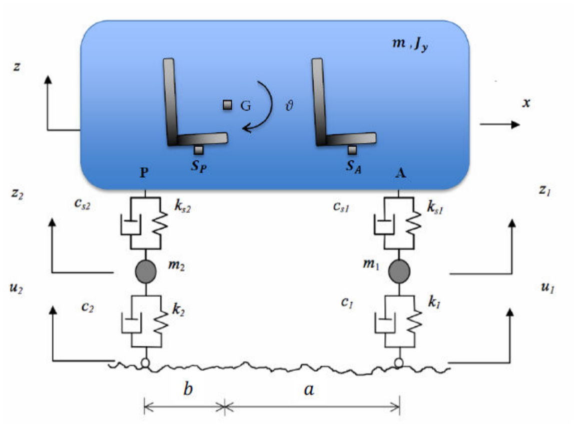
4DOF NVH & COMFORT

The 4DOF model is introduced so to evaluate also the concept of comfort for car's passengers. The difference with the previous model is that now the overall vehicle is considered. It is more of interest in this model to analyse the vertical dynamic, negleting the differences between left and right side of the vehicle (z symmetry). However a difference between front and rear axle is present since they have different stiffness *(1 = front, 2 = rear)*. Also the stiffness of the tyre is included in this model *(kp)*.



The roll motion neglected, insted pitch motion is analysed. Four variables observed: centre of mass vertical displacement (z_g), pitch angle (θ), vertical displacement of unsprung masses (z_1 front and z_2 rear).

The most important difference between the previous model is that the rear axle recive the input after a certain **delay**: model has two input shifted in time.

Equation of motion given, note that all matrices are symmetric.

Data can be declared

```
% 4 DOF Vehicle model for Comfort Analysis

%% Vehicle data
% vehicle data
clear; close all; clc;
warning('off','all');

%------------ masses and inertias ----------------------
% sprung mass [kg]
m =  1790;
% unsprung mass - front [kg]
m1 =  90;
% unsprung mass - rear [kg]
m2 =  70;
% mass moment of inertia I_yy [kg m^2]
Jy = 3.293e3;

%--------------- geometry --------------------------------
% wheelbase [m]
L = 2.885;
```

```
% front wheelbase [m]
a = 1.45;
% rear wheelbase [m]
b = L-a;
% % centre of mass height [m]
% h_g = 0.49;

%----------------- suspensions ----------------------------
% front axle stiffness [N/m]
ks1 = 2*29.1e3;
% rear axle stiffness [N/m]
ks2 = 2*31.5e3;
% front axle damping [Ns/m]
cs1 = 2*5.7e3;
% rear axle damping [Ns/m]
cs2 = 2*4.5e3;

%------------------- tyres (225/50 R 17) --------------------
% N.B.: axle stiffness (2*tyre stiffness)
% tyre stiffness [N/m]
k1 = 2*2.26954e5;
k2 = k1;
% tyre damping coefficient [Ns/m]
c1 = 2*50;
c2 = c1;
% % wheel moment of inertia [kg m^2] (2 wheels)
% Jns = 2*0.8;

% optimum damping
cott = sqrt(m*(ks1+ks2)/2*(k1+k2+2*(ks1+ks2))/(k1+k2));
cs1 = cott/(cs1+cs2)*cs1 /1.2;
cs2 = cott/(cs1+cs2)*cs2 /1.2;

%% System Matrices
% Mass
mm = diag([m,Jy,m1,m2])
```

```
mm = 4×4
      1790           0           0           0
         0        3293           0           0
         0           0          90           0
         0           0           0          70
```

```
% Damping
cc = [cs1+cs2   -cs1*a+cs2*b         -cs1    -cs2
    -cs1*a+cs2*b   cs1*a^2+cs2*b^2   cs1*a   -cs2*b
    -cs1               cs1*a          c1+cs1     0
    -cs2              -cs2*b              0       c2+cs2]
```

```
cc = 4×4
10^4 ×
    1.1540    0.0810   -0.5459   -0.6081
    0.0810    2.4000    0.7916   -0.8726
   -0.5459    0.7916    0.5559         0
   -0.6081   -0.8726         0    0.6181
```

```
% Stiffness
kk = [ks1+ks2   -ks1*a+ks2*b         -ks1     -ks2
     -ks1*a+ks2*b   ks1*a^2+ks2*b^2   ks1*a     -ks2*b
     -ks1              ks1*a          k1+ks1      0
     -ks2             -ks2*b             0       k2+ks2]
```

```
kk = 4×4
10^5 ×
    1.2120    0.0601   -0.5820   -0.6300
    0.0601    2.5210    0.8439   -0.9040
   -0.5820    0.8439    5.1211         0
   -0.6300   -0.9040         0    5.1691
```

Modal analysis

First requirement is to perform the modal analysis of the system, in other words it is required to solve the eigenproblem.

$$([K]-\omega^2[M])[V]=0$$

```
%% Modal analysis of the undamped system
% eigenvalues and eigenvectors
disp('eigenvetors and eigenvalues {ZG,theta,Z1,Z2}')
```

```
eigenvetors and eigenvalues {ZG,theta,Z1,Z2}
```

```
[vv,ww] = eig(kk,mm)
```

```
vv = 4×4
    0.0230   -0.0053   -0.0006   -0.0006
   -0.0039   -0.0170    0.0005   -0.0004
    0.0033    0.0022    0.1053   -0.0000
    0.0021   -0.0036    0.0000    0.1194
ww = 4×4
10^3 ×
    0.0592         0         0         0
         0    0.0679         0         0
         0         0    5.6981         0
         0         0         0    7.3936
```

```
% pause

% natural frequencies
om = sqrt(diag(ww));
f_n = om/2/pi
```

```
f_n = 4×1
    1.2247
    1.3112
   12.0139
   13.6851
```

```

% eigenvectors normalisation (first element of the vectors set equal to 1)
psi_1 = vv(:,1)./(vv(1,1));   % 1st eigenvector
psi_2 = vv(:,2)./(vv(1,2));   % 2nd eigenvector
psi_3 = vv(:,3)./(vv(1,3));   % 3rd eigenvector
```

```
psi_4 = vv(:,4)./(vv(1,4));   % 4th eigenvector

% modal matrix
disp('Modal Matrix')
```

```
Modal Matrix
```

```
psi = [psi_1 , psi_2, psi_3 , psi_4]
```

```
psi = 4×4
    1.0000    1.0000    1.0000    1.0000
   -0.1697    3.2024   -0.7888    0.7813
    0.1431   -0.4191 -173.1735    0.0504
    0.0930    0.6883   -0.0704 -208.1196
```

```

% position of the nodes related to each mode shape
disp('nodes: distance from G')
```

```
nodes: distance from G
```

```
x1_0 = psi_1(1)/psi_1(2)   %[m] node position of the first mode
```

```
x1_0 = -5.8943
```

```
x2_0 = psi_2(1)/psi_2(2)   %[m] node position of the second mode
```

```
x2_0 = 0.3123
```

```
x3_0 = psi_3(1)/psi_3(2)   %[m] node position of the 3rd mode
```

```
x3_0 = -1.2678
```

```
x4_0 = psi_4(1)/psi_4(2)   %[m] node position of the 4th mode
```

```
x4_0 = 1.2799
```

