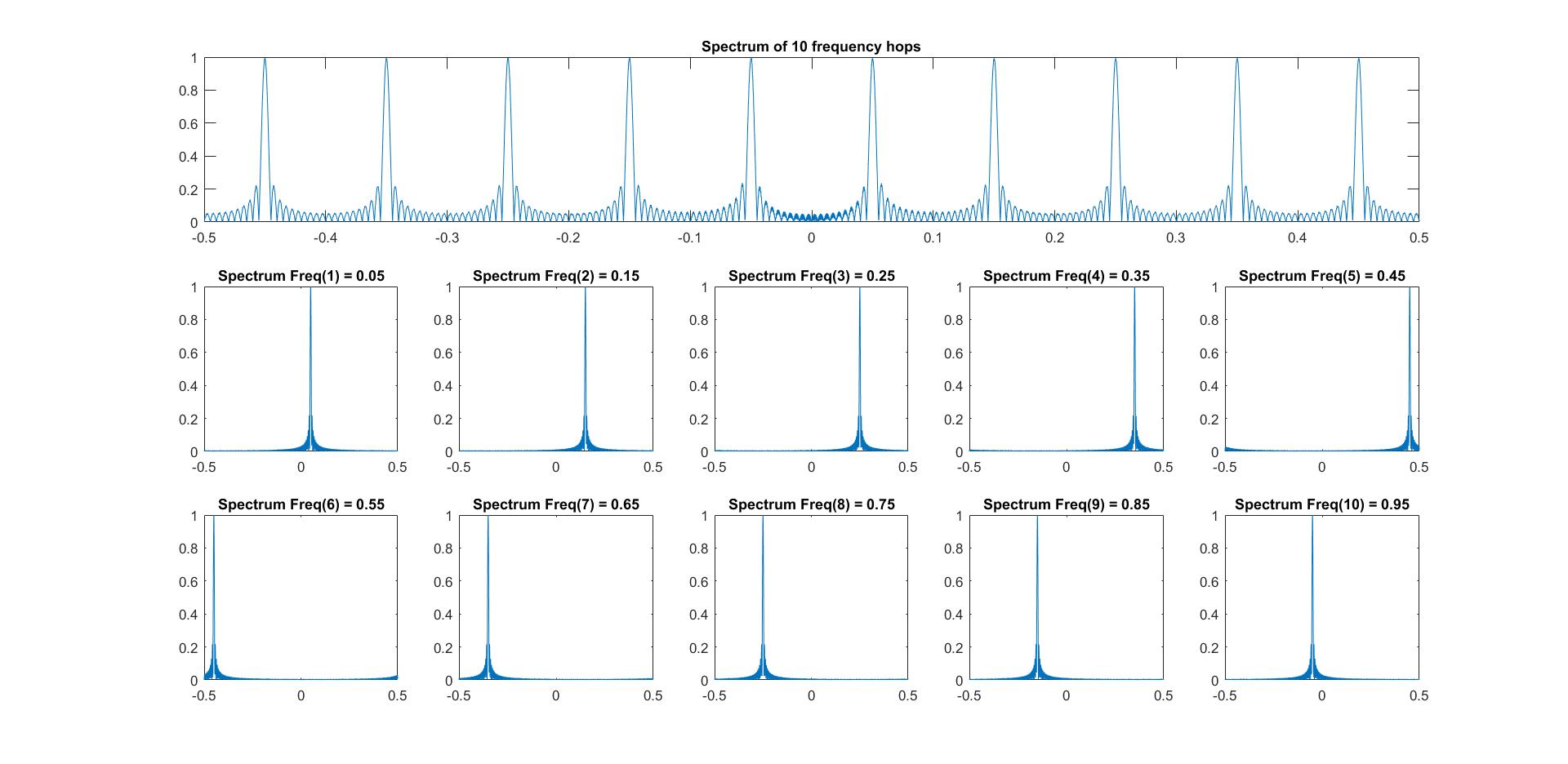
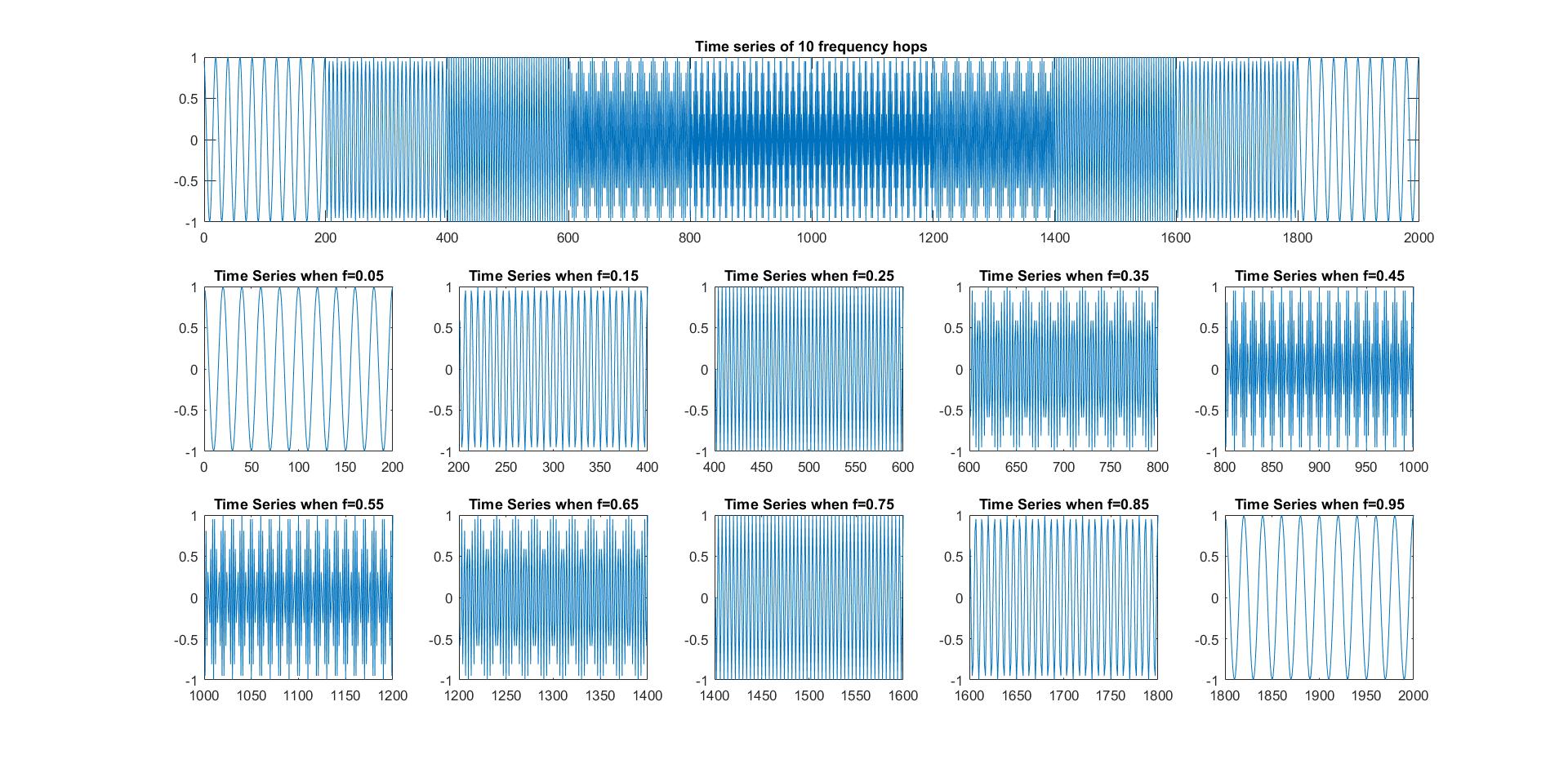
# EE654 – Adaptive Algorithms, Homework Assignment #1

