Capitolo 23
Vibrazioni meccaniche e sistemi vibranti
(Alessandro Fasana, Stefano Marchesiello)

I file che seguono contengono script Matlab® relativi alla risposta libera e forzata di sistemi con uno o molti gradi di libertà. Una breve descrizione è riportata per ciascuno di essi.

SDOF_FREE
Risposta libera di un sistema con un grado di libertà, sovra, sotto e criticamente smorzato (sez. 23.2 nel testo). L'utente può variare le condizioni iniziali, la pulsazione naturale e il fattore di smorzamento per osservare la loro influenza sulla risposta nel tempo.

SDOF_FRF
Risposta alla forzante armonica (FRF) di un sistema con un grado di libertà, sovra, sotto e criticamente smorzato (sez. 23.3.2). Si evidenzia l'effetto del fattore di smorzamento sulle curve di modulo e fase della FRF.

TRANSMISSIBILITY
Trasmissibilità, definita come rapporto tra la forza sul vincolo e la forzante esterna, di un sistema con un grado di libertà, sovra, sotto e criticamente smorzato (par. 23.4). Si evidenzia l'effetto del fattore di smorzamento sulle curve di modulo e fase della FRF.

TRANSMISSIBILITY_ECCENTRICAL
Trasmissibilità, in presenza di forzante dovuta alla rotazione di masse eccentriche, di un sistema con un grado di libertà, sovra, sotto e criticamente smorzato (par. 23.4). Si evidenzia l'effetto del fattore di smorzamento sulle curve di modulo e fase della FRF.

NDOF_FREE
Risposta libera di un sistema con molti gradi di libertà (par. 23.9), in funzione delle condizioni iniziali. L'utente può inserire le matrici di massa e rigidezza, scelto il numero di gradi di libertà, oppure leggere un file che le contenga: è fornito il file DO3.mat, che contiene le matrici **M** e **K** relative all'esempio 23.10.
Lo smorzamento viscoso è proporzionale e sono pertanto richiesti i coefficienti α e β per definire la matrice **C** dell'eq. (23.9.1).
Sono visualizzati gli andamenti nel tempo delle risposte in coordinate modali η (eq. (23.9.5)) e degli spostamenti **x** (eq. (23.9.2)).

NDOF_FRF
Risposta alla forzante armonica (FRF) di un sistema con molti gradi di libertà (par. 23.10). L'utente può inserire le matrici di massa e rigidezza, scelto il numero di gradi di libertà, oppure leggere un file che le contenga: è fornito il file DO3.mat, che contiene le matrici **M** e **K** relative all'esempio 23.10.
Lo smorzamento viscoso è proporzionale e sono pertanto richiesti i coefficienti α e β per definire la matrice **C** dell'eq. (23.9.1).
È possibile scegliere, per tracciare la FRF secondo l'eq. (23.10.4), la posizione del punto di ingresso *k* e di uscita *j*.

DOF3.mat
File contenente le matrici di massa **M** e di rigidezza **K** relative all'esempio 23.10.

