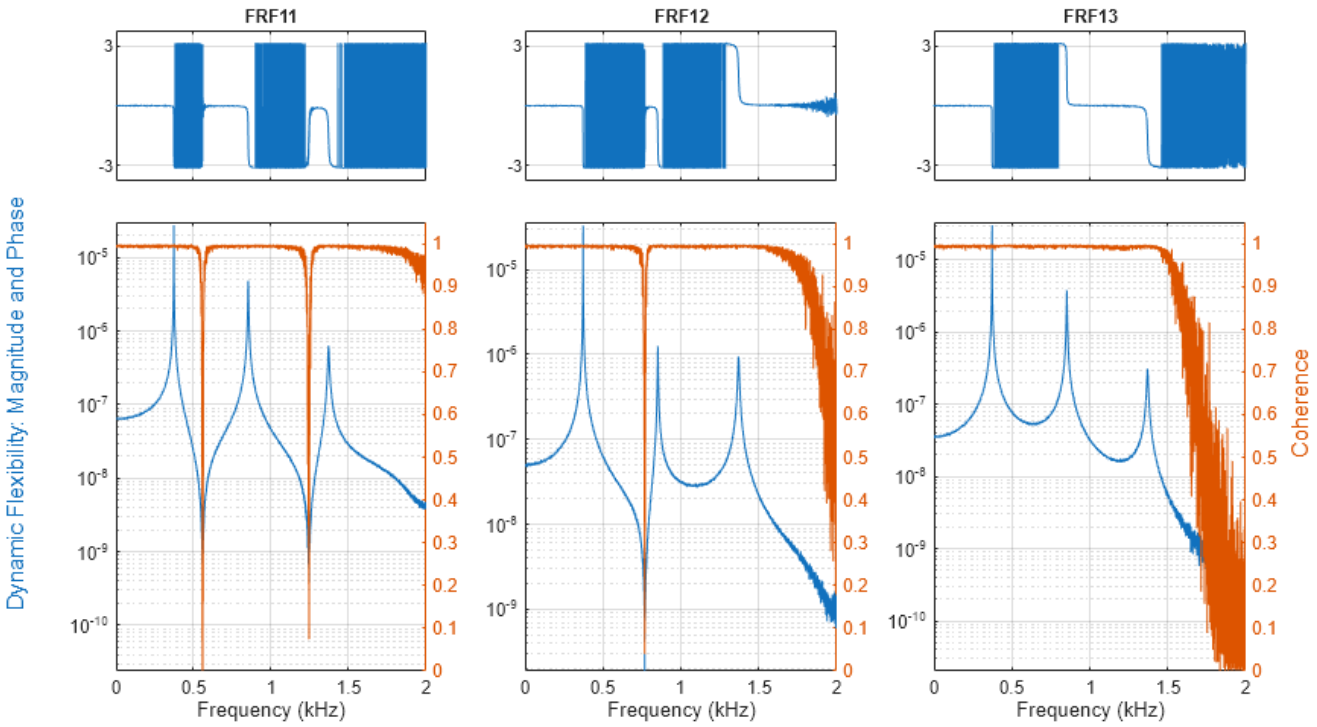


The reconstructed FRF agrees with the measured FRF of the first excitation and response channels. In the next section, two additional excitation locations are considered.

Roving Hammer Excitation

Compute and plot the FRFs for the responses of all three sensors using the default `'H1'` estimator. Specify the measurement type as `'rovinginput'` since we have a roving hammer excitation.

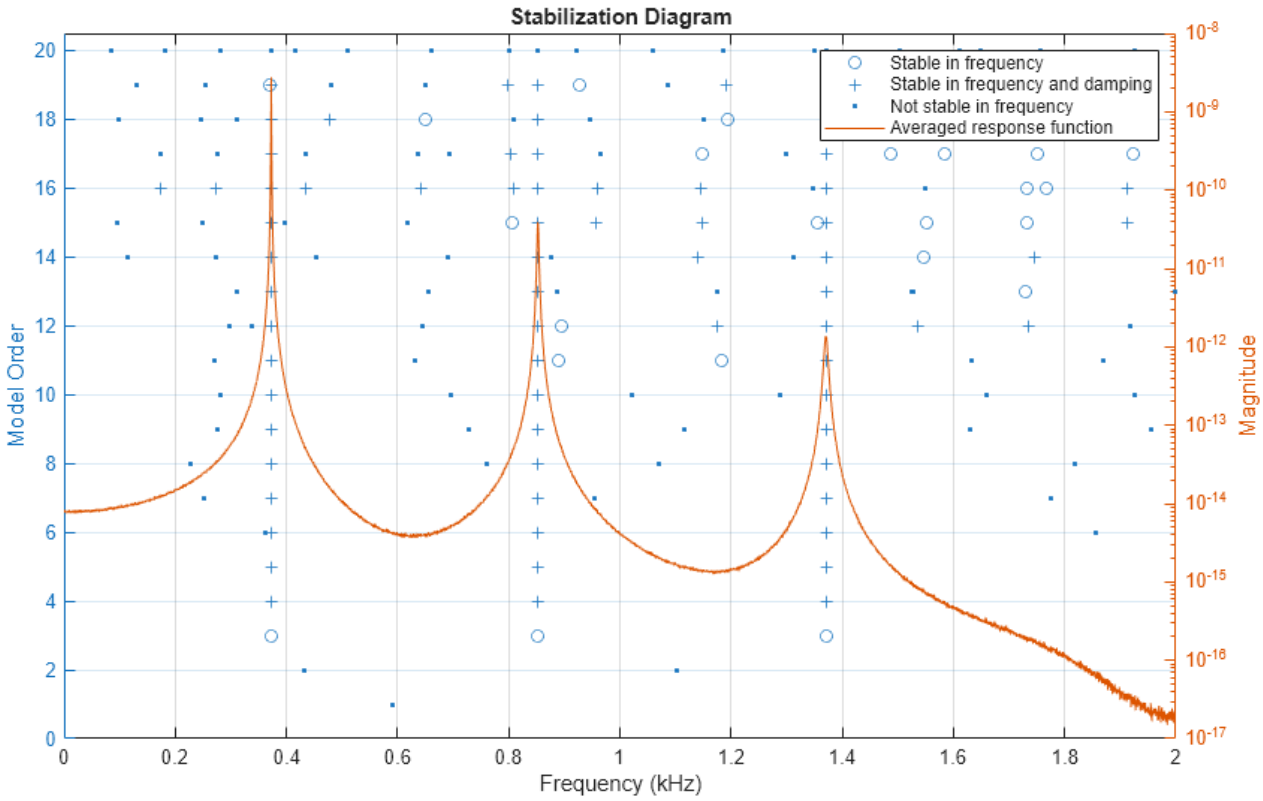
```
modalfrf(X1,Y1,fs,winLen,'Sensor','dis','Measurement','rovinginput')
```



In the previous section, a single set of modal parameters was computed from a single FRF. Now, estimate modal parameters using the least-squares complex exponential (LSCE) algorithm. The LSCE and LSRF algorithms generate a single set of modal parameters by analyzing multiple response signals simultaneously. These are global, multiple-degree-of-freedom (MDOF) methods, since the parameters for all modes are estimated simultaneously from multiple frequency-response functions.

The LSCE algorithm generates computational modes, which are not physically present in the structure. Use a stabilization diagram to identify physical modes by examining the stability of poles as the number of modes increases. The natural frequencies and damping ratios of physical modes tend to remain in the same place, or are 'stable'. Create a stabilization diagram and output the natural frequencies of those poles which are stable in frequency.

```
[FRF,f] = modalfrf(X1,Y1,fs,winLen,'Sensor','dis','Measurement','rovinginput');

fn = modalsd(FRF,f,fs,'MaxModes',20, 'FitMethod', 'lsce'); % Identify physical modes
```

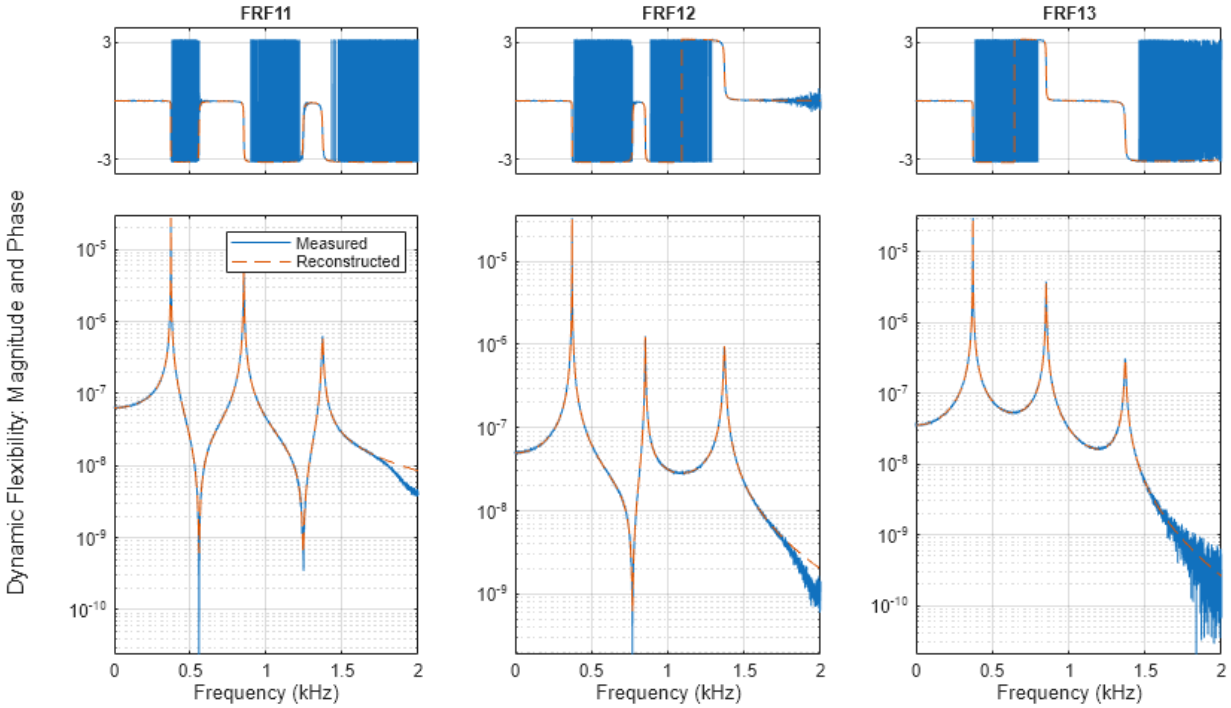


By default, poles are classified as stable in frequency if the natural frequency for the poles changes by less than one percent. Poles which are stable in frequency are further classified as stable in damping for a smaller than five percent change in damping ratio. Both criteria can be adjusted to different values. Based on the location of the stable poles, choose natural frequencies of 373, 852.5, and 1371 Hz. These frequencies are contained in the output `fn` of `modalsd`, along with natural frequencies of other frequency-stable poles. A higher model order than the number of modes physically present is generally needed to produce good modal parameters estimates using the LSCE algorithm. In this case, a model order of four modes indicates three stable poles. The frequencies of interest occur in the first three columns in the 4th row of `fn`.

```
physFreq = fn(4,[1 2 3]);
```

Estimate natural frequencies and damping and plot reconstructed and measured FRFs. Specify four modes and physical frequencies determined from the stability diagram, `'PhysFreq'`. `modalfit` returns modal parameters only for the specified modes.

```
modalfit(FRF,f,fs,4,'PhysFreq',physFreq)
```

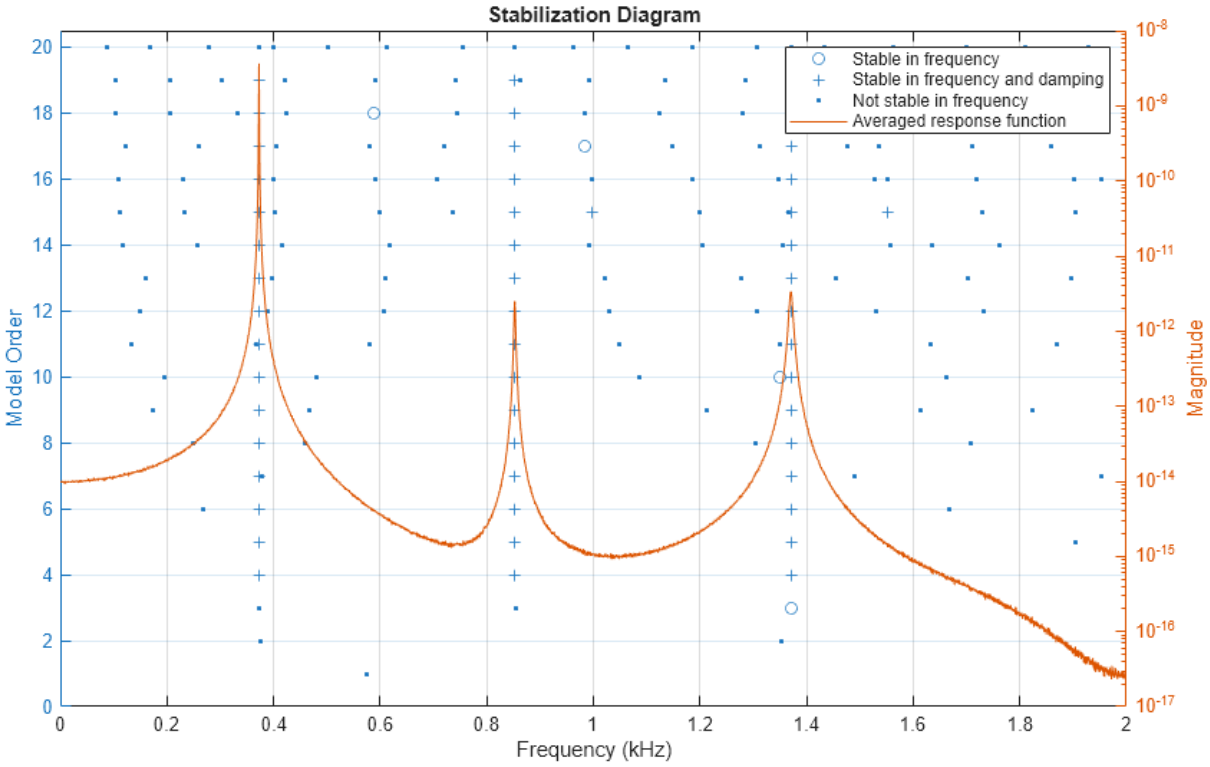



```
[fn1,dr1] = modalfit(FRF,f,fs,4,'PhysFreq',physFreq)
```

```
fn1 =

   1.0e+03 *

    0.3727
    0.8525
    1.3706


dr1 =

    0.0008
    0.0018
    0.0029
```

Next, compute FRFs and plot a stabilization diagram for a second set of hammer impacts with the sensor at a different location. Change the stability criterion to 0.1 percent for frequency and 2.5 percent for damping.

```
[FRF,f] = modalfrf(X2,Y2,fs,winLen,'Sensor','dis','Measurement','rovinginput');

fn = modalsd(FRF,f,fs,'MaxModes',20,'SCriteria',[0.001 0.025]);
```



With the more stringent criteria, the majority of poles are classified as not stable in frequency. The poles that are stable in frequency and damping align closely with the averaged FRF, suggesting they are present in the measured data.

```
physFreq = fn(4,[1 2 3]);
```

Extract modal parameters for this set of measurements and compare to the modal parameters for the first set of measurements. Specify the indices of the driving point FRF, corresponding to location where the excitation and response measurements coincide. The natural frequencies agree to within a fraction of a percent, and the damping ratios agree to within less than four percent, indicating that the modal parameters are consistent from measurement to measurement.

```
[fn2,dr2] = modalfit(FRF,f,fs,4,'PhysFreq',physFreq,'DriveIndex',[1 2])
```

```
fn2 =


   1.0e+03 *


    0.3727

    0.8525
```

```
    1.3705
```

```
dr2 =
```

```
    0.0008
    0.0018
    0.0029
```

```
Tdiff2 = table((fn1-fn2)./fn1,(dr1-dr2)./dr1,'VariableNames',{'diffFrequency','diffDamping'})
```

```
Tdiff2 =
```

```
  3×2 table

    diffFrequency    diffDamping
    _____________    ___________

      2.9972e-06      -0.031648
     -5.9335e-06     -0.0099076
       1.965e-05      0.0001186
```

A parametric method for modal parameter estimation can provide a useful alternative to peak-picking and the LSCE method when there is measurement noise in the FRF or the FRF shows high modal density. The least-squares rational function (LSRF) approach fits a shared denominator transfer function to the multi-input, multi-output FRF and thus obtains a single, global estimate of modal parameters [4]. The procedure for using LSRF approach is similar to that for LSCE. You can use stabilization diagram to identify stationary modes and extract modal parameters corresponding to the identified physical frequencies.

```
[FRF,f] = modalfrf(X1,Y1,fs,winLen,'Sensor','dis','Measurement','rovinginput');
fn = modalsd(FRF,f,fs,'MaxModes',20, 'FitMethod', 'lsrf'); % Identify physical modes using lsfr
physFreq = fn(4,[1 2 3]);
[fn3,dr3] = modalfit(FRF,f,fs,4,'PhysFreq',physFreq,'DriveIndex',[1 2],'FitMethod','lsrf')
```

