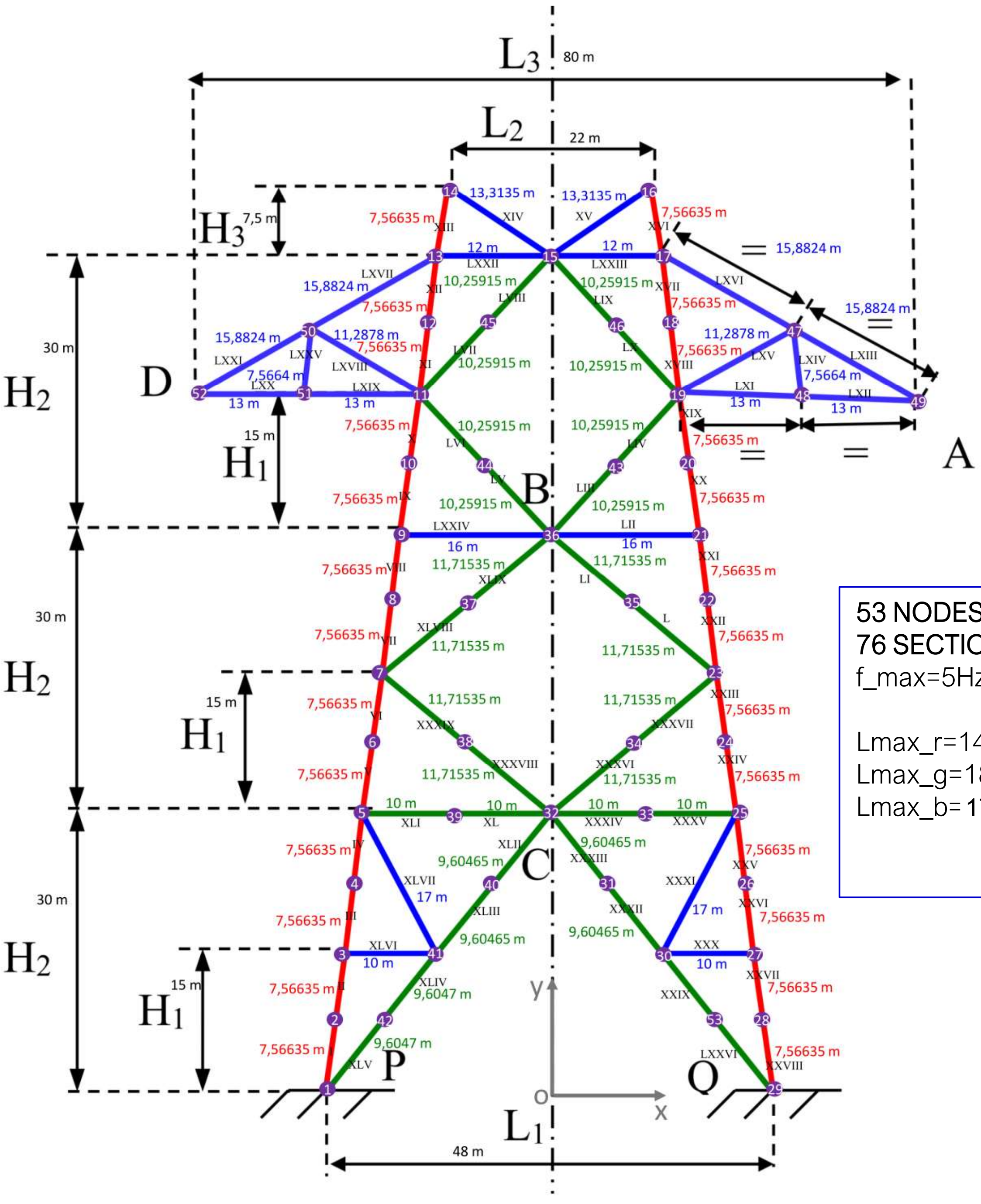
YEARWORK REPORT

- **Q1.** Geometrical analysis of the structure
- **Q1.** Matlab-reconstructed undeformed structure
- **Q2.** Natural frequencies and modes of vibration
- **Q3.** Structural damping script
- **Q3.** FRF
- **Q4.1.** DFT
- **Q4.2.** Time history:
  - Method 1 (Ficitious periodic)
    - Script
    - Results
  - Method 2 (Sampled outputs)
    - Script
    - Results
  - Method 3 (Frequency superimposition)
    - Script
    - Results
  - Method 4 (ode45 simulink integration)
    - Script
    - Results
  - 4 methods comparison
  - Different time period comparison
- **Q5.** Modified structure
- **Q5.** Results with modified structure

- Printouts of inp. File:
  - Initial structure
  - Modified structure
- Ode_simulink_extended_printout

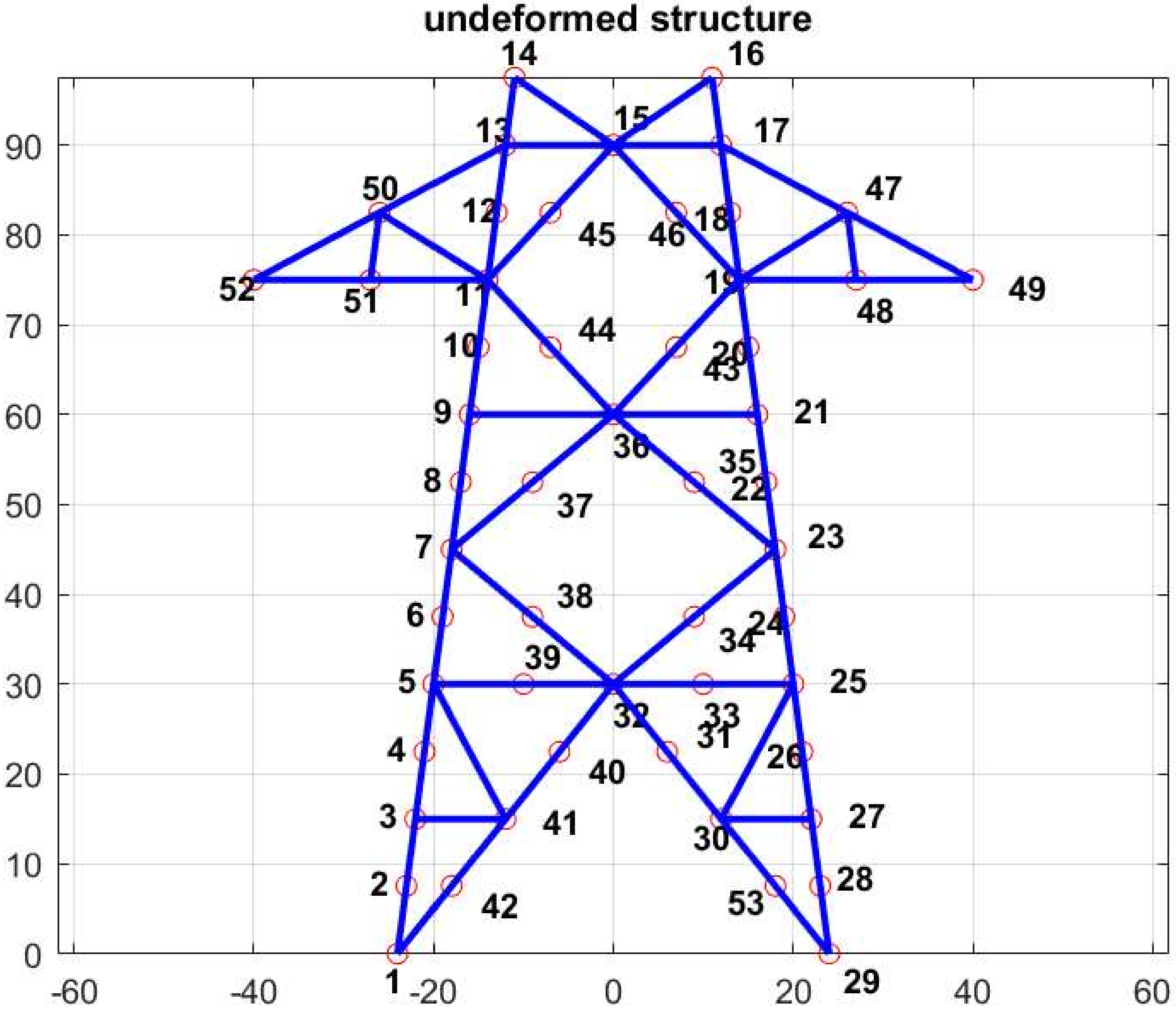
Q1. Geometrical analysis of the structure



53 NODES
76 SECTIONS
f_max=5Hz

Lmax_r=14 m (c=2,07)
Lmax_g=18 m (c=1,94)
Lmax_b= **17.4m (c=1,8)**

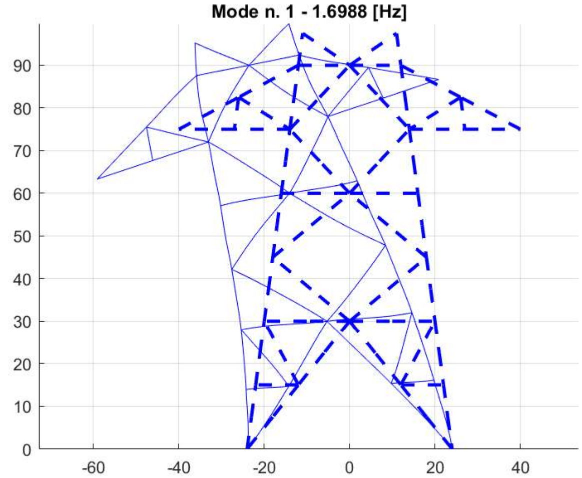
Q1. Matlab-reconstructed undeformed structure



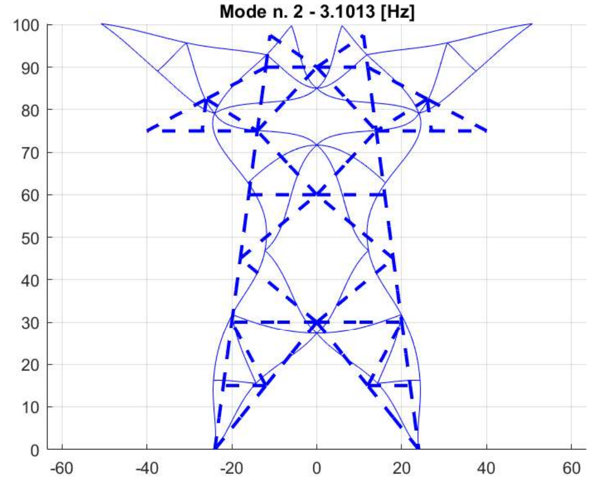

```
Total nodes number 53
Number of d.o.f. 153
Number of beam elements 76
Number of string elements 0
Number of tensile beam elements 0
Number of concentrated masses 0
Number of concentrated springs 0
Total mass [kg] 22104.8715
```

Q2. Natural frequencies and modes of vibration

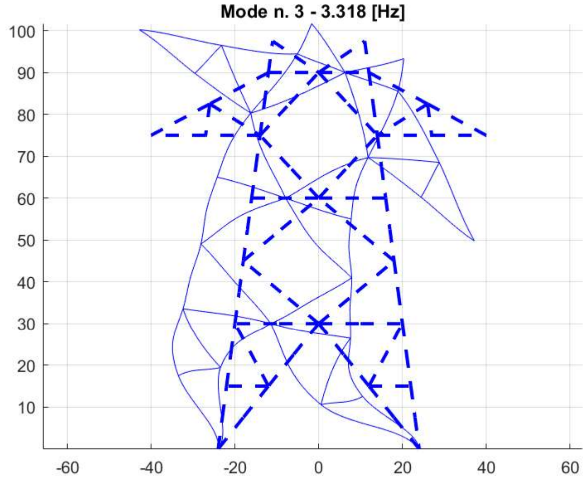
Mode n. 1 - 1.6988 [Hz]



