Liu Xiangyu G1802061L

**Continious Assignment 1 Result**

1. First order plus time delay model using least squares method under open loop step test in time domain.

**Matlab Program:**

%establish the desired tranfer function model and sample it using step test

s = tf('s');

Gp = (s+3.5)\*exp(-2.5\*s) / (((s+1)^3)\*((s+2)^2)\*(s^2+7\*s+10));

Ts=0.1;

Tf=100;

T=0:Ts:Tf;

yt=step(Gp,T);

%initialize the paramters

A=1; %the gain of input step A=1

nPt=30; %number of sample points nPt=30

ts=20\*Ts; %time interval between sample points ts=2s

t1=30; %start at t1=3s which is after the time delay L

t2=ts/Ts\*(nPt-1)+t1; %end at t2

tvec=t1:ts/Ts:t2; %tranform t1,ts,t2 into a vector, easier to program

Gamma=yt(tvec,:); %select the sample points from the vector of overall step test sampling results which is also Gamma matrix in this model

%calculate Phi matrix

Phi=zeros(nPt,3);

for i=1:nPt

intgr=0;

%compute the integral part using trapezoidal integral method

for k=1:i

if i==1

intgr=Gamma(k)\*ts+intgr;

elseif k==1||k==nPt

intgr=Gamma(k)\*ts\*0.5+intgr;

else

intgr=Gamma(k)\*ts+intgr;

end

end

%compose Phi matrix

Phi(i,1)=-intgr;

Phi(i,2)=-A;

Phi(i,3)=tvec(i)\*Ts\*A;

end

%calculate Theta matrix and get the values of a1,b1,L

Theta=inv((transpose(Phi)\*Phi))\*transpose(Phi)\*Gamma;

a1=Theta(1);

b1=Theta(3);

L=Theta(2)/Theta(3);

%compose the tranfer function of FOPTD model

gs=(b1)/(s+a1)\*exp(-L\*s);

%draw step response and Nyquist plot to compare

%red curve is the original function, green curve is the identified function

figure(1);

GpNyq=nyquistplot(Gp,'r')

setoptions(GpNyq, 'ShowFullContour', 'off');

hold

gsNyq=nyquistplot(gs,'g')

setoptions(gsNyq, 'ShowFullContour', 'off');

figure(2);

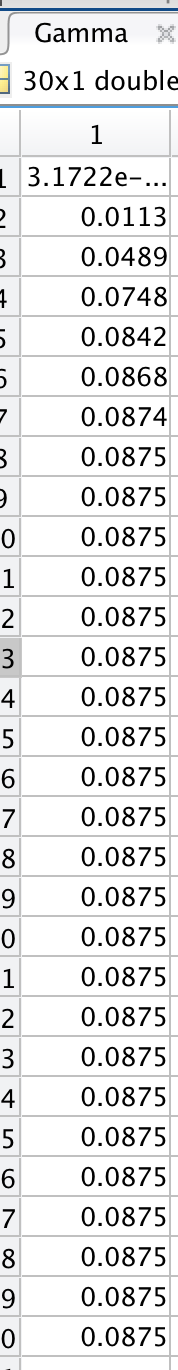
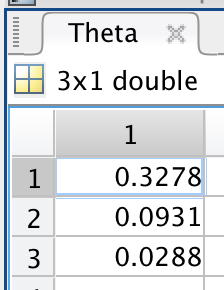
step(Gp,'r');