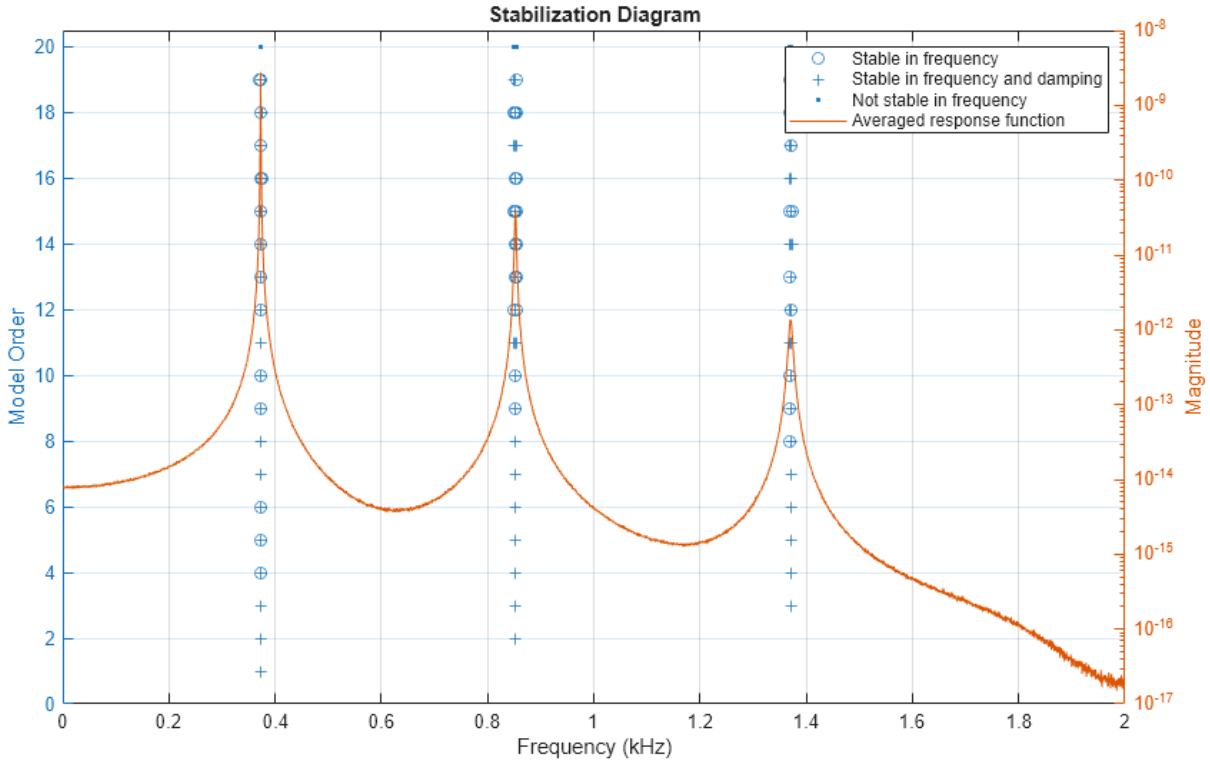
```
fn3 =
```

```
  372.6832
```

```
  372.9275
  852.4986
```

```
dr3 =
```

```
    0.0008
    0.0003
    0.0018
```




```
Tdiff3 = table((fn1-fn3)./fn1,(dr1-dr3)./dr1,'VariableNames',{'diffFrequency','diffDamping'})
```

```
Tdiff3 =
```

```
  3×2 table

    diffFrequency    diffDamping
    _____________    ___________
```

```
    -7.8599e-06      0.007982

        0.56255       0.83086

        0.37799       0.37626
```

A final note about parametric methods: the FRF estimation method ('subspace') and the modal parameter estimation method (`'lsrf'`) are similar to those used in System Identification Toolbox for fitting dynamic models to time-domain signals or to the frequency-response functions. If you have this toolbox available, you can identify models to fit your data using commands such as `tfest` and `ssest`. You can assess the quality of the identified models using `compare` and `resid` commands. Once you have validated the quality of the model, you can use them for extracting modal parameters. This is shown briefly using a state-space estimator.

```
Ts = 1/fs; % sample time

% Create a data object to be used for model estimation.

EstimationData = iddata(Y0(1:1000), X0(1:1000), 1/fs);

% Create a data object to be used for model validation

ValidationData = iddata(Y0(1001:2000), X0(1001:2000), 1/fs);
```

Identify a continuous-time state-space model of 6th order containing a feedthrough term.

```
sys = ssest(EstimationData, 6, 'Feedthrough', true)
```

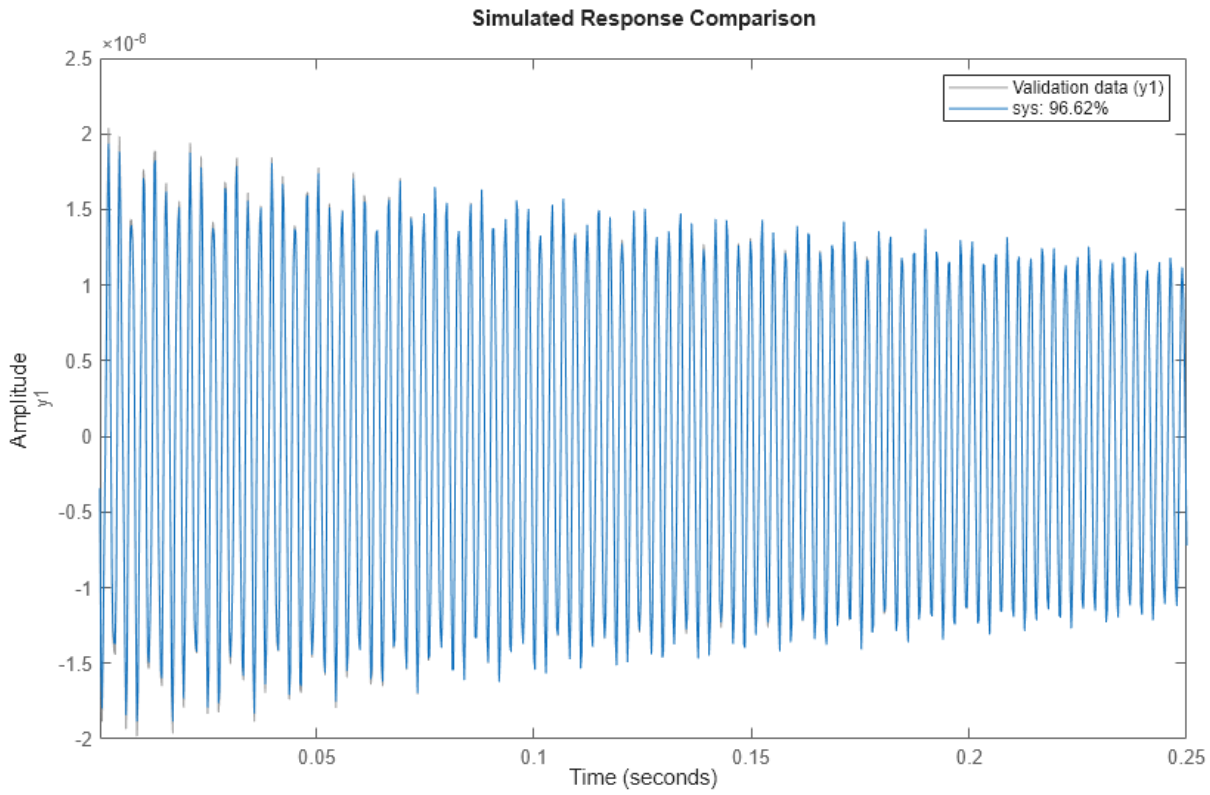
```
sys =

  Continuous-time identified state-space model:

      dx/dt = A x(t) + B u(t) + K e(t)

       y(t) = C x(t) + D u(t) + e(t)


  A =

           x1       x2       x3       x4       x5       x6

   x1    4.041     1765   -149.8     1880   -49.64     -358

   x2    -1764  -0.3434     2197   -232.5    438.3    128.4

   x3    152.4    -2198    2.839     4715   -255.9   -547.5

   x4    -1880    228.2    -4714   -15.91     1216    28.79

   x5    59.42   -440.9    275.5    -1217    35.03    -8508

   x6    363.7   -120.2    545.4    44.02     8508   -92.47


  B =

            u1

   x1   -0.2777
```

```
   x2   -0.6085
   x3   0.07123
   x4    -3.658
   x5  -0.04771
   x6     7.642

  C =
              x1          x2          x3          x4          x5          x6
   y1   -4.46e-05  -5.315e-06   -7.46e-06  -1.641e-05  -2.964e-06   5.871e-06

  D =
             u1
   y1  7.997e-09

  K =
              y1
   x1  -3.513e+07
   x2  -3.244e+06
   x3  -3.598e+07
   x4  -1.059e+07
   x5  -1.724e+08
   x6  -7.521e+06

Parameterization:
   FREE form (all coefficients in A, B, C free).
   Feedthrough: yes
   Disturbance component: estimate
   Number of free coefficients: 55
   Use "idssdata", "getpvec", "getcov" for parameters and their uncertainties.

Status:
Estimated using SSEST on time domain data "EstimationData".
Fit to estimation data: 99.26% (prediction focus)
```

```
FPE: 1.355e-16, MSE: 1.304e-16
```

Assess the quality of the model by checking how well it fits the validation data.

```
clf

compare(ValidationData, sys)  % plot shows good fit
```

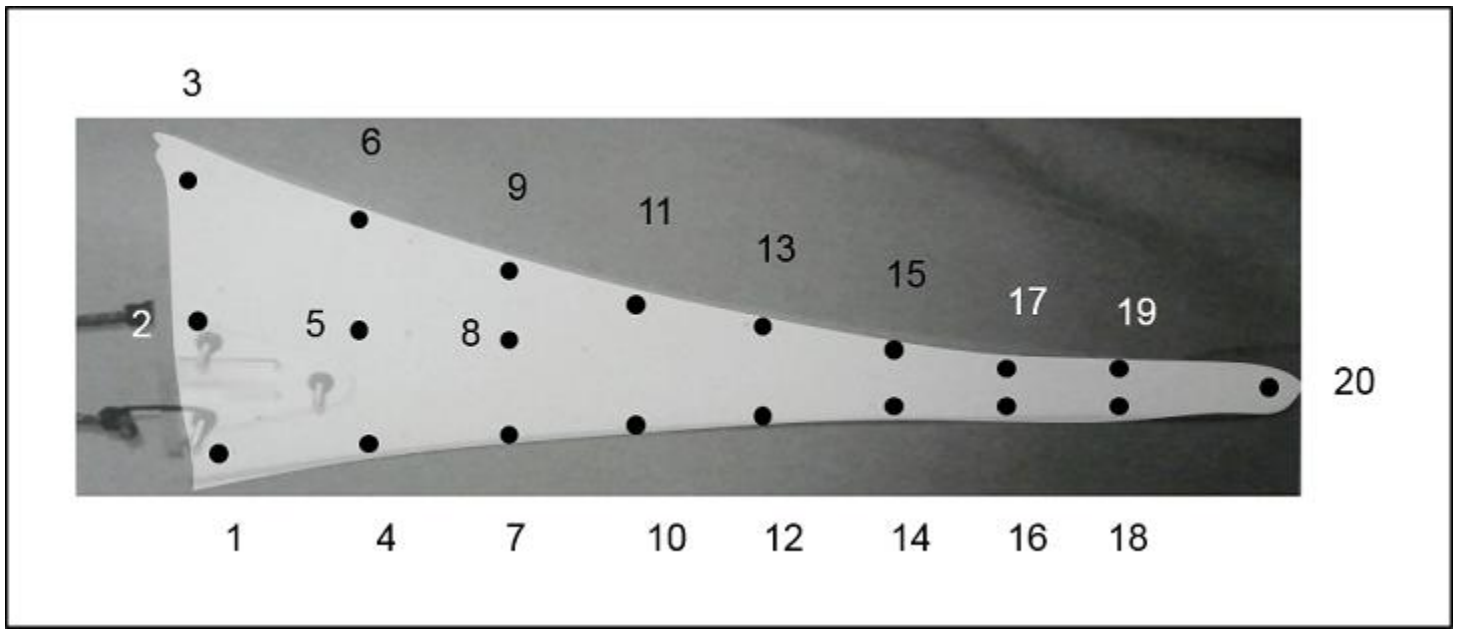


Use the model `sys` to compute modal parameters.

```
[fn4, dr4] = modalfit(sys, f, 3);
```

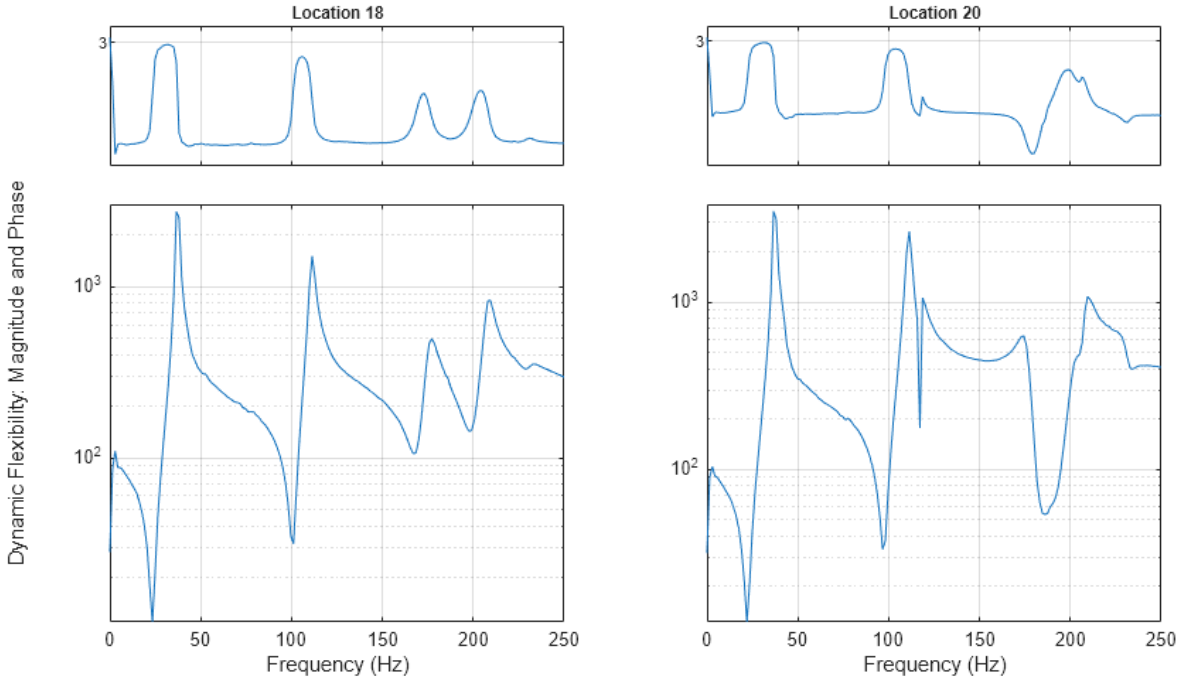
Mode Shape Vectors of a Wind Turbine Blade

Understanding the dynamic behavior of wind turbine blades is important to optimize operational efficiency and predict blade failure. This section analyzes experimental modal analysis data for a wind turbine blade and visualizes mode shapes of the blade. A hammer excites the turbine blade at 20 locations, and a reference accelerometer measures the responses at location 18. An aluminum block is mounted at the base of the blade, and the blade is excited in the flap-wise orientation, perpendicular to the flat part of the blade. An FRF is collected for each location. The FRF data are kindly provided by the Structural Dynamics and Acoustics Systems Laboratory at the University of Massachusetts, Lowell. First, visualize the spatial arrangement of the measurement locations.