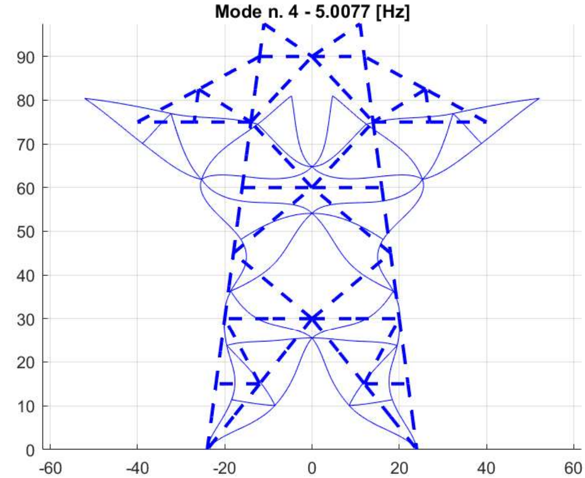
Mode n. 2 - 3.1013 [Hz]



Mode n. 3 - 3.318 [Hz]



Mode n. 4 - 5.0077 [Hz]



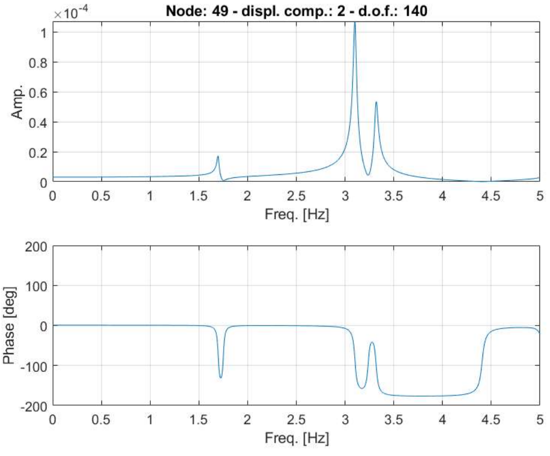
Q3. Structural damping script

Structural damping (initial structure): $\alpha = 0.1193, \beta = 2.5479e - 04$
Structural damping (modified structure for q5): $\alpha = 0.1135, \ \ \beta = 2.6514e - 04$

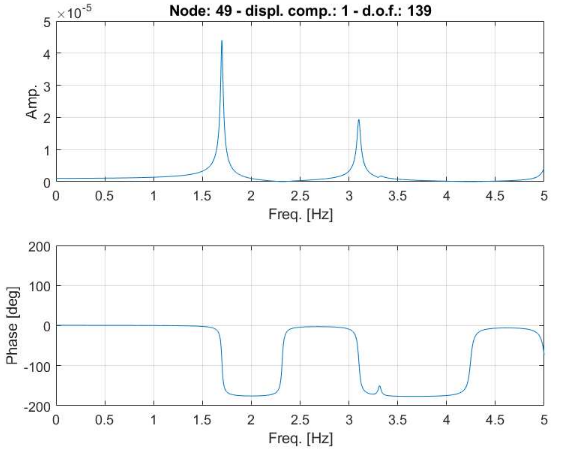
```
clear all;
%close all;

%% Computation of alpha and beta for structural
damping
%q4
ome1=1.698848*2*pi;
ome2=3.101310*2*pi;
ome3=3.317977*2*pi;
% %q5
% ome1=1.633141*2*pi;
% ome2=3.312939*2*pi;
% ome3=4.244738*2*pi;
a1=1/(2*ome1);
b1=ome1/2;
a2=1/(2*ome2);
b2=ome2/2;
a3=1/(2*ome3);
b3=ome3/2;
h1=7e-3;
h2=5e-3;
h3=6e-3;

A=[a1 b1;
    a2 b2;
    a3 b3];
b=[h1; h2; h3];
x=(A'*A)^(-1)*A'*b;
alpha=x(1)
beta=x(2)
```

Q3. FRF

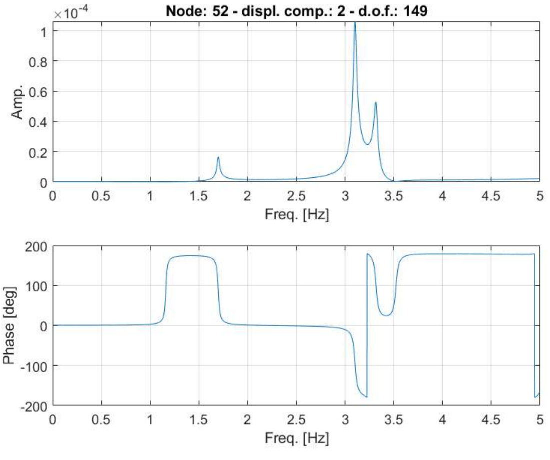
Q3.1. Vertical displacement of point A applying a vertical force at point A



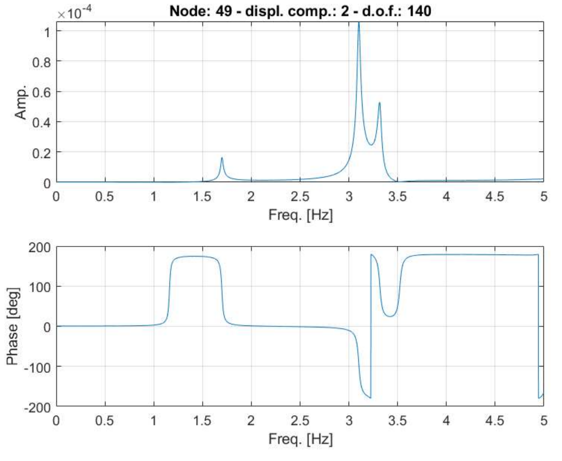
Q3.4. Horizontal displacement of point A applying a horizontal force at point A



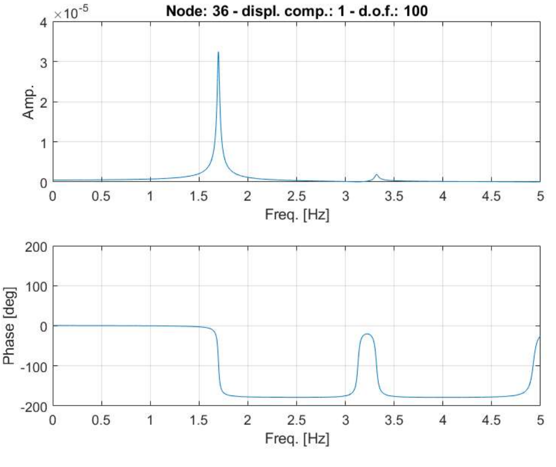
Q3.2. Vertical displacement of point D applying a vertical force at point A



Q3.3. Vertical displacement of point A applying a vertical force at point D

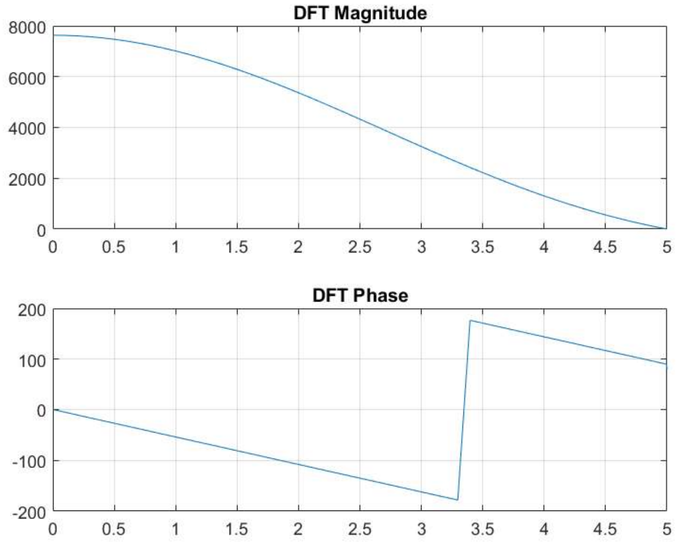


Q3.5. Horizontal displacement of point B applying a horizontal force at point A

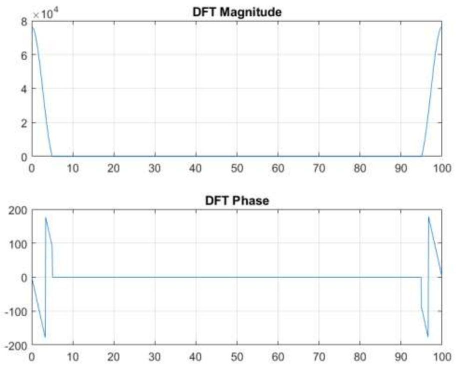


Q4.1. DFT

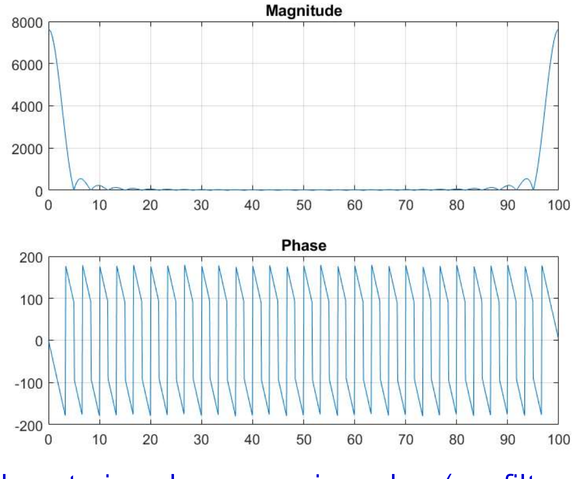
DFT [0-5Hz] (amplitude not normalized)



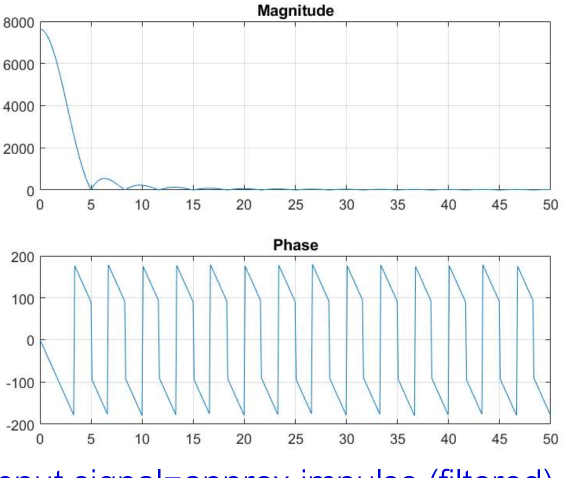
Extended (Filtered >5Hz)



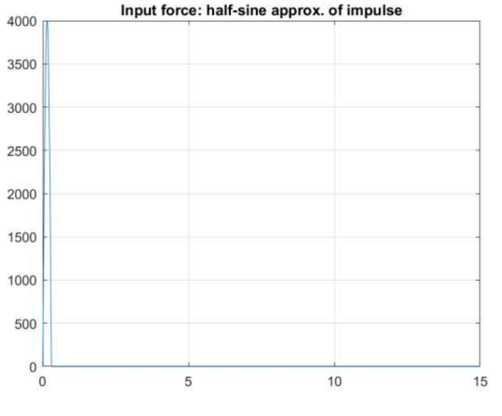
Extended (un-filtered)