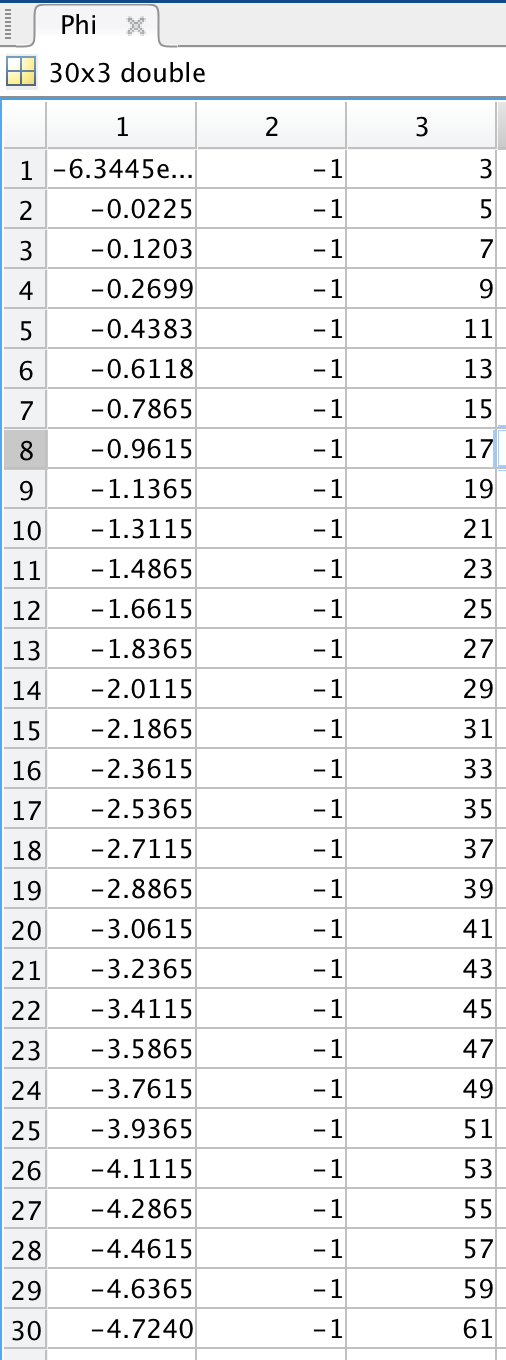
hold;

step(gs,'g');

**step by step results:**

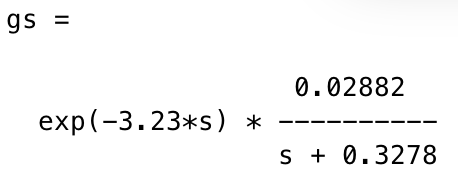
Phi Matrix: Gamma Matrix: Theta Matrix:



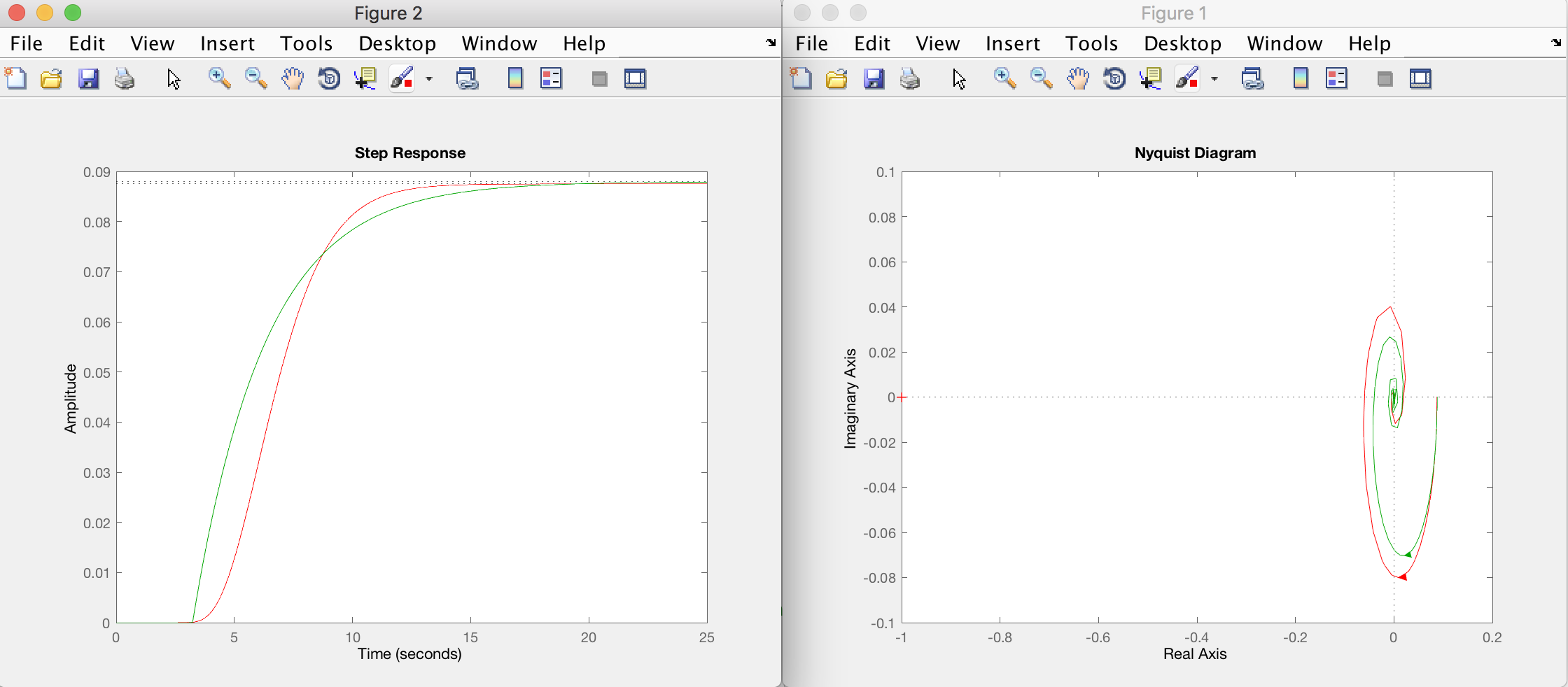


a1=0.3278, b1=0.02882, L=3.23;

and the identified transfer function is:



**Charts:**

****

1. First order plus time delay under open loop step test using least squares method in frequency domain.

**Matlab Program:**

%establish the desired tranfer function model and sample it using step test

s = tf('s');

Gp = (s+3.5)\*exp(-2.5\*s) / (((s+1)^3)\*((s+2)^2)\*(s^2+7\*s+10));

Ts=0.1;

Tf=100;

T=0:Ts:Tf;

yt=step(Gp,T);

%initialize the paramters

A=1; %the gain of input step A=1

M=10; %number of sample points in frequency domain M=10

nPt=30; %number of sample points in time domain nPt=30

ts=5\*Ts; %time interval between sample points ts=0.5s

t1=30; %start at t1=3s which is after the time delay L

t2=t1+(nPt-1)\*ts/Ts;%end at t2

tvec=t1:ts/Ts:t2; %tranform t1,ts,t2 into a vector, easier to program

t=(tvec-t1)\*Ts; %tranform tvec into a time vector starting from 0 in unit of 1s,

%easier to program

samples=yt(tvec,:); %select the sample points from the vector of overall step test sampling results

%intialize w0, phi0,w vector, phi vector, yss,

%delta\_yt,A1,A2,B1,B2,|G(jw)|^2

w0=0; phi0=0;

w=zeros(M,1); phi=zeros(M,1);

w(1)=10^-3;

yss=yt(length(yt));

delta\_yt=samples-yss;

A1=zeros(M,1);

A2=zeros(M,1);

B1=A;

B2=0;

G2jw=zeros(M,1);

%calculate the sampling pairs of w and phi

for i = 0:M-1

%compute w recursivly

if i==1

w(i+1)=w(i)-(i\*pi/(M-1)+phi(i))\*(w(i)-w0)/(phi(i)-phi0);

elseif i>1

w(i+1)=w(i)-((i\*pi/(M-1))+phi(i))\*(w(i)-w(i-1))/(phi(i)-phi(i-1));

end

%compute the integral part using trapezoidal integral method

dysin\_i=0;

for k=1:nPt

if k==1||k==nPt

dysin\_i=(delta\_yt(k))\*ts\*0.5\*sin(w(i+1)\*((k-1)\*ts))+dysin\_i;

else

dysin\_i=(delta\_yt(k))\*ts\*sin(w(i+1)\*((k-1)\*ts))+dysin\_i;

end

end

dycos\_i=0;

for k=1:nPt

if k==1||k==nPt

dycos\_i=(delta\_yt(k))\*ts\*0.5\*cos(w(i+1)\*((k-1)\*ts))+dycos\_i;

else

dycos\_i=(delta\_yt(k))\*ts\*cos(w(i+1)\*((k-1)\*ts))+dycos\_i;

end

end

A1(i+1)=yss+w(i+1)\*dysin\_i;