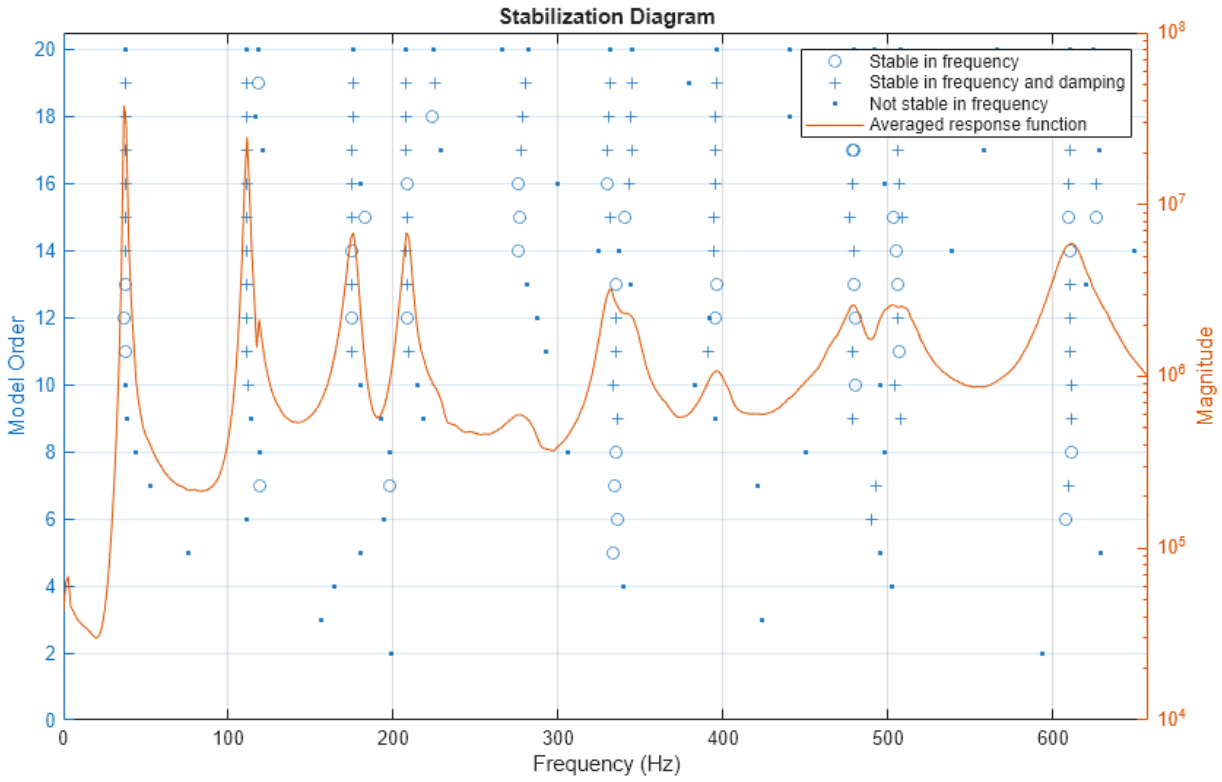
Load and plot the wind turbine blade FRF estimates for locations 18 and 20. Zoom in on the first few peaks.

```
[FRF,f,fs] = helperImportModalData();

helperPlotModalAnalysisExample(FRF,f,[18 20]);
```



The first two modes appear as peaks around 37 Hz and 111 Hz. Plot a stabilization diagram to estimate the natural frequencies. The first two values returned for a model order of 14 are stable in frequency and damping ratio.

```
fn = modalsd(FRF,f,fs,'MaxModes',20);

physFreq = fn(14,[1 2]);
```

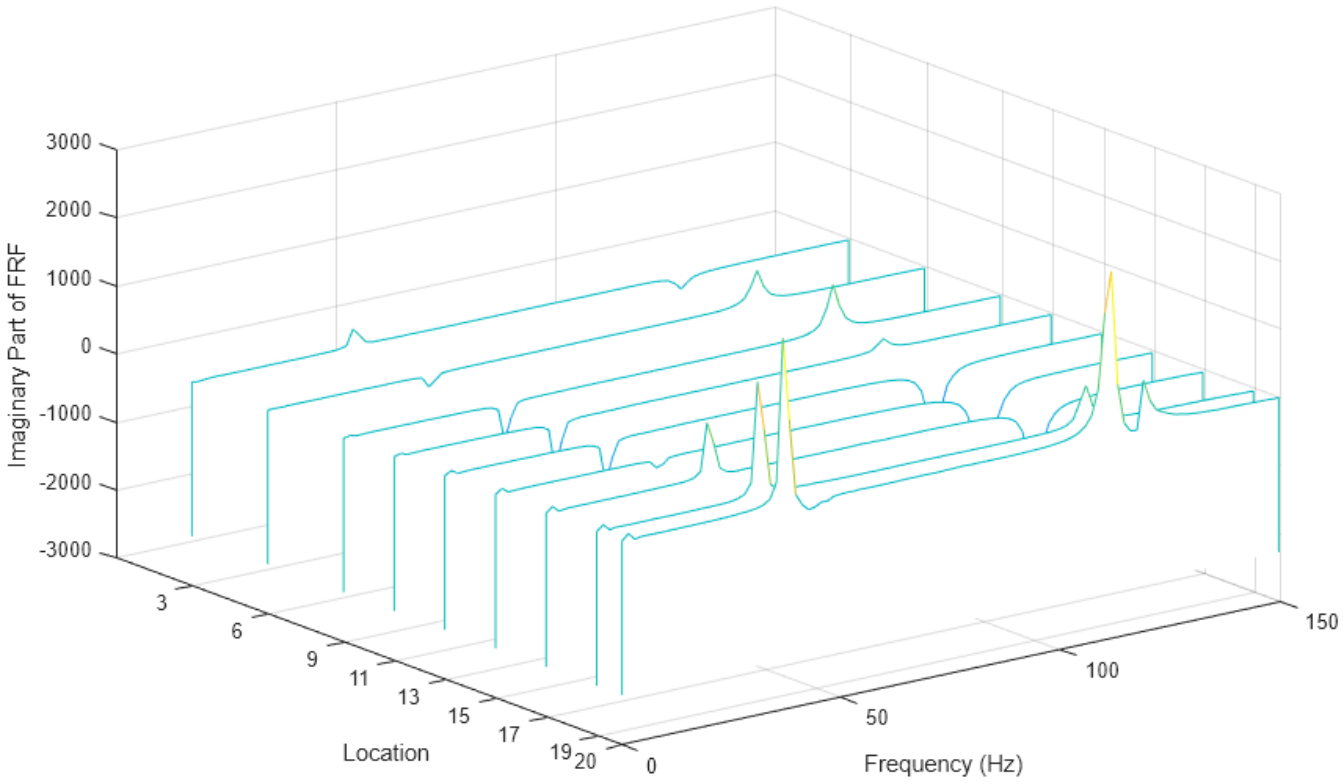


Next, extract the mode shapes for the first two modes using `modalfit`. Limit the fit to the frequency range from 0 to 250 Hz based on the previous plot.

```
[~,~,ms] = modalfit(FRF,f,fs,14,'PhysFreq',physFreq,'FreqRange',[0 250]);
```

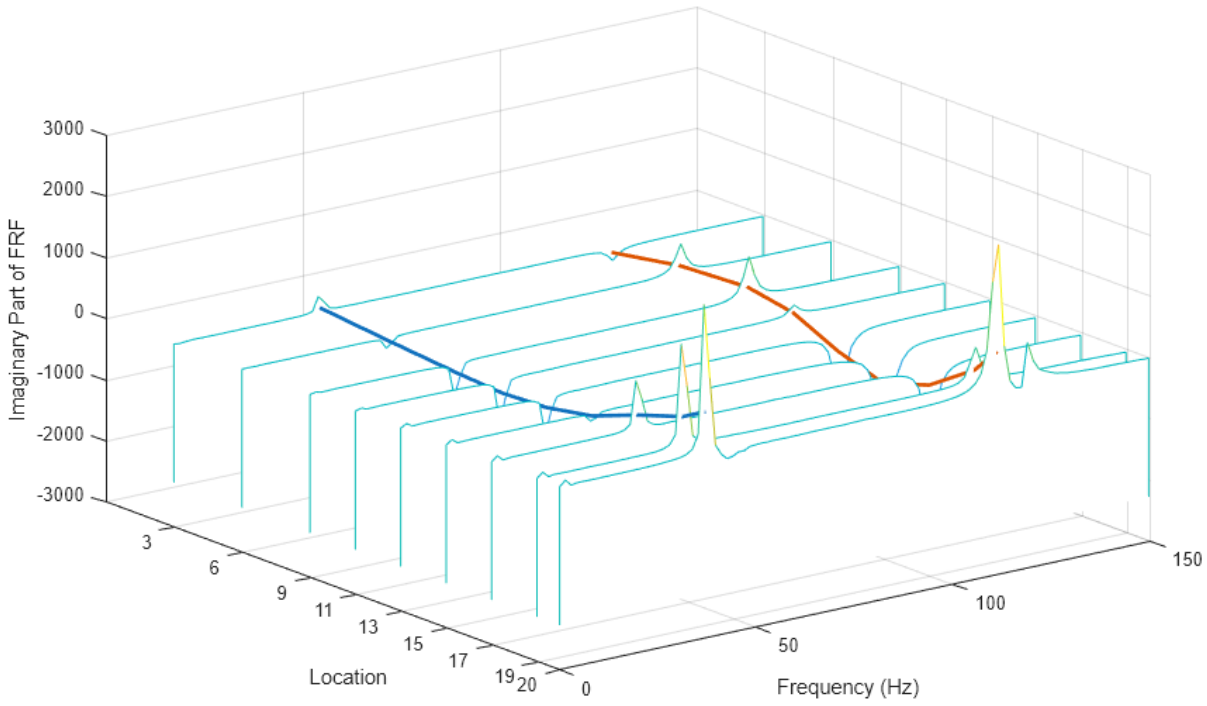
Mode shapes quantify the amplitude of motion for each mode of a structure at each location. To estimate a mode shape vector, one row or column of the frequency-response function matrix is needed. In practice, this means that either an excitation is needed at every measurement location of the structure (in this case, a roving hammer), or that a response measurement is needed at every location. Mode shapes can be visualized by examining the imaginary part of the FRF. Plot a waterfall diagram of the imaginary part of the FRF matrix for locations on one side of the blade. Limit the frequency range to a maximum of 150 Hz to examine the first two modes. The peaks of the plot represent mode shapes.

```
measlocs = [3 6 9 11 13 15 17 19 20]; % Measurement locations on blade edge

helperPlotModalAnalysisExample(FRF,f,measlocs,150);
```



The shapes indicated in the plot by the contour of the peaks represent the first and second bending moments of the blade. Next, plot the magnitude of mode shape vectors for the same measurement locations.

```
helperPlotModalAnalysisExample(ms,measlocs);
```



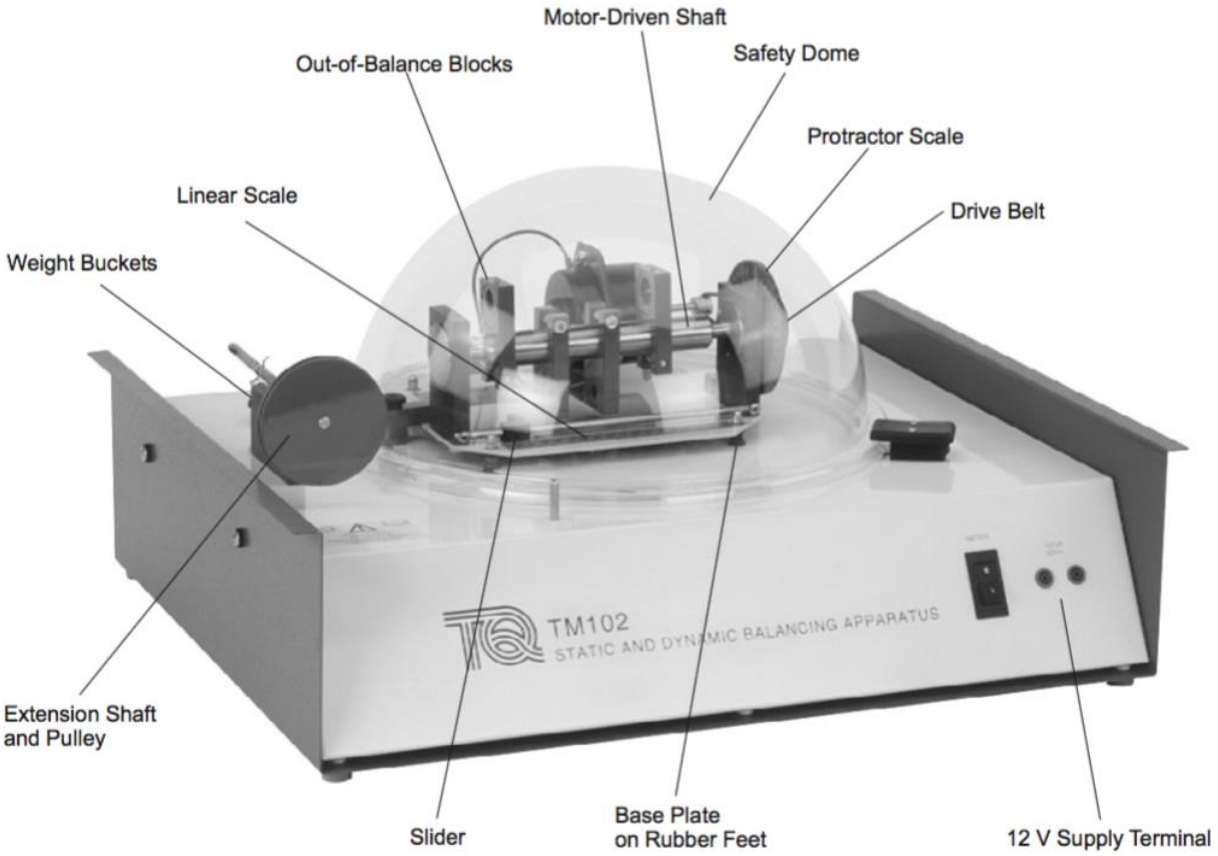
While the amplitudes are scaled differently (mode shape vectors are scaled to unity modal A), the mode shape contours agree in shape. The shape of the first mode has a large tip displacement and two nodes, where vibration amplitude is zero. The second mode also has a large tip displacement and has three nodes.

Summary

This example analyzed and compared simulated modal analysis datasets for a 3DOF system excited by a roving hammer. It estimated natural frequency and damping using a stabilization diagram and the LSCE and LSRF algorithms. The modal parameters were consistent for two sets of measurements. In a separate use case, mode shapes of a wind turbine blade were visualized using the imaginary part of the FRF matrix and mode shape vectors.

LAB PROJECT # 7:
Balancing of Machines: Static and Dynamic Balancing

APPARATUS: TM102 Static and Dynamic Balancing Apparatus



INTRODUCTION:

A shaft with masses mounted on it can be both statically and dynamically balanced. If it is statically balanced, it can stay in any angular position without rotating. If it is dynamically balanced, it can be rotated at any speed without vibration. It will be shown that if a shaft is dynamically balanced, it is automatically in static balance, but the reverse is not necessarily true.

- **Static Balancing**

Figure above shows a simple situation where two masses mounted on a shaft. If the shaft is to be statically balanced, the moment due to weight of mass 1, tending to rotate the shaft clockwise, must equal that of mass 2, trying to turn the shaft in the opposite direction.

Therefore, the static balance is:

$$W_1 r_r = W_2 r_2 \qquad 1$$

Similarly, the same principle holds for more than two masses, as shown in Figure 2.

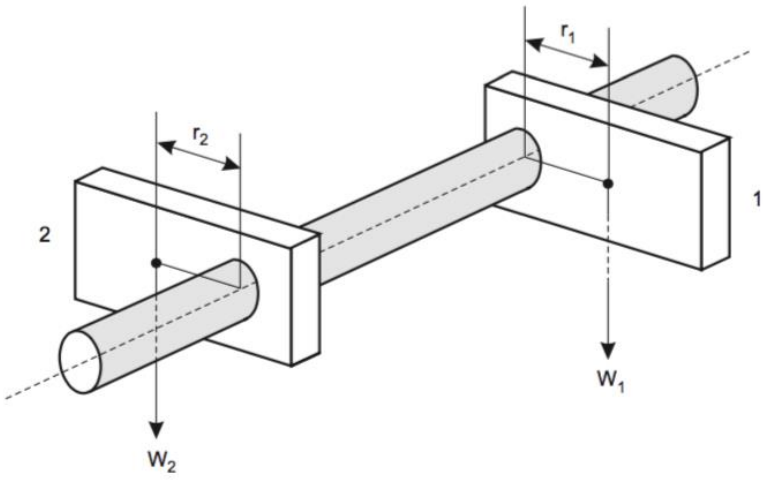


Figure 8: Static balancing of a simple two-masses system

And for the static balancing:

$$W_1 r_1 \cos \alpha_1 = W_2 r_2 \cos \alpha_2 + W_3 r_3 \cos \alpha_3 \qquad (2)$$

anti-clockwise clockwise

In general, the values of W, r and α have to be chosen such that the shaft is in balance. Nevertheless, for experiments using TM102 apparatus, the product Wr can be measured directly for each mass, and only the angular positions have to be determined for static balance. If the angular positions of two masses are fixed, then the position of the third mass can be found, either by trigonometry or drawing.

