Table of Contents

an example showing how to construct input, run the

simulink block 'plantForExperiment.slx' and extract the output.

```
clear all


Ts  = 0.3;
Fs = 1/Ts;
% The sampling period (Ts) must be defined for the
% simulink model to run
% warning: do not make Ts larger than 0.01.

%constructing the plant
numPlant_CT=[10 2];
denPlant_CT=[1 1 4.25];
plant_CT_TF=tf(numPlant_CT,denPlant_CT);
[plantFreqResp_CT,freqs_forPlant_CT_rad]=freqs(numPlant_CT,numPlant_CT);
%Beware: the plant is a discrete time transfer function
plant_DT_TF = c2d(plant_CT_TF,Ts,'zoh');
[numPlant_DT,denPlant_DT] = tfdata(plant_DT_TF,'v');

%1 : compute the true freq response, show Bode plot,
numFreqValues = 100;
[plantFreqResp,freqValuesHz_plant] =
 freqz(numPlant_DT,denPlant_DT,numFreqValues,Fs);
freqs_forPlant_rad = freqValuesHz_plant*2*pi;

figure
bode(plant_CT_TF,plant_DT_TF)
legend('CT','DT with zoh')

% [A,B,C,D] = tf2ss(numPlant_DT,denPlant_DT);
% plant_DT_SS = ss(A,B,C,D,Ts);
% state_dim  =length(denPlant_DT)-1;
% x0 = randn(state_dim,1);
%------


%construct input signal

N = 10000;
```

```
T = [0:1:N-1]'*Ts;
%input_type = 'GaussianWhite';
input_type = 'prbs';
%input_type = 'step';
% input_type = 'sweptsine';
noise_stdv = 0.1;

uLevel = 0.6; %max value of input allowed
switch input_type
    case 'prbs'
        u = idinput(length(T),'prbs',[0 1],[-uLevel, uLevel]);
    case 'GaussianWhite'
        u = randn(length(T),1)*sqrt(uLevel);
 case 'sweptsine'
        f0 = 0.002;
        u = uLevel*sin(2*pi*f0*T.*T);
    case 'step'
        u = -uLevel*ones(size(T));
        ind = floor(length(T)/5);
        u(ind:end) = uLevel;

    otherwise
        error('bad')
end
%clip:
u(find(u>uLevel))=uLevel;
u(find(u<-uLevel))=-uLevel;

%------

% u_struct =  [];
% t = [0:1:10000]'*Ts;
% u = sin(2*pi*0.01*t.^2);
u_struct.time = T;
u_struct.signals.values = u;
u_struct.signals.dimensions = 1;
t_final=T(end);

simOut=sim('plantForExperiment')

%extract y
y_struct = simOut.y_struct;
tout = y_struct.time;
y = y_struct.signals.values;

%plot
figure
subplot(211)
plot(T,u);ylabel('u');
subplot(212);
plot(T,y);ylabel('y');
```