A2(i+1)=w(i+1)\*dycos\_i;

%compute |G(jw)|^2 and phases

G(i+1)=A1(i+1)+A2(i+1)\*j;

G2jw(i+1)=(abs(G(i+1)))^2;

phi(i+1)=angle(G(i+1));

end

Phi=zeros(M,2);

%compose Gamma Matrix

Gamma=G2jw;

%compose Phi Matrix

for i = 1:M

Phi(i,1)=-w(i)^2\*G2jw(i);

Phi(i,2)=1;

end

%compute Theta Matrix

Theta=inv(transpose(Phi)\*Phi)\*transpose(Phi)\*Gamma

%get a1,b1

a1=sqrt(Theta(1));

b1=sqrt(Theta(2));

%compute time delay L

phase=zeros(M,1);

for i=1:M

phase(i)=-atan2(w(i),a1)-phi(i);

end

L=inv(transpose(w)\*w)\*transpose(w)\*phase;

%compose the identified function and draw plots

gs=b1/(a1\*s+1)\*exp(-L\*s)

figure(1);

GpNyq=nyquistplot(Gp,'r')

setoptions(GpNyq, 'ShowFullContour', 'off');

hold

gsNyq=nyquistplot(gs,'g')

setoptions(gsNyq, 'ShowFullContour', 'off');

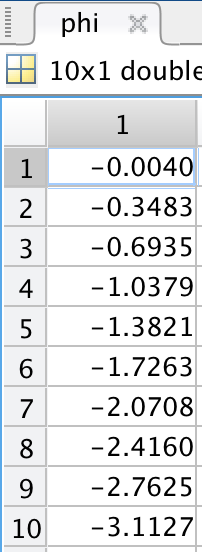
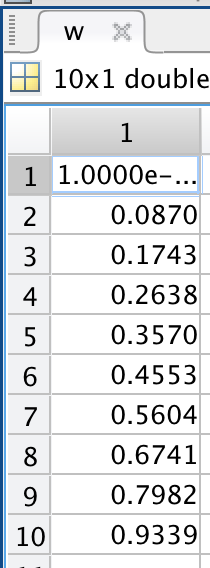
figure(2);

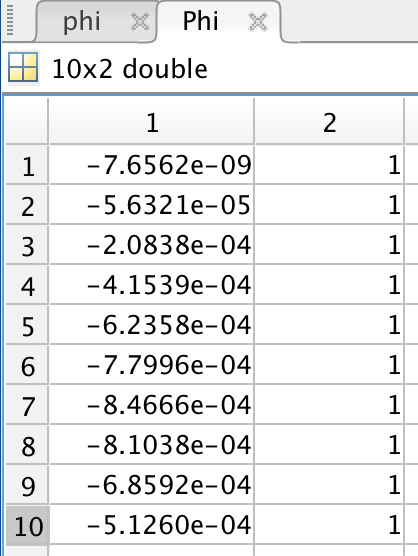
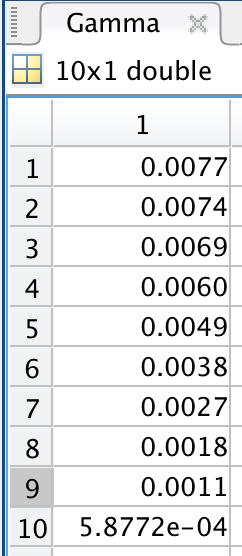
step(Gp,'r');

hold;

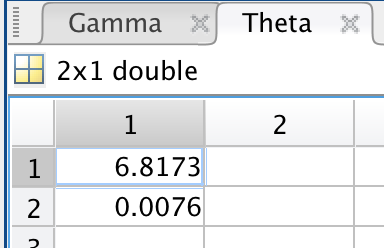
step(gs,'g');

