A p p e n d i c e C

CODICE MATLAB

Lo scopo di questa appendice è di fornire al lettore un utile riferimento del codice Matlab di maggior rilievo utilizzato. In particolare, si riportano i codici degli algoritmi per il calcolo della trasformata di Fourier implementata nella funzione *four.m* , per l'identificazione della funzione di trasferimento *calcolafdt.m* e per la costruzione di una famiglia di processo *famfdt.m* non presenti nei toolbox Matlab. Sia il codice che implementa l'algoritmo FFT che il codice che fa l'interpolazione tra due curve è built-in Matlab (funzione *fft*) e quindi non verrà ripotato in questa appendice.

C.1 Funzione *four.m*

```
function [Real,Imag,f,Medio,Freq,Mod,Tc]=four(t,u,period,j)

%Calcolo la scomposizione di Fourier
%period: è un vettore che viene dallo schema simulink,in cui l'ultimo
%       elemento è proprio il periodo del mio segnale
%u:     segnale di cui voglio calcolare la scomposizione
%j:     indice  che regola la sommatoria sui k
%A:     vettore contenente Aik,parte Reale

%B:     vettore contenente Bik,parte immaginaria
%Real:  parte reale del segnale ricostruito con la funzione
%Imag:  parte immaginaria del segnale ricostruito con la funzione
%Medio: valore medio o U(0) del segnale ricostruito con la funzione
%Freq:  vettore delle frequenze
%Mod:   modulo del segnale ricostruito con la funzione
%f:
```

$$u(t)=\sum_{k=1}^{\infty}\left[\left(\sum_{i=1}^{n}A_{ik}\right)\cos(k\Omega t)+\left(\sum_{i=1}^{n}B_{ik}\right)\sin(k\Omega t)\right]+\frac{A}{T}\sum_{i=1}^{n}(t_i-t_{i-1})_{i\_dispari}-\frac{A}{T}\sum_{i=1}^{n}(t_i-t_{i-1})_{i\_pari}$$

```
%Tc:    tempo di campionamento della simulazione
```

```
%=============================================================
====
%Implementazione della funzione
%Aik=-Ai/Pi*k*[sin(k*omega*ti[i-1]-sin(k*omega*ti[i])]
%Bik=Ai/Pi*k*[cos(k*omega*ti[i-1]-cos(k*omega*ti[i])]
%=============================================================
====
Tc=t(2)-t(1);
%Calcolo ti

n=1;
ti(1)=t(1);

len=length(u);
for i =2:len


   if(u(i)~=u(i-1))
      n=n+1;
      ti(n)=t(i);
   end
    if t(i)>=period(len)
      break
   end
end
len_ti=length(ti);
%################################
%Calcolo i valori di Omega
%################################
 l=length(period);
 omega=2*pi/(period(l));
%##########################################
%Calcolo del valore costante
%2*Amplitude*ti(disp)-2*Amplitude*ti(pari)
%##########################################

cost=0;
Amp=u(1);
%ti dispari
i=1;
m=1;
while i<=len_ti
 disp(m)=ti(i);
 m=m+1;
 i=i+2;
```

```
end

%ti pari
i=2;
m=1;
while i<=len_ti
 pari(m)=ti(i);
 m=m+1;
 i=i+2;
end

l_pari=length(pari);
l_dispari=length(disp);
if (mod(len_ti,2)==0) % se n è pari
    D=2*Amp*disp;
    E=2*Amp*pari(1:l_pari-1);
    cost=sum(D)-sum(E)-Amp*pari(l_pari);
else   % se n dispari
    D=2*Amp*disp(1:l_dispari-1);
    E=2*Amp*pari;
    cost=sum(D)-sum(E)+Amp*disp(l_dispari);
end
%#####################################################
% ti=ti(1:len_ti-1);
% len_ti=len_ti-1;
% %j=100;     %variabile che regola la sommatoria sui k
A(:,:)=zeros(len_ti,j); %10->len_ti
B(:,:)=zeros(len_ti,j); %10->len_ti

%#####################################################
%Calcolo di Aik_Devo ancora definire omega
%#####################################################

for k=1:j
    for i=1:len_ti %10->len_ti

        if(mod(i,2)==0)   %Guardo se è un indice pari o dispari
            Amp(i)=u(1);
        else Amp(i)=-u(1);
        end
        if i==1
        A(i,k)=(-Amp(i)/(pi*k))*(sin(k*0*omega)-sin(k*omega*ti(i)))   ;
        else     A(i,k)=(-Amp(i)/(pi*k))*(sin(k*omega*ti(i-1))-sin(k*omega*ti(i)));
        end
    end
end
```

```
A=sum(A);
%#####################################################
%Calcolo di Bik_Devo ancora definire omega
%#####################################################

for k=1:j
   for i=1:len_ti %10->len_ti

       if(mod(i,2)==0)
          Amp(i)=u(1);
       else Amp(i)=-u(1);
       end
       if i==1
       B(i,k)=(Amp(i)/(pi*k))*(cos(k*0*omega)-cos(k*omega*ti(i)));
       else     B(i,k)=(Amp(i)/(pi*k))*(cos(k*omega*ti(i-1))-cos(k*omega*ti(i)));
       end
   end
end
B=sum(B);  %in questo modo diventa un vettore riga 1*k
%#####################################################
%Costruisco il vettore di Campionamento
%#####################################################
t2=0:0.01:period(len); %Tcampionamento
len2=length(t2);

%#####################################################
%Costruisco il vettore dei cos(k*omega*t) _nella mia notazione al posto di t
%userò la variabile x,e il vettore lo chiamerò C
%#####################################################

for k=1:j
   for m=1:len2
   C(k,m)=cos(k*omega*t2(m));  %vettore dei coseni
   S(k,m)=sin(k*omega*t2(m));  %vettore dei seni
   end
end
% %C=C.'  %trasposta del vettore



%#####################################################
%Calcolo di u2(t)
%#####################################################
F=A*C; %Aik*cos(k*w*t)
G=B*S; %Bik*sin(k*w*t)
f=(F'+G')-cost/period(len); %f(t)
```

```
cost=-cost/period(len);
Medio=cost;
figure(1);
grid on;

plot(t2,f,'b');
%##############################################
u2=u(1:len2);
%#################################
hold on
plot(t2,u2,'r')
%plot(t,u,'r')
%#################################################
%Calcolo il modulo come Mod=sqrt2(Aik^2+Bik^2)
%Calcolo la Phase come Phase=arctg(Bik/Aik)
%Calcolo dei valori delle Frequenze
%#################################################


 Real=A/2;
 Imag=-B/2;
for k=1:j
  Mod(k)=sqrt(((A(k)/2)^2)+((B(k)/2)^2));
  Freq(k)=omega*k;

end
  %aggiungo la componente k=0
Mod=[abs(cost),Mod];
Freq=[0,Freq];
Phase_ing=unwrap(atan2(Imag,Real));
%Phase_ing=rad2deg(Phase_ing);
figure(2);
grid on;
f1 = (0:length(Freq)-1)'*100/length(Freq);
subplot(2,1,1),semilogx(f1,20*log10(Mod))
ylabel('Abs. Magnitude'), grid on
%
 %subplot(2,1,2), semilogx(f1(1:length(Freq)-1),Phase_ing)
% ylabel('Phase [degree]'), grid on
% xlabel('Frequency [rad/sec]')
```