a)



x=ones(1,200);

x1=[0.05\*x 0.15\*x 0.25\*x 0.35\*x 0.45\*x 0.55\*x 0.65\*x 0.75\*x 0.85\*x 0.95\*x];

x2=filter(1,[1 -1],x1);

x3=exp(j\*2\*pi\*x2);

N =2048;

i = 1;

fs = figure;

subplot(3,5,1:5)

X3 = fft(x3,4\*N);

plot(linspace(-0.5,0.5,4\*N), fftshift(abs(X3/max(X3))))

title('Spectrum of 10 frequency hops')

ts = figure;

subplot(3,5,1:5)

plot(0:length(x3)-1,x3)

title('Time series of 10 frequency hops')

for k = 1:200:2000

f = linspace(-0.5,0.5,N);

index = k:k+199;

figure(ts)

subplot(3,5,5+i)

plot(index, x3(index))

title(['Time Series when f=' num2str(x1(k))])

X3temp =fftshift(abs(fft(x3(index),N)));

figure(fs)

subplot(3,5,5+i)

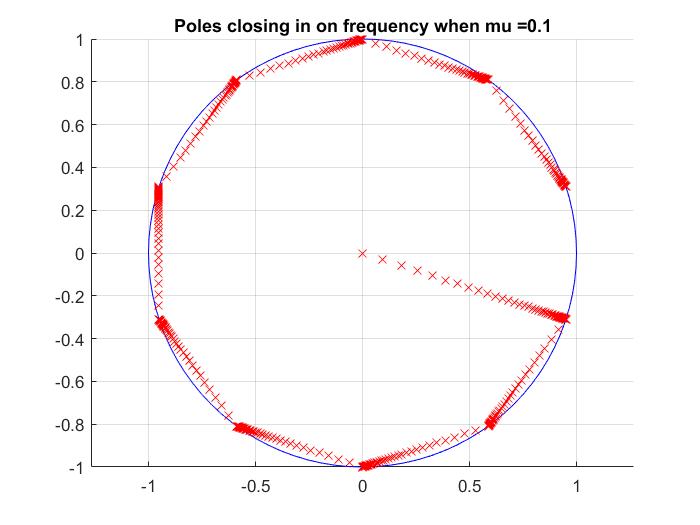
plot(f,X3temp/max(X3temp))

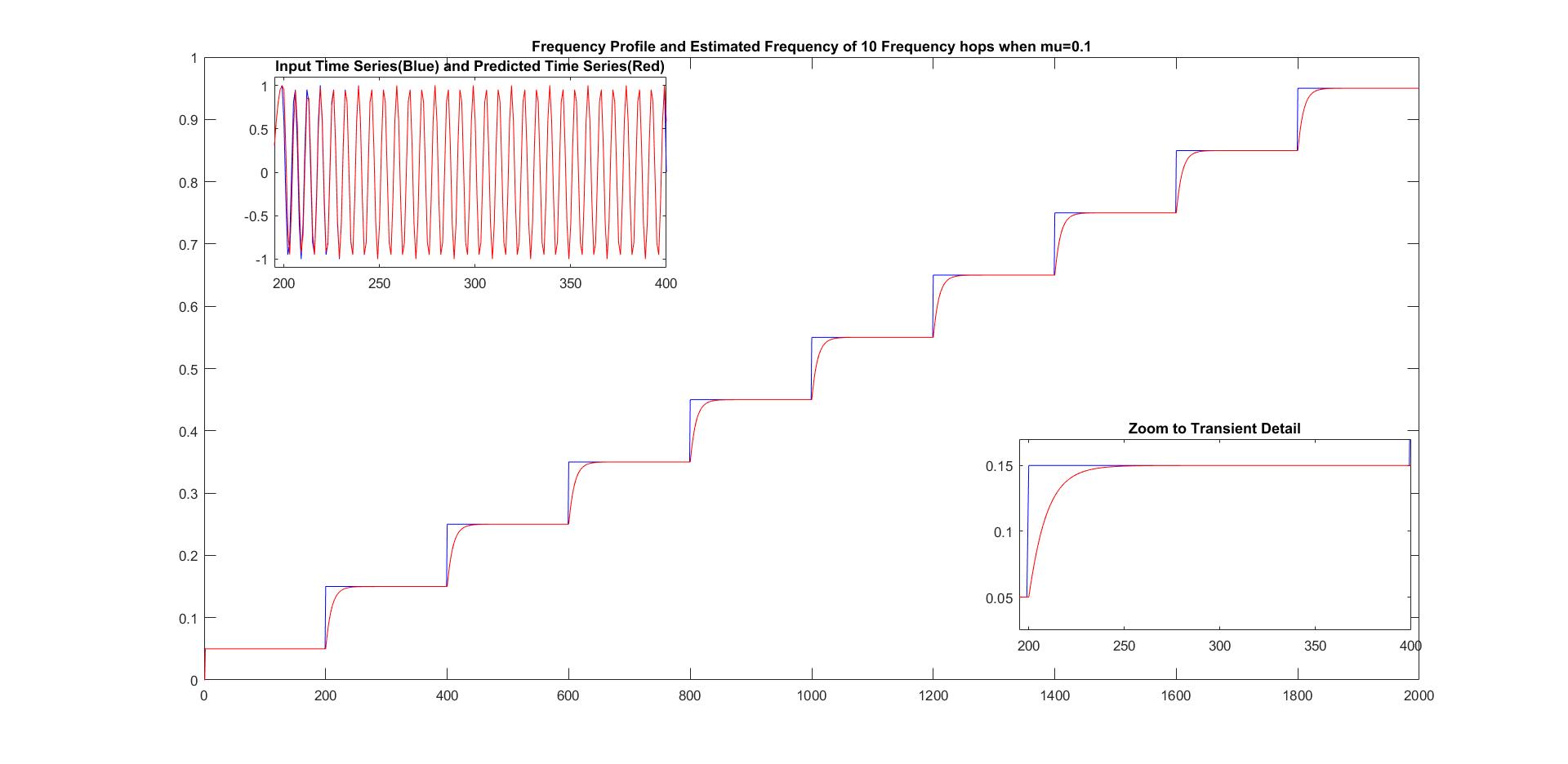
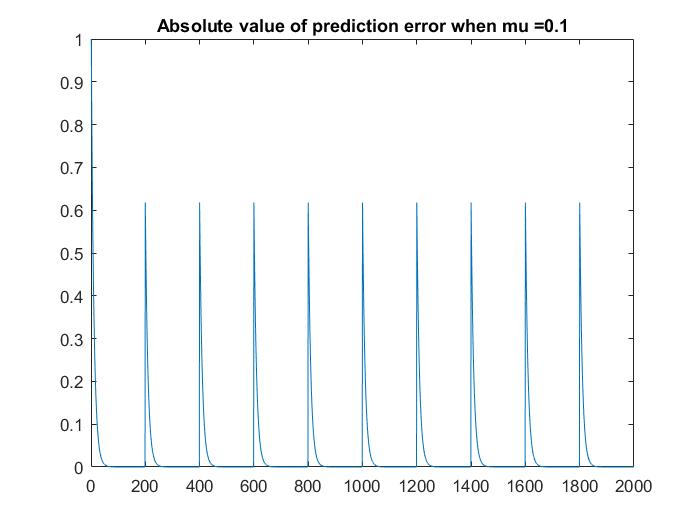
title(['Spectrum Freq(' num2str(i) ') = ' num2str(x1(k))])