**step by step results:**

vector of w: vector of phi:

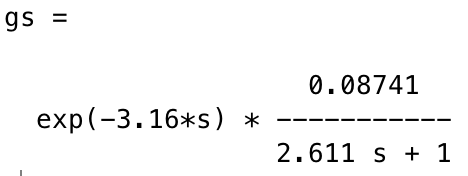
Phi Matrix: Gamma Matrix:

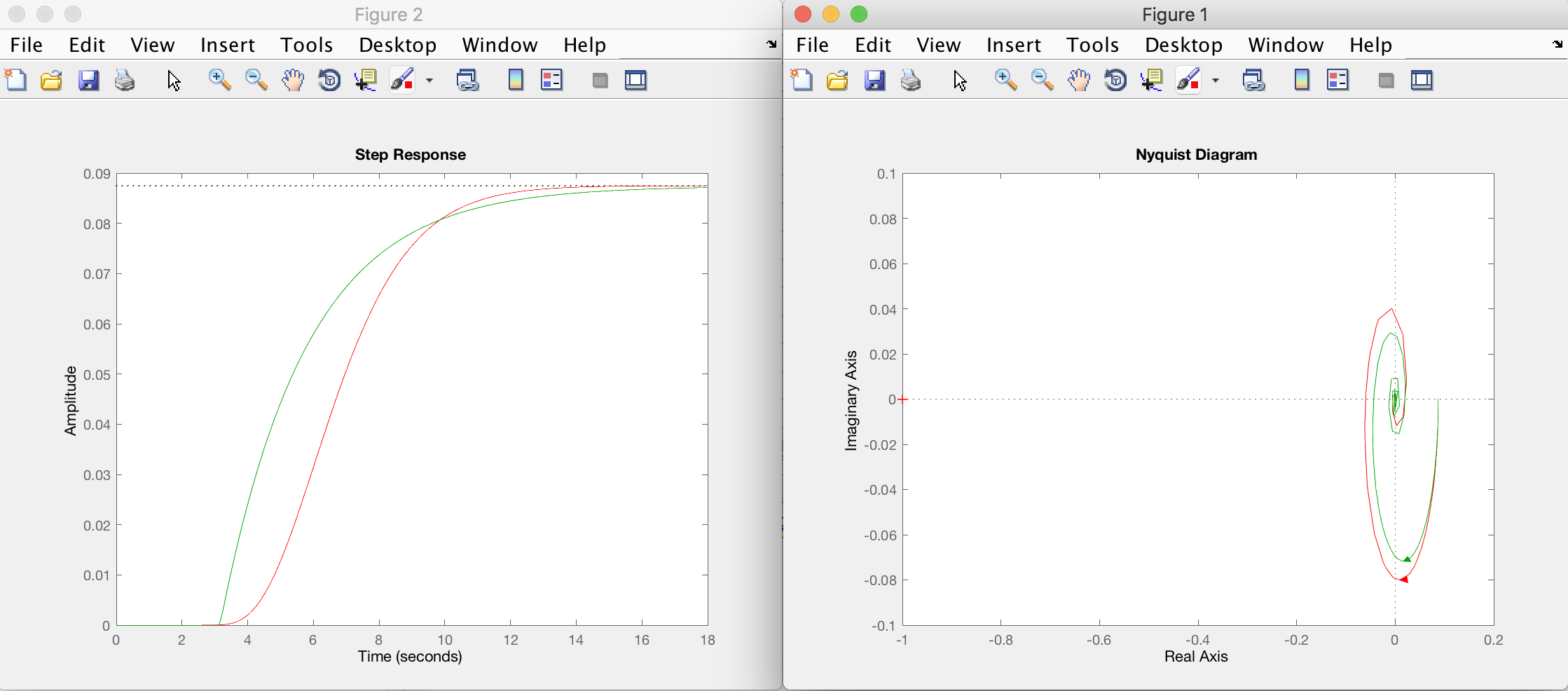
Theta Matrix:



a1= 2.611, b1=0.08761, L=3.16

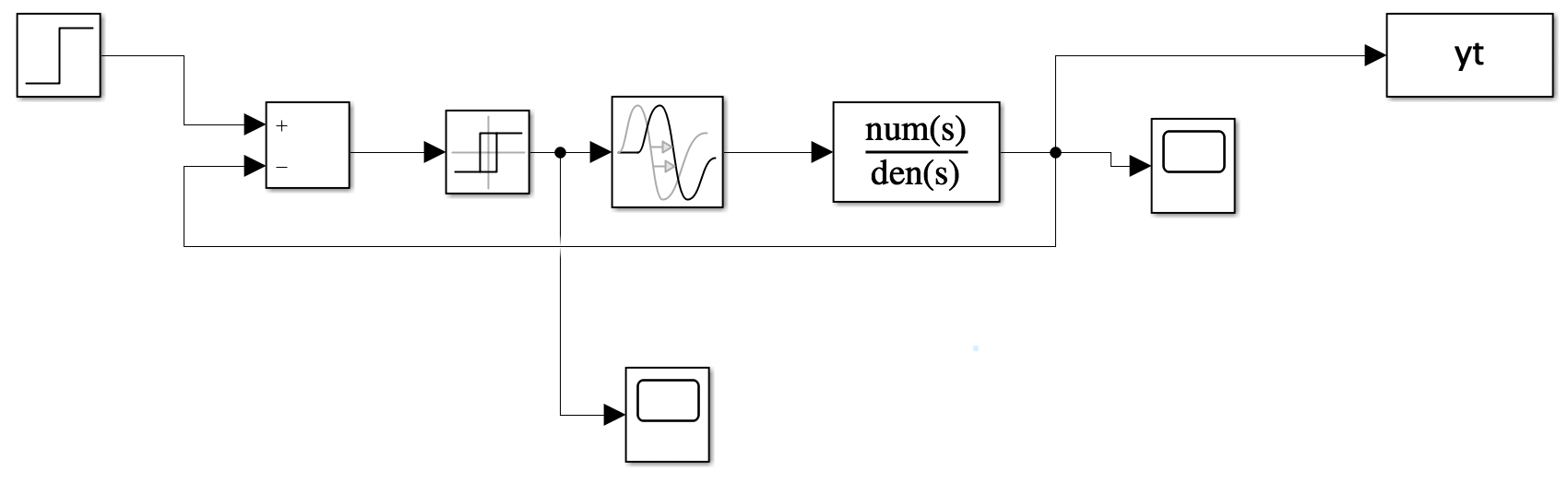
and the identified transfer function is

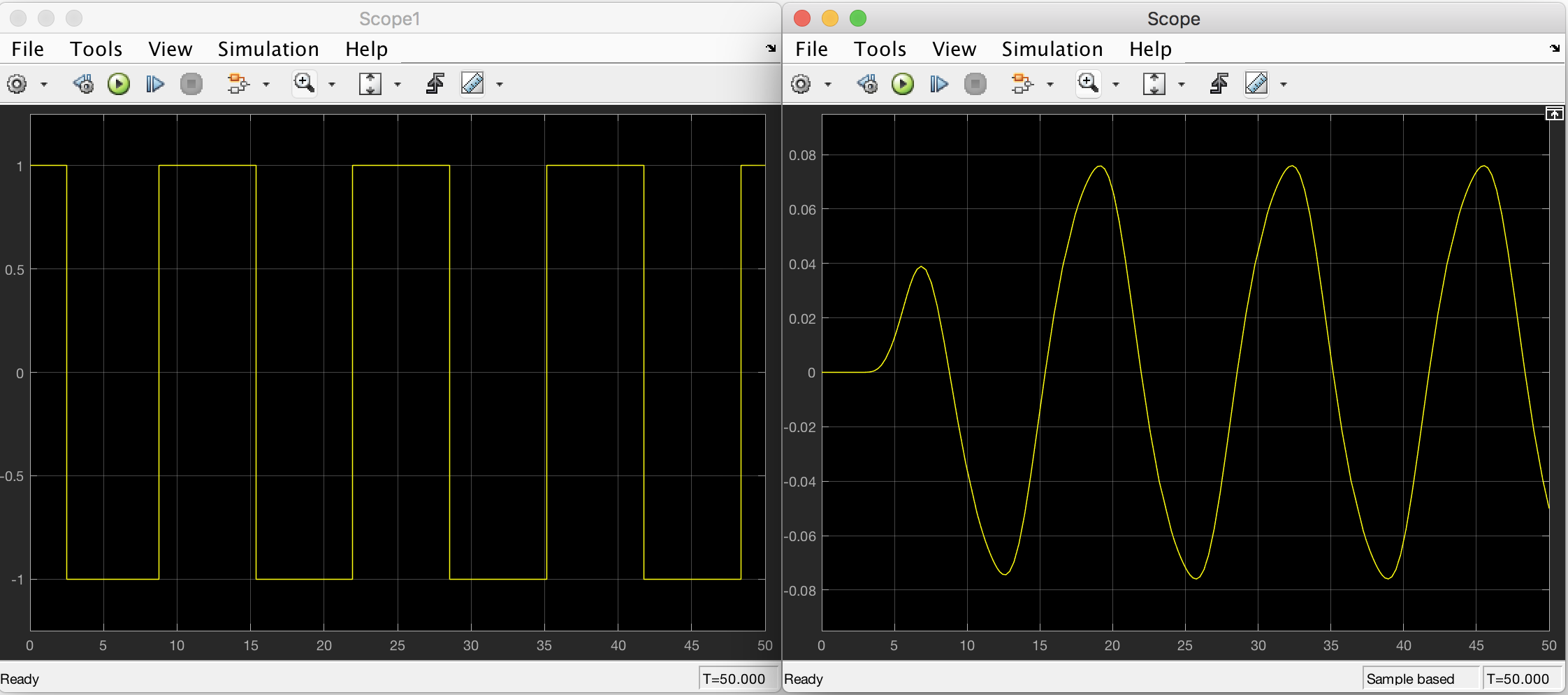


**Charts:**

1. Using relay feedback to generate sustained oscillation and using available information to calculate the parameters of first order plus time delay model.

**Simulink Model:**

****

****

**Matlab Program:**

%observed from the relay feedback test waveform

h=1;

L=3.5;

Ku=4\*h/(0.076\*pi);

wu=2\*pi/(32.3525-19.2051);

Kp=0.0875;

%calculate T

T=sqrt((Kp\*Ku)^2-1)/wu;