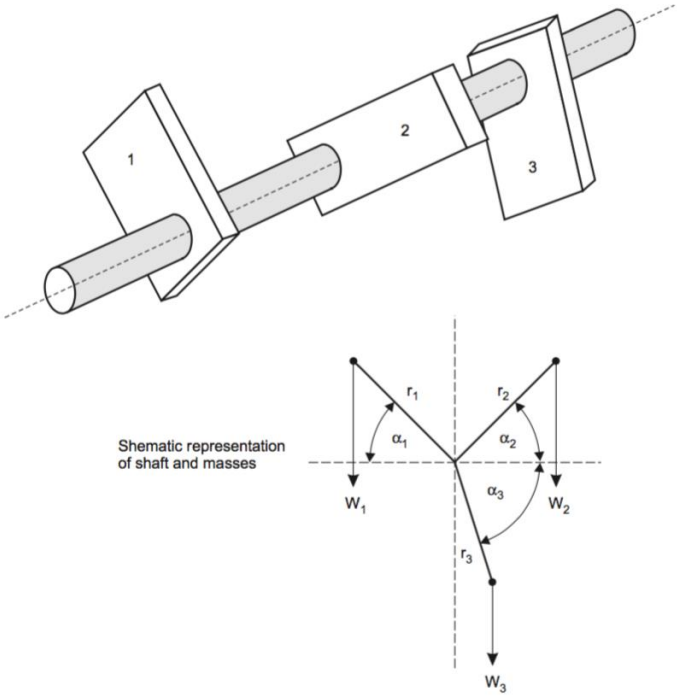


Figure 9: Static balancing of three-masses system

- **Dynamic Balancing**

As the shaft rotate the masses are going to be subjected to centrifugal forces. If the shaft is not to vibrate as it rotates, then **two conditions** must be satisfied:

1. There must be no out-of-balance centrifugal force trying to deflect the shaft
2. There must be no out-of-balance moment or couple trying to twist the shaft

The shaft will not be dynamically balanced until these conditions are satisfied.

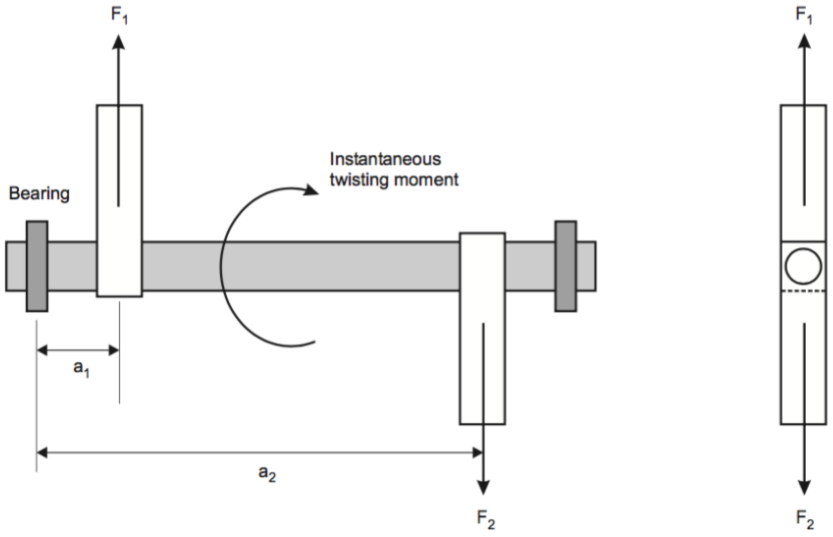


Figure 10:

Applying condition 1 to the shaft in figure 3 gives:

$F_1 = F_2$ 3

The centrifugal force is:

$$mr\omega^2 \text{ or } \frac{W}{g}r\omega^2$$

Therefore;

$$\frac{W_1}{g}r_2\omega^2 = \frac{W_2}{g}r_2\omega^2$$

4

The angular speed of the rotation is the same for each mass, so that for the dynamic balance:

$W_1r_1 = W_2r_2$ 5

Note: If a shaft is dynamically balanced, it will also be statically balanced.

The second condition will be satisfied by taking moments about some desired datum, like one of the bearings. Hence:

$a_1F_1 = a_2F_2$ 6

Remember, $F_1 = F_2$, so as $a_1 = a_2$. Thus, in this case, dynamic balance can only be achieved if the two masses are mounted at the same point along the shaft.

Note: If a shaft is statically balanced, it does not mean that it is dynamically balanced.

- **Dynamic Balancing of three Masses**

It has been shown that dynamic balance can only be achieved for a two-mass system if the masses are mounted at the same point along the shaft.

It can also be shown that there are special conditions which must be satisfied for the dynamic balancing of three masses.

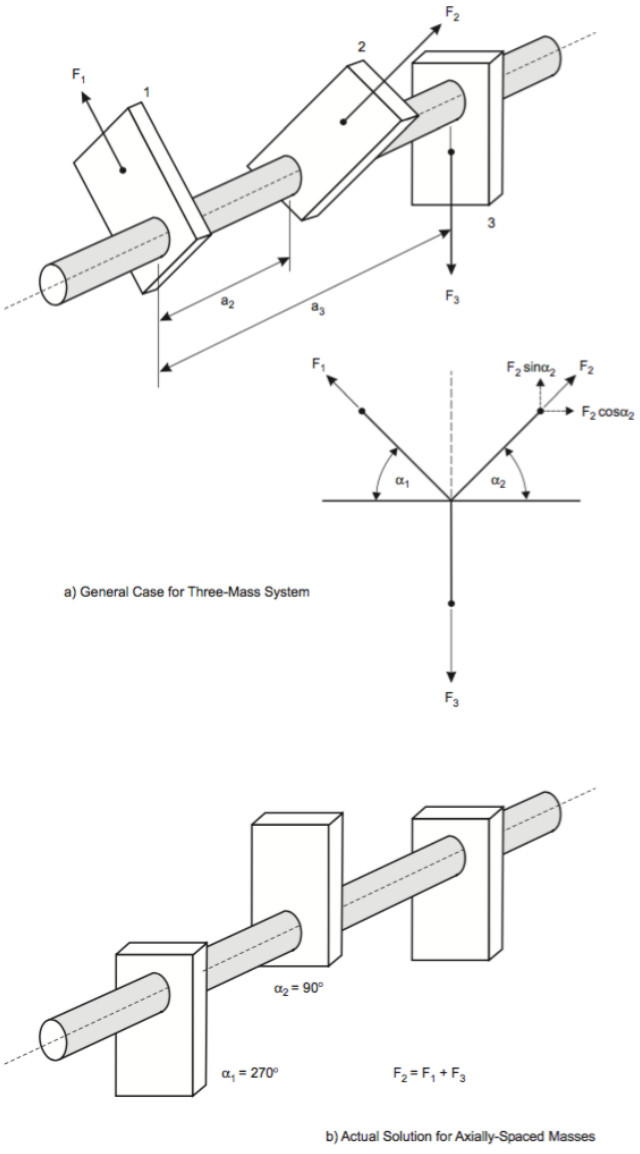


Figure 11: Dynamic balance for three mass system

From figure 4(a), mass 3 is positioned vertically for convenience. From condition 2, dynamic balance can be expressed mathematically by equating moments for centrifugal forces in both vertical and horizontal planes. In order to simplify the equations, it is convenient to take moments about mass 1, so that moments due to forces on this mass are eliminated.

Horizontal: $a_2F_2\cos\alpha_2 = 0$ 7

Vertical: $a_2F_2\sin\alpha_2 = a_3F_3$ 8

Conditions that can satisfy equation 7 are either $a_2 = 0$, or $\alpha_2 = 90$ or 270 (that is $\cos\alpha_2 = 0$). The following conditions are obtained after substituting these in to equation 8:

a) $\alpha_2 = 0$

a_3 must be zero for this condition. Thus, for any arbitrary values of α_2 and α_3, all three masses must be located at the same point along the shaft.

b) $\alpha_2 = 90$ or 270

For these conditions, it is necessary to write down further equations to obtain solutions.

Applying condition 1 for dynamic balance:

Horizontal: $F_1\cos\alpha_1 = F_2\cos\alpha_2$ 9

Vertical: $F_3 = F_1\sin\alpha_1 + F_2\sin\alpha_2$ 10

If $\alpha_2 = 90$, equation 9 gives $\alpha_1 = 90$ or 270. Let's assume $\alpha_1 = 90$, then equation 10 reduces to:

$F_3 = F_1 + F_2$

Similarly, equation 8 reduces to:

$a_2F_2 = a_3F_3$ 11

Combine above equations to solve for F_1.

Therefore, If the masses are distributed along the shaft, the following conditions must b satisfied for dynamic balancing:

1. Central mass at 180 degrees to other two masses
2. Masses chosen such that:

   $F_2 = F_1 + F_3$
3. Masses distributed along the shaft such that:

   $a_2F_2 = a_3F_3$

- **Dynamic Balancing of More Than Three Masses**

If there are more than three masses, there are no special restrictions which apply to the angular and shaft-wise distributions of the masses, and the general conditions for dynamic balance have to be applied to solutions.

The angular positions of the masses can be determined by applying conditions for either static balance or for dynamic balance. The distribution of the masses along the shaft is then found by applying condition 1 for dynamic balance. This can be done by calculation.

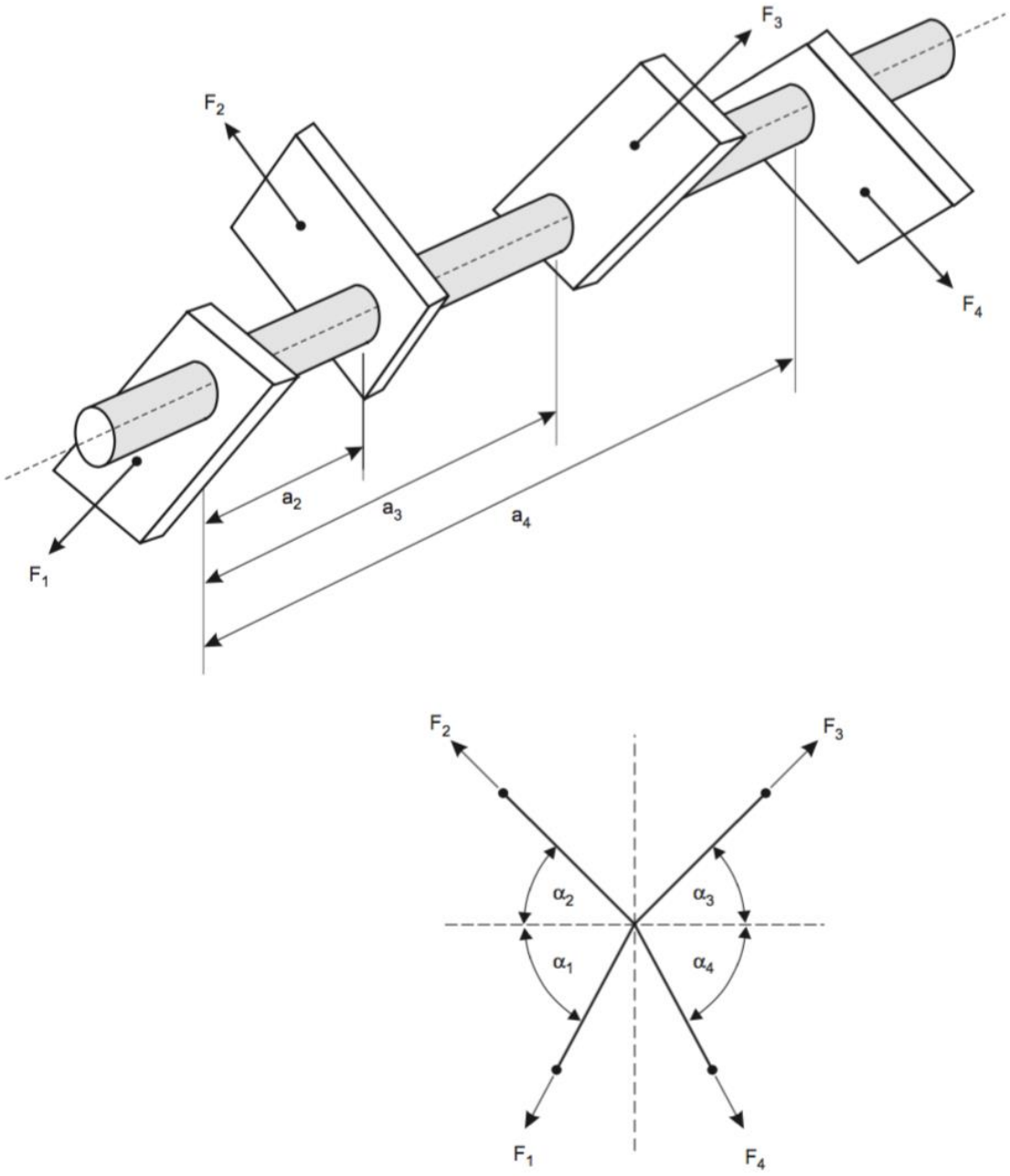


Figure 12: Nomenclature for Arbitrary Four-Mass System

The rotating moments are resolved into components tending to twist the shaft in the horizontal planes. The net moment on each plane must be zero if the shaft is to be dynamically balanced. As shown from figure 5, the appropriate equations for a four-mass system are:

Horizontal moments about mass 1

$$-a_2F_2\cos\alpha_2 + a_3F_3\cos\alpha_3 + a_4F_4\cos\alpha_4 = 0 \quad (12)$$

Vertical moments about mass 1

$$a_2F_2\sin\alpha_2 + a_3F_3\sin\alpha_3 - a_4F_4\sin\alpha_4 = 0 \quad (13)$$

PROCEDURE:

- **Demonstration of Static Balance by Dynamic Imbalance**
  - Remove the Perspex dome and the shaft drive belt.
  - Remove the disc from the four rectangle blocks using the small hexagon key. ***Always take the blocks off the shaft before removing or replacing the discs.***
  - Set up two of the blocks, as shown in figure 6, with the relative angular displacement of 180 degrees and shaft-wise displacement of 120mm. use the slider to set and read off the positions of the blocks of the blocks.
  - Observe that the shaft will remain in any angular position and is therefore statically balanced.
  - Connect the apparatus to a 12 VDC supply. Make sure that the slider adjacent to the linear scale is clear of the blocks, then replace the shaft drive-belt and the Perspex dome.
  - Run the motor. Note the severe imbalance of the shaft.

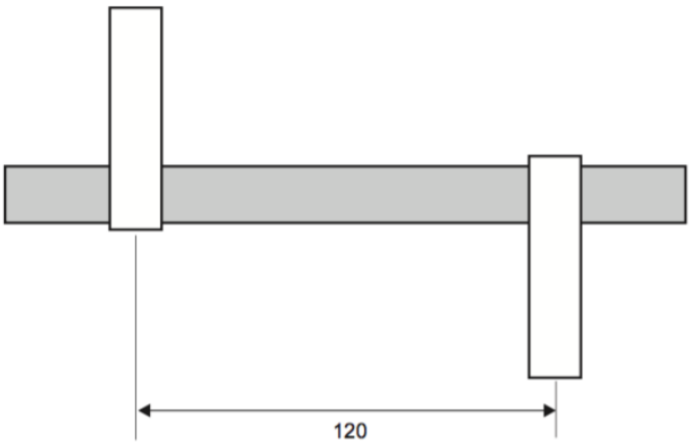


Figure 13: Static Balance for Tow-Mass system

- **Dynamic Balance using Four Masses**
  - Remove the Perspex dome and set up the four rectangular blocks as shown in figure 7
  - Make sure that theoretically the shaft is both statically and dynamically balanced (refer back to **"Introduction and Theory"**).
  - Test the shaft for static balance.
  - Replace the drive-belt and the Perspex dome.
  - Run the motor and observe the lack of vibration, thus, showing that the shaft is dynamically balanced.