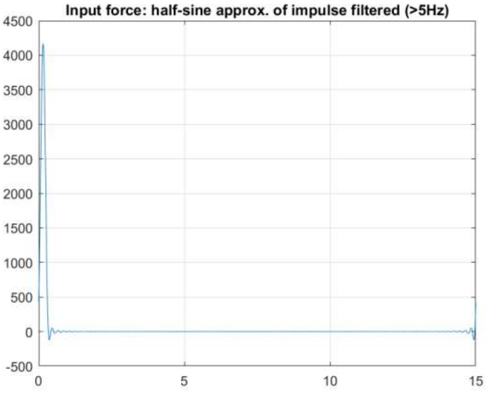
Half-cut (un-filtered)



Input signal=approx.impulse (un-filtered)



Input signal=approx.impulse (filtered)



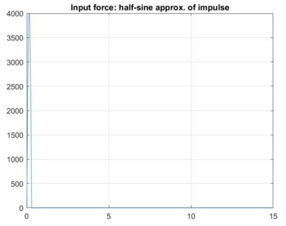
Q4.1. DFT-script

```
%% Impulse (half-sine) definition

fs=1000; % sample frequency
dt=1/fs; % time step
T=15; %in the range [6,24] s
t=0:dt:T;
Fmax=4000;
Ti=2*0.3; % period of the half-sine function
fi=1/Ti; % freq. of the half-sine function
for j=1:length(t)
    if t(j)<=0.3
        imp(j)=Fmax*sin(2*pi*fi*t(j));
    else
        imp(j)=0;
    end

end

% PLOTTING %
figure;plot(t,imp);grid;title('Input force: half-sine approx. of impulse');
%% 4.1. DFT %%

%% DFT of input force=approx.impulse
N = length(imp); % number of samples in the signal being analyzed by the DFT

% define the frequency components of the series
bin_num = 0:N-1;  %freq.expressed in samples
freq = bin_num*fs/N; %freq. expressed in Hz

Fk = fft(imp, N);
fmax=5;

% PLOTTING %
% Showing just till fmax
figure; subplot(2,1,1); plot(freq,abs(Fk));grid; xlim([0 fmax]); title('DFT Magnitude up to 5Hz')
subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; xlim([0 fmax]); title('DFT Phase up to 5Hz')
% % Showing the extended spectrum
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase')
% % Showing just half of the symmetric spectrum (!! Attention will cut also Fk vector in 2 !!)
% cutOff = ceil(N/2); Fk = Fk(1:cutOff); freq = freq(1:cutOff);
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude symmetric half')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase symmetric half')
% pause

%% Removing the elements of vector Fk related to a frequency span higher than 2*pi*fmax

for k=2:N
    if k<N/2+1
        Omk=(k-1)*2*pi/T;
    else
        Omk=(N-k+1)*2*pi/T;
    end
    if Omk>2*pi*fmax
        Fk(k)=0;
    end
end

sign=ifft(Fk);
% PLOTTING %
figure;plot(t,sign);grid;title('Input force: half-sine approx. of impulse filtered (>5Hz)');
```

Q4.2. Time history – Method 1

Approximation to a fictious periodic function

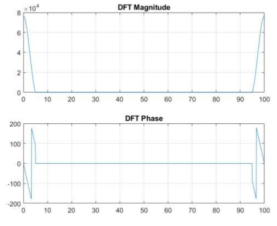
$$f = approx.impulse$$



INPUT

FFT

$$\overline{F_k}$$



Switch to periodic case analysis

$$for\ k = 1,2,\ldots,\frac{N}{2}+1:$$

$$F_0 = c_0 = \frac{1}{N}\overline{F_0}\ \ (for\ matlab - index\ k = 1)$$

$$F_k = 2|c_k| = \frac{2}{N}|\overline{F_k}|\quad \left(for\ k = 2,\ldots.,\frac{N}{2}+1\right)$$

$$\varphi_k = \angle c_k = \angle\overline{F_k}\left(for\ k = 2,\ldots.,\frac{N}{2}+1\right)$$

Fourier Series of the output (subjected to a fictious periodic force)

$$x(t) = X_0 + \sum_{2}^{\frac{N}{2}+1} X_k \cos((k-1)\Omega_0 t + \psi_k)$$

OUTPUT

$$X_0 = G(i0)F_0$$

$$X_k = |G(i\Omega_k)|F_k$$

$$\psi_k = \angle G(i\Omega_k) + \varphi_k \text{ n}$$

$$\Omega_k = (k-1)\Omega_0 = \frac{(k-1)2\pi}{T}$$

$$\begin{cases} x_A, y_A \rightarrow \mathrm{G_A}(\mathrm{i}\Omega_k) = A^{-1} \\ R \rightarrow \mathrm{G_R}(\mathrm{i}\Omega_k) = A_{CF}A^{-1} \end{cases}$$

$$A_{CF} = (-\Omega_K^2[M_{CF}] + i\Omega_k[C_{CF}] + [K_{CF}])$$

$$A = (-\Omega_K^2[M_{FF}] + i\Omega_k[C_{FF}] + [K_{FF}])$$

Being the vector of displacement $x = A^{-1}F$

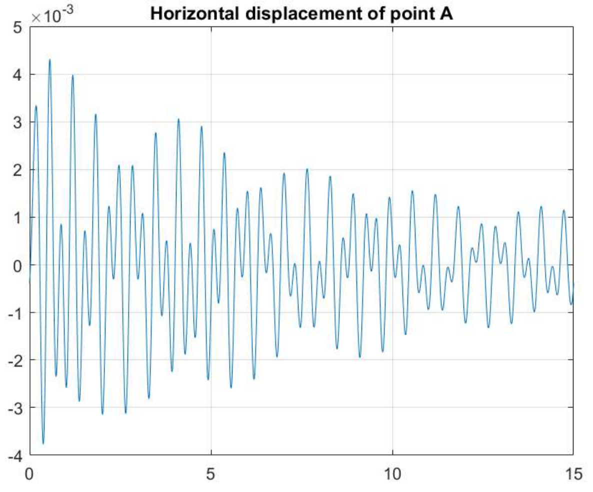
Then $R = \mathrm{G_R}(\mathrm{i}\Omega_k)F = A_{CF}\ x$

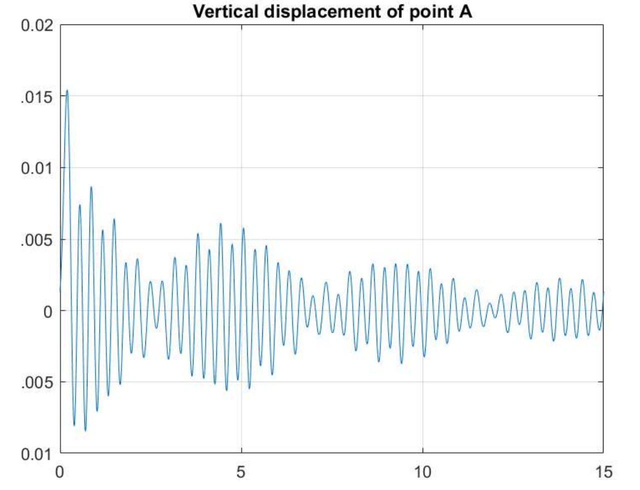
Q4.2. Time history – Method 1
Script

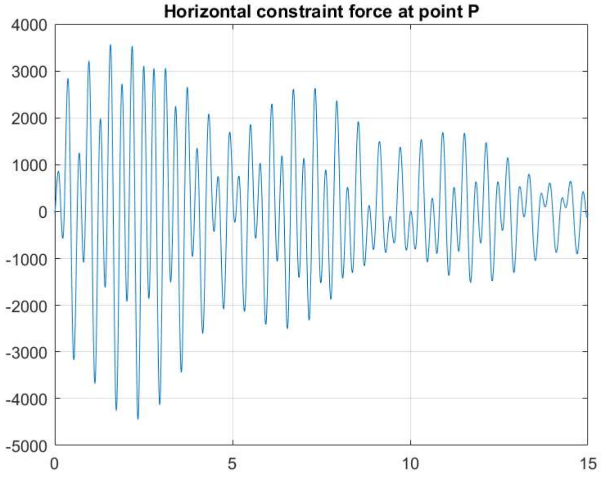
```
clear all;
close all;

%% Matrix partioning
%q1-q4
load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q1\initial_structure_mkr.mat');
%q5
% load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q5\q5_55n_82s_mkr.mat');
%q1-q4
ntot=3*53; %159
ndof=153;
% %q5
% ntot=3*55;
% ndof=159;
MFF=M(1:ndof,1:ndof);
KFF=K(1:ndof,1:ndof);
CFF=R(1:ndof,1:ndof);
MCF=M(ndof+1:ntot,1:ndof);
CCF=R(ndof+1:ntot,1:ndof);
KCF=K(ndof+1:ntot,1:ndof);

%% Impulse (half-sine) definition

fs=1000; % sample frequency
dt=1/fs; % time step
T=15; %in the range [6,24] s
t=0:dt:T;
Fmax=4000;
Ti=2*0.3; % period of the half-sine function
fi=1/Ti; % freq. of the half-sine function
for j=1:length(t)
    if t(j)<=0.3
        imp(j)=Fmax*sin(2*pi*fi*t(j));
    else
        imp(j)=0;
    end

end

% % PLOTTING %
% plot(t,imp);grid
% title('Input force: half-sine approx. of impulse')
%% 4.1. DFT %%

%% DFT of input force=approx.impulse
N = length(imp); % number of samples in the signal being analyzed by the DFT

% define the frequency components of the series
bin_num = 0:N-1;  %freq.expressed in samples
freq = bin_num*fs/N; %freq. expressed in Hz

Fk = fft(imp, N);
fmax=5;

% PLOTTING %
% Showing just till fmax
figure; subplot(2,1,1); plot(freq,abs(Fk));grid; xlim([0 fmax]); title('DFT Magnitude up to 5Hz')
subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; xlim([0 fmax]); title('DFT Phase up to 5Hz')
% % Showing the extended spectrum
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase')
% % Showing just half of the symmetric spectrum (!! Attention will cut also Fk vector in 2 !!)
% cutOff = ceil(N/2); Fk = Fk(1:cutOff); freq = freq(1:cutOff);
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude symmetric half')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase symmetric half')
% pause

%% 4.2 TIME HISTORY %%
%% Removing the elements of vector Fk related to a frequency span higher than 2*pi*fmax

for k=2:N
    if k<N/2+1
        Omk=(k-1)*2*pi/T;
    else
        Omk=(N-k+1)*2*pi/T;
    end
    if Omk>2*pi*fmax
        Fk(k)=0;
    end
end
%% Creating the Fourier coefficient Fkt an phases to use in the third form of the Fourier series
Fkt=zeros(size(Fk));
phik=zeros(size(Fk));
for k=1:N/2+1
    Fkt(k)=2*abs(Fk(k))/N;
    phik(k)=angle(Fk(k)/N);
    if k==1
        Fkt(k)=Fk(k)/N;
        phik(k)=0;
    end
end
```