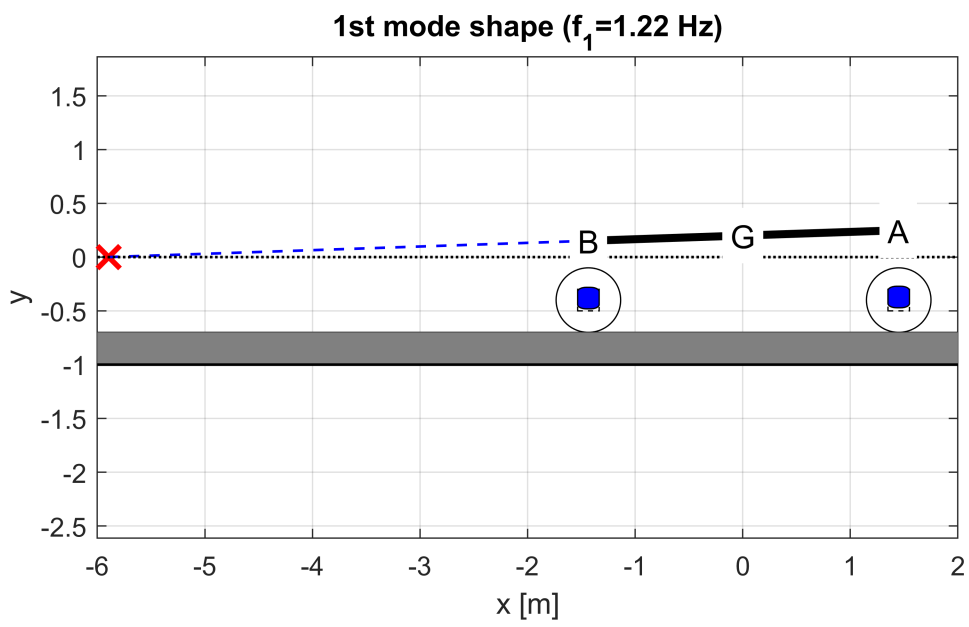
The script required for the modal analysis is directly given by the professor, so it more important to focus over the results, trying to understand what is going on with the system.

In particular, the script given plot each mode.

Mode 1

```
%% Plot mode shape

%% Mode 1
scale_factor = 2e-1;

h_fig_MS = figure('Name','Modal shapes','units','normalized','outerposition',[0 0.5 0.5 0.5]);
x = -b:0.01:a;     % array of the coordinate along beam length
y1 = scale_factor*(psi_1(1) - x*psi_1(2)); % vertical coordinates of the beam (1st mode)

% plot mode shapes
plot(x,y1,'-k','linewidth',3); hold on % 1st mode
plot([x1_0], [0],'xr','linewidth',2,'markersize',12) % nodes
```

```matlab
% extension up to the node of mode 1
x_node = x1_0:0.01:-b;
y1_node = scale_factor*(psi_1(1) - x_node*psi_1(2));
plot(x_node,y1_node,'--b','linewidth',1); hold on

% Plot horizontal line
xlim1 = xlim; ylim1 = ylim;
line(xlim1.',[0; 0],...
    'linewidth',1,...
    'color',[0,0,0],'linestyle',':')
text([x(1),0, x(end)], [y1(1),scale_factor*psi_1(1),y1(end)],{'B','G','A'} ...
    , 'HorizontalAlignment','center',...
    'BackgroundColor',[1 1 1],'FontSize',12)
% Plot vertical line
axis equal
title(['1st mode shape (f_1=',num2str(round(f_n(1)*100)/100),' Hz)']);
xlabel('x [m]'); ylabel('y');


% Plot sprung mass
hold on;
R = 0.1;
offsetZ_m_ns = -0.4;

h3 = rectangle('Position',[-b-R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h4 = rectangle('Position',[-b-R,offsetZ_m_ns+scale_factor*psi_1(4)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
h5 = rectangle('Position',[a-1*R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h6 = rectangle('Position',[a-1*R,offsetZ_m_ns+scale_factor*psi_1(3)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
grid on
% wheels
R = 0.3;
h3 = rectangle('Position',[-b-R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor', ...
    'none','Curvature',[1,1]);
h5 = rectangle('Position',[a-1*R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor', ...
    'none','Curvature',[1,1]);

% road
area(xlim,offsetZ_m_ns-R*[1 1]-0.3,offsetZ_m_ns-R,'linewidth',1,'FaceColor', ...
    [128 128 128]./256); hold on;
```



The first natural frequency: $f_1 = 1.22Hz$. From the chart it is possible to observe how this mode is resambles **a perfect heave motion** since the pitch centre of the system is located very far away from the car.

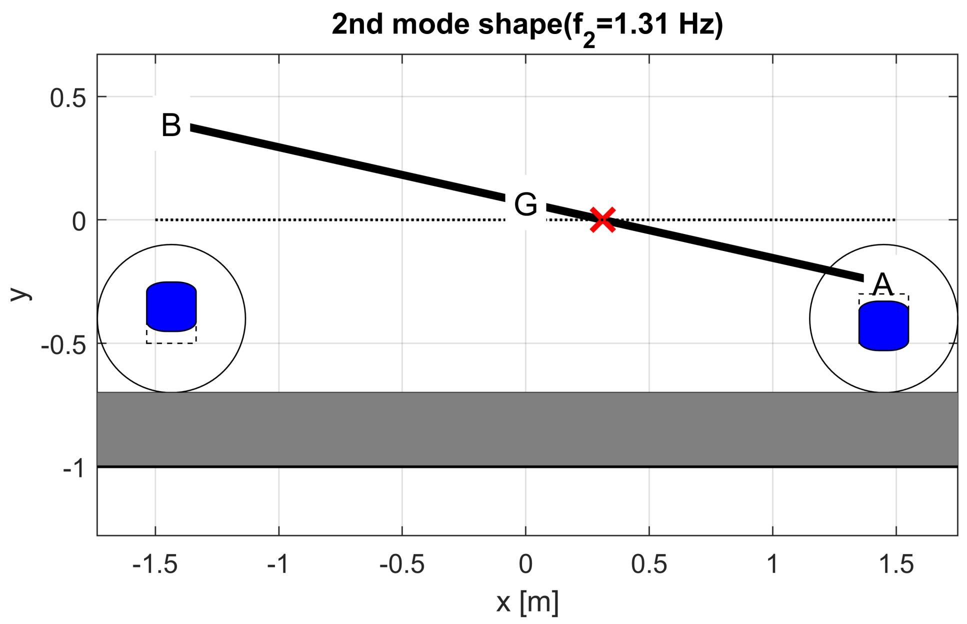
Heave motion correspond to that movement for which all the tyres are subjected to the same vertical displacement, like a floating boat.

However, it is possible to declare that this is a perfect one since the vehicle is slightly rotation.