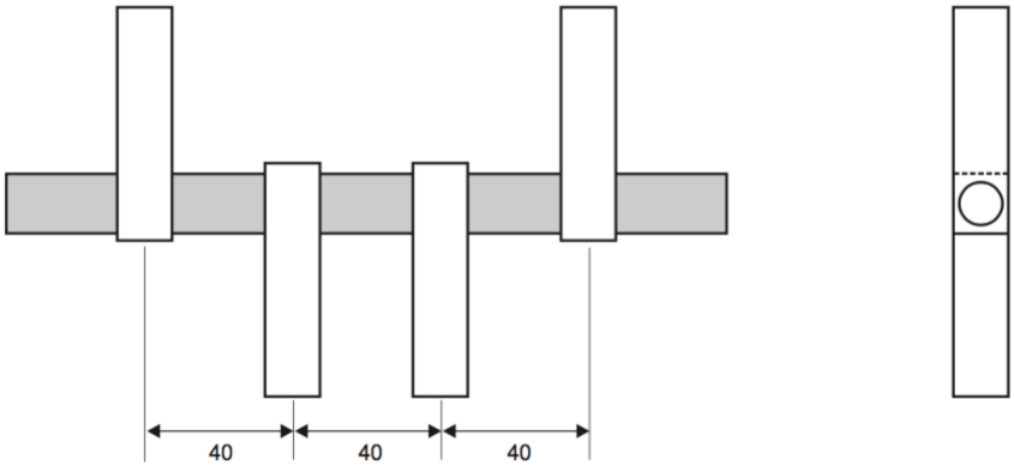
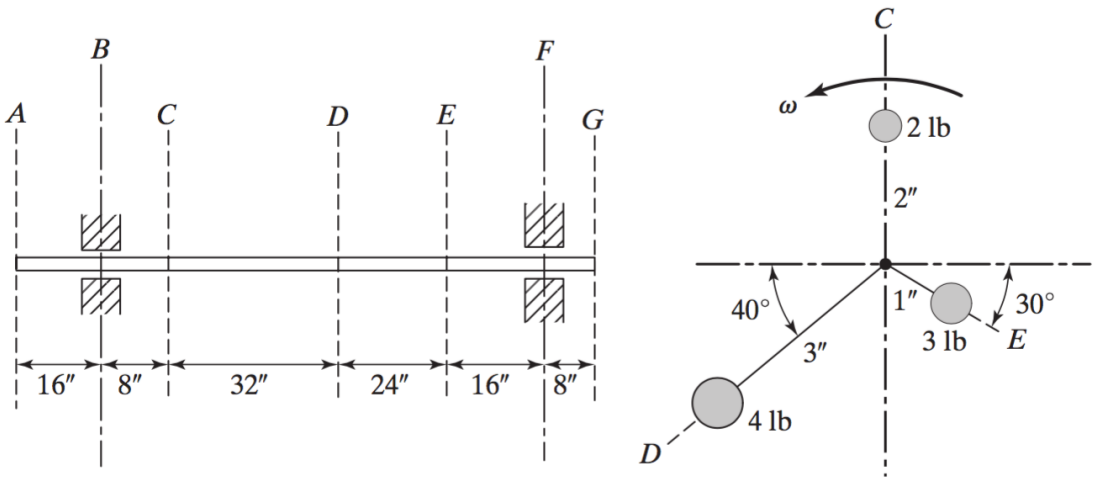


Figure 14: Static and Dynamic Balance for Four-Mass System

Problem to be solved

Weights of 2 lb., 4 lb., and 3 lb. are located at radii 2in., 3in. and 1in. in the planes C, D, and E, respectively, on a shaft supported at the bearings B and F, as shown in figure below. Find the weights and angular locations of the two balancing weights to be placed in the end planes A and G so that the dynamic load on the bearings will be zero.



QUESTIONS:

Briefly explain/ discuss the following:

1. **What are the various methods available for vibration control?**
2. **Why does dynamic balancing imply static balancing? and Explain why dynamic balancing can never be achieved by a static test alone**
3. **What is the difference between stationary damping and rotary damping?**
4. **What causes instability in a rotor system?**
5. **Why does a rotating shaft always vibrate? What is the source of the shaking force**

LAB PROJECT # 8:
Vibration Analysis of Rotating Machinery

This example shows how to analyze vibration signals from a gearbox using time-synchronous averaging and envelope spectra. These functions are especially useful in the predictive maintenance of gearboxes, which contain multiple rotating components: gears, shafts and bearings.

This example generates and analyzes vibration data for a gearbox whose shafts rotate at a fixed speed. Time-synchronous averaging is used to isolate vibration components associated with a specific shaft or gear and average out all other components. Envelope spectra are especially useful in identifying localized bearing faults that cause high-frequency impacts.

Consider an idealized gearbox that consists of a 13-tooth pinion meshing with a 35-tooth gear. The pinion is coupled to an input shaft connected to a prime mover. The gear is connected to an output shaft. The shafts are supported by roller bearings on the gearbox housing. Two accelerometers, A_1 and A_2, are placed on the bearing and gearbox housings, respectively. The accelerometers operate at a sample rate of 20 kHz.

The pinion rotates at a rate f_{Pinion} = 22.5 Hz or 1350 rpm. The rotating speed of the gear and output shaft is

$$f_{\text{Gear}} = f_{\text{Pinion}} \times \frac{\text{Number of pinion teeth } (N^p)}{\text{Number of gear teeth } (N^g)}.$$

The tooth-mesh frequency, also called gear-mesh frequency, is the rate at which gear and pinion teeth periodically engage:

$$f_{\text{Mesh}} = f_{\text{Pinion}} \times N_p = f_{\text{Gear}} \times N_g.$$

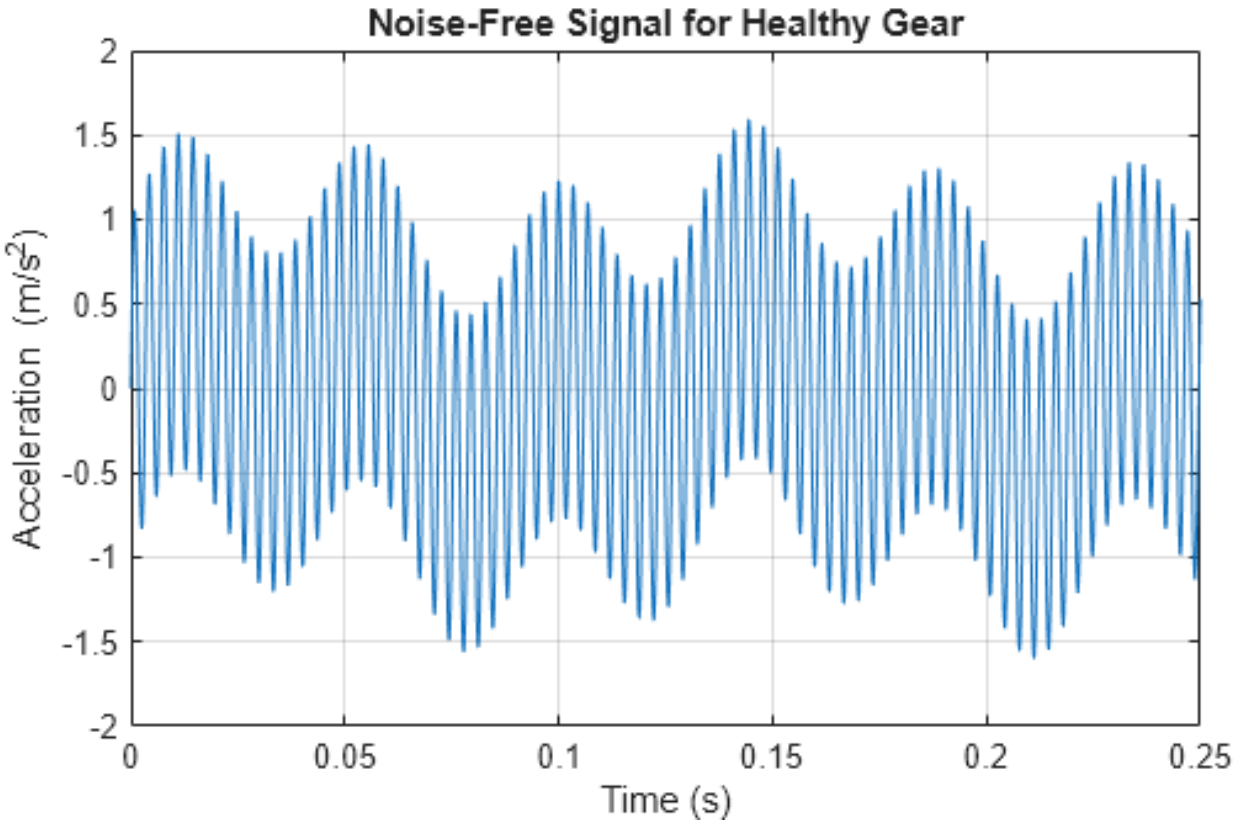
```
rng default
fs = 20e3;          % Sample Rate (Hz)
Np = 13;            % Number of teeth on pinion
Ng = 35;            % Number of teeth on gear

fPin = 22.5;        % Pinion (Input) shaft frequency (Hz)
fGear = fPin*Np/Ng; % Gear (Output) shaft frequency (Hz)
fMesh = fPin*Np;    % Gear Mesh frequency (Hz)
```

Generate vibration waveforms for the pinion and the gear. Model the vibrations as sinusoids occurring at primary shaft gear mesh frequencies. Analyze 20 seconds of vibration data.

The gear-mesh waveform is responsible for transmitting load and thus possesses the highest vibration amplitude. A_2 records vibration contributions from the two shafts and the gear-mesh. For this experiment, the contributions of the bearing rolling elements to the vibration signals recorded by A_2 are considered negligible. Visualize a section of noise-free vibration signal.

```
t = 0:1/fs:20-1/fs;

afIn = 0.4*sin(2*pi*fPin*t);     % Pinion waveform
afOut = 0.2*sin(2*pi*fGear*t);   % Gear waveform
```

```
aMesh = sin(2*pi*fMesh*t);      % Gear-mesh waveform

aHealthy = afIn + afOut + aMesh; % Healthy gear signal

plot(t, afIn+afOut+aMesh)
xlim([0 0.25])
title("Noise-Free Signal for Healthy Gear")
xlabel("Time (s)")
ylabel("Acceleration (m/s^2)")
grid on
```



Generate High-Frequency Impacts Caused by a Local Fault on a Gear Tooth

Assume that one of the teeth of the gear is suffering from a local fault such as a spall. This results in a high-frequency impact occurring once per rotation of the gear.

The local fault causes an impact that has a duration shorter than the duration of tooth mesh. A dent on the tooth surface of the gear generates high-frequency oscillations over the duration of the impact. The frequency of impact is dependent on gearbox component properties and its natural frequencies. In this example, it is arbitrarily assumed that the impact causes a 2 kHz vibration signal and occurs over a duration of about 8% of 1/fMesh, or 0.25 milliseconds. The impact repeats once per rotation of the gear.

```
ipf = fGear;
fImpact = 2000;

tImpact = 0:1/fs:2.5e-4-1/fs;
```

```
xImpact = sin(2*pi*fImpact*tImpact)/3;
```
Make the impact periodic by modeling it as a periodic train of pulses.

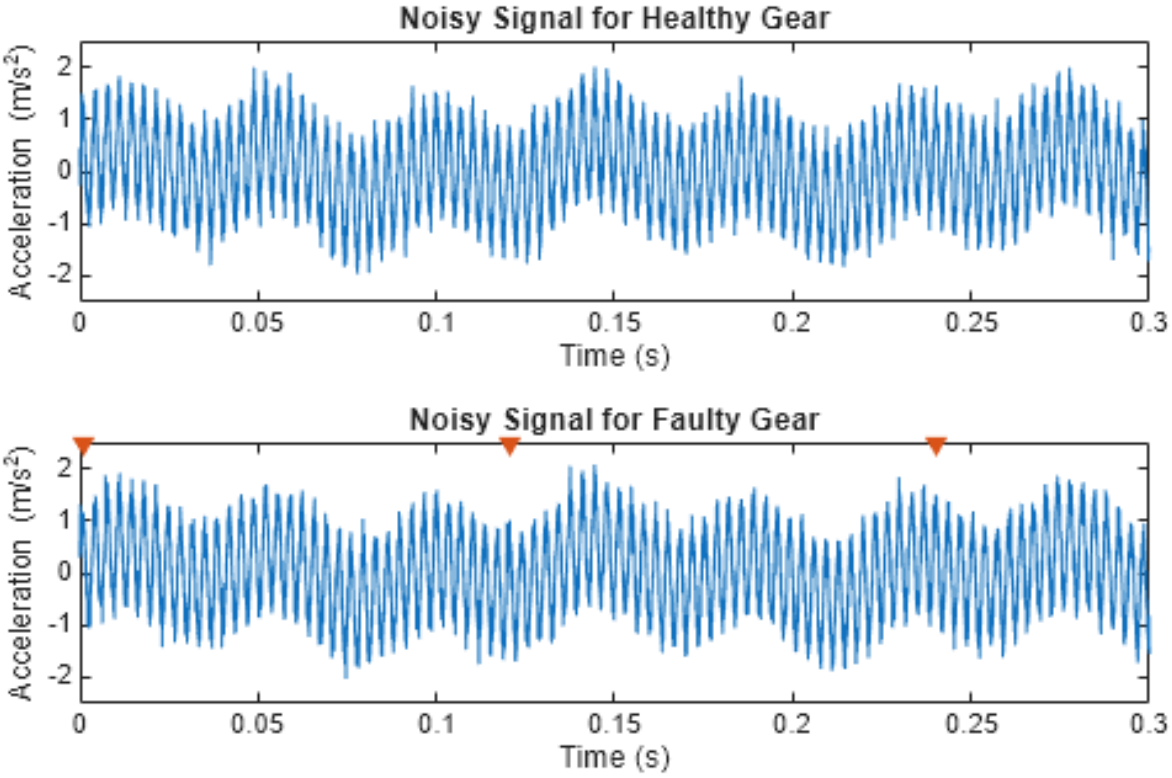
```
xImpactPer = 2*pulstran(t,0:1/ipf:t(end),xImpact,fs);
```

Add the fault signal xImpactPer to the shaft signal. Add white Gaussian noise to the output signals for both the fault-free and the faulty gear to model the output from A_2.

```
aFaulty = aHealthy + xImpactPer;


aHealthyNoisy = aHealthy + randn(size(t))/5;
aFaultyNoisy = aFaulty + randn(size(t))/5;
```

Visualize a segment of the time history. The impact locations are indicated on the plot for the faulty gear by the inverted red triangles. They are almost indistinguishable.

```
figure
tiledlayout flow


nexttile
plot(t,aHealthyNoisy)
title("Noisy Signal for Healthy Gear")
xlabel("Time (s)")
ylabel("Acceleration (m/s^2)")
axis([0.0 0.3 -2.5 2.5])


nexttile
plot(t,aFaultyNoisy)
hold on
Ind = (0.25*fs/fMesh):(fs/ipf):length(t);
Ind = round(Ind);
MarkX = t(Ind(1:3));
MarkY = 2.5;
scatter(MarkX,MarkY,"v","filled", ...
    MarkerFaceColor="#D95319")
hold off
title("Noisy Signal for Faulty Gear")
xlabel("Time (s)")
ylabel("Acceleration (m/s^2)")
axis([0.0 0.3 -2.5 2.5])
```



Compare Power Spectra for Both Signals

Localized tooth faults cause distributed sidebands to appear in the neighborhood of the gear mesh frequency:

$$f_{\text{sideband,Pinion}} = f_{\text{Mesh}} \pm m \times f_{\text{Pinion}} \qquad \forall\, m \in \{1, 2, 3, ..\}$$

$$f_{\text{sideband,Gear}} = f_{\text{Mesh}} \pm m \times f_{\text{Gear}} \qquad \forall\, m \in \{1, 2, 3, ..\}$$

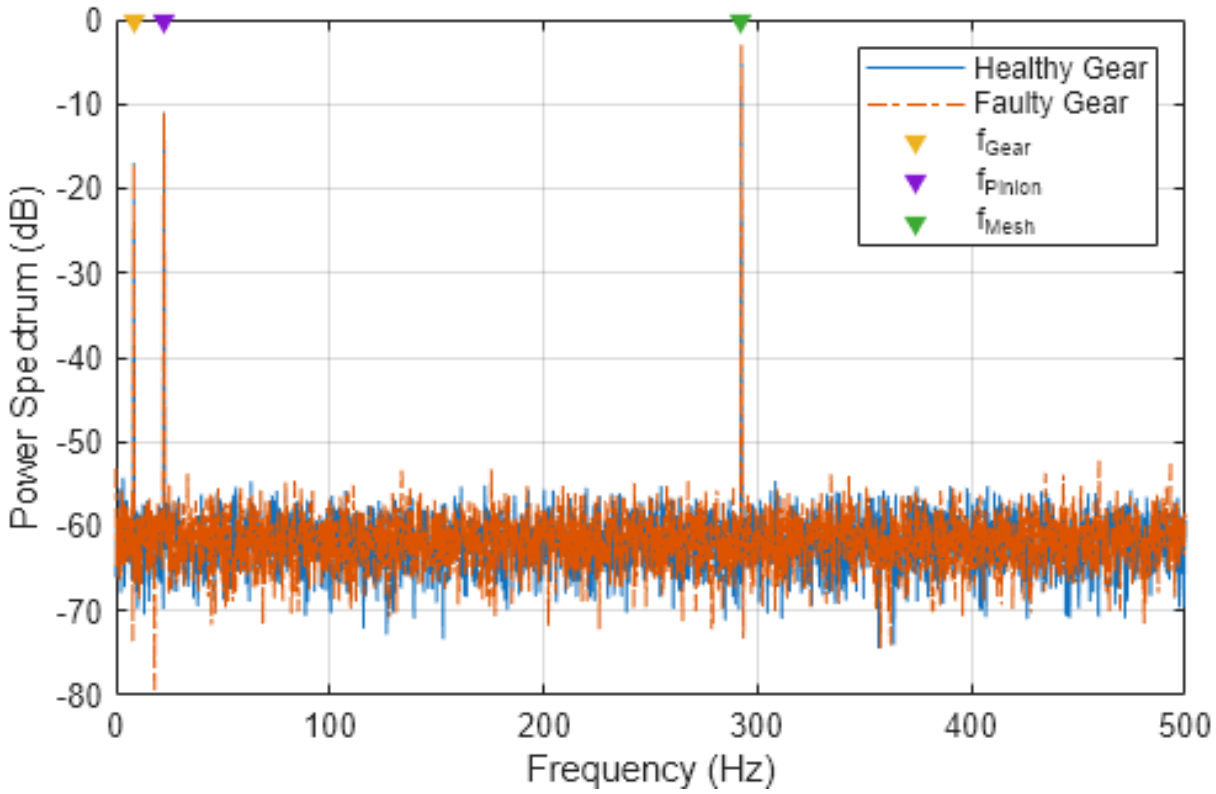
Calculate the spectrum of the healthy and faulty gears. Specify a frequency range that includes the shaft frequencies at 8.35 Hz and 22.5 Hz and the gear-mesh frequency at 292.5 Hz.

```
[Spect,f] = pspectrum([aHealthyNoisy' aFaultyNoisy'], ...
    fs,FrequencyResolution=0.2,FrequencyLimits=[0 500]);
```

Plot the spectra. Because the fault is on the gear and not the pinion, sidebands are expected to appear at $f_{\text{sideband,Gear}}$ and spaced f_{Gear} apart on the spectra. The spectra show the expected peaks at fGear, fPin, and fMesh. However, the presence of noise in the signal makes the sideband peaks at $f_{\text{sideband,Gear}}$ indistinguishable.