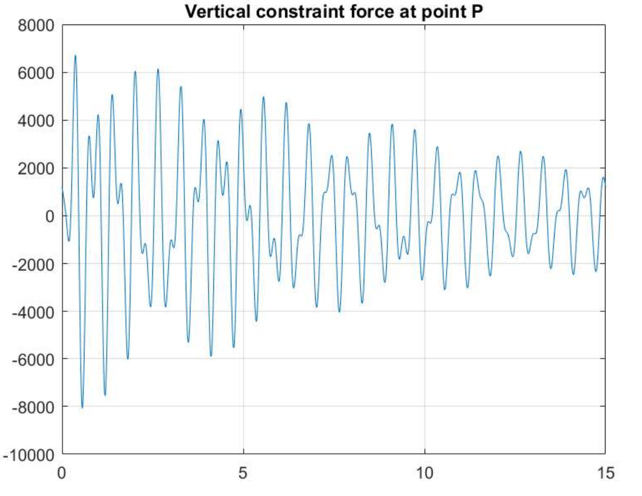
```matlab
%% Creating the Fourier series expressing a fictious periodic input force
dof_ya=idb(49,2); %ya (A=node 49)
dof_xa=idb(49,1); %xa (A=node 49)

% Initializing of time domain vectors for outputs
xa=zeros(1,N);
ya=zeros(1,N);
Rx=zeros(1,N);
Ry=zeros(1,N);
Mp=zeros(1,N);

% Fourier series
for k=1:N/2+1
    ome=(k-1)*2*pi/T;
    Om0=2*pi/T;
%     ck=Fk(k)/N;
    A=-ome^2*MFF+i*ome*CFF+KFF;
    A_cf=-ome^2*MCF+i*ome*CCF+KCF;
    G_a=A^(-1); % FRF matrix for displacements
    G_r=A_cf*A^(-1); % FRF matrix for displacements

    % computing the different components of the sum based on the value of intex k
    if k==1 %x0 computation
        %ome=0 %A=KFF %A_cf=KCF
        G_a=KFF^(-1);
        G_r=KCF*KFF^(-1);
    end
    % Fourier coeff. for each output
    xk_xa=abs(G_a(dof_xa,dof_ya))*Fkt(k);
    xk_ya=abs(G_a(dof_ya,dof_ya))*Fkt(k);
    xk_Rx=abs(G_r(1,dof_ya))*Fkt(k); %looking at idb the #dof of clamp P are in the end but still
before than the ones of clamp Q
    xk_Ry=abs(G_r(2,dof_ya))*Fkt(k);
    xk_Mp=abs(G_r(3,dof_ya))*Fkt(k);
    % Definig the phase contributions
    psik_xa=angle(G_a(dof_xa,dof_ya))+phik(k);
    psik_ya=angle(G_a(dof_ya,dof_ya))+phik(k);
    psik_Rx=angle(G_r(1,dof_ya))+phik(k);
    psik_Ry=angle(G_r(2,dof_ya))+phik(k);
    psik_Mp=angle(G_r(3,dof_ya))+phik(k);

    if 1<k<=N/2+1
%         % in time domain outputs
        xa=xa+xk_xa*cos((k-1)*Om0*t+psik_xa);
        ya=ya+xk_ya*cos((k-1)*Om0*t+psik_ya);
        Rx=Rx+xk_Rx*cos((k-1)*Om0*t+psik_Rx);
        Ry=Ry+xk_Ry*cos((k-1)*Om0*t+psik_Ry);
        Mp=Mp+xk_Mp*cos((k-1)*Om0*t+psik_Mp);
    end

end


% PLOTTING %
%ya
figure; plot(t,ya); grid; title('Vertical displacement of point A')
%xa
figure; plot(t,xa); grid; title('Horizontal displacement of point A')
%Rx
figure; plot(t,Rx); grid; title('Horizontal constraint force at point P')
%Ry
figure; plot(t,Ry); grid; title('Vertical constraint force at point P')
%Mp
figure; plot(t,Mp); grid; title('Constraint moment at point P')
```

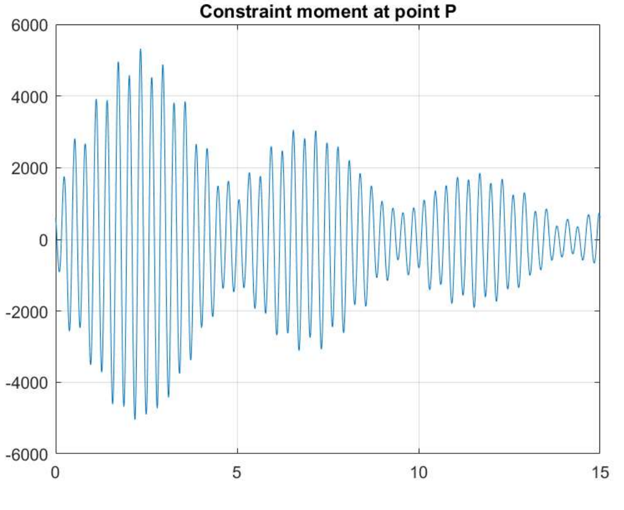
Q4.2. Time history – Method 1
Results







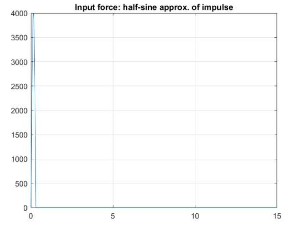




Q4.2. Time history – Method 2

Discrete samples of output

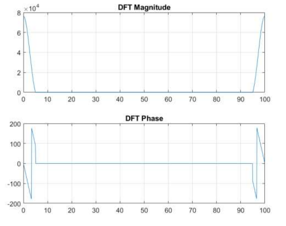
$$f = approx.\,impulse$$



INPUT

FFT

$$\overline{F_k}$$



$$for\ k = 1,2,\ldots,\frac{N}{2}+1:$$

$$\overline{X_0} = G(i0)\overline{F_0}$$

$$\overline{X_k} = G(i\Omega_k)\overline{F_k} \qquad \overline{X}_{N-k} = conj(\overline{X_k})$$

$$\Omega_k = (k-1)\Omega_0 = \frac{(k-1)2\pi}{T}$$

$$\begin{cases} x_A, y_A \rightarrow \mathrm{G_A}(\mathrm{i}\Omega_k) = A^{-1} \\ R \rightarrow \mathrm{G_R}(\mathrm{i}\Omega_k) = A_{CF}A^{-1} \end{cases}$$

$$A_{CF} = (-\Omega_K^2[M_{CF}] + i\Omega_k[C_{CF}] + [K_{CF}])$$

$$A = (-\Omega_K^2[M_{FF}] + i\Omega_k[C_{FF}] + [K_{FF}])$$

Being the vector of displacement $x = A^{-1}F$
Then $R = \mathrm{G}_R(\mathrm{i}\Omega_k)F = A_{CF}\,x$

IFFT

Fourier Series of the sampled output

$$x_j = \frac{1}{N}\sum_{j=0}^{N-1}\bar{X}_k e^{i\frac{2\pi jk}{N}} \quad (j = 1,2,\ldots,N)^*$$

$$x(t) = [x_1,..,x_j,\ldots,x_N]$$

OUTPUT

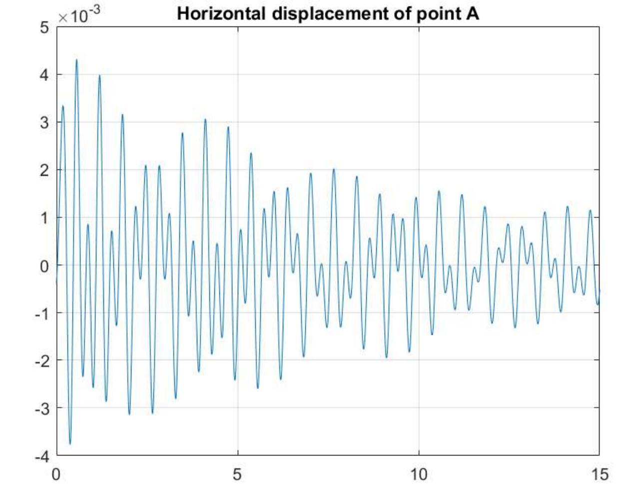
Q4.2. Time history – Method 2

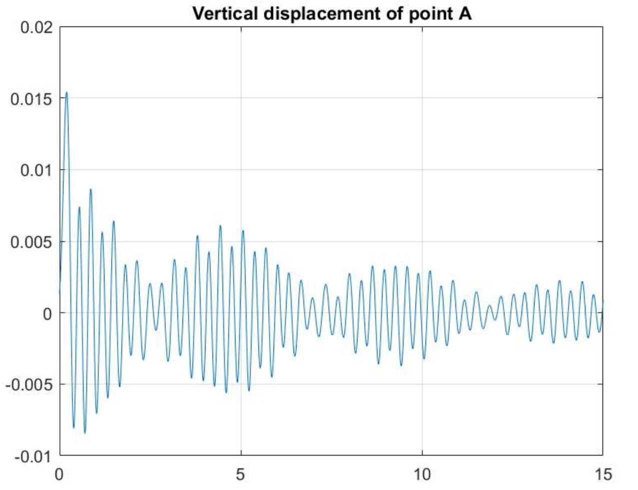
Script

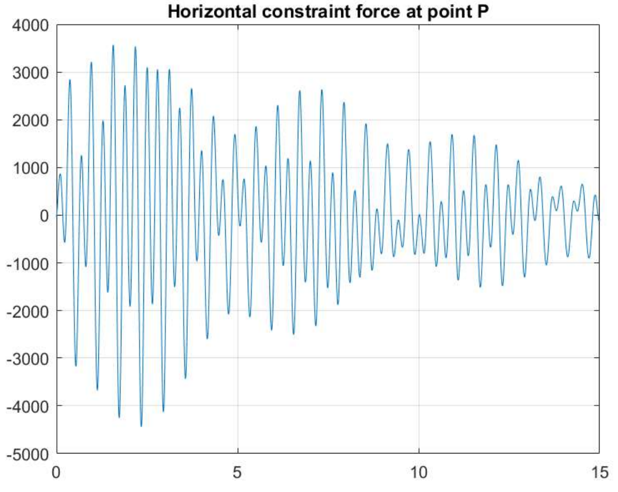
```
clear all;
close all;

%% Matrix partioning
%q1-q4
load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q1\initial_structure_mkr.mat');
%q5
% load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab_labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q5\q5_55n_82s_mkr.mat');
%q1-q4
ntot=3*53; %159
ndof=153;
% %q5
% ntot=3*55;
% ndof=159;
MFF=M(1:ndof,1:ndof);
KFF=K(1:ndof,1:ndof);
CFF=R(1:ndof,1:ndof);
MCF=M(ndof+1:ntot,1:ndof);
CCF=R(ndof+1:ntot,1:ndof);
KCF=K(ndof+1:ntot,1:ndof);

%% Impulse (half-sine) definition

fs=1000; % sample frequency
dt=1/fs; % time step
T=15; %in the range [6,24] s
t=0:dt:T;
Fmax=4000;
Ti=2*0.3; % period of the half-sine function
fi=1/Ti; % freq. of the half-sine function
for j=1:length(t)
    if t(j)<=0.3
        imp(j)=Fmax*sin(2*pi*fi*t(j));
    else
        imp(j)=0;
    end

end

% % PLOTTING %
% figure;plot(t,imp);grid;title('Input force: half-sine approx. of impulse');

%% 4.1. DFT %%

%% DFT of input force=approx.impulse
N = length(imp); % number of samples in the signal being analyzed by the DFT

% define the frequency components of the series
bin_num = 0:N-1;  %freq.expressed in samples
freq = bin_num*fs/N; %freq. expressed in Hz

Fk = fft(imp, N);
fmax=5;

% PLOTTING %
% Showing just till fmax
figure; subplot(2,1,1); plot(freq,abs(Fk));grid; xlim([0 fmax]); title('DFT Magnitude up to 5Hz')
subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; xlim([0 fmax]); title('DFT Phase up to 5Hz')
% % Showing the extended spectrum
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase')
% % Showing just half of the symmetric spectrum (!! Attention will cut also Fk vector in 2 !!)
% cutOff = ceil(N/2); Fk = Fk(1:cutOff); freq = freq(1:cutOff);
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude symmetric half')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase symmetric half')
% pause

%% 4.2 TIME HISTORY %%

%% Removing the elements of vector Fk related to a frequency span higher than 2*pi*fmax

for k=2:N
    if k<N/2+1
        Omk=(k-1)*2*pi/T;
    else
        Omk=(N-k+1)*2*pi/T;
    end
    if Omk>2*pi*fmax
        Fk(k)=0;
    end
end
% sign=ifft(Fk);
% % PLOTTING %
% figure;plot(t,sign);grid;title('Input force: half-sine approx. of impulse filtered (>5Hz)');
```

