Noah Choe

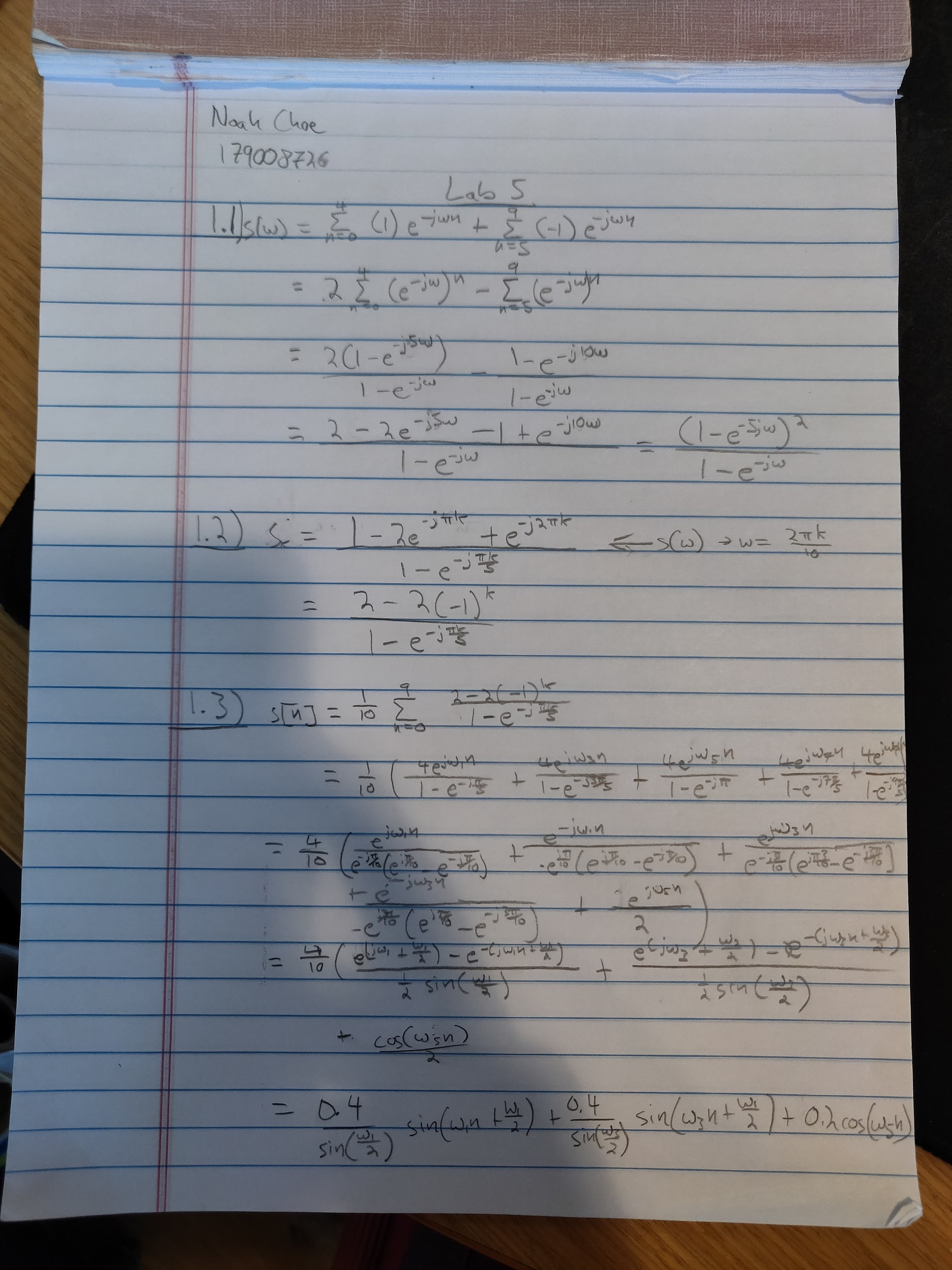
179008726

Lab 5

Problem 1:

The purpose of this problem is to see the relationship between DFT and DTFT. We did this by analytically solving the DTFT and DFT of a period sine wave and showing the results through matlab calculations and graphs.