Mode 2

```
%% Mode 2
scale_factor = 7e-2;

h_fig_MS2 = figure('Name','Modal shapes','units','normalized','outerposition', ...
    [0.5 0.5 0.5 0.5]);
x = -b:0.01:a;     % array of the coordinate along beam length
y2 = scale_factor*(psi_2(1) - x*psi_2(2)); % vertical coordinates of the beam (2nd mode)

% plot mode shapes
plot(x,y2,'-k','linewidth',3);hold on  % 2nd mode
plot([x2_0], [0],'xr','linewidth',2,'markersize',12) % nodes

% Plot horizontal line
xlim1 = xlim; ylim1 = ylim;
line(xlim1.',[0; 0],...
    'linewidth',1,...
    'color',[0,0,0],'linestyle',':')
text([x(1), x(end), 0], [y2(1),y2(end),scale_factor*psi_1(1)],{'B','A','G'}, ...
    'HorizontalAlignment','center',...
    'BackgroundColor',[1 1 1],'FontSize',12)
% Plot vertical line
axis equal
title(['2nd mode shape(f_2=',num2str(round(f_n(2)*100)/100),' Hz)']);
xlabel('x [m]'); ylabel('y');
```

```
% Plot unsprung mass
hold on;
R = 0.1;
% offset Z_m_ns = -0.3;
h3 = rectangle('Position',[-b-R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h4 = rectangle('Position',[-b-R,offsetZ_m_ns+scale_factor*psi_2(4)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
h5 = rectangle('Position',[a-1*R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h6 = rectangle('Position',[a-1*R,offsetZ_m_ns+scale_factor*psi_2(3)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
grid on
% wheels
R = 0.3;
h3 = rectangle('Position',[-b-R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
h5 = rectangle('Position',[a-1*R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
% road
area(xlim,offsetZ_m_ns-R*[1 1]-0.3,offsetZ_m_ns-R,'linewidth',1,'FaceColor', ...
    [128 128 128]./256); hold on;
```



The second mode occurs at the frequency $f_2 = 1.31Hz$ and it is instead a **pitch motion**, since the pitch centre is inside the vehicle, pretty close to the centre of mass of the vehicle.

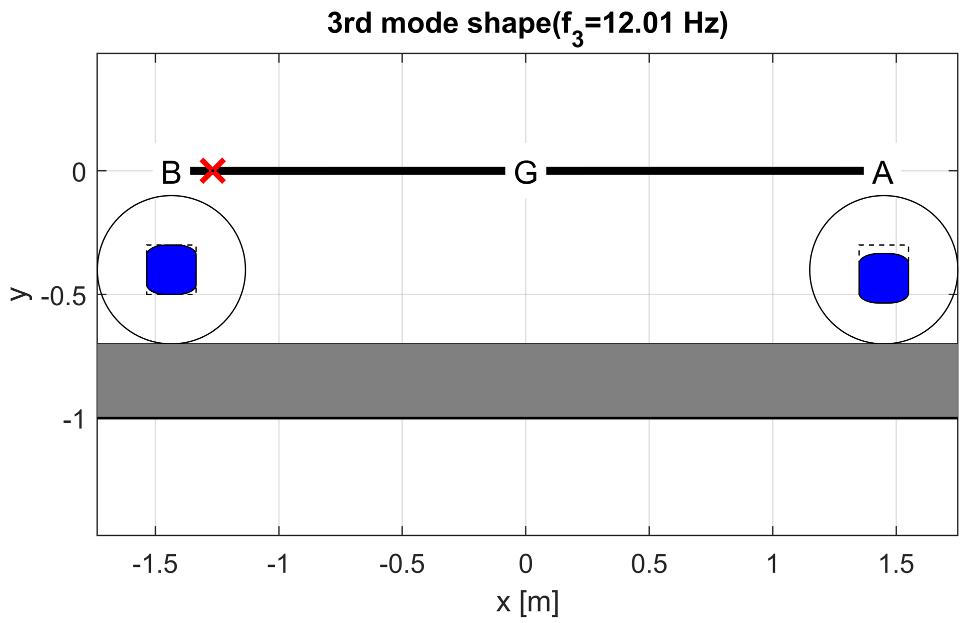
The first two modes are normally refered as: heave mode and pitch mode.

Mode 3

The other two modes are refered to the unsprung masses vibrations.

NOTE: there is some congruency between the variables and the modes. Being the 4DOF centre of mass displacement, pitch and unsprung displacements, the system will have for each of them a particular mode of vibration.

```
%% Mode 3
scale_factor = 2e-4;

h_fig_MS3 = figure('Name','Modal shapes','units','normalized','outerposition',[0 0 0.5 0.5]);
x = -b:0.01:a;     % array of the coordinate along beam length
y3 = scale_factor*(psi_3(1) - x*psi_3(2)); % vertical coordinates of the beam (1st mode)

% plot mode shapes
plot(x,y3,'-k','linewidth',3); hold on % 1st mode
plot([x3_0], [0],'xr','linewidth',2,'markersize',12) % nodes

% extension up to the node of mode 1
x_node = x3_0:0.01:-b;
y3_node = scale_factor*(psi_3(1) + x_node*psi_3(2));
plot(x_node,y3_node,'--b','linewidth',1); hold on

% Plot horizontal line
xlim1 = xlim; ylim1 = ylim;
line(xlim1.',[0; 0],...
    'linewidth',1,...
    'color',[0,0,0],'linestyle',':')
text([x(1),0, x(end)], [y3(1),scale_factor*psi_3(1),y3(end)],{'B','G','A'}, ...
    'HorizontalAlignment','center',...
    'BackgroundColor',[1 1 1],'FontSize',12)
% Plot vertical line
axis equal
title(['3rd mode shape(f_3=',num2str(round(f_n(3)*100)/100),' Hz)']);
xlabel('x [m]'); ylabel('y');

% Plot unsprung mass
hold on;
R = 0.1;
h3 = rectangle('Position',[-b-R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h4 = rectangle('Position',[-b-R,offsetZ_m_ns+scale_factor*psi_3(4)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
h5 = rectangle('Position',[a-1*R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h6 = rectangle('Position',[a-1*R,offsetZ_m_ns+scale_factor*psi_3(3)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
grid on
% wheels
R = 0.3;
h3 = rectangle('Position',[-b-R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
h5 = rectangle('Position',[a-1*R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
% road
area(xlim,offsetZ_m_ns-R*[1 1]-0.3,offsetZ_m_ns-R,'linewidth',1,'FaceColor', ...
    [128 128 128]./256); hold on;
```

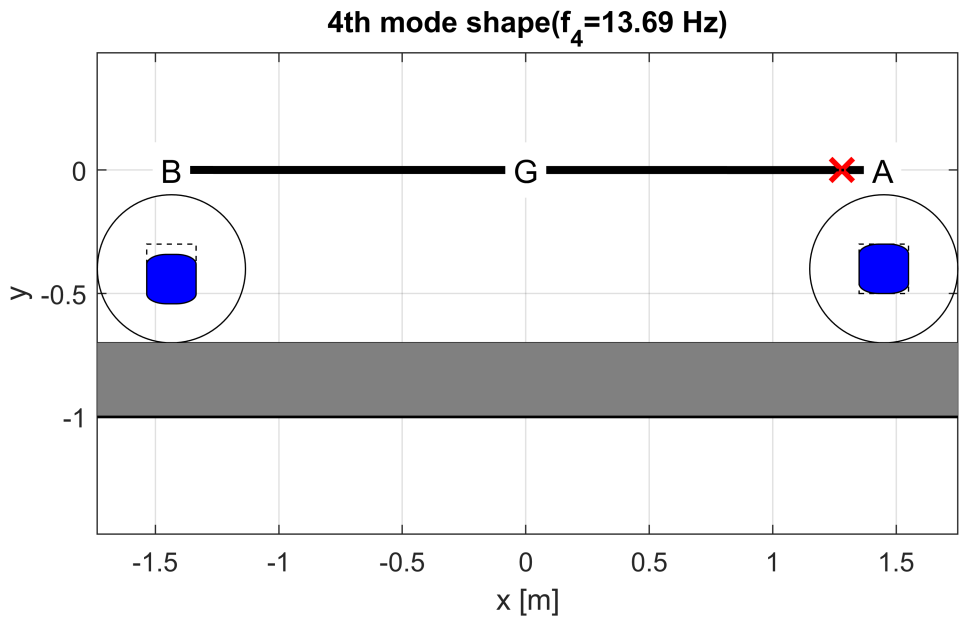


The third natural frequency is $f_3 = 12.01 Hz$, which is an order of magnitude bigger than the first one (this ratio of x10 is always present in suspensions!). This mode referes to the front unsprung mass, therefore how the front tyres will vibrates.