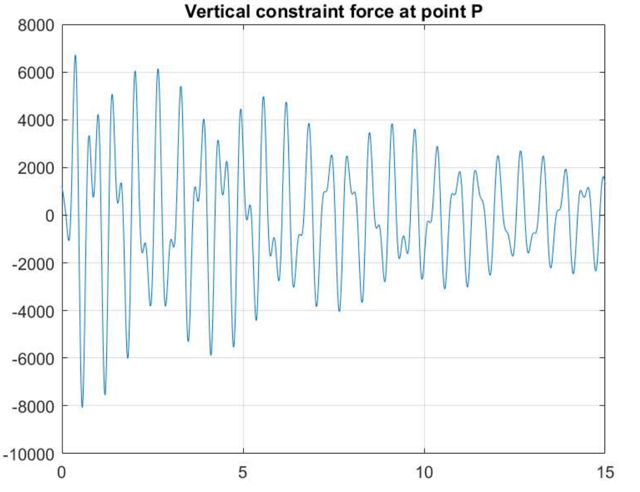
```matlab
%% Creating the Fourier series
dof_ya=idb(49,2); %ya (A=node 49)
dof_xa=idb(49,1); %xa (A=node 49)

% Initializing of time domain vectors for outputs
xa=zeros(1,N);
ya=zeros(1,N);
Rx=zeros(1,N);
Ry=zeros(1,N);
Mp=zeros(1,N);

% Fourier series
for k=1:N
    if k<N/2+1
        ome=(k-1)*2*pi/T;
    elseif k>N/2+1
        ome=-(N-k+1)*2*pi/T;
%     else
%         ome=0;
    end
    Om0=2*pi/T;
    A=-ome^2*MFF+i*ome*CFF+KFF;
    A_cf=-ome^2*MCF+i*ome*CCF+KCF;
    G_a=A^(-1); % FRF matrix for displacements
    G_r=A_cf*A^(-1); % FRF matrix for displacements

    % computing the different components of the sum based on the value of intex k
    if k==1 %x0 computation
        %ome=0 %A=KFF %A_cf=KCF
        G_a=KFF^(-1);
        G_r=KCF*KFF^(-1);
    end
    % Fourier coeff. for each output
    xk_xa(k)=G_a(dof_xa,dof_ya)*Fk(k);
    xk_ya(k)=G_a(dof_ya,dof_ya)*Fk(k);
    xk_Rx(k)=G_r(1,dof_ya)*Fk(k); %looking at idb the #dof of clamp P are in the end but still
before than the ones of clamp Q
    xk_Ry(k)=G_r(2,dof_ya)*Fk(k);
    xk_Mp(k)=G_r(3,dof_ya)*Fk(k);
    % in time domain outputs
    xa=ifft(xk_xa);
    ya=ifft(xk_ya);
    Rx=ifft(xk_Rx);
    Ry=ifft(xk_Ry);
    Mp=ifft(xk_Mp);
end

% PLOTTING %
%ya
figure; plot(t,ya); grid; title('Vertical displacement of point A')
%xa
figure; plot(t,xa); grid; title('Horizontal displacement of point A')
%Rx
figure; plot(t,Rx); grid; title('Horizontal constraint force at point P')
%Ry
figure; plot(t,Ry); grid; title('Vertical constraint force at point P')
%Mp
figure; plot(t,Mp); grid; title('Constraint moment at point P')
```

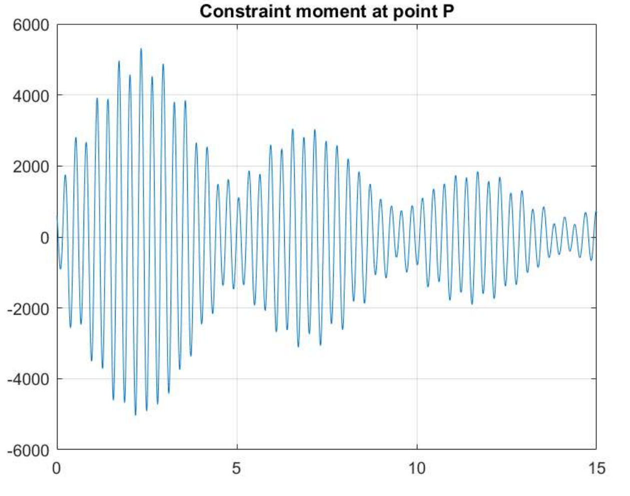
Q4.2. Time history – Method 2
Results





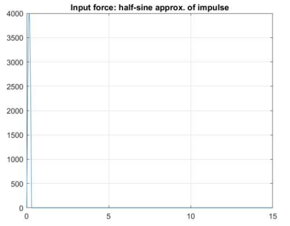






Q4.2. Time history – Method 3
Frequency superimposition

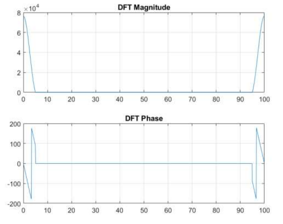
$$f = approx.impulse$$



INPUT

FFT

$$\overline{F_k}$$



In frequency domain

$for\ k = 1,2, \ldots, \frac{N}{2} + 1:$

$F = |F_k| e^{i\angle F_k}, where\ F_k = \frac{\overline{F_k}}{\frac{N}{2}}$

$x = A^{-1}F, with\ A = (-\Omega_K^2 [M_{FF}] + i\Omega_k [C_{FF}] + [K_{FF}])$ → x_{Ak}, y_{Ak} (in freq. domain)

$R = A_{CF}x, with A_{CF} = (-\Omega_K^2 [M_{CF}] + i\Omega_k [C_{CF}] + [K_{CF}])$ → R_{pxk}, R_{pyk}, M_{pk} (in freq. domain)

In time domain:

The generic output y_i can be obtained in time domain like:

$$y_i(t) = \sum_{k=1}^{\frac{N}{2}+1} |y_{ik}| \cos(\Omega_k + \angle y_{ik})$$

OUTPUT

Q4.2. Time history – Method 3

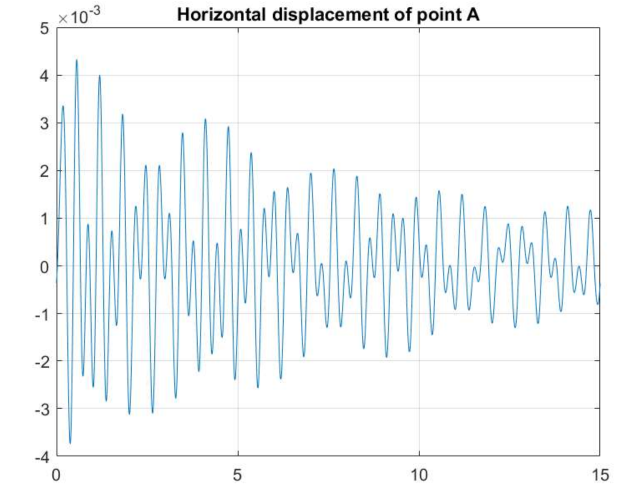
Script

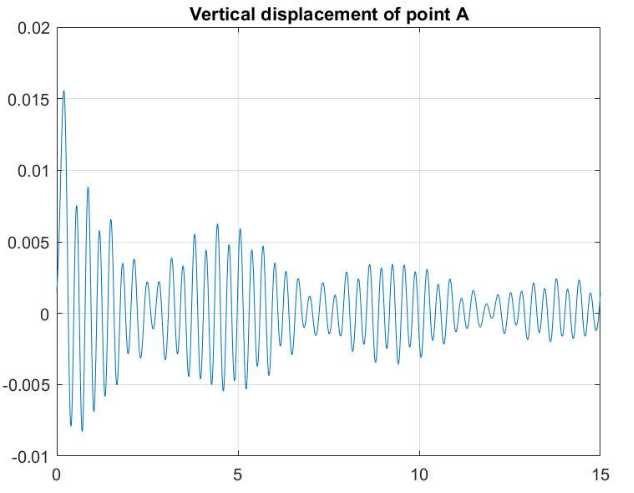
```
clear all;
close all;

%% Matrix partioning
%q1-q4
load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical
system\esercizi\matlab labs\Yearwork-
20211118\FINAL_COMPLETE_SOLUTION\Q1\initial_structure_mkr.mat');
%q5
% load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of
mechanical system\esercizi\matlab labs\Yearwork-
20211118\FINAL_COMPLETE_SOLUTION\Q5\q5_55n_82s_mkr.mat');
%q1-q4
ntot=3*53; %159
ndof=153;
%%q5
% ntot=3*55;
% ndof=159;
MFF=M(1:ndof,1:ndof);
KFF=K(1:ndof,1:ndof);
CFF=R(1:ndof,1:ndof);
MCF=M(ndof+1:ntot,1:ndof);
CCF=R(ndof+1:ntot,1:ndof);
KCF=K(ndof+1:ntot,1:ndof);

%% Impulse (half-sine) definition

fs=1000; % sample frequency
dt=1/fs; % time step
T=15; %in the range [6,24] s
t=0:dt:T;
Fmax=4000;
Ti=2*0.3; % period of the half-sine function
fi=1/Ti; % freq. of the half-sine function
for j=1:length(t)
    if t(j)<=0.3
        imp(j)=Fmax*sin(2*pi*fi*t(j));
    else
        imp(j)=0;
    end

end

% % PLOTTING %
% plot(t,imp);grid
% title('Input force: half-sine approx. of impulse')

%% 4.1. DFT %%

%% DFT of input force=approx.impulse
N = length(imp); % number of samples in the signal being analyzed by the DFT

% define the frequency components of the series
bin_num = 0:N-1;  %freq.expressed in samples
freq = bin_num*fs/N; %freq. expressed in Hz

Fk = fft(imp, N);
fmax=5;

% PLOTTING %
% Showing just till fmax
figure; subplot(2,1,1); plot(freq,abs(Fk));grid; xlim([0 fmax]); title('DFT Magnitude up to 5Hz')
subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; xlim([0 fmax]); title('DFT Phase up to 5Hz')
% % Showing the extended spectrum
figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude')
subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase')
% % Showing just half of the symmetric spectrum (!! Attention will cut also Fk vector in 2 !!)
% cutOff = ceil(N/2); Fk = Fk(1:cutOff); freq = freq(1:cutOff);
% figure; subplot(2,1,1); plot(freq,abs(Fk));grid; title('DFT Magnitude symmetric half')
% subplot(2,1,2); plot(freq,angle(Fk)*180/pi);grid; title('DFT Phase symmetric half')
% pause

%% 4.2 TIME HISTORY %%
%% Removing the elements of vector Fk related to a frequency span higher than 2*pi*fmax and
normalizing the amplitude

for k=1:floor(N/2)+1
        Omk=(k-1)*2*pi/T;

    if Omk>2*pi*fmax
        Fk(k)=0;
    else
        Fk(k)=Fk(k)/(N/2+1);
    end
end
```