SD0F_FREE

```
Free response of a m-c-k system from non zero initial conditions
m-c-k parameters are defined by the user together with initial
displacement and velocity
Section 23.2
```

```
%
clear all
close all hidden
set(0,'DefaultAxesFontSize',18)

x0=1; %Initial displacement
v0=50; %Initial velocity
wn=30; % Natural frequency
T=2*pi/wn; % Period
fs=100*(wn/2/pi); % Sampling frequency
dt = 1/fs; % Sampling period
t=[0:dt:10*T]; % Time axis

% zeta=1
  x=(x0 + v0*t + x0*wn*t).*exp(-wn*t);
  p1=plot(t,x);
  set(p1,'Linestyle','-', 'LineWidth',1,'Color','b');
  hold on
  % zeta<1
  zeta=0.05;
  wd = wn*sqrt(1-zeta^2);
  pot = -zeta*wn;
  esp = exp(pot*t);
  x = (x0*cos(wd*t) + (v0+zeta*wn*x0)/wd*sin(wd*t)).*esp;
  A = sqrt(x0^2+((v0+zeta*wn*x0)/wd)^2);
  p2=plot(t,x);
  set(p2,'Linestyle','-', 'LineWidth',1.0,'Color','k');
  hold on
  % zeta>1
  zeta=2;
  rad = sqrt(zeta^2 - 1);
  A = (x0*wn*(zeta + rad) + v0)/(2*wn*rad);
  B = x0 - A;
  ws = wn*rad;
  pot = ws*t;
  x = (A*(exp(pot)) + B*(exp(-pot))).*exp(-zeta*wn*t);
  p3=plot(t,x);
  set(p3,'Linestyle','--', 'LineWidth',1,'Color','r');

  Tmax=t(end);
  set(gca,'Xlim',[0 Tmax])
  xlabel('Time (s)')
  ylabel('Free response (m)')
  legend('\zeta=1','\zeta<1','\zeta>1')
  grid on
  zoom on
```

SDOF_FRF

```
FRF of a SDOF system with viscous damping
Eq. (23.3.7)

%
% Plot are given for various damping factors
clear all
close all hidden
set(0,'DefaultAxesFontSize',18)

colours='bgrcmykbgrcmykbgrcmyk';cont=0;
zetas=[0.05 0.1 0.2 0.4 sqrt(2)/2 1.0 2 ];
for zeta=zetas
%for zeta=[0.05 0.35 0.65 0.95 1.25 1.55]
```

```
cont=cont+1;
nc=4000;    % Number of points
rf=50;            %Max ratio omega/omega_n
dr=rf/nc;   % Frequency resolution
r=linspace(0,5,nc);

Reale=(1-r.^2)./((1-r.^2).^2 + (2*zeta*r).^2);
Immag=-2*zeta*r./((1-r.^2).^2 + (2*zeta*r).^2);
FRF=Reale+i*Immag;
y=(abs(FRF));
fi=-atan(Immag./Reale);
for k=1:nc
  if fi(k)<0
    fi(k)=pi+fi(k);
  end
end
%Fase=180*fi/pi;
Fase=fi;

figure(1)
plot(r,y,colours(cont))
hold on

figure(2)
plot(r,-Fase,colours(cont))
hold on

end
if length(zetas)==7
    figure(1),legend(['\zeta=' num2str(zetas(1),2)],['\zeta='
num2str(zetas(2),2)],...
        ['\zeta=' num2str(zetas(3),2)],['\zeta='
num2str(zetas(4),2)],...
        ['\zeta=' num2str(zetas(5),2)],['\zeta='
num2str(zetas(6),2)],['\zeta=' num2str(zetas(7),2)]);
end
figure(1)
x1=xlabel('\Omega/\omega_n');;
y1=ylabel('A/(f_o/k)');
set(gca,'YLim',[0 10])
set(gca,'XTick',[0 1 2 3 4 5]);
set(gca,'YTick',[0 2 4 6 8 10]);
grid on

figure(2)
x1=xlabel('\Omega/\omega_n');
y1=ylabel('Phase (rad)');
set(gca,'XTick',[0 1 2 3 4 5]);
grid on
```