%compose the identified transfer function

s = tf('s');

gs=Kp/(T\*s+1)\*exp(-L\*s)

Gp = (s+3.5)\*exp(-2.5\*s) / (((s+1)^3)\*((s+2)^2)\*(s^2+7\*s+10));

%draw plots

figure(1);

GpNyq=nyquistplot(Gp,'r')

setoptions(GpNyq, 'ShowFullContour', 'off');

hold

gsNyq=nyquistplot(gs,'g')

setoptions(gsNyq, 'ShowFullContour', 'off');

figure(2);

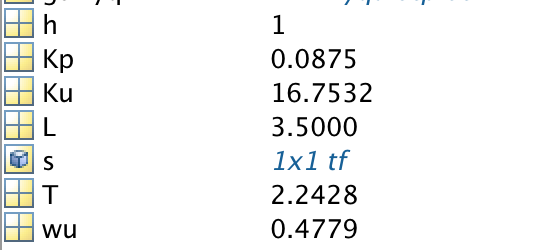
step(Gp,'r');

hold;

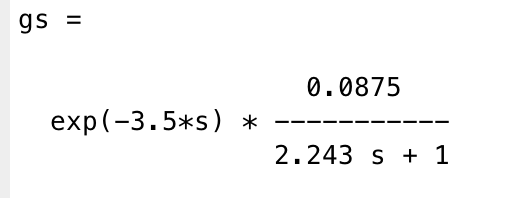
step(gs,'g');

**step by step result:**

Paramters:



The identified transfer function is:



**Charts**