Mode 4

```
%% Mode 4
scale_factor = 2e-4;

h_fig_MS4 = figure('Name','Modal shapes','units','normalized','outerposition',[0.5 0 0.5 0.5]);
x = -b:0.01:a;     % array of the coordinate along beam length
y4 = scale_factor*(psi_4(1) - x*psi_4(2)); % vertical coordinates of the beam (2nd mode)

% plot mode shapes
plot(x,y4,'-k','linewidth',3);hold on  % 2nd mode
plot([x4_0], [0],'xr','linewidth',2,'markersize',12) % nodes

% Plot horizontal line
xlim1 = xlim; ylim1 = ylim;
line(xlim1.',[0; 0],...
    'linewidth',1,...
    'color',[0,0,0],'linestyle',':')
text([x(1), x(end), 0], [y4(1),y4(end),scale_factor*psi_4(1)],{'B','A','G'} ...
    , 'HorizontalAlignment','center',...
    'BackgroundColor',[1 1 1],'FontSize',12)
% Plot vertical line
axis equal
title(['4th mode shape(f_4=',num2str(round(f_n(4)*100)/100),' Hz)']);
xlabel('x [m]'); ylabel('y');

% Plot unsprung mass
hold on;
R = 0.1;
```

```
h3 = rectangle('Position',[-b-R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h4 = rectangle('Position',[-b-R,offsetZ_m_ns+scale_factor*psi_4(4)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
h5 = rectangle('Position',[a-1*R,offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[0,0],'linestyle','--','linewidth',0.5);
h6 = rectangle('Position',[a-1*R,offsetZ_m_ns+scale_factor*psi_4(3)-R,2*R,2*R], ...
    'FaceColor','b','Curvature',[0.8,0.4]);
grid on
% wheels
R = 0.3;
h3 = rectangle('Position',[-b-R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
h5 = rectangle('Position',[a-1*R,1*offsetZ_m_ns-R,2*R,2*R],'FaceColor','none', ...
    'Curvature',[1,1]);
% road
area(xlim,offsetZ_m_ns-R*[1 1]-0.3,offsetZ_m_ns-R,'linewidth',1,'FaceColor', ...
    [128 128 128]./256); hold on;
```



The last frequency is $f_4 = 13.69 Hz$ and referes to the rear vibration mode.

Effect of shock absorber damping

After evaluating the natural frequencies and the modes correspoding to them, it is possible to evaluate the modal analysis of the damped system and see the influence of the damping over the natural frequencies. Modal analysis for damping systems cannot be done using `eig` since woth the contribution of the damping factor there is no more an eigenproblem. However, Matlab provides already a function which run modal analysis for damped systems: `damp()`.

```
%% Modal analysis of the damped system
% eigenvalues and eigenvectors (both complex)
n = 4;   % number of system degrees of freedom
A = [ zeros(n,n)  diag(ones(n,1))
```

```
    -mm^-1*kk        -mm^-1*cc]; % state matrix of state-space representation
[V,D] = eig(A)  % produces a diagonal matrix lambda of generalized eigenvalues and a
```

```
V = 8×8 complex
  -0.0002 + 0.0005i  -0.0002 - 0.0005i   0.0003 - 0.0005i   0.0003 + 0.0005i ⋯
  -0.0001 + 0.0004i  -0.0001 - 0.0004i  -0.0003 + 0.0004i  -0.0003 - 0.0004i
  -0.0001 + 0.0001i  -0.0001 - 0.0001i   0.0059 + 0.0123i   0.0059 - 0.0123i
  -0.0065 - 0.0101i  -0.0065 + 0.0101i  -0.0002 + 0.0001i  -0.0002 - 0.0001i
  -0.0238 - 0.0325i  -0.0238 + 0.0325i   0.0194 + 0.0360i   0.0194 - 0.0360i
  -0.0178 - 0.0261i  -0.0178 + 0.0261i  -0.0164 - 0.0283i  -0.0164 + 0.0283i
   0.0007 - 0.0137i   0.0007 + 0.0137i  -0.9984 + 0.0000i  -0.9984 + 0.0000i
   0.9985 + 0.0000i   0.9985 + 0.0000i   0.0040 - 0.0175i   0.0040 + 0.0175i
D = 8×8 complex
 -44.7984 +69.7387i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i ⋯
   0.0000 + 0.0000i -44.7984 -69.7387i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i -31.4599 +66.1251i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i -31.4599 -66.1251i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
%     full matrix vv whose columns are the corresponding eigenvectors so that A*V = V*D
[Wn,Z,P] = damp(A);
% [Wn,Z,P] = damp(SYS) returns vectors Wn, Z and P containing the natural frequencies, damping
f_damp = Wn/(2*pi)
```

```
f_damp = 8×1
   13.1920
   13.1920
   11.6545
   11.6545
    1.2622
    1.2622
    1.3605
    1.3605
```