i = i+1;

end

b)





To solve this problem, two functions were created. The first one, **line\_canceller()** takes an input and a mu value, and returns vectors for weights, predictions, and errors. The code follows:

function [ws, ps, errs] = line\_canceller(x3, mu)

%line\_canceller One-tap adaptive line canceller

% Computes the adaptive line canceller algorithm for a sinusoid input.

% Returns the weights, predictions, and errors for each sample.

w = 0;

w\_new = 0;

input\_reg = 0;

w\_sv = zeros(1,length(x3));

p\_sv = zeros(1,length(x3));

err\_sv = zeros(1,length(x3));

for nn = 1:length(x3)

w = w\_new;

w\_sv(nn) = w;

p = conj(w)\*input\_reg;

p\_sv(nn) = p;

err = x3(nn) - p;

err\_sv(nn) = err;

w\_new = conj(err)\*input\_reg\*mu+w;

input\_reg=x3(nn);

end

ws = w\_sv;

ps = p\_sv;

errs = err\_sv;

end

The second function, **plot\_results(),** takes the results obtained from **line\_canceller()** and produces the plots that were required for the problem. The code follows:

function [null] = plot\_results(x1,x3,ws,ps,errs,mu,str)

%plot\_results Plots results of adaptive line canceller.

figure

pos = [-1 -1 2 2];

unitcircle =rectangle('Position',pos,'Curvature',[1 1]);

set(unitcircle,'edgecolor','b');

axis equal

grid on

hold on

plot(ws,'rx')

title(['Poles closing in on frequency when mu =' num2str(mu) str])

figure

plot(0:length(x1)-1,x1, 'b')

hold on

plot(0:length(ws)-1,unwrap(angle(ws))/(-2\*pi),'r')

title(['Frequency Profile and Estimated Frequency of 10 Frequency hops when mu=' num2str(mu) str])

axes('position',[0.65 0.175 0.25 0.25])

box on

plot(195:400,x1(196:401),'b')

hold on

plot(195:400,unwrap(angle(ws(196:401)))/(-2\*pi),'r')

xlim([195 400])

ylim([0.025 0.17])

title('Zoom to Transient Detail')

axes('position',[0.175 0.65 0.25 0.25])

box on

plot(195:400,x3(196:401),'b')

hold on

plot(195:400,ps(196:401),'r')

xlim([195 400])

ylim([-1.1 1.1])

title('Input Time Series(Blue) and Predicted Time Series(Red)')

figure

plot(0:length(errs)-1,abs(errs))

title(['Absolute value of prediction error when mu =' num2str(mu) str])

end

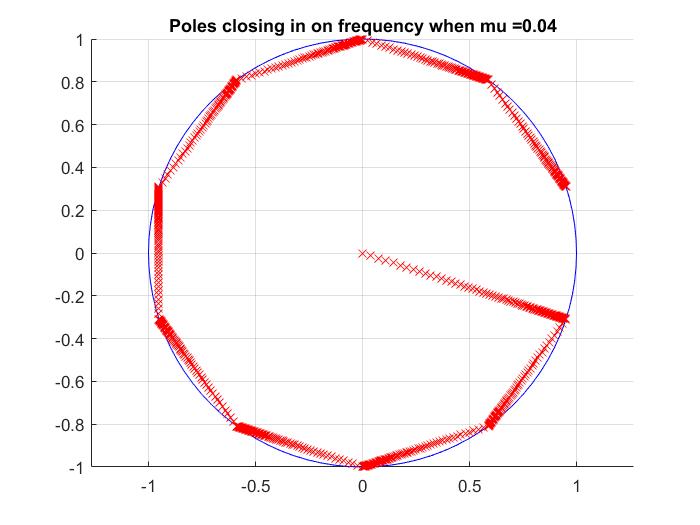
And this is the script which runs these functions with the parameters given in the problem.