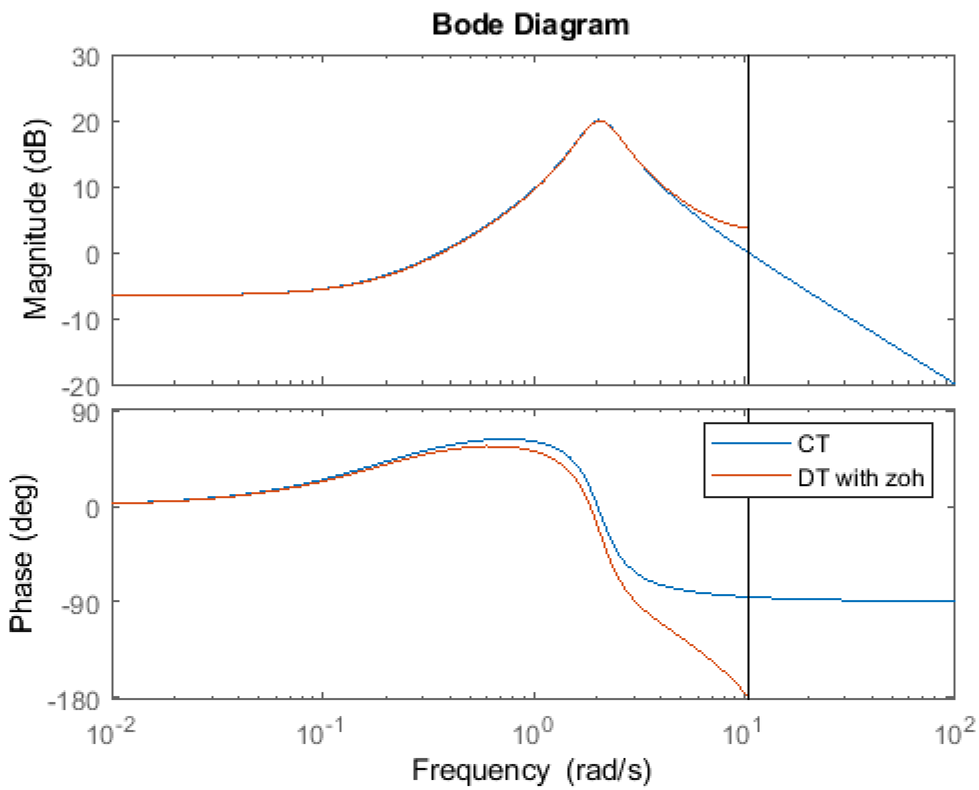
Warning: The PRBS signal delivered is the 10000 first values of a full sequence

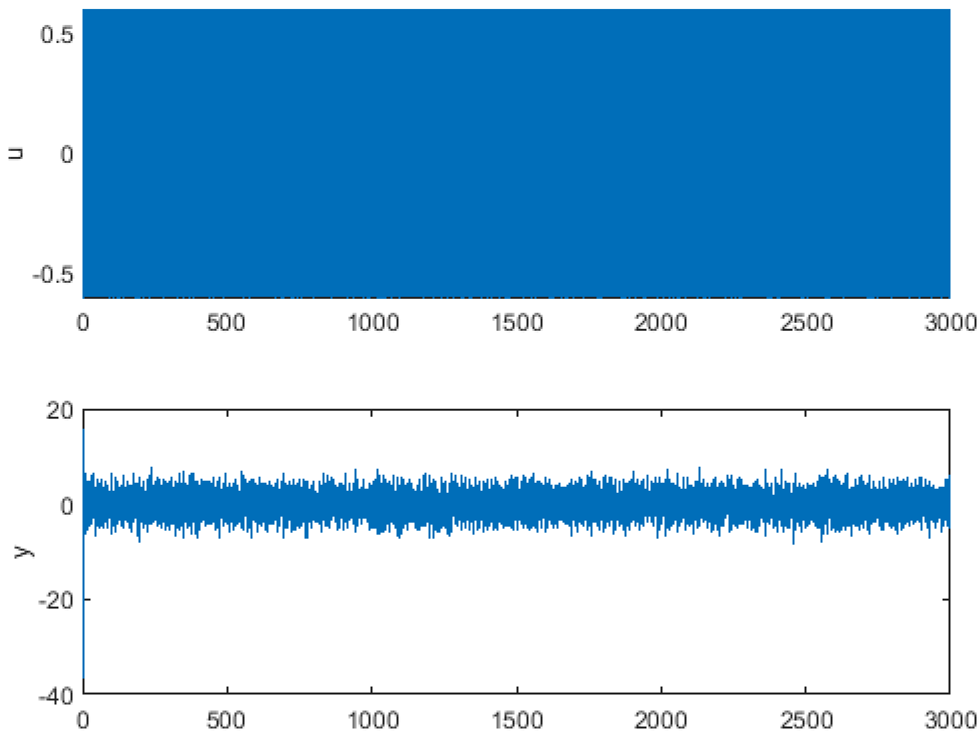
```
of length 16383.

simOut =

  Simulink.SimulationOutput:
                   tout: [10002x1 double]
               y_struct: [1x1 struct]

     SimulationMetadata: [1x1 Simulink.SimulationMetadata]
           ErrorMessage: [0x0 char]
```