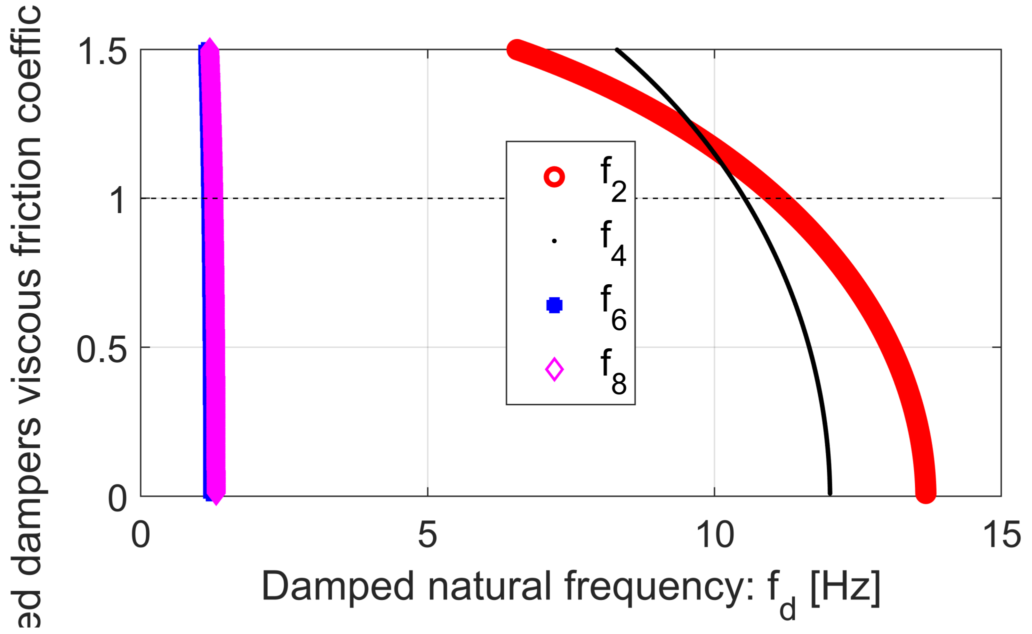
```
abs(P)/(2*pi)
```

```
ans = 8×1
   13.1920
   13.1920
   11.6545
   11.6545
    1.2622
    1.2622
    1.3605
    1.3605
```

```
Z
```

```
Z = 8×1
    0.5405
    0.5405
    0.4296
    0.4296
    0.3279
    0.3279
    0.3560
    0.3560
```

```
%% Sensitivity analysis of the damping ratio for different viscous coefficient
flag_plot_comp_mode = 0;

switch flag_plot_comp_mode
    case 1
        h_fig_mode_comp = figure('units','normalized','Outerposition',[0 0 0.5 1]); hold all
        h_fig_freq = figure('units','normalized','Outerposition',[0.5 0 0.5 1]); hold all
end

cs1_vet = cs1*linspace(0.01,1.5,10000);
cs2_vet = cs2*linspace(0.01,1.5,10000);

cs1_n = cs1;
cs2_n = cs2;

% Initialization
f_c = zeros(length(cs1_vet),length(P));
zeta = zeros(length(cs1_vet),length(P));

for cont1=1:length(cs1_vet)
    cs1 = cs1_vet(cont1); cs2 = cs2_vet(cont1);

    % Damping matrix update
    cc = [cs1+cs2   -cs1*a+cs2*b         -cs1    -cs2
        -cs1*a+cs2*b   cs1*a^2+cs2*b^2   cs1*a   -cs2*b
        -cs1               cs1*a           c1+cs1     0
        -cs2               -cs2*b              0       c2+cs2];

    A = [ zeros(n,n)  diag(ones(n,1))
        -mm^-1*kk       -mm^-1*cc];
    [V,D] = eig(A);  % produces a diagonal matrix lambda of generalized eigenvalues and a
    %     full matrix vv whose columns are the corresponding eigenvectors so that A*V = V*D

    psi_1 = V(:,1)./(V(1,1));   % 1st eigenvector
    psi_2 = V(:,2)./(V(1,2));   % 2nd eigenvector
    psi_3 = V(:,3)./(V(1,3));   % 1st eigenvector
    psi_4 = V(:,4)./(V(1,4));   % 2nd eigenvector

    psi = [psi_1 psi_2 psi_3 psi_4];

    [Wn,Z,P] = damp(A);
    f_damp = Wn/(2*pi);

    f_c(cont1,:) = [f_damp];
    zeta(cont1,:) = [Z];
end

flag_sort_f = 0;
switch flag_sort_f
    case 1
        [f_c_ord, in] = sort(f_c,2);
        for cont2=1:length(in)
            in(cont2,:);
            zeta_ord(cont2,:) = zeta(cont2,in(cont2,:));
        end
    case 0
        f_c_ord = f_c;
        zeta_ord = zeta;
end

%% Plot f e zeta(c)
figure('units','normalized','outerposition',[0 0.5 0.5 0.5]);
plot(f_c_ord(:,2).*sqrt(1-zeta_ord(:,2).^2),cs1_vet/cs1_n,'or','linewidth',2); hold on
plot(f_c_ord(:,4).*sqrt(1-zeta_ord(:,4).^2),cs1_vet/cs1_n,'.k','linewidth',3);
plot(f_c_ord(:,6).*sqrt(1-zeta_ord(:,6).^2),cs1_vet/cs1_n,'+b','linewidth',4);
plot(f_c_ord(:,8).*sqrt(1-zeta_ord(:,8).^2),cs1_vet/cs1_n,'dm','linewidth',1);
plot(xlim,[1 1],'--k')
% ylim([0 5])
leg = legend('f_2','f_4','f_6','f_8','Location','best'); set(leg,'Fontsize',14)
ylabel('normalized dampers viscous friction coefficient: c_s/c_{s,nom}','Fontsize',14);
xlabel('Damped natural frequency: f_d [Hz]'); grid on;
set(gca,'Fontsize',14)
```

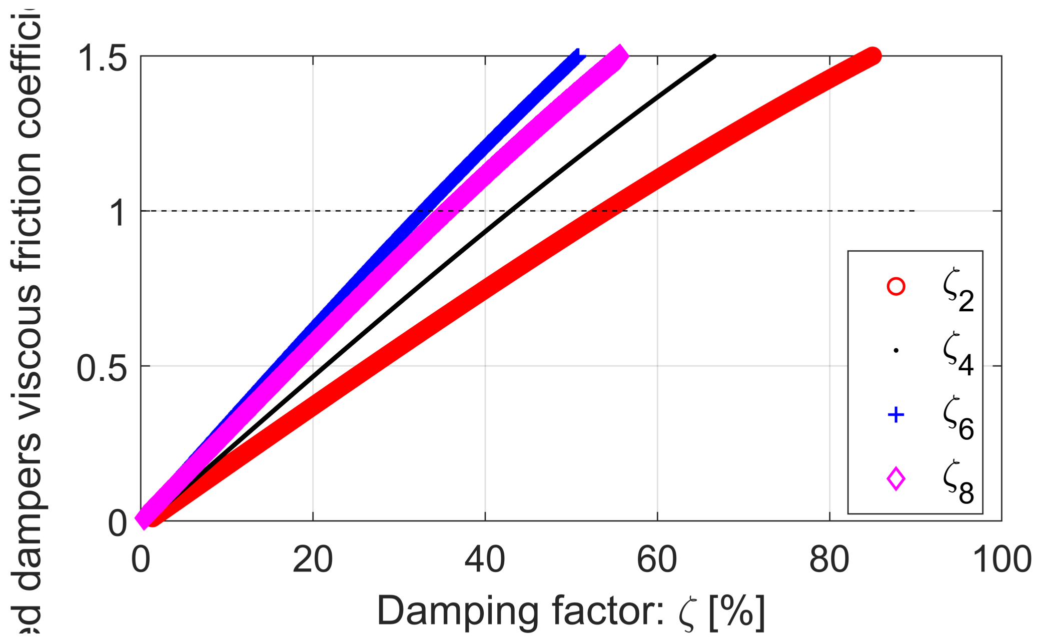


On the y axis the damping friction coefficents are normalaised to the nominal value of the damping (remeber the nromalaised one is equal to the optimal damping).

The first two modes are pratically uninfluenced by the damping, meanwhile the other two decrease for bigger values of damping.

```
% Plot zeta
figure('units','normalized','outerposition',[0.5 0.5 0.5 0.5]);
plot(zeta_ord(:,2)*100,cs1_vet/cs1_n,'or','linewidth',1); hold on
plot(zeta_ord(:,4)*100,cs1_vet/cs1_n,'.k','linewidth',1);
plot(zeta_ord(:,6)*100,cs1_vet/cs1_n,'+b','linewidth',1);
plot(zeta_ord(:,8)*100,cs1_vet/cs1_n,'dm','linewidth',1);
plot(xlim,[1 1],'--k')
ylabel('normalized dampers viscous friction coefficient: c_s/c_{s,nom}','Fontsize',14);
xlabel('Damping factor: \zeta [%]'); grid on;
```

```
leg = legend('\zeta_2','\zeta_4','\zeta_6','\zeta_8','Location','best');
set(leg,'Fontsize',14)
set(gca,'Fontsize',14)
```



This chart shows that increasing the damping factor, it will influence the four modes in a different way, thererfore the variation is not linear (ot it is but with different linearization for each mode).

NOTE: The damped system implies that the mode analysis becomes complex analysis. For this reason there are a total of 8 natural frequencies, but since they are all complex and conuigate (I think is due to the fact that all matrices are symmetrical) in the end only half of them are considered.