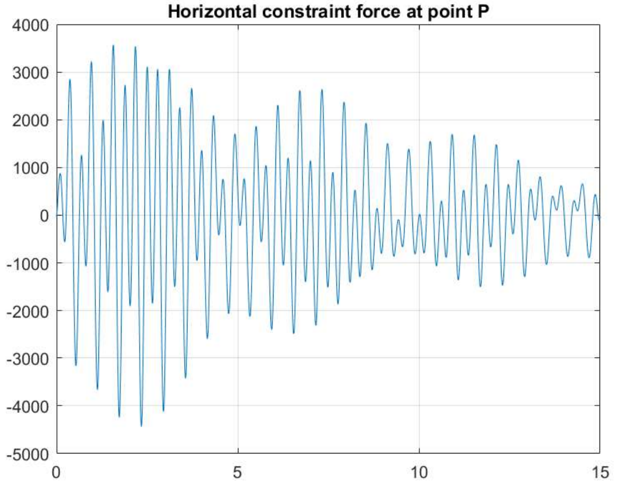
```matlab
%% Creating the Fourier series expressing a fictious periodic input force
dof_ya=idb(49,2); %ya (A=node 49)
dof_xa=idb(49,1); %xa (A=node 49)

% Initializing of time domain vectors for outputs
xa_t=zeros(1,N);
ya_t=zeros(1,N);
Rx_t=zeros(1,N);
Ry_t=zeros(1,N);
Mp_t=zeros(1,N);
F=zeros(ndof,1);
% Fourier series
for k=1:N/2+1
    ome=(k-1)*2*pi/T;
    Om0=2*pi/T;
    F(dof_ya)=abs(Fk(k))*exp(i*angle(Fk(k)));
    A=-ome^2*MFF+i*ome*CFF+KFF;
    x=A\F;
    A_cf=-ome^2*MCF+i*ome*CCF+KCF;
    Rp=A_cf*x;
    % Output in freq. domain
    xa=x(dof_xa);
    ya=x(dof_ya);
    Rx=Rp(1);
    Ry=Rp(2);
    Mp=Rp(3);
    % Output in time domain
    xa_t=xa_t+abs(xa)*cos(ome*t+angle(xa));
    ya_t=ya_t+abs(ya)*cos(ome*t+angle(ya));
    Rx_t=Rx_t+abs(Rx)*cos(ome*t+angle(Rx));
    Ry_t=Ry_t+abs(Ry)*cos(ome*t+angle(Ry));
    Mp_t=Mp_t+abs(Mp)*cos(ome*t+angle(Mp));

end


% PLOTTING %
%ya
figure; plot(t,ya_t); grid; title('Vertical displacement of point A')
%xa
figure; plot(t,xa_t); grid; title('Horizontal displacement of point A')
%Rx
figure; plot(t,Rx_t); grid; title('Horizontal constraint force at point P')
%Ry
figure; plot(t,Ry_t); grid; title('Vertical constraint force at point P')
%Mp
figure; plot(t,Mp_t); grid; title('Constraint moment at point P')
```

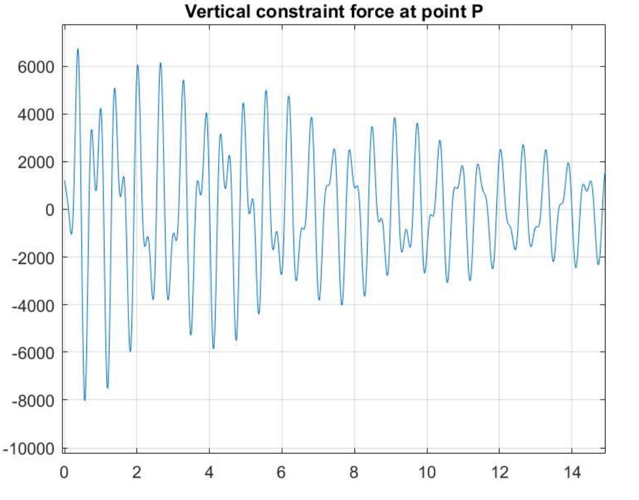
Q4.2. Time history – Method 3

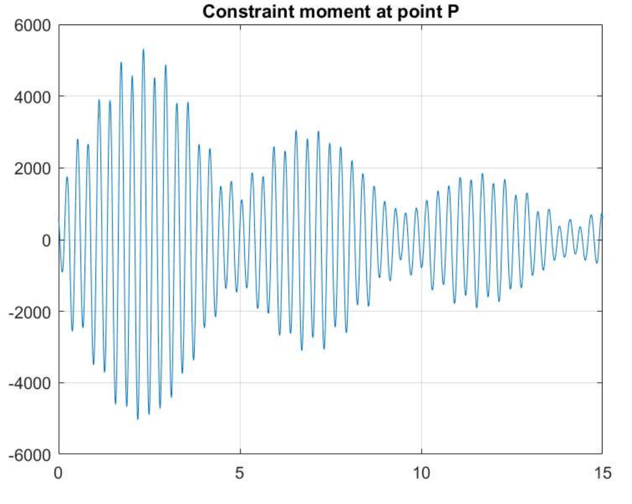
Results





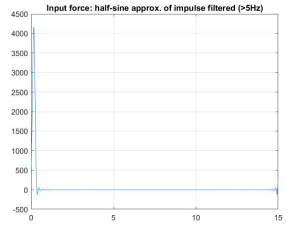






Q4.2. Time history – Method 4
Runge-Kutta 2° order (ode45 simulink)

$$f(t) = approx.impulse$$



INPUT

$$M\ddot{\mathbf{x}}(t)+C\dot{\mathbf{x}}(t)+K\mathbf{x}(t)=\mathbf{f}(t) \qquad \mathbf{x}(0)=\mathbf{x}_0,\ \dot{\mathbf{x}}(0)=\dot{\mathbf{x}}_0$$

For different output here:

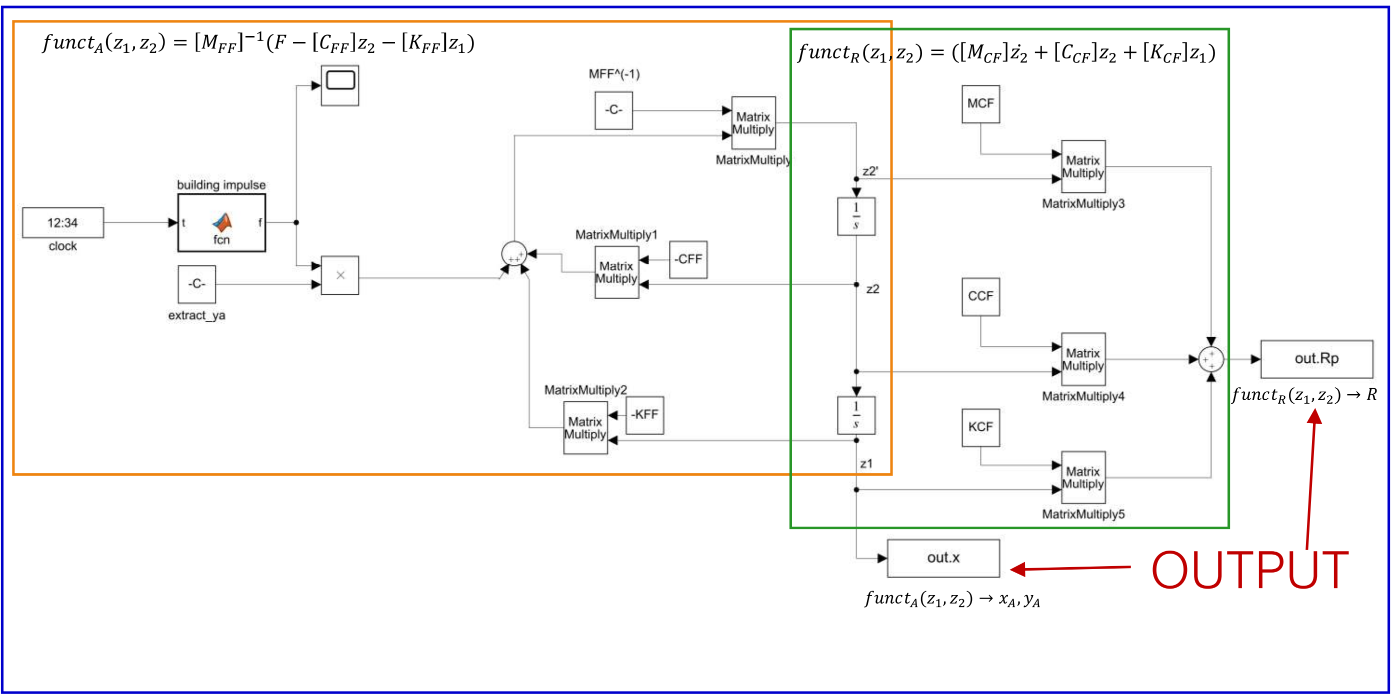
- $M = [M_{FF}], [M_{CF}]$
- $C = [C_{FF}], [C_{CF}]$
- $K = [K_{FF}], [K_{CF}]$

$$\mathbf{z}(t)=\begin{bmatrix}\mathbf{x}(t)\\ \dot{\mathbf{x}}(t)\end{bmatrix} \qquad \mathbf{z}_1=\mathbf{x}(t) \qquad \dot{\mathbf{z}}_1=\mathbf{z}_2$$

$$\mathbf{z}_2=\dot{\mathbf{x}}(t) \qquad \dot{\mathbf{z}}_2=-M^{-1}K\mathbf{z}_1-M^{-1}C\mathbf{z}_2+M^{-1}\mathbf{f}(t)$$

$$\mathbf{z}(0)=\begin{bmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}_0\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$$

SIMULINK MODEL:



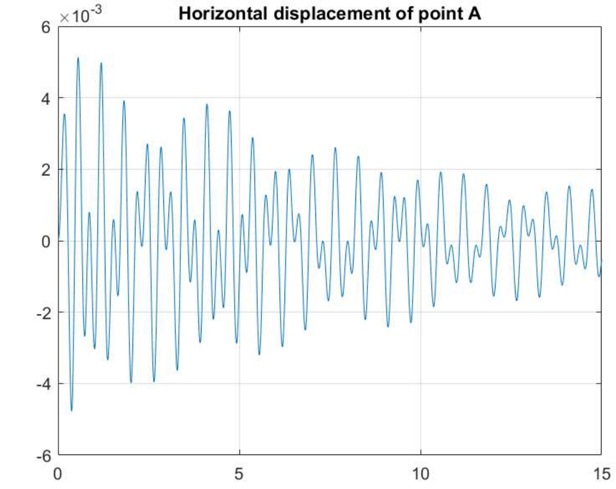
OUTPUT

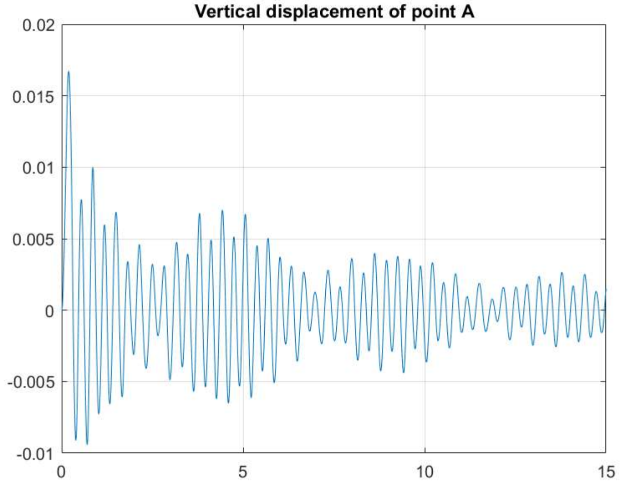
Q4.2. Time history – Method 4
Script

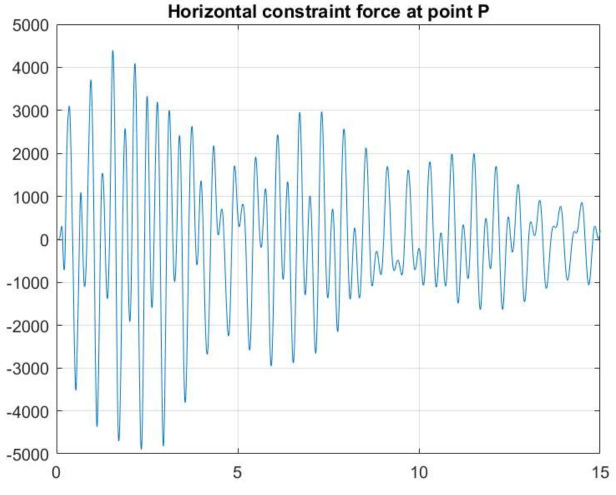
```
function f = fcn(t) %in the simulink function block
%% Impulse (half-sine) definition


Fmax=4000;
Ti=2*0.3; % period of the half-sine function
fi=1/Ti; % freq. of the half-sine function
f=zeros(size(t));
for j=1:length(t)
    if t(j)<=0.3
        f(j)=Fmax*sin(2*pi*fi*t(j));
    else
        f(j)=0;
    end

end
```