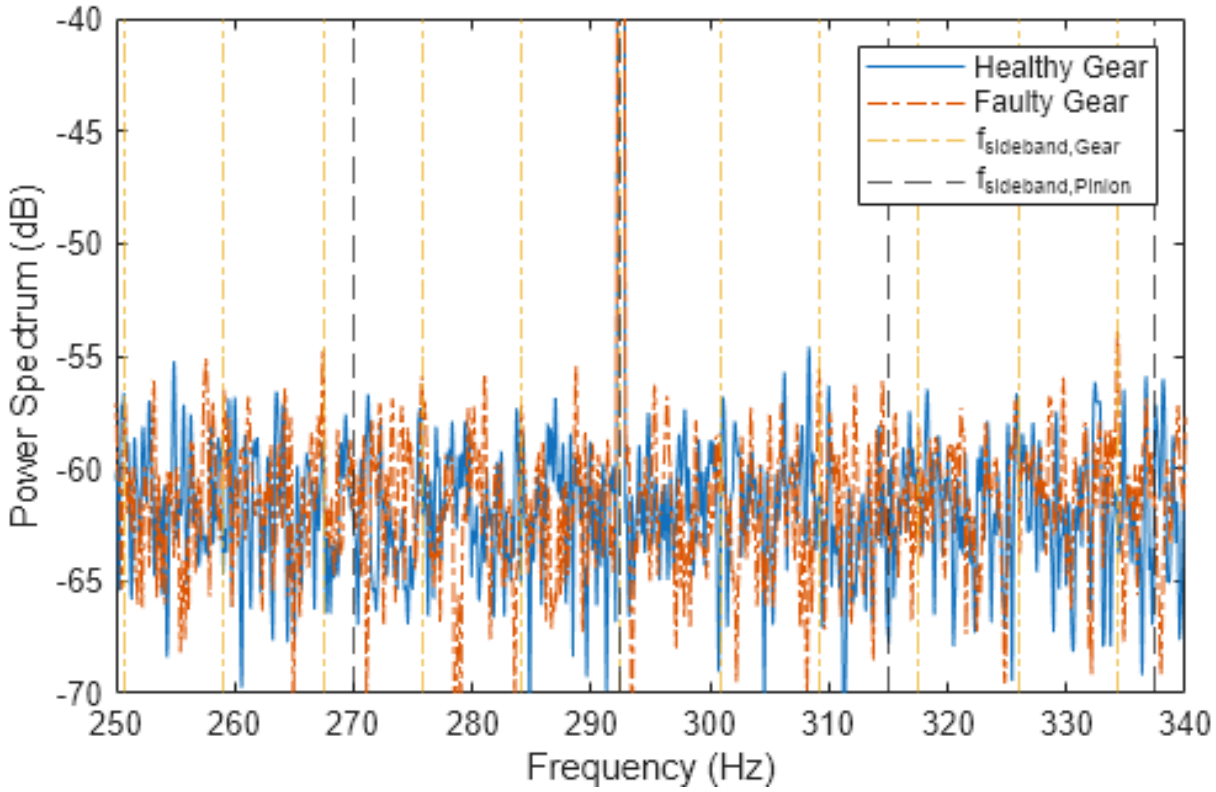
```
figure
plot(f,pow2db(Spect(:,1)), ...
    f,pow2db(Spect(:,2)),"-.")
hold on
scatter([fGear fPin fMesh],0,"v","filled")
hold off
xlabel("Frequency (Hz)")
ylabel("Power Spectrum (dB)")
legend("Healthy Gear","Faulty Gear","f_{Gear}","f_{Pinion}","f_{Mesh}")
```

```
grid on
```



Zoom in on the neighborhood of the gear-mesh frequency. Create a grid of gear and pinion sidebands at $f_{\text{sideband,Gear}}$ and $f_{\text{sideband,Pinion}}$.

```
figure
p1 = plot(f,pow2db(Spect(:,1)));
hold on
p2 = plot(f,pow2db(Spect(:,2)),"-.");

harmonics = -5:5;
SBandsGear = (fMesh+fGear.*harmonics);
SBandsPinion = (fMesh+fPin.*harmonics);
p3 = xline(SBandsGear,"-.",Color="#EDB120");
p4 = xline(SBandsPinion,"--");
hold off

axis([250 340 -70 -40])
xlabel("Frequency (Hz)")
ylabel("Power Spectrum (dB)")
legend([p1 p2 p3(1) p4(1)], ...
   ["Healthy Gear";"Faulty Gear"; ...
   "f_{sideband,Gear}";"f_{sideband,Pinion}"])
```



It is not clear if the peaks align with the gear sidebands $f_{\text{sideband,Gear}}$.

Apply Time-Synchronous Averaging to the Output Vibration Signal

Note that it is difficult to separate the peaks at the gear sidebands, $f_{\text{SideBand, Gear}}$, and the pinion sidebands, $f_{\text{SideBand, Pinion}}$. The previous section demonstrated difficulty in separating peaks and determining if the pinion or the gear is affected by faults. Time-synchronous averaging averages out zero-mean random noise and any waveforms not associated with frequencies of the particular shaft. This makes the process of fault detection easier.

Use the function tsa to generate time-synchronized waveforms for both the pinion and the gear.

Specify time-synchronized pulses for the pinion. Calculate the time-synchronous average for 10 rotations of the pinion.

```
tPulseIn = 0:1/fPin:max(t);
taPin = tsa(aFaultyNoisy,fs,tPulseIn,NumRotations=10);
```

Specify time-synchronized pulses for the gear. Calculate the time-synchronous average for 10 rotations of the gear.

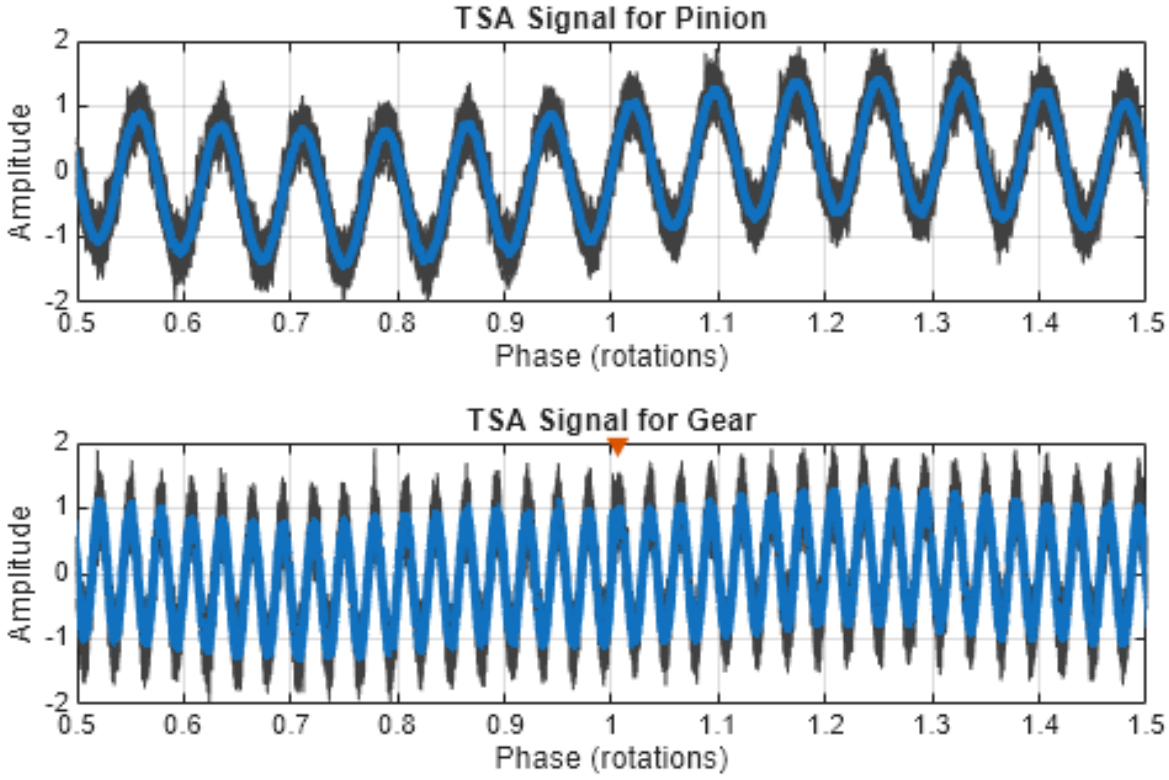
```
tPulseOut = 0:1/fGear:max(t);
taGear = tsa(aFaultyNoisy,fs,tPulseOut,NumRotations=10);
```

Visualize the time-synchronized signals for a single rotation. The impact is comparatively easier to see on the time-synchronous averaged signal for the gear, while it is averaged out for the pinion shaft. The location of the impact, indicated on the plot with a marker, has a higher amplitude than neighboring gear-mesh peaks.

The tsa function without output arguments plots the time-synchronous average signal and the time-domain signals corresponding to each signal segment in the current figure.

```
figure
```

```
tiledlayout flow

nexttile
tsa(aFaultyNoisy,fs,tPulseIn,NumRotations=10)
axis([0.5 1.5 -2 2])
title("TSA Signal for Pinion")

nexttile
tsa(aFaultyNoisy,fs,tPulseOut,NumRotations=10)
hold on
scatter(1.006,2,"v","filled")
hold off
axis([0.5 1.5 -2 2])
title("TSA Signal for Gear")
```

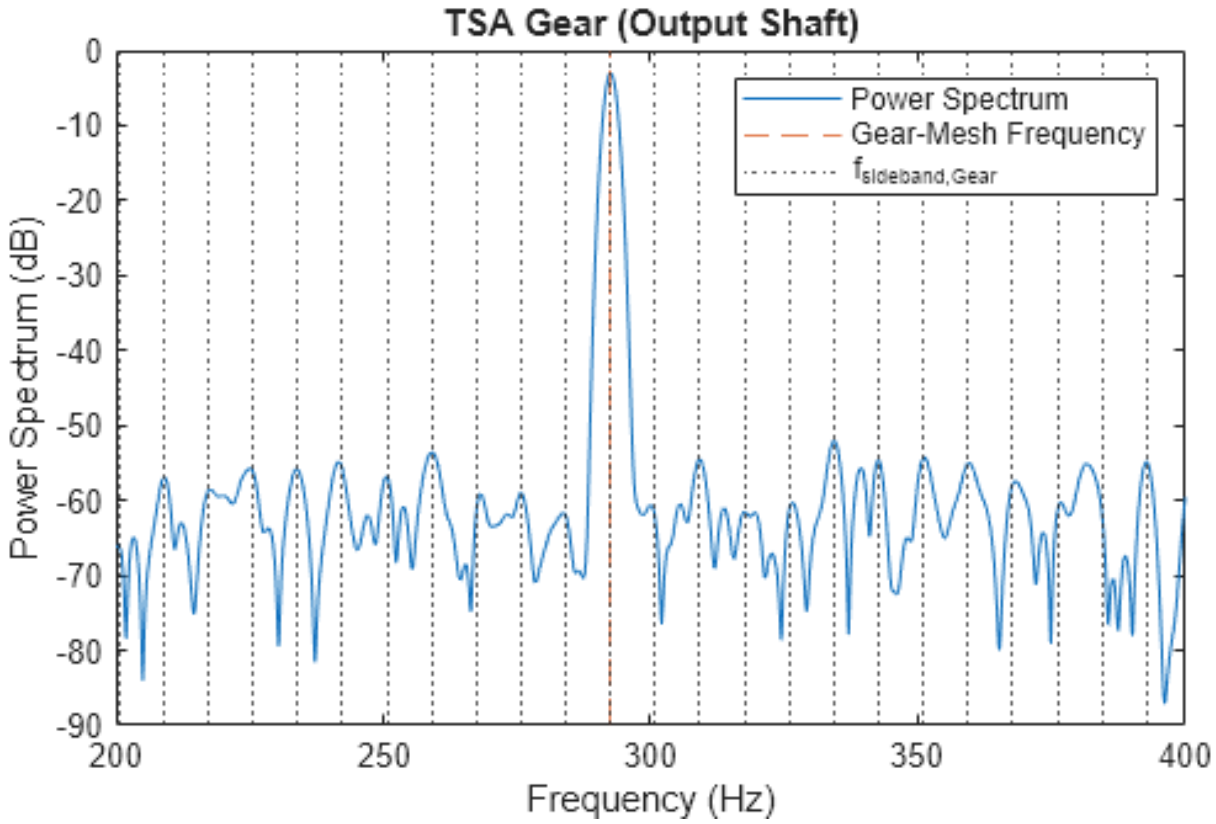


Visualize the Power Spectra for Time-Synchronous Averaged Signals

Calculate the power spectrum of the time-synchronous averaged gear signal. Specify a frequency range that covers 15 gear sidebands on either side of the gear mesh frequency of 292.5 Hz. Notice the peaks at $f_{\text{sideband,Gear}}$.