Mode	Damping factor: $c_{s,nom}$	Damping factor: $c_{s,nom}/2$	Damped frequency for $c_{s,nom}$ [Hz]:	Damped frequency for $c_{s,nom}/2$ [Hz]:	Natural undamped frequency [Hz]
Shake (Blue)	32,79	16,13	1,19	1,21	1,22
Pitch (Green)	35,6	17,44	1,27	1,30	1,31
Front Unsprung Mass (Black)	42,96	21,43	10,52	11,65	12,01
Rear Unsprung Mass (Red)	54,05	26,78	11,1	13,07	13,68

This table reasumes the effects of the damped system over the natural frequencies and damping factor. Firstly it is important to notice how having the damping factor equal to the nominal one implies different values of damping factors, suggesting that each mode has its own reaction tot he damping variation. Furthemore it is possible to notice how the reduction of the damping value implies variation in the damped natural frequencies whoch are not linear (unsprung modes are parabolic).

Frequency responce to road excitation

The frequency responce for this 4DOF system is a sinusoidal input which can be written as a complex number: $h_0 \sin(\Omega t) \rightarrow h_0 e^{i\Omega t}$,. This input can be applied simultaneously $u1 = u2 = u(t) = h_0 e^{i\Omega t}$ to both wheel or with some delay on the second axle respect tot he first one. In the latter situation, the input becomes: $u_2 = u_1(t - \tau)$, where the delay is $\tau = \frac{L}{V}$.

The frequency responce can be calculated analytically by exploiting the advantages of the complex numbers. The final solution is: $\{\ddot{q}_0\} = -\Omega^2[(-\Omega^2[M] + i\Omega[C]) + [K])^{-}1]([K_u] + i\Omega[C_u])\{u_0\}$, or defining the **trasmissibility matrix**: $\{\ddot{q}_0\} = \{\ddot{z}_G \ \ddot{\theta} \ \ddot{z}_f \ \ddot{z}_R\}^T = [T(\Omega)]\{1 \ \ e^{-i\Omega\tau}\}^T h_0$. The terms $([K_u] + i\Omega[C_u])$ refers to the wheel stiffness and damping.

Please note that the trasmissibility is just the receptance matrix of the car multiplied by the receptance matrix of the tyre. Furthemore the receptance matrix is always defined in the same way indipendelty on the of degrees of freedom of the system.

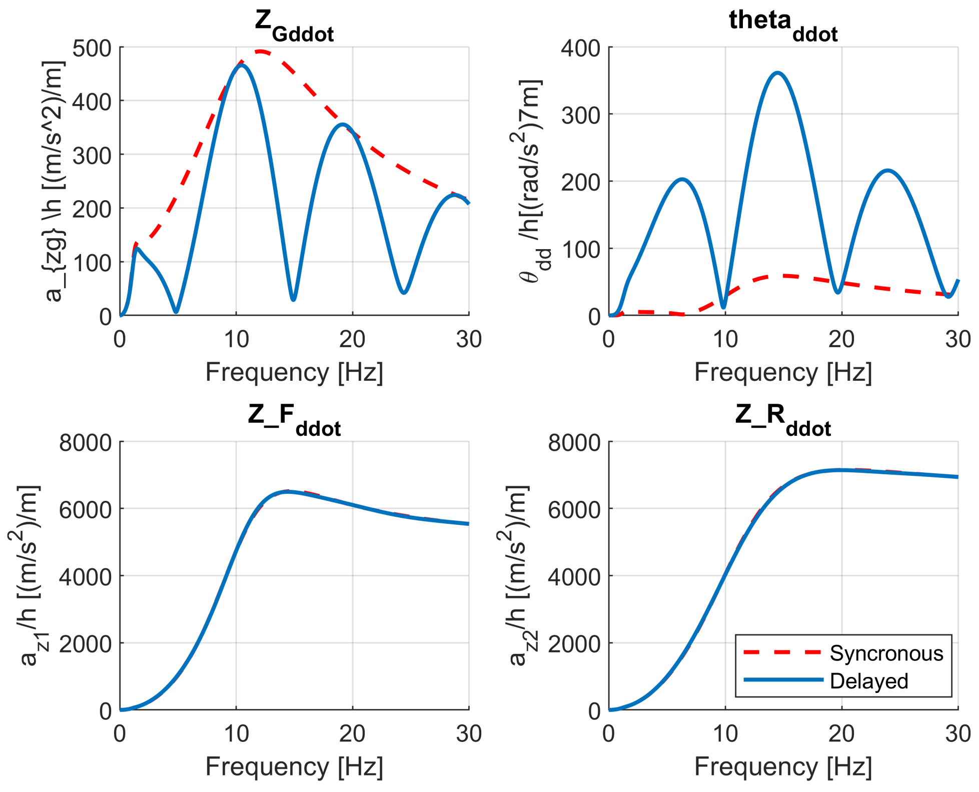
The transfer fucntion can be computed simply exploiting `MATLAB` capacities to perform matrix calculation. Please not that the transfer function wanted is considering as input h, not simply u, so please remeber to divide the vector u_0.

```
%% FRF
% Damping
cc = [cs1_n+cs2_n   -cs1_n*a+cs2_n*b         -cs1_n    -cs2_n
    -cs1_n*a+cs2_n*b   cs1_n*a^2+cs2_n*b^2   cs1_n*a    -cs2_n*b
    -cs1_n               cs1_n*a          c1+cs1_n     0
    -cs2_n              -cs2_n*b            0       c2+cs2_n];
%Variables and initialization
V = 100/3.6;
tau = L/V;

OM = 2*pi*[0:1e-2:30]';      % rad/s
lambda = 2*pi*V./OM;         % m

Ku = [0 0; 0 0; k1 0; 0 k2];
Cu = [0 0; 0 0; c1 0; 0 c2];

%  initialization of FRF vectors
Z_G_ddot_syn      = ones(length(OM),1);
theta_ddot_syn     = ones(length(OM),1);
Z_F_ddot_syn       = ones(length(OM),1);
Z_R_ddot_syn       = ones(length(OM),1);

h0 = 1/100; %1 cm of amplitude, seen in the slides
```