TRANSMISSIBILITY

Eq. (23.4.1)

```
%
clear all
close all hidden
set(0,'DefaultAxesFontSize',18)

nc=4000;    % Points
rf=50;            % Max ratio omega/omega_n
```

```
dr=rf/nc;   % Frequency resolution
r=linspace(10^-1,10^1,nc);

colours='bgrcmykbgrcmykbgrcmyk';cont=0;
zetas=[0.05 0.1 0.2 0.4 sqrt(2)/2 1.0 2];
for zeta=zetas
  cont=cont+1;
  num=1+i*(2*zeta*r);
  den=(1-r.^2) + i*(2*zeta*r);
  REC=num./den;
  y=20*log10(abs(REC));

  figure(1)
  semilogx(r,y,colours(cont))
  hold on
end
figure(1)
xl=xlabel('\Omega/\omega_n');
yl=ylabel('|T(\Omega)|_d_B');
grid on
set(gca,'YLim',[-40 20])
set(gca,'XTick',[0.1 1 10]);
set(gca,'XTickLabel',[0.1 1 10]);
set(gca,'YTick',[-40 -20 0 20]);
if length(zetas)==7
    figure(1),legend(['\zeta=' num2str(zetas(1),2)],['\zeta='
num2str(zetas(2),2)],...
        ['\zeta=' num2str(zetas(3),2)],['\zeta='
num2str(zetas(4),2)],...
        ['\zeta=' num2str(zetas(5),2)],['\zeta='
num2str(zetas(6),2)],['\zeta=' num2str(zetas(7),2)]);
end
```

TRANSMISSIBILITY_ECCENTRICAL

Rotating eccentrical mass

Eq. (23.4.2)

```
%
clear all
close all hidden
set(0,'DefaultAxesFontSize',18)

nc=4000;    % Points
rf=50;           % Max ratio omega/omega_n
dr=rf/nc;   % Frequency resolution
r=linspace(10^-1,10^1,nc);

colours='bgrcmykbgrcmykbgrcmyk';cont=0;
zetas=[0.05 0.1 0.2 0.4 sqrt(2)/2 1.0 2];
for zeta=zetas
    cont=cont+1;
    num=(r.^2).*(1+i*(2*zeta*r));
    den=(1-r.^2)+i*(2*zeta*r);
    FRF=num./den;
    y=20*log10(abs(FRF));
    %y=(abs(FRF));
    fi=-atan(imag(FRF)./real(FRF));

    for k=1:nc
        if fi(k)<0
```

```
            fi(k)=pi+fi(k);
        end
    end
    %  Fase=180*fi/pi;
    Fase=fi;

    figure(1)
    semilogx(r,y,colours(cont))
    hold on

    figure(2)
    semilogx(r,-Fase,colours(cont))
    hold on

    figure(3)
    semilogx(r,abs(FRF),colours(cont))
    hold on

end

figure(1)
t1=xlabel('\Omega/\omega_n');
t1=ylabel('|T(\Omega)|_{dB}');
grid on
set(gca,'YLim',[-40 20])
set(gca,'XTick',[0.1 1 10]);
set(gca,'XTickLabel',[0.1 1 10]);
set(gca,'YTick',[-40 -20 0 20  ]);
if length(zetas)==7
    figure(1),legend(['\zeta=' num2str(zetas(1),2)],['\zeta='
num2str(zetas(2),2)],...
        ['\zeta=' num2str(zetas(3),2)],['\zeta='
num2str(zetas(4),2)],...
        ['\zeta=' num2str(zetas(5),2)],['\zeta='
num2str(zetas(6),2)],['\zeta=' num2str(zetas(7),2)]);
    figure(3),legend(['\zeta=' num2str(zetas(1),2)],['\zeta='
num2str(zetas(2),2)],...
        ['\zeta=' num2str(zetas(3),2)],['\zeta='
num2str(zetas(4),2)],...
        ['\zeta=' num2str(zetas(5),2)],['\zeta='
num2str(zetas(6),2)],['\zeta=' num2str(zetas(7),2)]);
end
figure(2)
xlabel('\Omega/\omega_n');
ylabel('Phase (rad)');
grid on
figure(3)
t1=xlabel('\Omega/\omega_n');
t1=ylabel('|T(\Omega)|');
grid on
%set(gca,'YLim',[-40 20])
% set(gca,'XTick',[0.1 1 10]);
% set(gca,'XTickLabel',[0.1 1 10]);
% set(gca,'YTick',[-40 -20 0 20  ]);
```