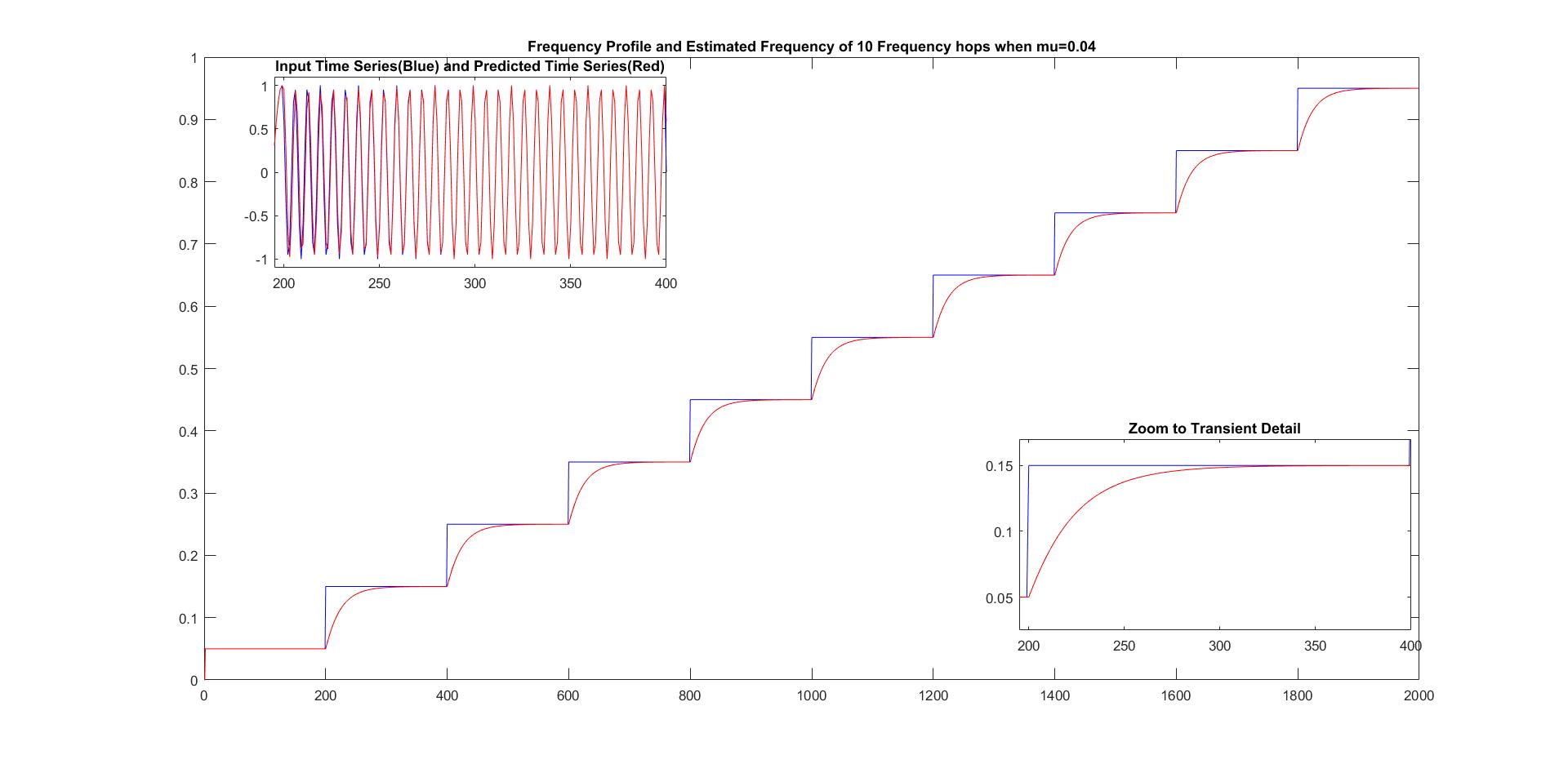
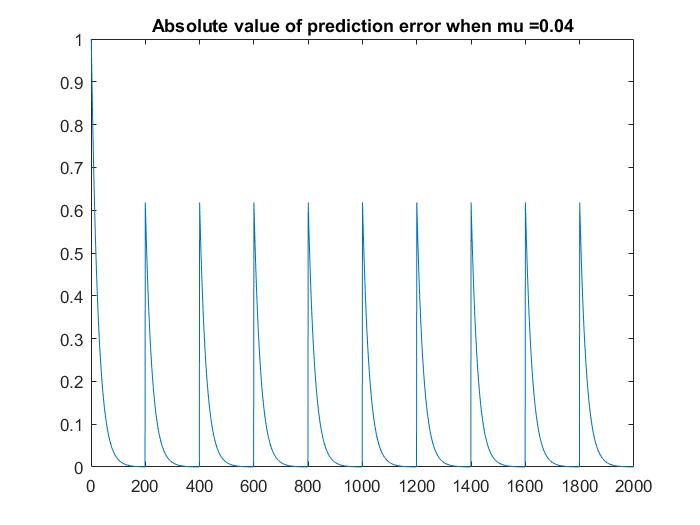
mu = 0.1;