C.2 Funzione *calcolafdt.m*

```
%function [Modg,Tc]=funz(Mod,Freq,Tc,out,t)
%Calcola la trasformata di Fourier dell'uscita,la funzione di trasferimento
function [Mod_usc,f_usc,Modg,Freq]=calcolafdt(Mod,Freq,Tc,y,t)
figure(1)
plot(Freq,Mod);
title('Spettro di ampiezza dell ingresso')

x=y;
lent=length(x);

Yb=fft(x,lent)/lent;

Mod_usc=abs(Yb);

c=find(Mod_usc<0);
Mod_usc(c)=0;

f_usc=[0:(lent/2-1)']*2*pi/(lent*Tc); %Freq sono in rad/sec prendiamo solo metà
spettro
figure(2);

plot(f_usc,Mod_usc(1:(lent/2)));
title('spettro di ampiezza dell uscita')


Mod_usc_F_ing=interp1(f_usc,Mod_usc(1:(lent/2)),Freq)

figure(3)
hold on
plot(Freq,Mod_usc_F_ing,'r')
title('spettro di ampiezza dell uscita alle frequenze di ing')

Modg=Mod_usc_F_ing./Mod;

figure(4)
plot(Freq,Modg)
title('spettro di ampiezza della funzione di trasferimento')
xlabel('Frequency [rad/sec]')
```

```
figure(5)
d=find(Modg>0);
c=Modg(d);
ModgdB=20*log10(c);
semilogx(Freq(d),ModgdB)
title('20Log della funzione di trasferimento')
xlabel('Frequency [rad/sec]')
% figure(6)
% plot(Freq(1:length(Freq)),Phase)
```

C.3 Funzione *famgfdt.m*

```
function [absg]=famgfdt(n,Freq)

%In questa funzione sono state unite dell G(s)
%caso 1: G(s)=1/s*(s+3)
%caso 2: G(s)=1/(s^3+2*s^2+s)
%caso 3: G(s)=s+3/(s^3+2*s^2+5*s)

w=Freq;


%######################################################################
% Calcolo dello spettro di ampiezza della funzione di trasferimento
% Grafico dello spettro di ampiezza
%######################################################################


switch  (n)
   case 1  %G(s)=1/s*(s+3)
      disp ('G(s)=1/s*(s+3)')
      g=1./((i*w).^2+3*(i*w));
      absg=abs(g);
      figure(4)
      hold on
      plot(Freq,absg,'r');
      title('Spettro di ampiezza fdt 1/(s^2+3*s)')
   case 2 %G(s)=1/(s^3+2*s^2+s)
      disp ('G(s)=1/(s^3+2*s^2+s)')
```

```
        g=1./((i*w).^3+2*(i*w).^2+(i*w));
        absg=abs(g);
        figure(4)
        hold on
        plot(Freq,absg,'r');
        title('Spettro di ampiezza fdt 1/(s^3+2*s^2+s)')
    case 3  % G(s)=s+3/(s^3+2*s^2+5*s)
        disp(' G(s)=s+3/(s^3+2*s^2+5*s')
        g=(i*w+3)./((i*w).^3+(i*w).^2+5*(i*w));
        absg=abs(g);
        figure(4)
        hold on
        plot(Freq,absg,'r');
        title('Spettro di ampiezza fdt s+3/(s^3+2*s^2+5*s)')
    otherwise
      disp('Unknown method.')
end

%######################################################################
% Diagramma di Bode
%######################################################################

figure(5)
hold on
semilogx(Freq,20*log10(absg),'r')
```