NDOF_FREE

Free response of a "N" degrees of freedom lumped system
with proportional viscous damping

File DOF3.mat contains mass and stiffness matrices as in the

```
Example 23.10

%

clear all;
close all hidden;
set(0,'DefaultAxesFontSize',18)

disp(' ')
disp(' Type a valid file name [e.g. DOF3]')
nome=input(' or press [return] to enter matrices: ','s');
vuoto=isempty(double(nome));

if vuoto==1
    n = input(' Number of dofs: ');
    n = abs(n);

    % Mass matrix is typed (upper triangular part)
    disp(' ')
    disp(' MASS MATRIX')
    disp(' ')
    for i=1:n
        for j=i:n
            jj=num2str(j);
            ii=num2str(i);
            pp=[' m[',ii,' ',jj,']= '];
            m(i,j) = input(pp);
            m(j,i) = m(i,j);
        end
    end

    % Stiffness matrix is typed (upper triangular part)
    disp(' ')
    disp(' STIFFNESS MATRIX')
    for i=1:n
        for j=i:n
            jj=num2str(j);
            ii=num2str(i);
            pp=[' k[',ii,' ',jj,']= '];
            k(i,j) = input(pp);
            k(j,i) = k(i,j);
        end
    end

    % Matrices are saved
    disp(' ')
    disp(' SAVE MATRICES')
    nome=input(' File name [NO extension needed] : ','s');
    savefile=['save ',nome,'.mat',' m',' k'];
    eval(savefile);
else
    disp(' ');
    loadfile=['load ',nome,'.mat'];
    eval(loadfile);
    n = length(diag(m));
end

% Damping matrix is proportional to M and K
disp(' ')
alfa = input(' Constant \alpha: ');
beta = input(' Constant \beta: ');
```

```
c = alfa*m + beta*k;

% Eigenproblem eq. (23.8.3)
[psi,lam]=eig(k,m);
lam = diag(lam);

% Initital conditions
disp(' ')
disp(' Initital conditions')
for i=1:n
    ii=num2str(i);
    pp=[' Initital displacement    x0[',ii,']= '];
    x0(i) = input(pp);
end
for i=1:n
    ii=num2str(i);
    pp=[' Initital velocity     v0[',ii,']= '];
    v0(i) = input(pp);
end

x0=x0';
v0=v0';

% Modal matrices
M = psi'*m*psi;
K = psi'*k*psi;
C = alfa*M + beta* K;

%
omega = sqrt(lam); % Natural frequencies
T = 2*pi/min(omega); % Max period of oscillation
dt = T/100; % Sampling period
NP=40; % Number of periods to compute
t=(0:dt:NP*T);
for r=1:n
    om = omega(r);
    zeta = C(r,r)/(2*om);
    od = om*sqrt(1 - zeta^2);
    A(r)=psi(:,r)'*m*x0/M(r,r);  %eq. (23.9.6)
    B(r)=psi(:,r)'*m*(v0+zeta*om*x0)/(M(r,r)*od); % eq. (23.9.6)
    eta(r,:) = exp(-zeta*om*t).*(A(r)*cos(od*t)+B(r)*sin(od*t)); %
Modal coordinates: eq. (23.9.5)
end

% Modal transformation
x = psi*eta; % eq. (23.9.2)

% Graphical representation
eta=eta';
x=x';

figure(1)
clf;
hold on
xlabel('Time');
ylabel('');
grid on
zoom on
title('MODAL COORDINATES  \eta')
if n==1
    xx=eta(:,1);
```

```
    plot(t,xx,'r');
elseif n==2
    xx=eta(:,1);
    plot(t,xx,'r');
    xx=eta(:,2);
    plot(t,xx,'b');
else
    xx=eta(:,1);
    plot(t,xx,'r');
    xx=eta(:,2);
    plot(t,xx,'b');
    xx=eta(:,3);
    plot(t,xx,'g');
    for i=4:n
        xx=eta(:,i);
        plot(t,xx,'k')
    end
    legend('\eta_1','\eta_2','\eta_3')
end
hold off

figure(2)
clf;
hold on
tx=xlabel('Time');
ty=ylabel('Displacements');
grid on
zoom on
title('PHYSICAL COORDINATES - DISPLACEMENTS x')
xx=x(:,1);
p1=plot(t,xx);
set(p1,'Linestyle','-', 'LineWidth',1.,'Color','r');
xx=x(:,2);
p1=plot(t,xx);
set(p1,'Linestyle','-.', 'LineWidth',1.,'Color','k');
if n>2
    for cc=3:n
    xx=x(:,cc);
    p1=plot(t,xx);
    set(p1,'Linestyle','--', 'LineWidth',1.,'Color','b');
    legend('x_1','x_2','x_3')
    end
end
hold off
```