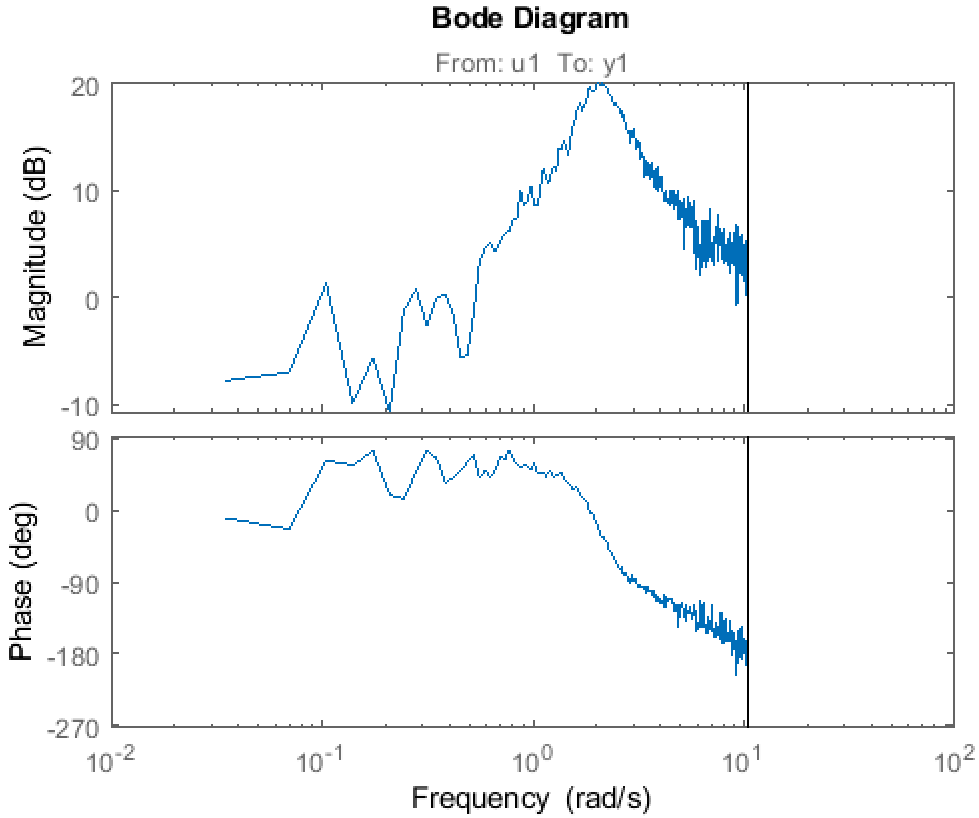




Step 2: estimate G(jw) = Y(jw)/U(jw) from u(t),y(t),

by using the etfe command Step 2.1: put the data in the right format that etfe wants: idDataForThePlant = iddata(y,u,Ts); Phat_DT = etfe(idDataForThePlant);

```
figure
Q=6000;
numFreqRes=300;
uetfe=u(end-Q+1:end);
yetfe=y(end-Q+1:end);
idDataForThePlant = iddata(yetfe,uetfe,Ts);
smoothingWindowSize = Q/10;
Phat_DT = etfe(idDataForThePlant,smoothingWindowSize,numFreqRes);
bode(Phat_DT)
```



compare with the true plant's frequency response.

Beware: the plant and the estimated plant are both discrete time transfer functions, so the frequency axis only goes up to the Nyquist frequency, which is half of the sampling frequency.

```
plantEstFreqResp=Phat_DT.ResponseData;
plantEstFreqResp = plantEstFreqResp(:);
if ~strcmp(Phat_DT.FrequencyUnit,'rad/TimeUnit')
    freqs_forETFE_rad = 2*pi*Phat_DT.Frequency;
else
    freqs_forETFE_rad = Phat_DT.Frequency;
end
```