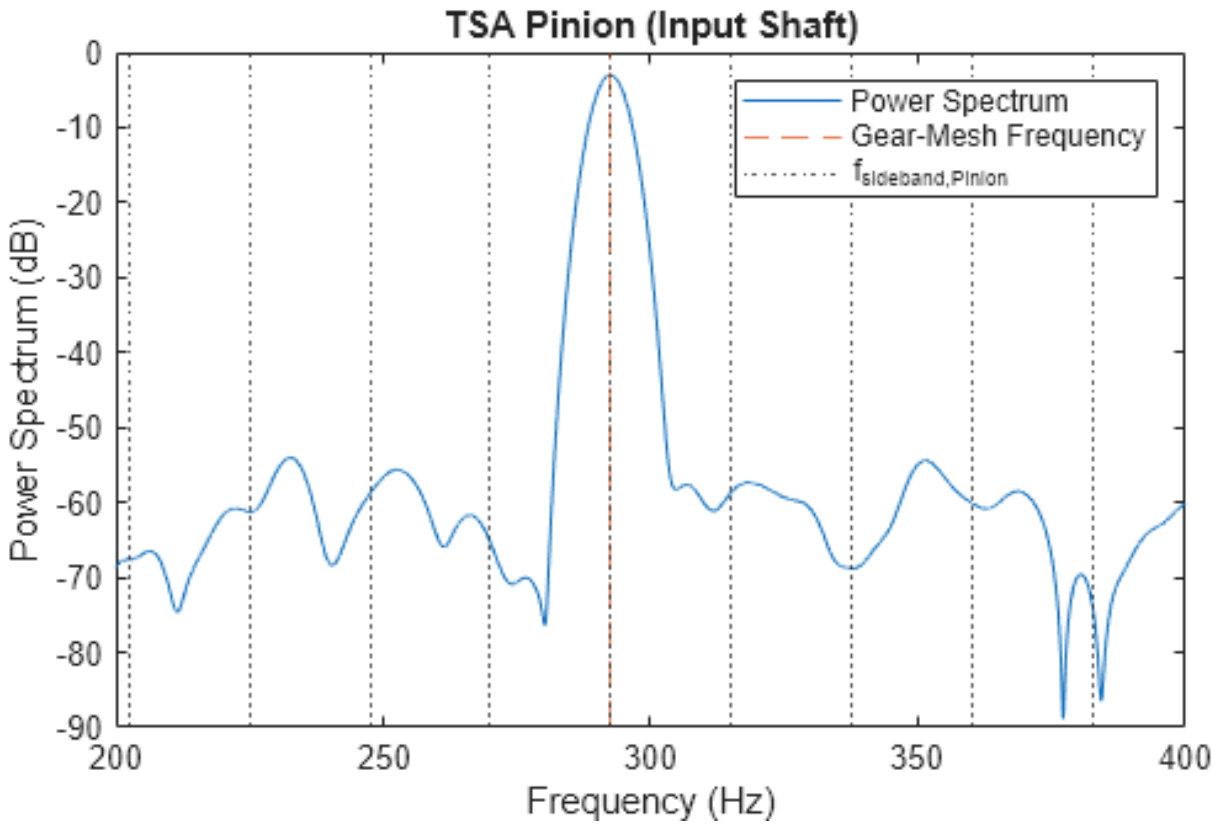
```
figure
pspectrum(taGear,fs, ...
   FrequencyResolution=2.2,FrequencyLimits=[200 400])

harmonics = -15:15;
```

```
SBandsGear=(fMesh+fGear.*harmonics);
xline(fMesh,"--",Color="#D95319")
xline(SBandsGear,":")

grid off
title("TSA Gear (Output Shaft)")
legend("Power Spectrum","Gear-Mesh Frequency","f_{sideband,Gear}")
```



Visualize the power spectra of the time-synchronous averaged pinion signal in the same frequency range. This time, plot grid lines at $f_{\text{sideband,Pinion}}$ frequency locations.

```
figure
pspectrum(taPin,fs, ...
   FrequencyResolution=5.8,FrequencyLimits=[200 400])

SBandsPinion = (fMesh+fPin.*harmonics);
xline(fMesh,"--",Color="#D95319")
xline(SBandsPinion,":")

grid off
title("TSA Pinion (Input Shaft)")
legend("Power Spectrum","Gear-Mesh Frequency","f_{sideband,Pinion}")
```



Notice the absence of prominent peaks at $f_{\text{sideband,Pinion}}$ in the plot.

The power spectra of the original signal contains waveforms from two different shafts, as well as noise. It is difficult to distinguish the sideband harmonics. However, observe the prominent peaks at the sideband locations on the spectrum of the time-synchronous averaged gear signal. Also observe the nonuniformity in sideband magnitudes, which are an indicator of localized faults on the gear. On the other hand, sideband peaks are absent from the spectrum of the time-synchronous averaged pinion signal. This helps us conclude that the pinion is potentially healthy.

By averaging out the waveforms that are not relevant, the tsa function helps identify the faulty gear by looking at sideband harmonics. This functionality is especially useful when it is desirable to extract a vibration signal corresponding to a single shaft, from a gearbox with multiple shafts and gears.

Add a Distributed Fault in the Pinion and Incorporate its Effects into the Vibration Signal

A distributed gear fault, such as eccentricity or gear misalignment [1], causes higher-level sidebands that are narrowly grouped around integer multiples of the gear-mesh frequency.

To simulate a distributed fault, introduce three sideband components of decreasing amplitude on either side of the gear-mesh frequency.

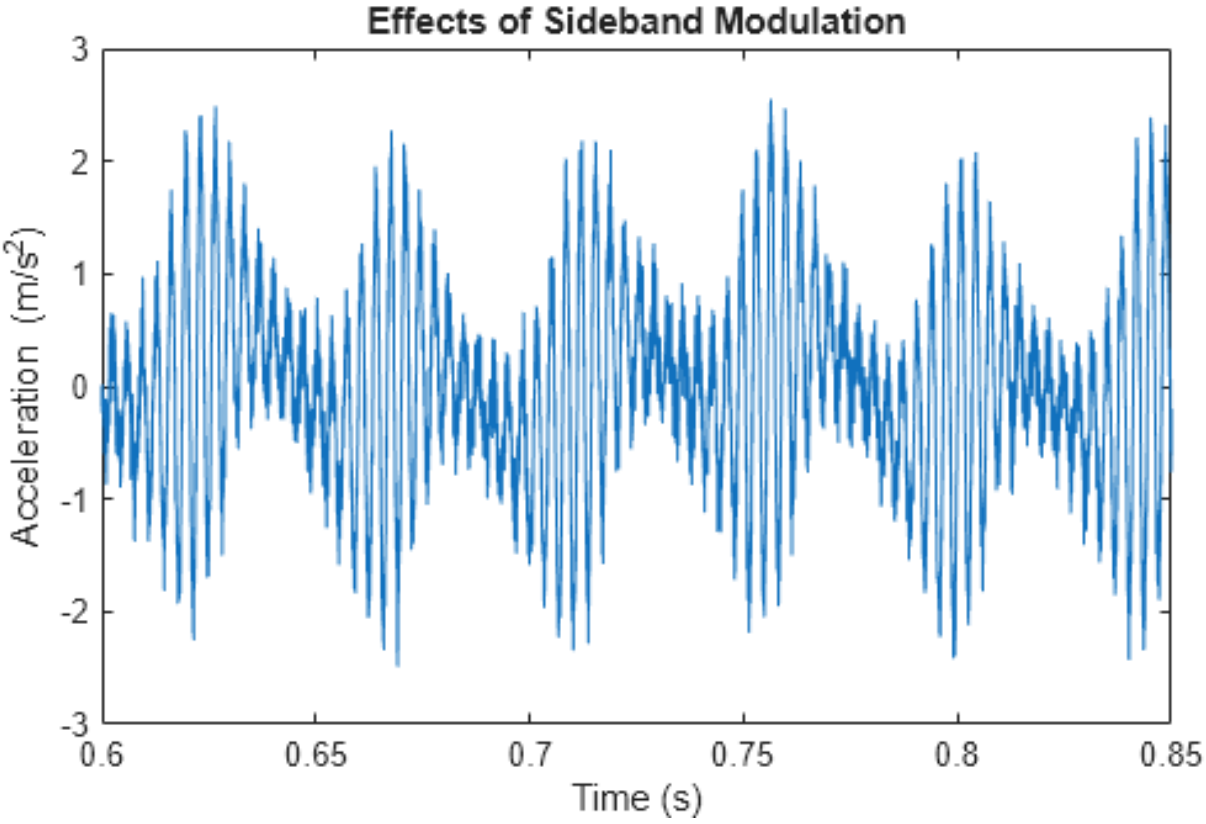
```
SideBands = -3:3;
SideBandAmp = [0.02 0.1 0.4 0 0.4 0.1 0.02]; % Sideband amplitudes
SideBandFreq = fMesh + SideBands*fPin;       % Sideband frequencies


vSideBands = SideBandAmp*sin(2*pi*SideBandFreq'.*t);
```

Add the sideband signals to the vibration signal. This results in amplitude modulation.

```
vPinFaultNoisy = aFaultyNoisy + vSideBands;
```

Visualize a section of the time history for the gearbox affected by the distributed fault.

```
plot(t,vPinFaultNoisy)
xlim([0.6 0.85])
title("Effects of Sideband Modulation")
xlabel("Time (s)")
ylabel("Acceleration (m/s^2)")
```



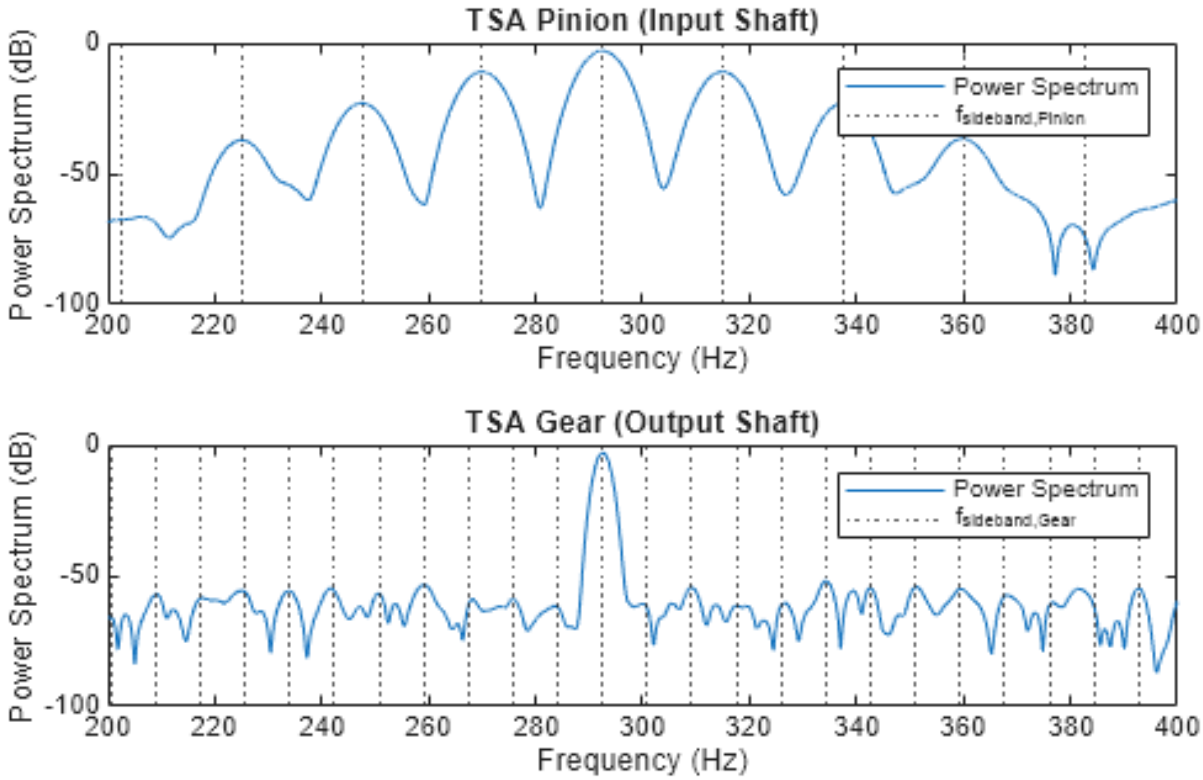
Recompute the time-synchronous averaged signal for the pinion and the gear.

```
taPin = tsa(vPinFaultNoisy,fs,tPulseIn,NumRotations=10);
taGear = tsa(aFaultyNoisy,fs,tPulseOut,NumRotations=10);
```

Visualize the power spectrum of the time-synchronous averaged signal. The three sidebands in the time-synchronous averaged signal of the pinion are more pronounced which indicate the presence of distributed faults. However, the spectrum of the time-synchronous averaged gear signal remains unchanged.

```
figure
tiledlayout flow


nexttile
pspectrum(taPin,fs, ...
    FrequencyResolution=5.8,FrequencyLimits=[200 400])
xline(SBandsPinion,":")
grid off
```

```
title ("TSA Pinion (Input Shaft)")
legend("Power Spectrum","f_{sideband,Pinion}")

nexttile
pspectrum(taGear,fs, ...
   FrequencyResolution=2.2,FrequencyLimits=[200 400])
xline(SBandsGear,":")
grid off
title ("TSA Gear (Output Shaft)")
legend("Power Spectrum","f_{sideband,Gear}")
```



In conclusion, the tsa function helps extract the gear and pinion contributions from the overall vibration signal. This in turn helps identify the specific components that are affected by localized and distributed faults.

Vibration Analysis of Rolling Element Bearing Faults

Localized faults in a rolling element bearing may occur in the outer race, the inner race, the cage, or a rolling element. Each of these faults is characterized by its own frequency, which is usually listed by the manufacturer or calculated from the bearing specifications. An impact from a localized fault generates high-frequency vibrations in the gearbox structure between the bearing and response transducer [2]. Assume that the gears in the gearbox are healthy and that one of the bearings supporting the pinion shaft is affected by a localized fault in the inner race. Neglect the effects of radial load in the analysis.

The bearing, with a pitch diameter of 12 cm, has eight rolling elements. Each rolling element has a diameter of 2 cm. The angle of contact θ is 15°. It is common practice to place the accelerometer on a bearing-housing while analyzing

bearing vibration. Acceleration measurements are recorded byA_1, an accelerometer located on the faulty bearing housing.

Define the parameters for the bearing.

```
n = 8;        % Number of rolling element bearings
d = 0.02;     % Diameter of rolling elements
p = 0.12;     % Pitch diameter of bearing
thetaDeg = 15; % Contact angle in degrees
```

The impacts occur whenever a rolling element passes the localized fault on the inner race. The rate at which this happens is the ball pass frequency-inner race (BPFI). The BPFI can be calculated using

$$f_{\mathrm{BPFI}} = \frac{n \times f^{\mathrm{Pin}}}{2}\left(1 + \frac{d}{p}\cos\theta\right).$$

```
bpfi = n*fPin/2*(1 + d/p*cosd(thetaDeg))
```

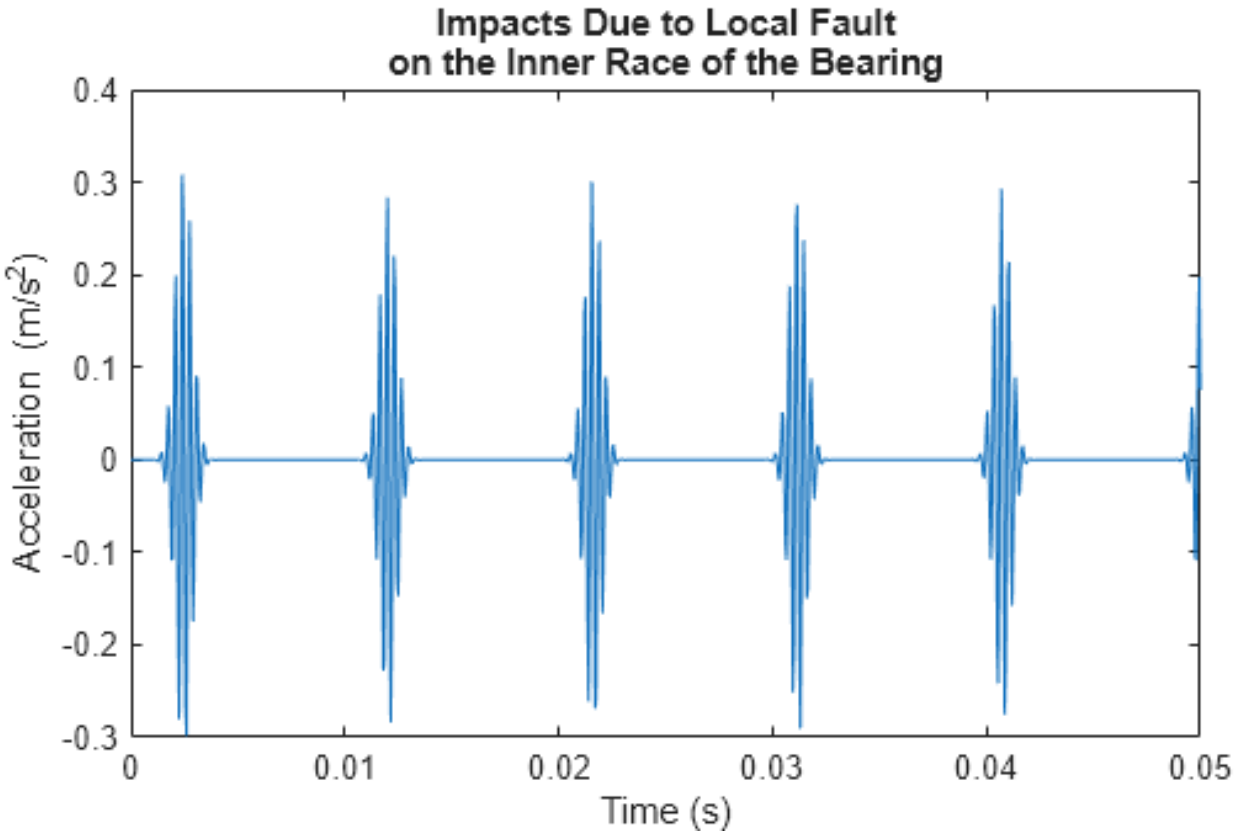
```
bpfi =

104.4889
```

Model each impact as a 3 kHz sinusoid windowed by a Kaiser window. The defect causes a series of 5-millisecond impacts on the bearing. Impulses in the early stages of pits and spalls cover a wide frequency range up to about 100 kHz [2]. Model the impacts as a periodic train of 3 kHz sinusoids windowed by a Kaiser window. Since A_1 is closer to the bearing, adjust the amplitude of the impact such that it is prominent with respect to the gearbox vibration signal recorded by A_2.

```
fImpact = 3000;
tImpact = 0:1/fs:5e-3-1/fs;
xImpact = sin(2*pi*fImpact*tImpact) ...
    .*kaiser(length(tImpact),40)';


xImpactBper = 0.33*pulstran(t,0:1/bpfi:t(end),xImpact,fs);
```