NDOF_FRF

```
Frequency Response Function (Receptance) of a "N" degrees of freedom
lumped system with proportional viscous damping

File DOF3.mat contains mass and stiffness matrices as in the
Example 23.10

%
clear all;
close all hidden;
set(0,'DefaultAxesFontSize',18)

disp(' ')
disp(' Type a valid file name [e.g. DOF3]')
nome=input(' or press [return] to enter matrices: ','s');
```

```
vuoto=isempty(double(nome));

% disp(' ')
% disp(' E'possibile leggere le matrici da file (se salvate in
precedenza)')
% disp(' ')
% disp(' Nome file [NON dare alcuna estensione]')
% nome=input(' oppure [return -> input da tastiera] : ','s');
% vuoto=isempty(double(nome));
%
% if vuoto==1
%   disp(' ')
%   n = input(' Numero di gradi di liberta'' : ');
%   n = abs(n);
%
%   % Matrice di massa del sistema fisico
%   disp(' ')
%   disp(' Matrice massa')
%   disp(' ')
%   for i=1:n
%     for j=i:n
%       jj=num2str(j);
%       ii=num2str(i);
%       pp=[' m[',ii,' ',jj,']= '];
%       m(i,j) = input(pp);
%       m(j,i) = m(i,j);
%     end
%   end
%
%   % Matrice di rigidezza del sistema fisico
%   disp(' ')
%   disp(' Matrice rigidezza')
%   for l=1:n
%     for j=l:n
%       jj=num2str(j);
%       ll=num2str(l);
%       kk=[' k[',ll,' ',jj,']= '];
%       k(l,j) = input(kk);
%       k(j,l) = k(l,j);
%     end
%   end
%
%   % Salvo le matrici in un file
%   disp(' ')
%   disp(' Salvo le matrici')
%   nome=input(' Nome file [NON dare alcuna estensione] : ','s');
%   savefile=['save ',nome,'.mat',' m',' k'];
%   eval(savefile);
% else
%   disp(' ');
%   disp(' Leggo le matrici')
%   loadfile=['load ',nome,'.mat'];
%   eval(loadfile);
%   n = length(diag(m));
% end
%
% % Costanti di proporzionalit? per la matrice smorzamento
% alfa = input(' Costante di proporzionalit? per [m] = ');
% beta = input(' Costante di proporzionalit? per [k] = ');
% c = alfa*m + beta*k;
%
```

```
% % Scelgo i punti su cui calcolare la frf
% disp(' ')
% mm = input(' Posizione della forzante = ');
% disp(' ')
% ll = input(' Posizione dell''uscita = ');
%
% % Costruisco la matrice di cui calcolare gli autovettori
% [psi,lam]=eig(k,m);
% lam = diag(lam);
%
% % Calcolo la matrice di massa modale
% M = psi'*m*psi;

if vuoto==1
  n = input(' Number of dofs: ');
  n = abs(n);

  % Mass matrix is typed (upper triangular part)
  disp(' ')
  disp(' MASS MATRIX')
  disp(' ')
  for i=1:n
    for j=i:n
      jj=num2str(j);
      ii=num2str(i);
      pp=[' m[',ii,' ',jj,']= '];
      m(i,j) = input(pp);