compare the true with estimated

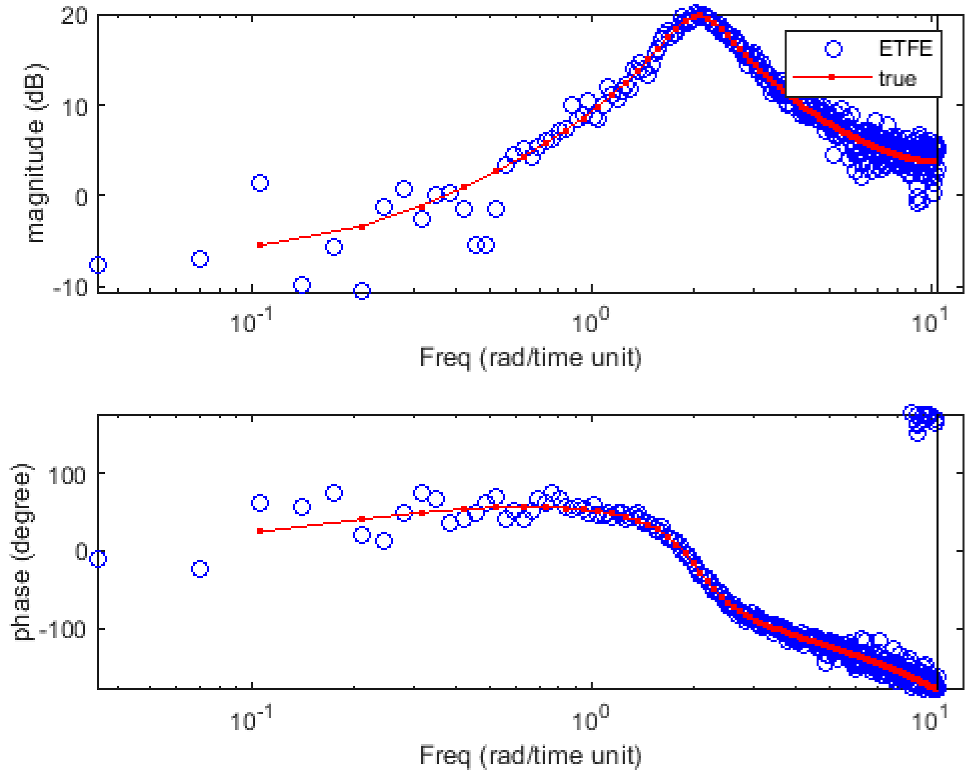
```
figure
gainph=[];
phaph=[];

subplot(211);%mag plot
%true plant:
```

```
gainph(1)  =
 semilogx(freqs_forETFE_rad,20*log10(abs(plantEstFreqResp)),'bo');
 hold on
gainph(2)  =
 semilogx(freqs_forPlant_rad,20*log10(abs(plantFreqResp)),'r.-');
% gainph(3)  =
 semilogx(freqs_forPlant_CT_rad,20*log10(abs(plantFreqResp_CT)),'r-');
% show the freq = pi location
plot(Fs*[pi, pi],20*log10([min(abs(plantFreqResp))
 max(abs(plantFreqResp))]),'k-');
axis tight
yy = ylim;
plot(Fs*[pi, pi],yy,'k-');
xlim([0 1.2*Fs*pi]);
xlabel('Freq (rad/time unit)');
ylabel('magnitude (dB)');
legend(gainph,'ETFE','true','true(CT)');

subplot(212); %phase plot
phaph(1)  = semilogx(freqs_forETFE_rad,angle(plantEstFreqResp)*180/
pi,'bo'); hold on
phaph(2)  = semilogx(freqs_forPlant_rad,angle(plantFreqResp)*180/
pi,'r.-');
% phaph(3)  =
 semilogx(freqs_forPlant_CT_rad,angle(plantFreqResp_CT)*180/pi,'r-');
% show the freq = pi location
axis tight
yy = ylim;
plot(Fs*[pi, pi],yy,'k-');
xlim([0 1.2*Fs*pi]);
xlabel('Freq (rad/time unit)');
ylabel('phase (degree)');
```

Warning: Ignoring extra legend entries.



Published with MATLAB® R2020a