Plots

```
%% SYNCRONOUS
tau = 0;
for count1=1:length(OM)
    ww      = OM(count1);
    u1     = 1;
    u2     = 1;    %To obtian the trasnfer function of the system of our
    % interest it is necessary to consider the input that has been imposed.
    % In syncronous case it is u0 defined as unit column
    Kdyn    = kk + i*ww*cc - ww^2*mm;
    Rec     = inv(Kdyn);
    Trans   = Rec*(Ku + i*ww*Cu);

    Z_G_ddot_syn (count1,1)   = -ww^2*Trans(1,:)*[u1 ; u2];
    theta_ddot_syn (count1,1) = -ww^2*Trans(2,:)*[u1 ; u2];
    Z_F_ddot_syn (count1,1)   = -ww^2*Trans(3,:)*[u1 ; u2];
    Z_R_ddot_syn (count1,1)   = -ww^2*Trans(4,:)*[u1 ; u2];
end

% Plots
close all;
figure(7);
subplot(2,2,1); hold all;
plot(OM/(2*pi),abs(Z_G_ddot_syn ),'r--','Linewidth',1.5); grid on;
title('Z_G_{ddot}'); xlabel('Frequency [Hz]'); ylabel('Z_G_{ddot} [mm/s^2]');

subplot(2,2,2); hold all;
plot(OM/(2*pi),abs(theta_ddot_syn ),'r--','Linewidth',1.5); grid on;
title('theta_{ddot}'); xlabel('Frequency [Hz]'); ylabel('\theta_{ddot} [rad^2]');

subplot(2,2,3); hold all;
plot(OM/(2*pi),abs(Z_F_ddot_syn ),'r--','Linewidth',1.5); grid on;
title('Z\_F_{ddot}'); xlabel('Frequency [Hz]'); ylabel('Z_F_{ddot} [mm/s^2]')

subplot(2,2,4); hold all;
plot(OM/(2*pi),abs(Z_R_ddot_syn ),'r--','Linewidth',1.5); grid on;
title('Z\_R_{ddot}'); xlabel('Frequency [Hz]'); ylabel('Z_R_{ddot} [mm/s^2]')

%% DELAYED
Z_G_ddot_del      = ones(length(OM),1);
theta_ddot_del    = ones(length(OM),1);
Z_F_ddot_del      = ones(length(OM),1);
Z_R_ddot_del      = ones(length(OM),1);
tau = L/V;

for count1=1:length(OM)
    ww      = OM(count1);

    u1     = 1;
    u2     = exp(-i*ww*tau);

    Kdyn    = kk + i*ww*cc - ww^2*mm;
    Rec     = inv(Kdyn);
    Trans   = Rec*(Ku + i*ww*Cu);

    Z_G_ddot_del(count1,1) = -ww^2*Trans(1,:)*[u1 ; u2];
    theta_ddot_del(count1,1) = -ww^2*Trans(2,:)*[u1 ; u2];
    Z_F_ddot_del(count1,1)= -ww^2*Trans(3,:)*[u1 ; u2];
    Z_R_ddot_del(count1,1)= -ww^2*Trans(4,:)*[u1 ; u2];
```

```
end

% Plots
figure(7);
subplot(221); hold all;
plot(OM/(2*pi),abs(Z_G_ddot_del),'color',[0, 0.4470, 0.7410],'Linewidth',1.5); grid on;
ylabel('a_{zg} \h [(m/s^2)/m]')
%  plot(x,y,'*','color',[.5 .4 .7])

subplot(222); hold all;
plot(OM/(2*pi),abs(theta_ddot_del),'color',[0, 0.4470, 0.7410],'Linewidth',1.5); grid on;
ylabel('\theta_{dd} /h[(rad/s^2)7m]')

subplot(223); hold all;
plot(OM/(2*pi),abs(Z_F_ddot_del),'color',[0, 0.4470, 0.7410],'Linewidth',1.5); grid on;
xlabel('Frequency [Hz]')
ylabel('a_{z1}/h [(m/s^2)/m]')

subplot(224); hold all;
plot(OM/(2*pi),abs(Z_R_ddot_del),'color',[0, 0.4470, 0.7410],'Linewidth',1.5); grid on;
xlabel('Frequency [Hz]')
ylabel('a_{z2}/h [(m/s^2)/m]')
legend('Syncronous','Delayed','Location','best')
```



Delaying the input has effects only over the sprung trasnfer functions, meanwhile the unsprung masses shows any changes. The delayed soluton makes reference to a pitch motion (plot on the top-right) meanwhile the syncronous one is a heave motion (top-left).

The transfer function of syncronous motion (heave) is the envelop of the delayed motion (pitch). The syncronous motion can be see as the maximum of the heave motion and the minimum of the pitch one.

When there is the maximum of the pitch motion there is the minimum of the heave and viceversa. By empirical conclusion, having a certain speed can imply heave motion but changing it the motion can become a pitch one: these two motion are complementary.

Random road profile

Since now all the inputs have been modeled as linear or harmonic functions, however in the real world this is not true at all. For this reason a random road profile should be adopted. In order to do that, it is necessary that the process satifies two conditions:

- Stationary: propeties are constant in time.
- Ergotic: process can be deduced from a sample.

In order to deal with randomness, the **spatial correlation function** will be used.

General concept: Spatial autocorrelation is an important concept in spatial statistics, as it allows for spatial interpolation. Autocorrelation (whether spatial or not) is a measure of similarity (correlation) between nearby observations. Measures of spatial autocorrelation describe the degree two which observations (values) at spatial locations (whether they are points, areas, or raster cells), are similar to each other. So we need two things: observations and locations. *“The first law of geography: Everything is related to everything else, but near things are more related than distant things.”*

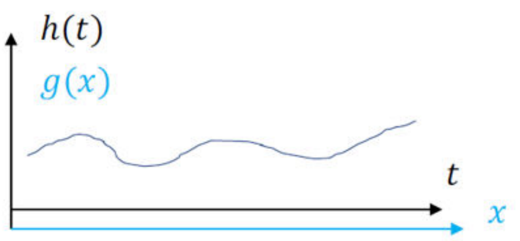
In the case analysed in this exercise, the spatial correlationfunction is defined as follows:

$$R_g(\xi) = \lim_{L\to\infty} \frac{1}{L} \int_{-L/2}^{L/2} g(x)g(x+\xi)dx$$

From it, the Fourier First Transform is applied, therefore a simple switch to frequency domain is done, obtaining the so called **PSD** (power spectral density). A Power Spectral Density (PSD) is the measure of signal's power content versus frequency.

$$S_g(s) = \int_{-\infty}^{+\infty} R_g(\xi)\, e^{-\iota s\xi}\, d\xi$$

Where S_g is measured in $\frac{m^2}{cycle/m}$ or m^3. Up the spatial refernce has been used, but since the road profile should be a temporal signal, by dividing for the velocity it is possible to pass to the "time domain", or to be more precise, to the temporal dependency of the signal.



$g(x) \rightarrow$ [m] and $h(t) \rightarrow$ [m]

$V = x/t$ and $h(t) = g(Vt)$

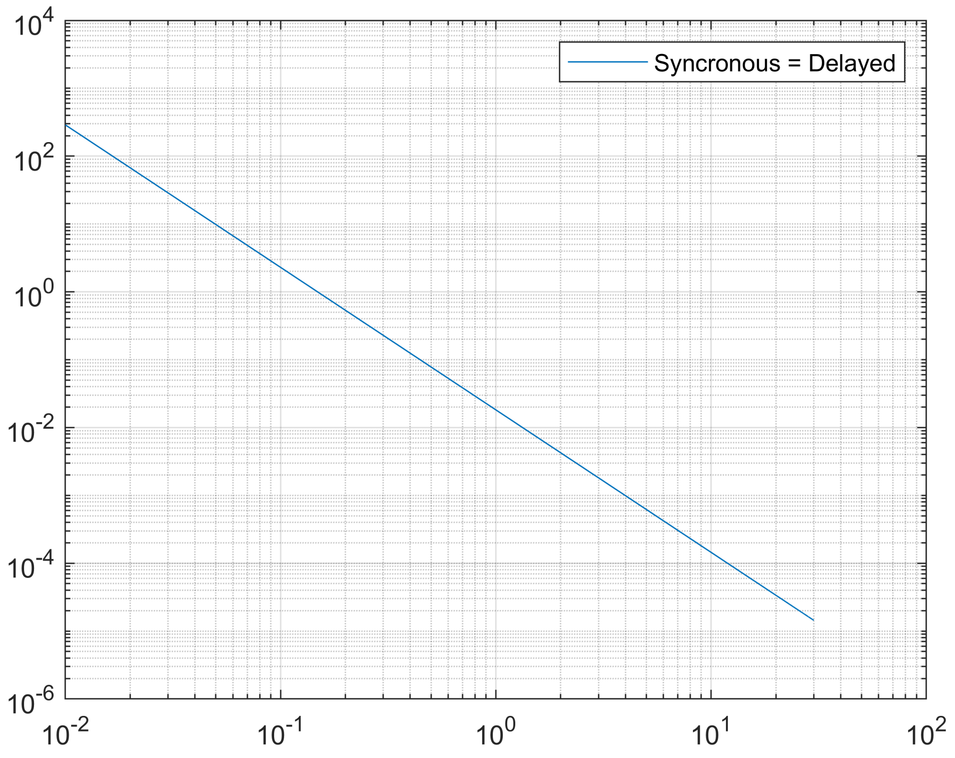
$$S_g(s) \qquad \rightarrow \qquad S_h(f) = \frac{S_g(s)}{V} = \frac{S_g(f/V)}{V} \quad [\mathrm{m}^2/\mathrm{Hz}]$$