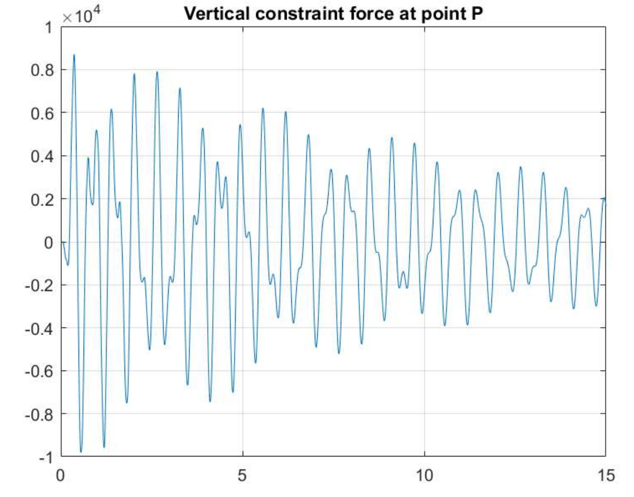
```
%% Script to upload the data and plot the outputs

clear all;
close all;
%% Matrix partioning
%q1-q4
load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab_labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q1\initial_structure_mkr.mat');
%q5
% load('C:\Users\Gianciccetto\Desktop\INGEGNERD\Magistrale\1 anno\1 semestre\dynamics of mechanical system\esercizi\matlab_labs\Yearwork-20211118\FINAL_COMPLETE_SOLUTION\Q5\q5_55n_82s_mkr.mat');
%q1-q4
ntot=3*53; %159
ndof=153;
% %q5
% ntot=3*55;
% ndof=159;
MFF=M(1:ndof,1:ndof);
KFF=K(1:ndof,1:ndof);
CFF=R(1:ndof,1:ndof);
MCF=M(ndof+1:ntot,1:ndof);
CCF=R(ndof+1:ntot,1:ndof);
KCF=K(ndof+1:ntot,1:ndof);

dof_ya=idb(49,2); %ya (A=node 49)
dof_xa=idb(49,1); %xa (A=node 49)

extract_ya=zeros(ndof,1); extract_ya(dof_ya)=1;%vector that defines the position at which the approx. impulse force is applied

sim('q4_ode');
% Output definition

ya=ans.x(:,dof_ya);
xa=ans.x(:,dof_xa);
Rx=ans.Rp(:,1);
Ry=ans.Rp(:,2);
Mp=ans.Rp(:,3);

% PLOTTING %
t=ans.tout;
%ya
figure; plot(t,ya); grid; title('Vertical displacement of point A')
%xa
figure; plot(t,xa); grid; title('Horizontal displacement of point A')
%Rx
figure; plot(t,Rx); grid; title('Horizontal constraint force at point P')
%Ry
figure; plot(t,Ry); grid; title('Vertical constraint force at point P')
%Mp
figure; plot(t,Mp); grid; title('Constraint moment at point P')
```

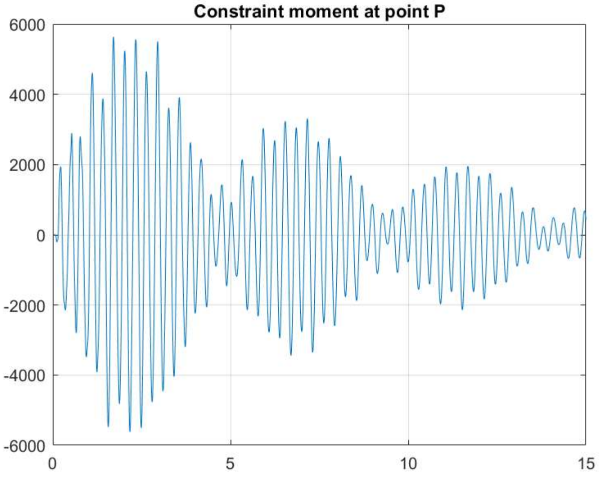
Q4.2. Time history – Method 4
Results





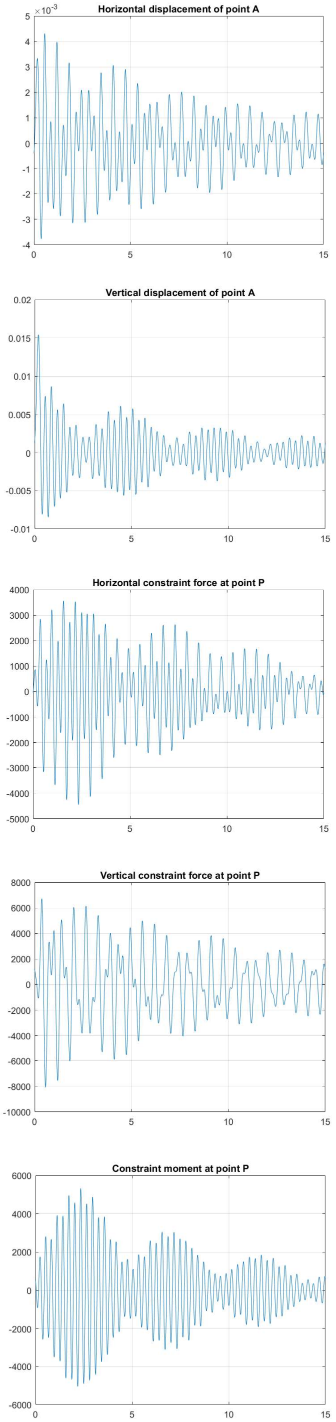




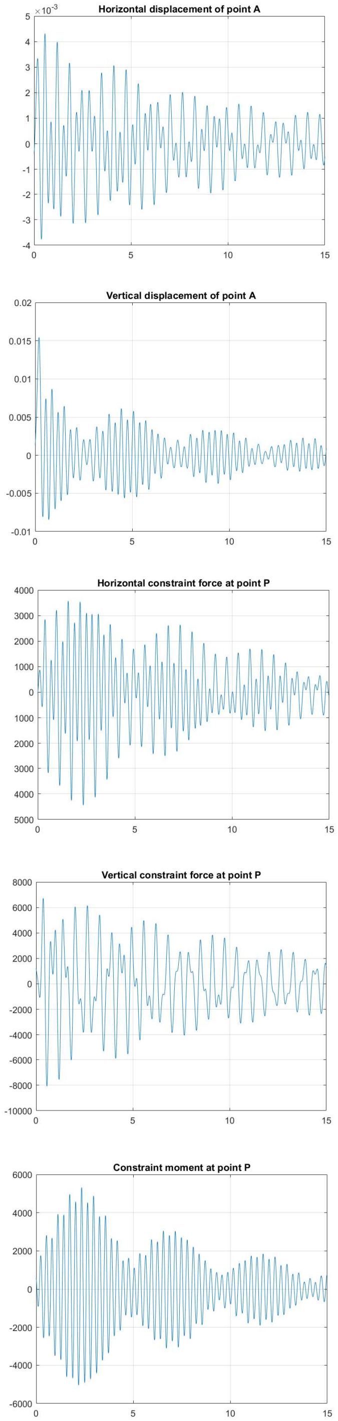


Q4.2. 4 methods comparison (T=15 s)