Visualize the impact signal.

```
figure
plot(t,xImpactBper)
xlim([0 0.05])
title(["Impacts Due to Local Fault" ...
   "on the Inner Race of the Bearing"])
xlabel("Time (s)")
ylabel("Acceleration (m/s^2)")
```

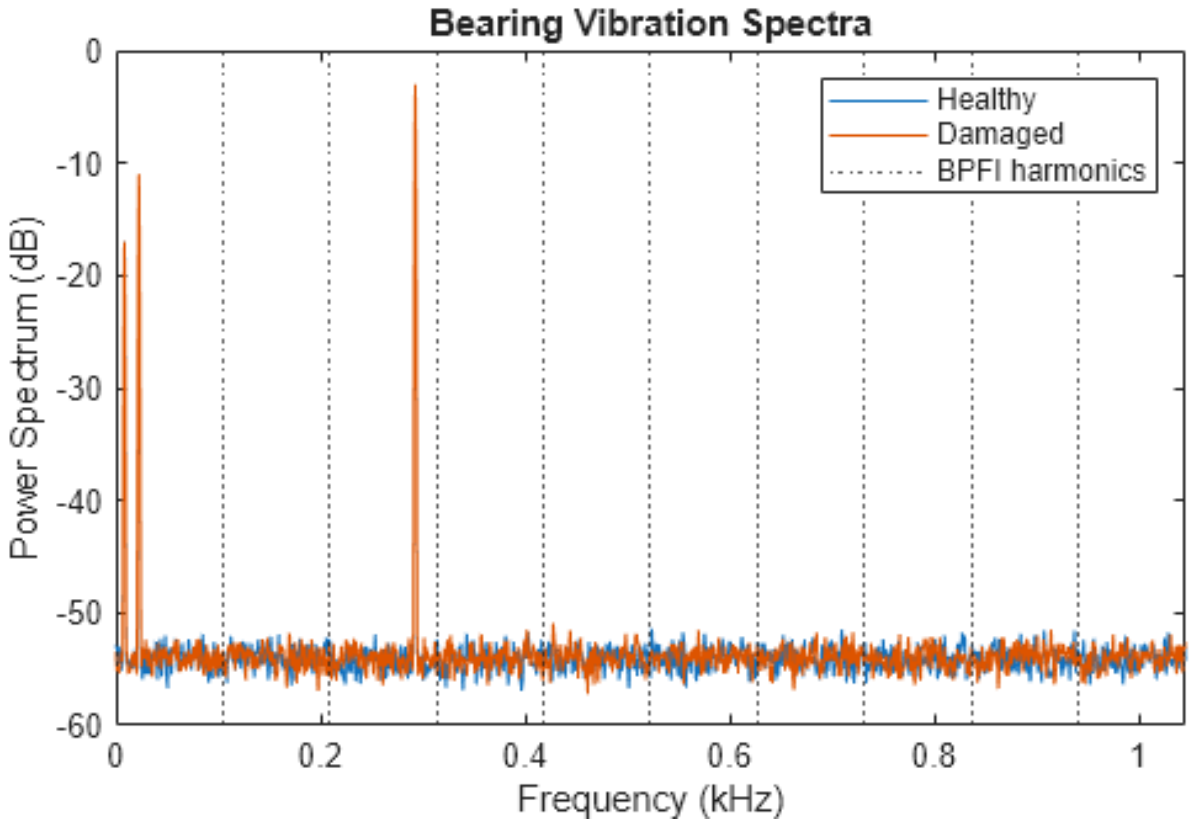


Add the periodic bearing fault to the vibration signal from the healthy gearbox.

```
vNoBFaultNoisy = aHealthy + randn(size(t))/5;
vBFaultNoisy = xImpactBper + aHealthy + randn(size(t))/5;
```

Compute the spectra of the signals. Visualize the spectrum at lower frequencies. Create a grid of the first ten BPFI harmonics.

```
pspectrum([vBFaultNoisy' vNoBFaultNoisy'],fs, ...
    FrequencyResolution=1,FrequencyLimits=[0 10*bpfi])
grid off

harmImpact = (0:10)*bpfi;
xline(harmImpact/1000,":")
xlim([0 10*bpfi]/1000)
title("Bearing Vibration Spectra")
legend("Healthy","Damaged","BPFI harmonics")
```



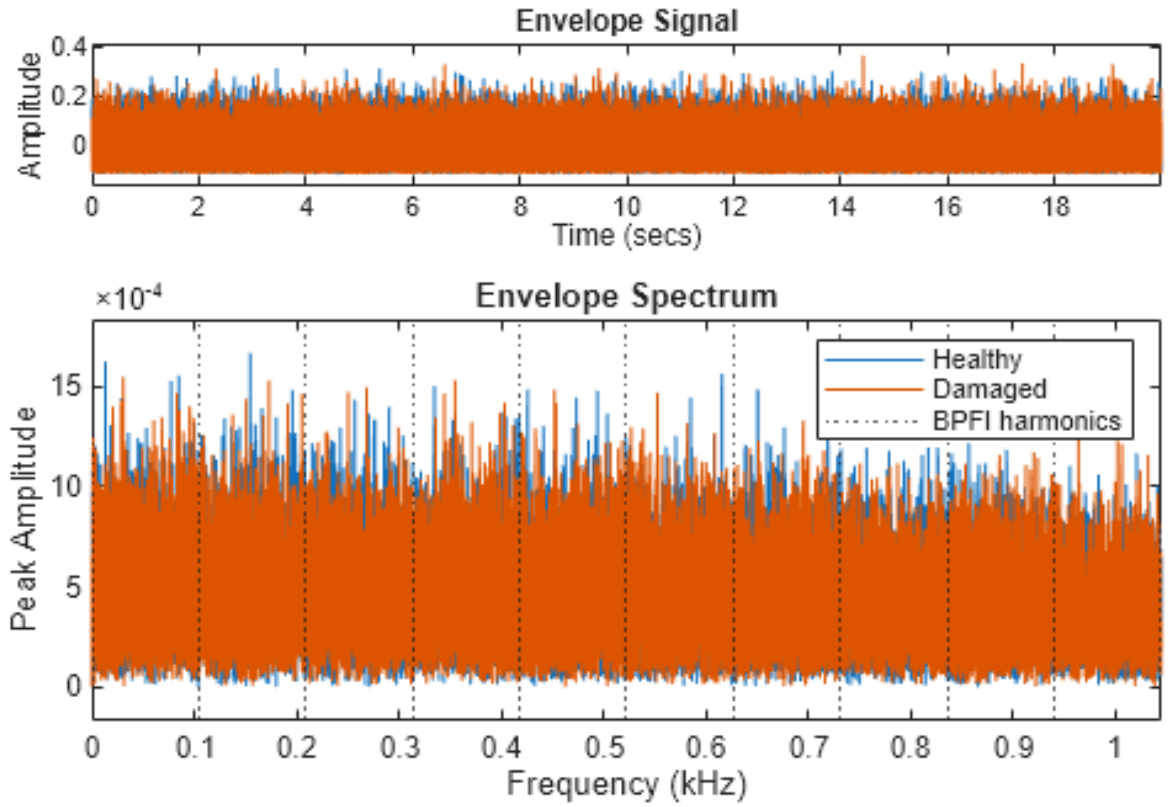
At the lower end of the spectrum, the shaft and mesh frequencies and their orders obscure other features. The spectrum of the healthy bearing and the spectrum of the damaged bearing are indistinguishable. This flaw highlights the necessity for an approach that can isolate bearing faults.

BPFI is dependent on the ratio d/p and the cosine of the contact angle θ. An irrational expression for BPFI implies that bearing impacts are not synchronous with an integer number of shaft rotations. The tsa function is not useful in this case because it averages out the impacts. The impacts do not lie on the same location in every averaged segment.

The function envspectrum (envelope spectrum) performs amplitude demodulation, and is useful in extracting information about high-frequency impacts.

Compute and plot the envelope signals and their spectra. Compare the envelope spectra for the signals with and without the bearing fault. Visualize the spectrum at lower frequencies. Create a grid of the first ten BPFI harmonics.

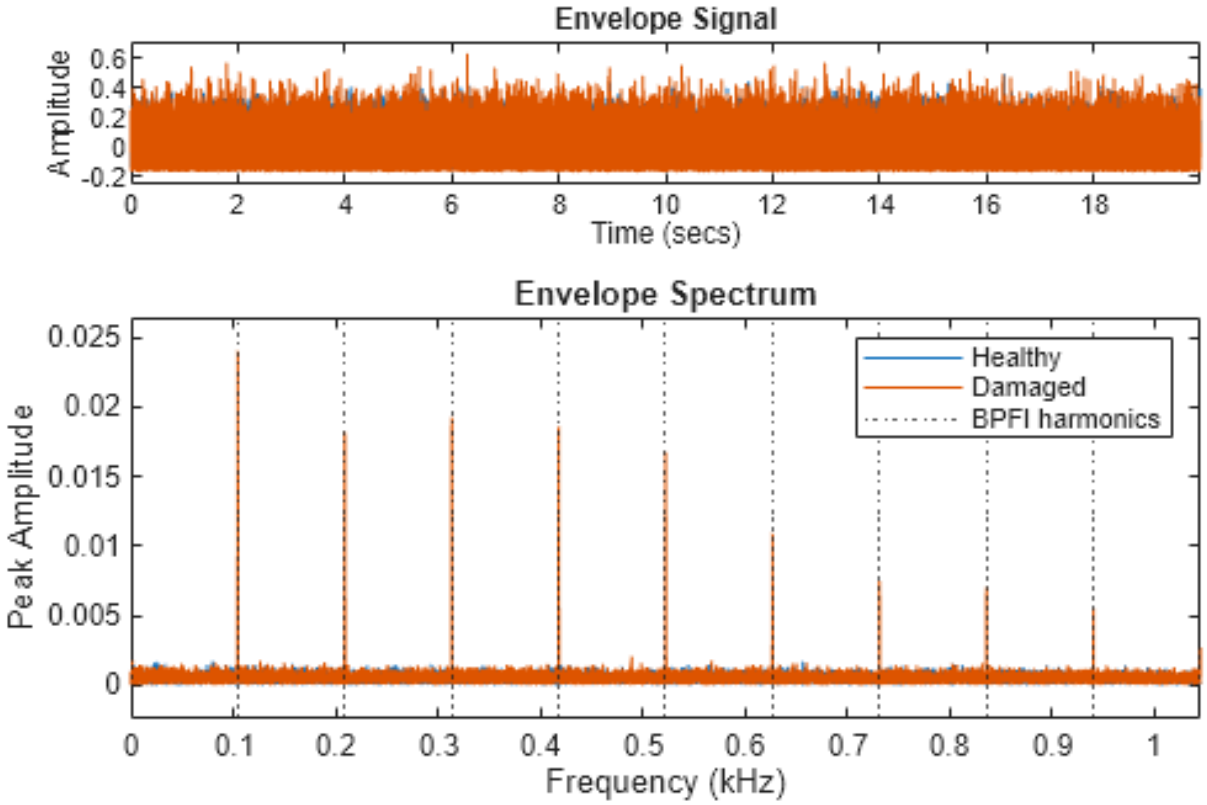
```
figure
envspectrum([vNoBFaultNoisy' vBFaultNoisy'],fs)


xline(harmImpact/1000,":")
xlim([0 10*bpfi]/1000)
legend("Healthy","Damaged","BPFI harmonics")
```



Observe that BPFI peaks are not prominent in the envelope spectrum because the signal is polluted by noise. Recall that performing tsa to average out noise is not useful for bearing-fault analysis because it also averages out the impact signals.

The envspectrum function offers a built-in filter that can be used to remove noise outside the band of interest. Apply a bandpass filter of order 200 centered at 3.125 kHz and 4.167 kHz wide.

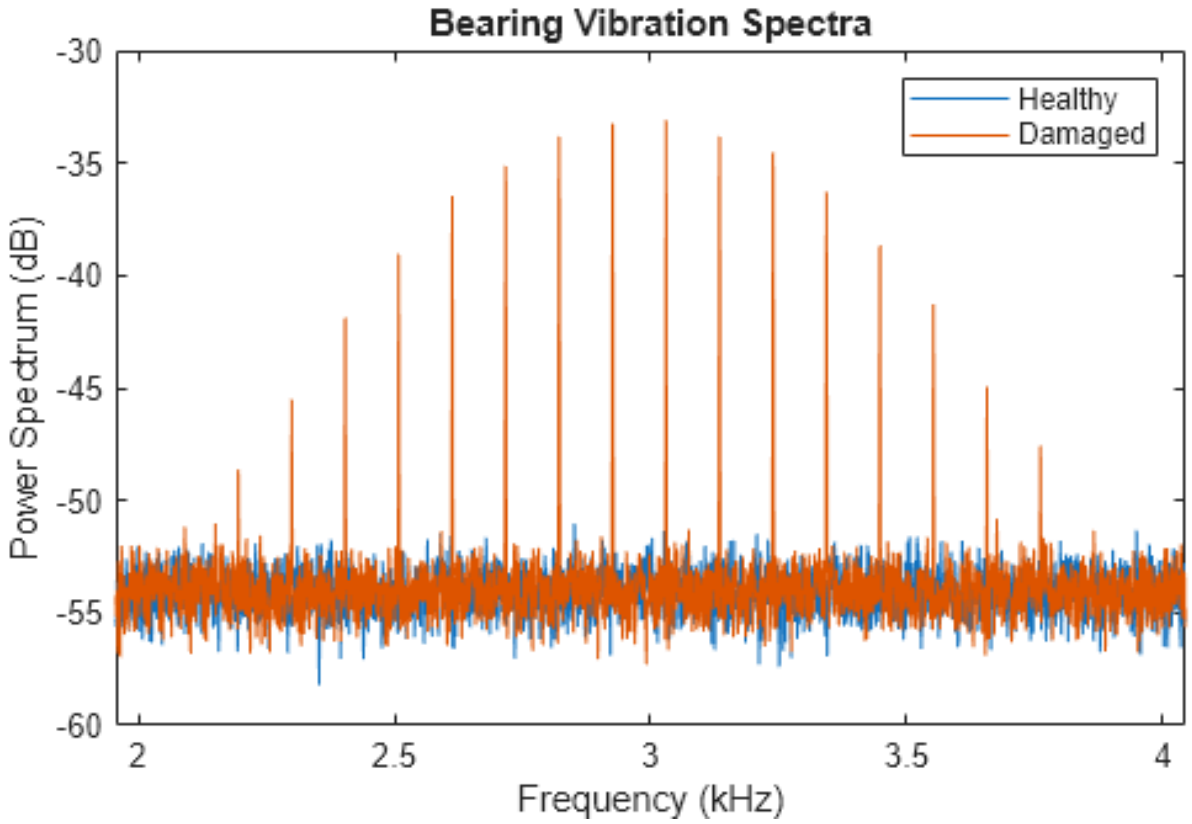
```
Fc = 3125;
BW = 4167;


envspectrum([vNoBFaultNoisy' vBFaultNoisy'],fs, ...
    Method="hilbert",FilterOrder=200,Band=[Fc-BW/2 Fc+BW/2])


harmImpact = (0:10)*bpfi;
xline(harmImpact/1000,":")
xlim([0 10*bpfi]/1000)
legend("Healthy","Damaged","BPFI harmonics")
```



The envelope spectrum effectively brings in the passband content to baseband, and therefore shows the presence of prominent peaks at the BPFI harmonics below 1 kHz. This helps conclude that the inner race of the bearing is potentially damaged.

In this case, the frequency spectrum of the faulty bearing clearly shows BPFI harmonics modulated by the impact frequency. Visualize this phenomenon in the spectra, close to the impact frequency of 3 kHz.

```
figure
pspectrum([vNoBFaultNoisy' vBFaultNoisy'],fs, ...
   FrequencyResolution=1, ...
   FrequencyLimits=(bpfi*[-10 10]+fImpact))
grid off
legend("Healthy","Damaged")
title("Bearing Vibration Spectra")
```



Observe that the separation in frequency between peaks is equal to BPFI.

Conclusions

This example used time-synchronous averaging to separate vibration signals associated with both a pinion and a gear. In addition, tsa also attenuated random noise. In cases of fluctuating speed (and load [2]), order tracking can be used as a precursor to tsa to resample the signal in terms of shaft rotation angle. Time-synchronous averaging is also used in experimental conditions to attenuate the effects of small changes in shaft speed.

Broadband frequency analysis may be used as a good starting point in the fault analysis of bearings [3]. However, its usefulness is limited when the spectra in the neighborhood of bearing impact frequencies contain contributions from other components, such as higher harmonics of gear-mesh frequencies in a gearbox. Envelope analysis is useful under such circumstances. The function envspectrum can be used to extract envelope signals and spectra for faulty bearings, as an indicator of bearing wear and damage.