Then it is possible to compute each possible transfer function having as input the random road profile, just multiplying the vehicle transfer function (squared) by the S_h obtained . Please remeber that all this process is still performed into the frequency domain, therefore if there is a double derivative term (like the sprung mass acceleration), the pulsation should be multiplied.

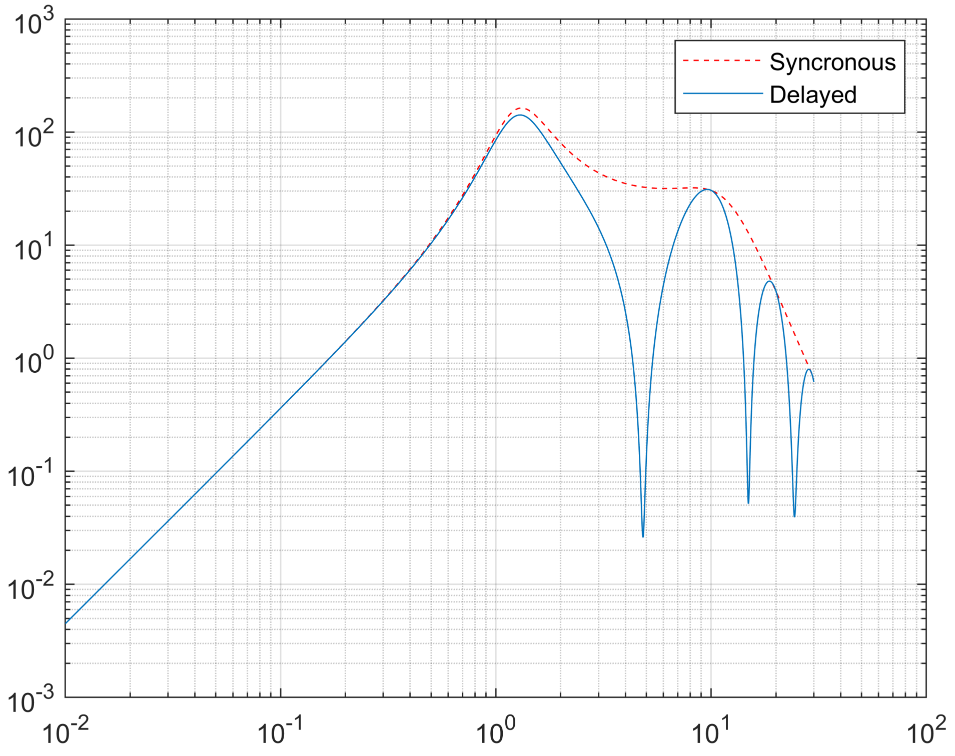
$$S_r(\omega) = G^2(\omega)\, \underset{\displaystyle \downarrow \atop \displaystyle S = c\, V^{n-1} f^{-n}}{S(\omega)} \quad \text{: vehicle response to road random input}$$

The values of the variables are given as input, since they have been evaluated empirically.

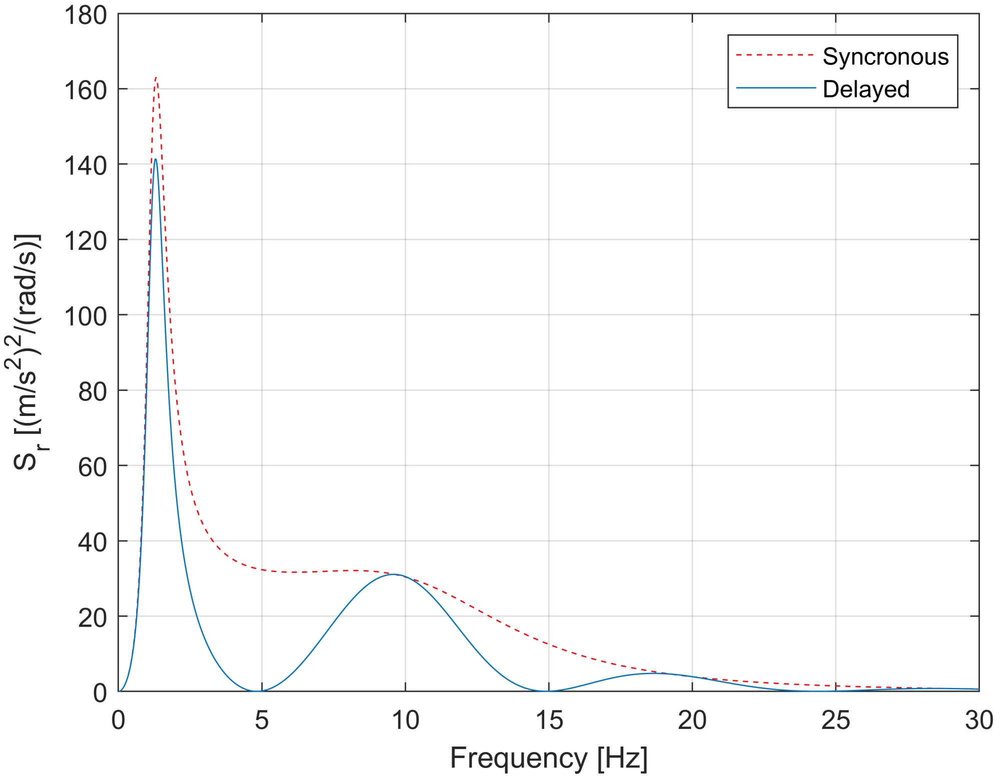
```
c = 4.7*10^-4; n=2.1;
f = OM./(2*pi);

S = c*V^(n-1).*f.^-n;
S_r_syn = Z_G_ddot_syn.^2.*S;
S_r_del = Z_G_ddot_del.^2.*S;

% plots
close all;
figure(9); loglog(f,abs(S)); grid on; legend('Syncronous = Delayed')
```



```
figure(10); loglog(f,abs(S_r_syn),'r--'); grid on; hold on;
figure(10); loglog(f,abs(S_r_del),'color',[0, 0.4470, 0.7410]);legend('Syncronous','Delayed')
```



```
figure(11); plot(f,abs(S_r_syn),'r--'); grid on; hold on;
figure(11); plot(f,abs(S_r_del),'color',[0, 0.4470, 0.7410]); hold on;
xlabel('Frequency [Hz]'); ylabel('S_r [(m/s^2)^2/(rad/s)]');
legend('Syncronous','Delayed')
```



Also in the plot of the transfer functions, it is possible to observe how the syncronous being a heave motion has its TF as the envelop of the delayed one, which resamble a pitch motion instead. Results coming from the TF with sinusoidal input maintain their validity even for a random road input.