[ws ps errs] = line\_canceller(x3, mu);

plot\_results(x1,x3,ws,ps,errs,mu,'')

c)





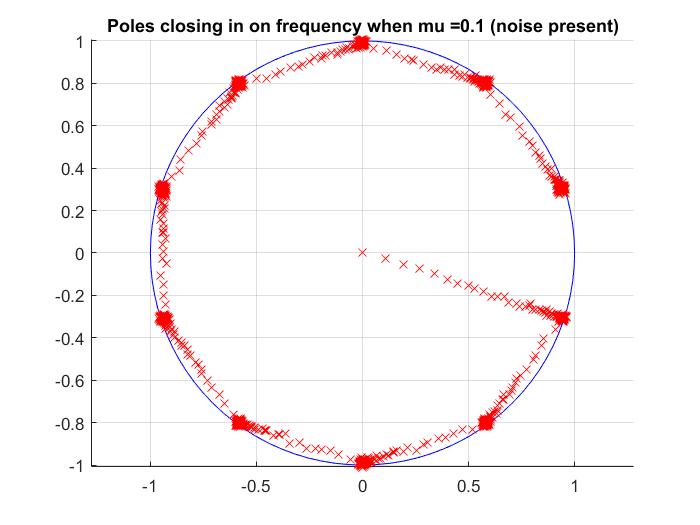
For this task, the same functions as in b) were used, but with different parameters.

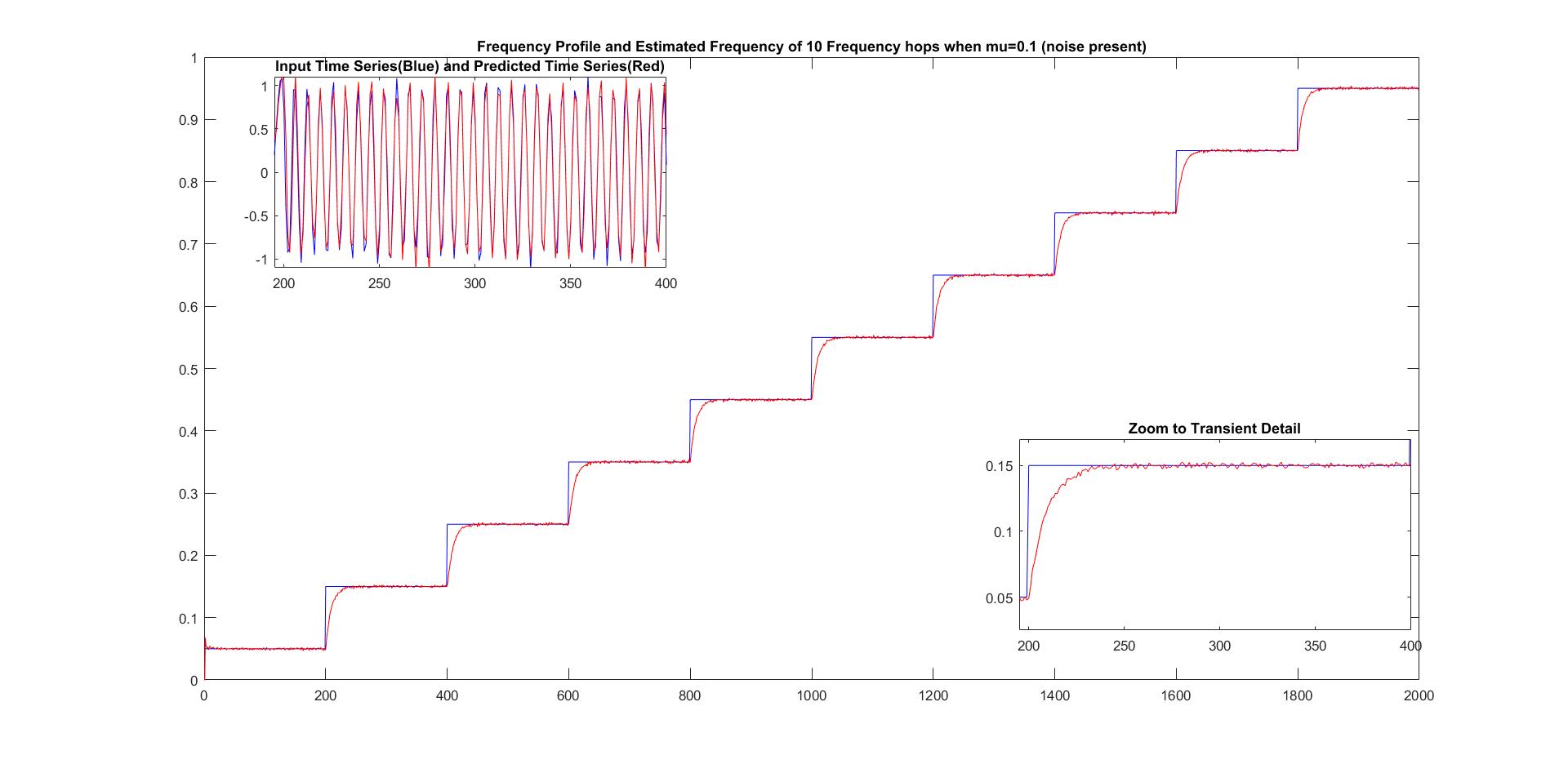
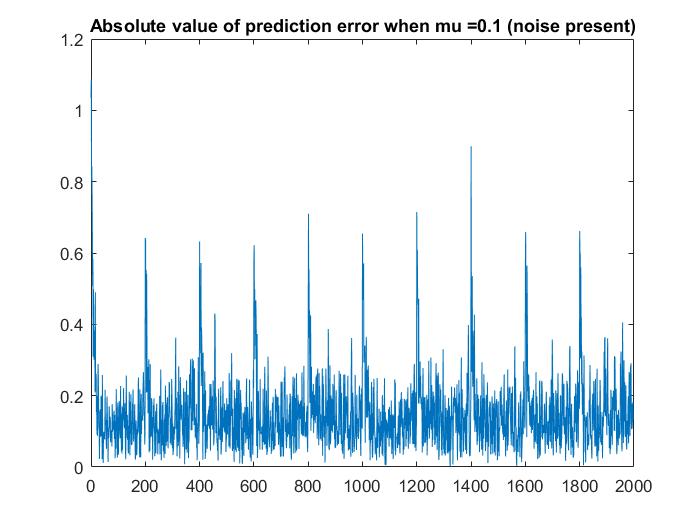
mu = 0.04;