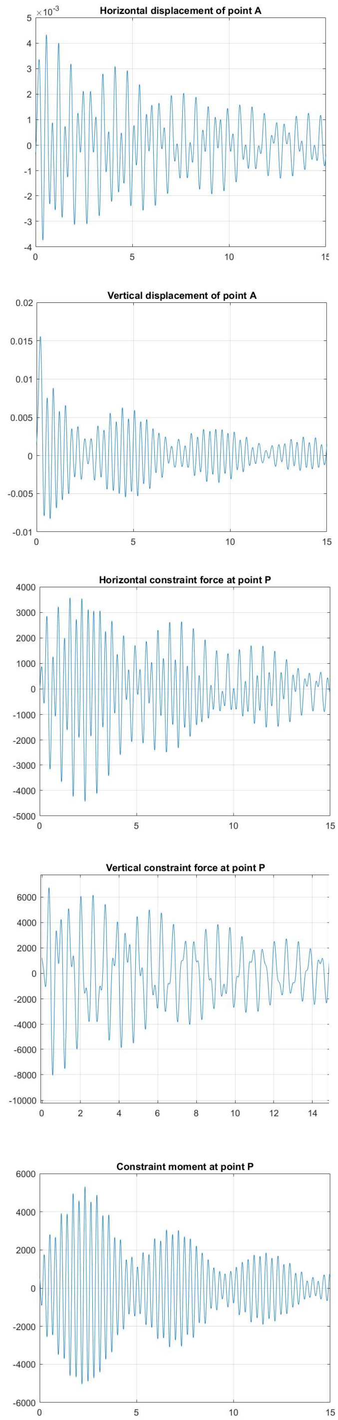
METHOD 1



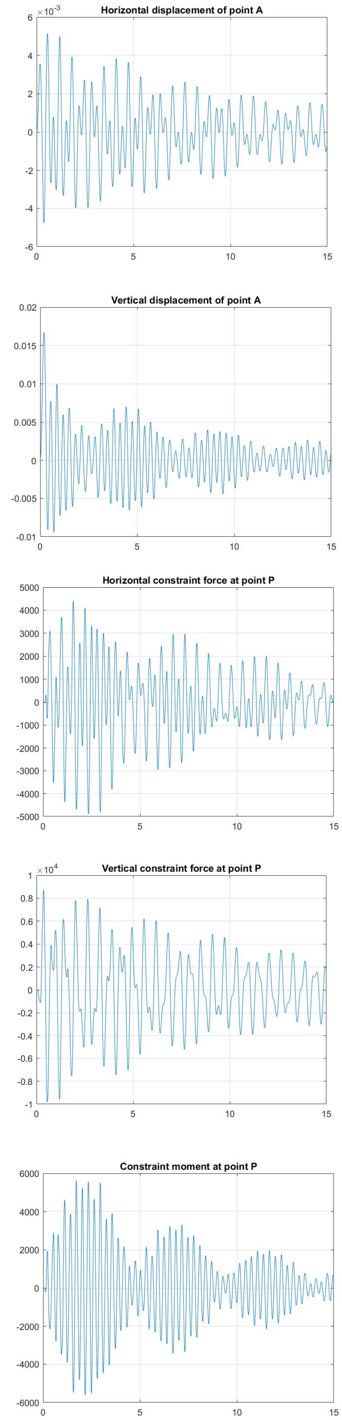
METHOD 2



METHOD 3



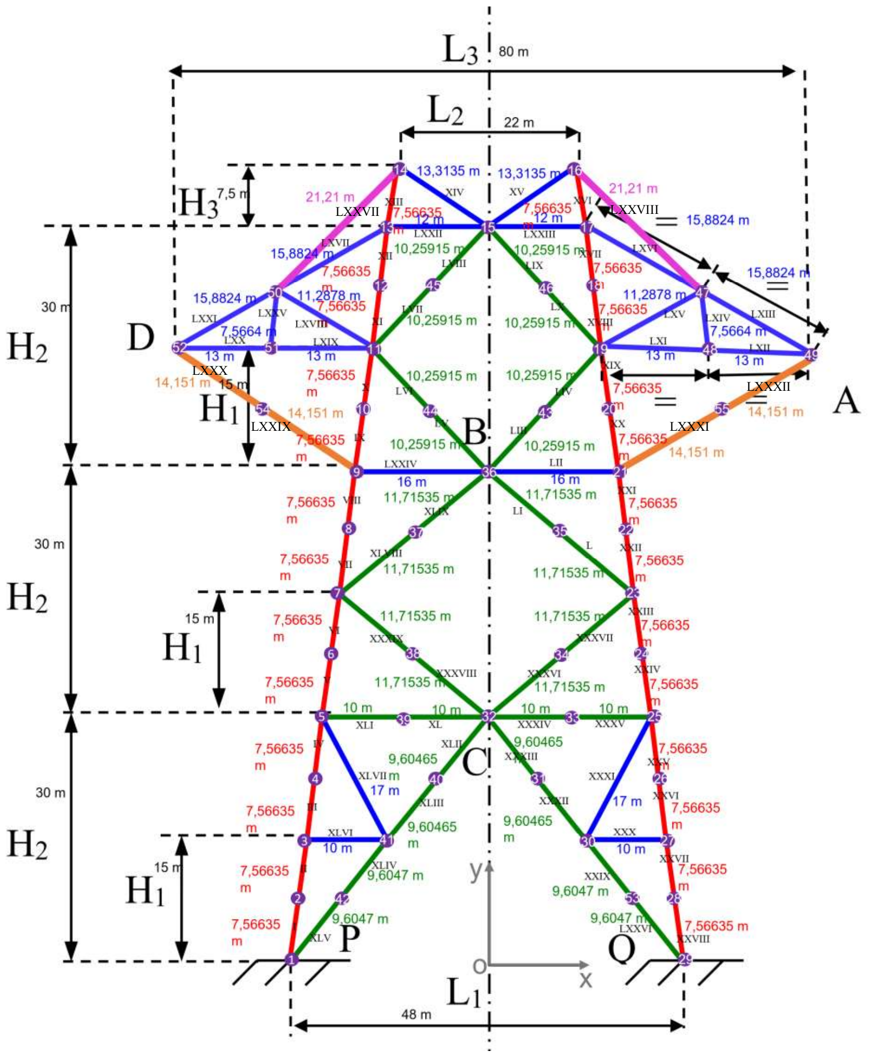
METHOD 4



Q4.2. Different time period comparison (Method 1)

T=12s T=18s T=24s

Q5. Modified structure



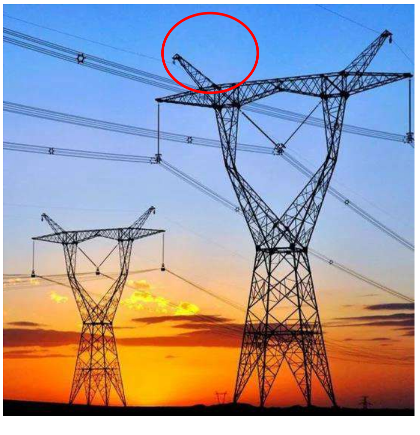
6 more sections (2 fuchsia, 4 orange) and 2 additional nodes (54,55)

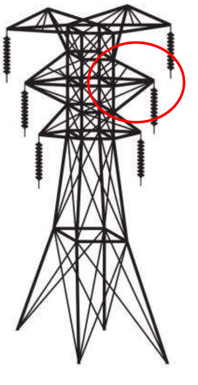
55 NODES
82 SECTIONS
f_max=5Hz

Lmax_r=14 m (c=2,07)
Lmax_g=18 m (c=1,94)
Lmax_b=17 m (c=1,88)
Lmax_f=21.25 m (c=1.7)
Lmax_o=19 m (c=1,99)

Idea (main goal feasibility in real life and cost saving):

- I used for the 6 additional beams with m_o=11.5 Kg/m, EA_o=4 e8N, EI_o=6 e7Nm2, m_f=10 Kg/m, EA_f=4 e8N, EI_f=6 e7Nm2
- I used a simmetric shape for the cross arms already existing in real trasmission towers / technical models

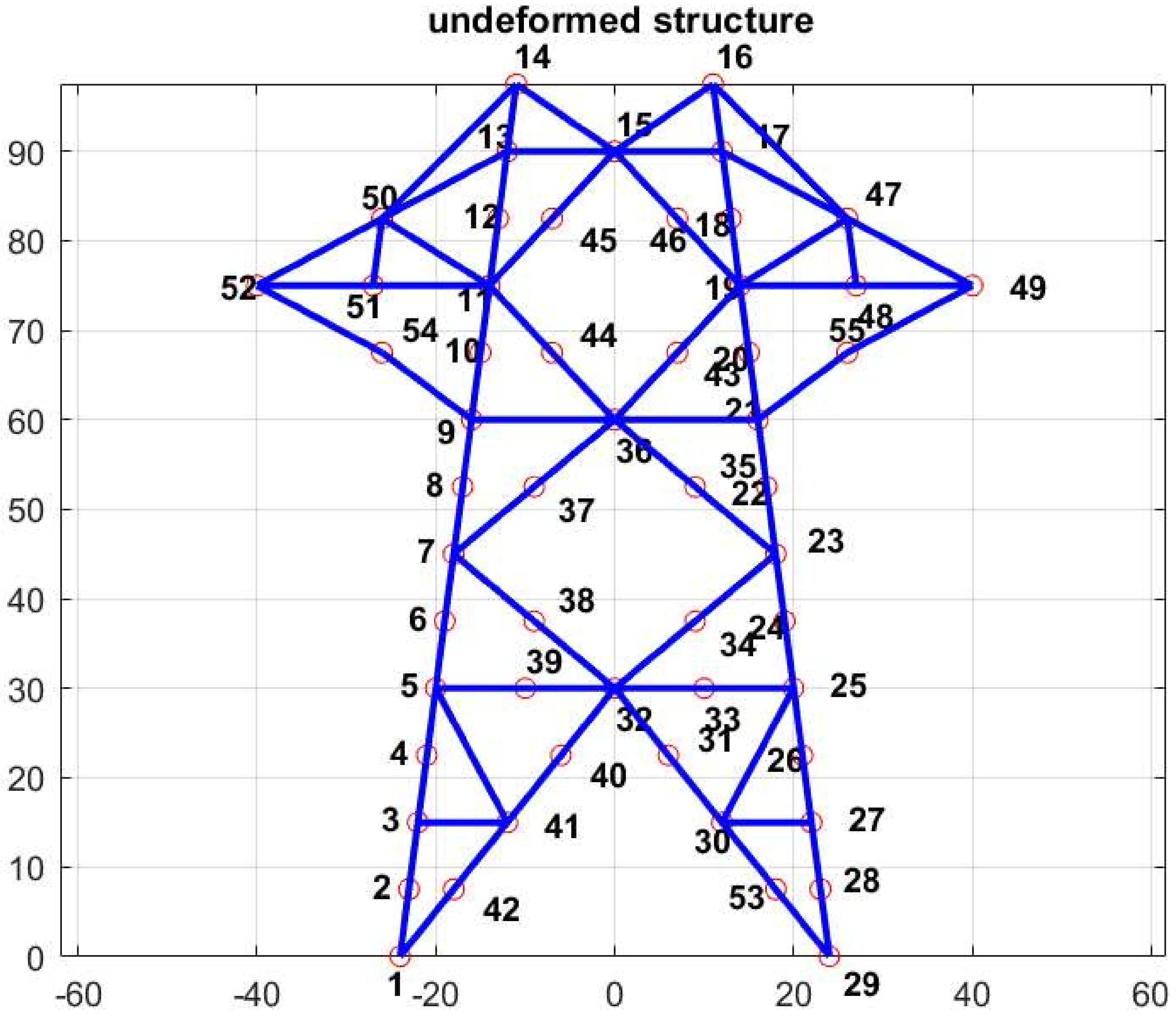




- I kept the total mass in the +5% range of increment (22104.8715→23181.9303= **+4.64%**)

Q5. Modified structure

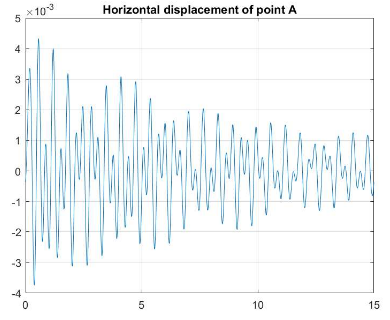
Undeformed structure:

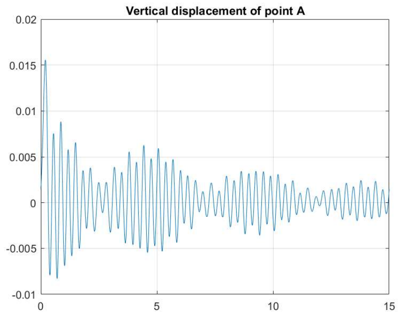


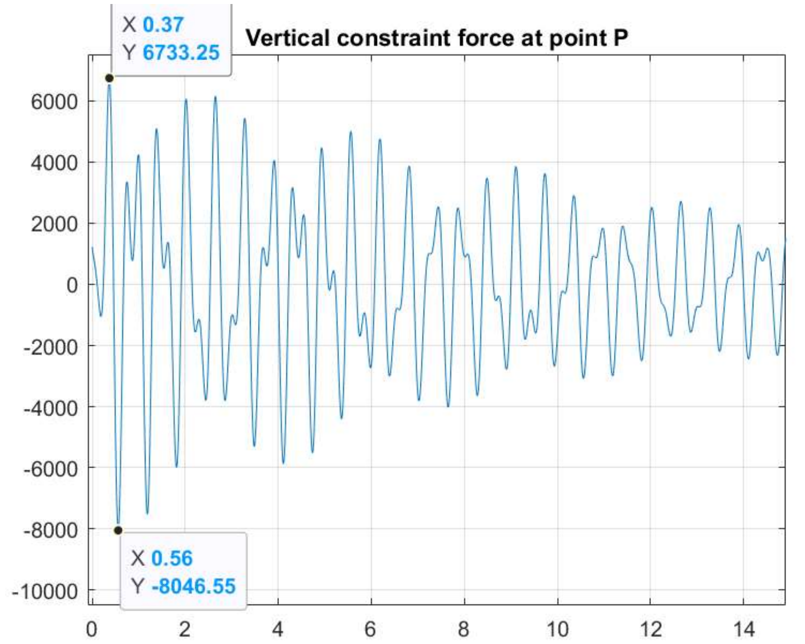

```
Total nodes number 55
Number of d.o.f. 159
Number of beam elements 82
Number of string elements 0
Number of tensile beam elements 0
Number of concentrated masses 0
Number of concentrated springs 0
Total mass [kg] 23181.9303
```

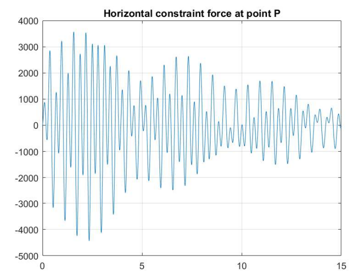
Q5. Initial structure vs. Modified structure(T=15 s)

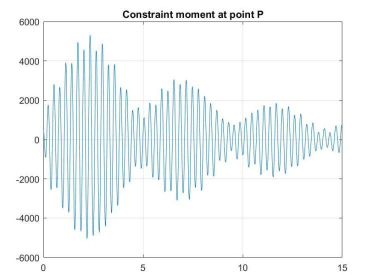
INITIAL STRUCTURE











MODIFIED STRUCTURE

~-33.4% wrt the initial amplitude