      m(j,i) = m(i,j);
    end
  end

  % Stiffness matrix is typed (upper triangular part)
  disp(' ')
  disp(' STIFFNESS MATRIX')
  for i=1:n
    for j=i:n
      jj=num2str(j);
      ii=num2str(i);
      pp=[' k[',ii,' ',jj,']= '];
      k(i,j) = input(pp);
      k(j,i) = k(i,j);
    end
  end

  % Matrices are saved
  disp(' ')
  disp(' SAVE MATRICES')
  nome=input(' File name [NO extension needed] : ','s');
  savefile=['save ',nome,'.mat',' m',' k'];
  eval(savefile);
else
  disp(' ');
  loadfile=['load ',nome,'.mat'];
  eval(loadfile);
  n = length(diag(m));
end

% Damping matrix is proportional to M and K
disp(' ')
alfa = input(' Constant \alpha: ');
beta = input(' Constant \beta: ');
```

```
c = alfa*m + beta*k;

% INPUT and OUTPUT positions
disp(' ')
mm = input(' INPUT position: #  ');
disp(' ')
ll = input(' OUTPUT position: #  ');

% Eigenproblem eq. (23.8.3)
[psi,lam]=eig(k,m);
lam = diag(lam);

% Modal mass matrix
M = psi'*m*psi;

%
for r=1:n
  fi(:,r) = psi(:,r)/sqrt(M(r,r));
end

%
M = fi'*m*fi; % Identity matrix
C = fi'*c*fi;
K = fi'*k*fi;

% Modal damping factors
zeta = diag(C)./(2*sqrt(lam));

% Frequency axis limit for the FRF
om_r = sqrt(lam);  % Natural frequencies
ommin = min(om_r);
ommax = max(om_r);
dw = (round(ommin*10))/1000;  % Frequency resolution
wmax = (round(ommax*10))*0.15; %
w=(0:dw:wmax);
w=w';

% FRF: receptance eq. (23.10.4)
x=zeros(length(w),n);
v=zeros(length(w),n);
for r = 1:n,
  a1 = om_r(r)^2-w.^2;
  a2 = sqrt(-1)*2.0*zeta(r)*om_r(r)*w;
  x(:,r)=(fi(ll,r)*fi(mm,r))./(a1+a2);
end
x=conj(x');
[row col]=size(x);
if row>1
      xadd = sum(x);
else
      xadd=x;
end

% Graphical representation
x = abs(x)';
x=20*log10(x);
freq=w/(2*pi);

clf;
figure(1)
if n>=1
```

```
  plot(freq,20*log10(abs(xadd)),'k-')
end
grid on
hold on
tx=xlabel('Frequency (Hz)');
ty=ylabel('|FRF| (dB, ref = 1 m/N)');
zoom on
```

DOF3.mat

```
4d41 544c 4142 2035 2e30 204d 4154 2d66
696c 652c 2050 6c61 7466 6f72 6d3a 204d
4143 4936 342c 2043 7265 6174 6564 206f
6e3a 2054 6875 2053 6570 2031 3720 3133
3a34 303a 3433 2032 3031 3520 2020 2020
2020 2020 2020 2020 2020 2020 2020 2020
2020 2020 2020 2020 2020 2020 2020 2020
2020 2020 2020 2020 2020 2020 0001 494d
0f00 0000 2900 0000 789c e363 6060 7000
6236 20e6 80d2 20c0 0ae5 3343 3123 10e6
0269 2620 e604 f321 9809 aa1e 002a 5600
f40f 0000 0034 0000 0078 9ce3 6360 6088
0062 3620 e680 d220 c00a e533 4331 2310
6643 c555 80d8 a094 81e1 c38d ffff 6178
411b 230a 1f24 0f02 0083 c81a 39
```