[ws ps errs] = line\_canceller(x3, mu);

plot\_results(x1,x3,ws,ps,errs,mu,'')

d)





For this task, the same functions as in b) were used, but with different parameters.

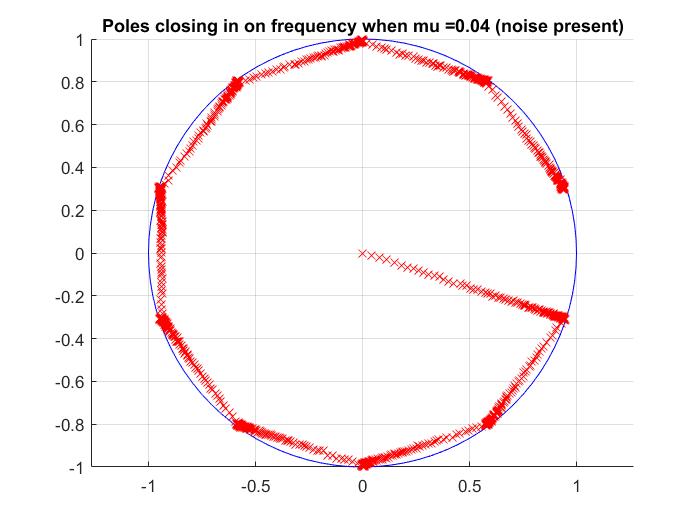
X4=x3+0.1\*(randn(1,2000)+j\*randn(1,2000))/sqrt(2);

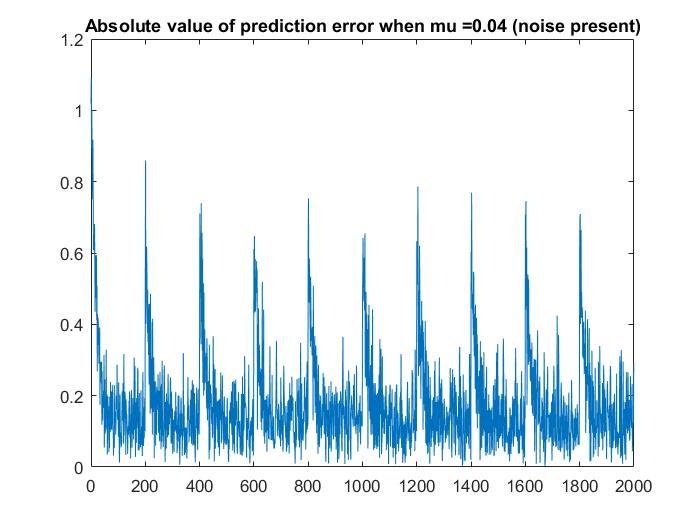
mu = 0.1;

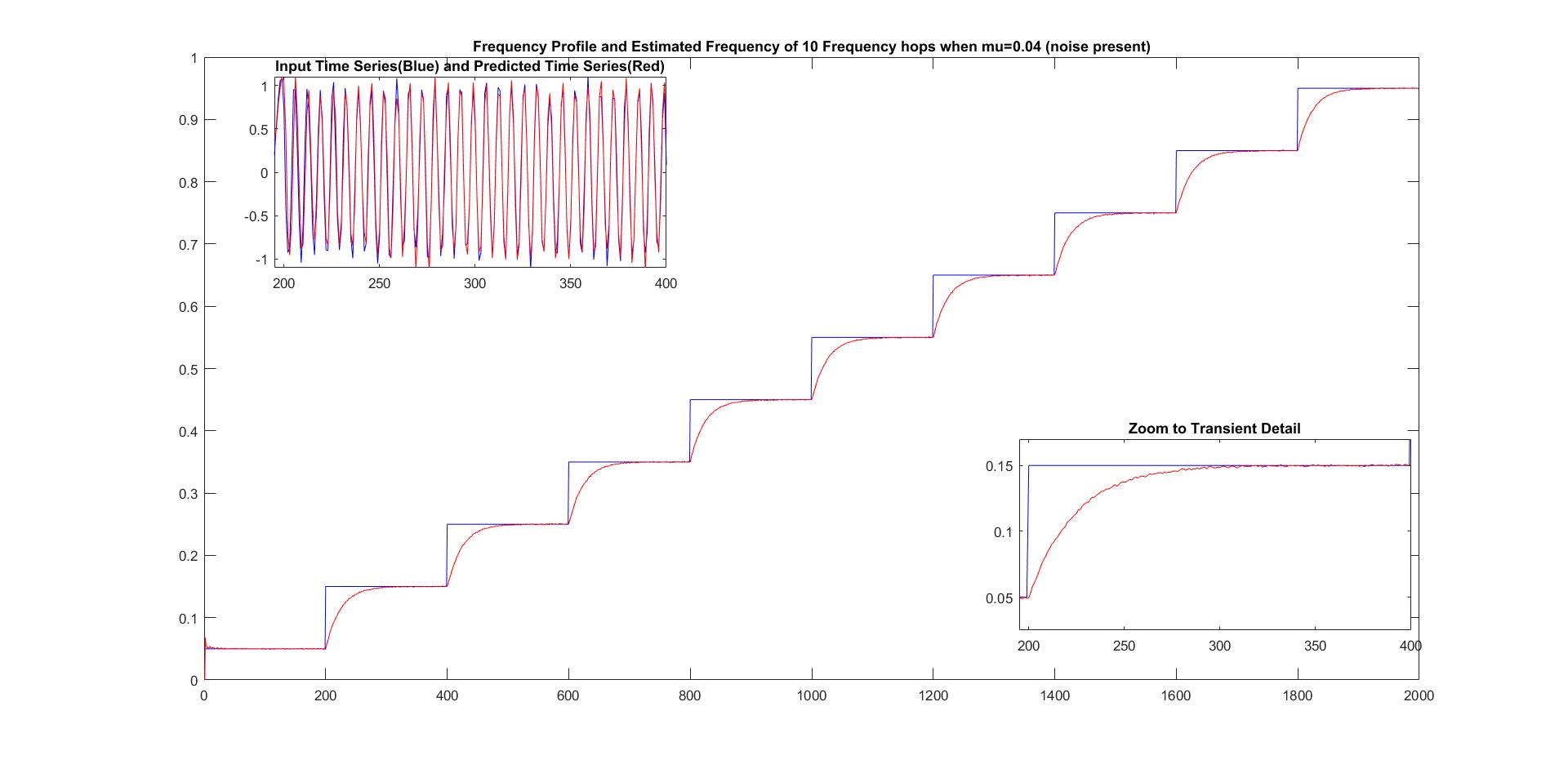
[ws ps errs] = line\_canceller(X4, mu);

plot\_results(x1,x4,ws,ps,errs,mu,' (noise present)')

e)







For this task, the same functions as in b) were used, but with different parameters.

mu = 0.04;

[ws ps errs] = line\_canceller(X4, mu);

plot\_results(x1,X4,ws,ps,errs,mu,' (noise present)')