Problem 1.1 - 1.3 (analytical):



Problem 1.3 (verification):

w1 = 2\*pi/10;

w3 = 6\*pi/10;

w5 = pi;

s =@(n) (0.4/(sin(w1/2)))\*sin(w1.\*n+w1/2) + (0.4/(sin(w3/2)))\*sin(w3.\*n+w3/2) + 0.2\*cos(w5.\*n);

n = 0:9;

display(s(n));

s(n) =

1.0000 1.0000 1.0000 1.0000 1.0000 -1.0000 -1.0000 -1.0000 -1.0000 -1.0000

Problem 1.4:

Swfreqz =@(w) (1-exp(-5\*1i.\*w)).^2./(1-exp(-1i.\*w));

w = linspace(0, 2\*pi, 201);

Sk =@(k) 2\*(1-(-1).^k)./(1-exp(-1i\*pi.\*k./5));

k = 1:9;

dft = [0, abs(Sk(k))];

k=0:9;

figure;

plot(w/pi, abs(Swfreqz(w)))

hold on;

plot((2\*k)/10, dft, 'r.');

xlim([0 2]);

ylim([0 8]);

title('DTFT and DFT');

hold off;

figure;

plot((10\*w)./(2\*pi), abs(Swfreqz(w)));

hold on;

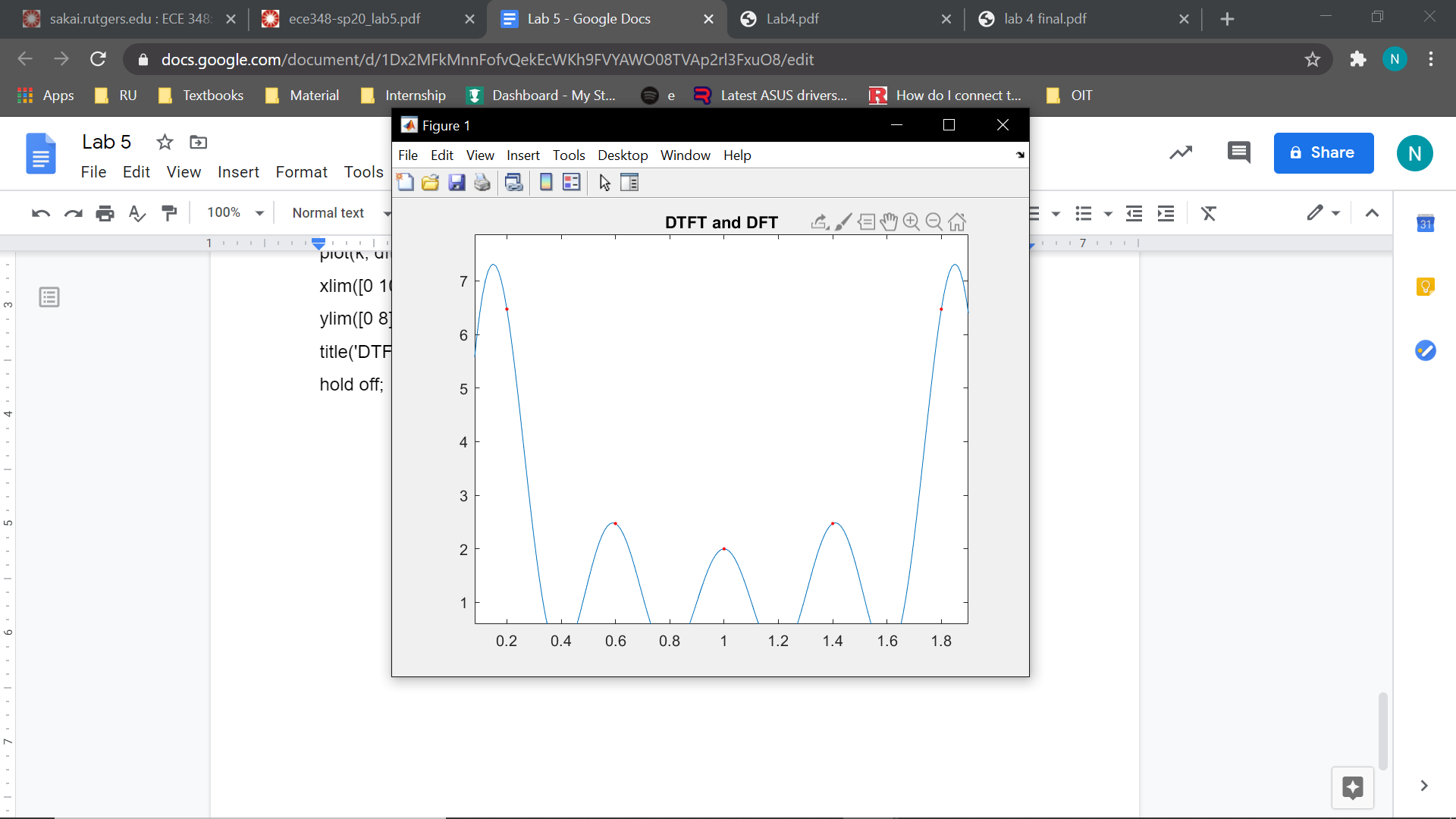
plot(k, dft, 'r.');

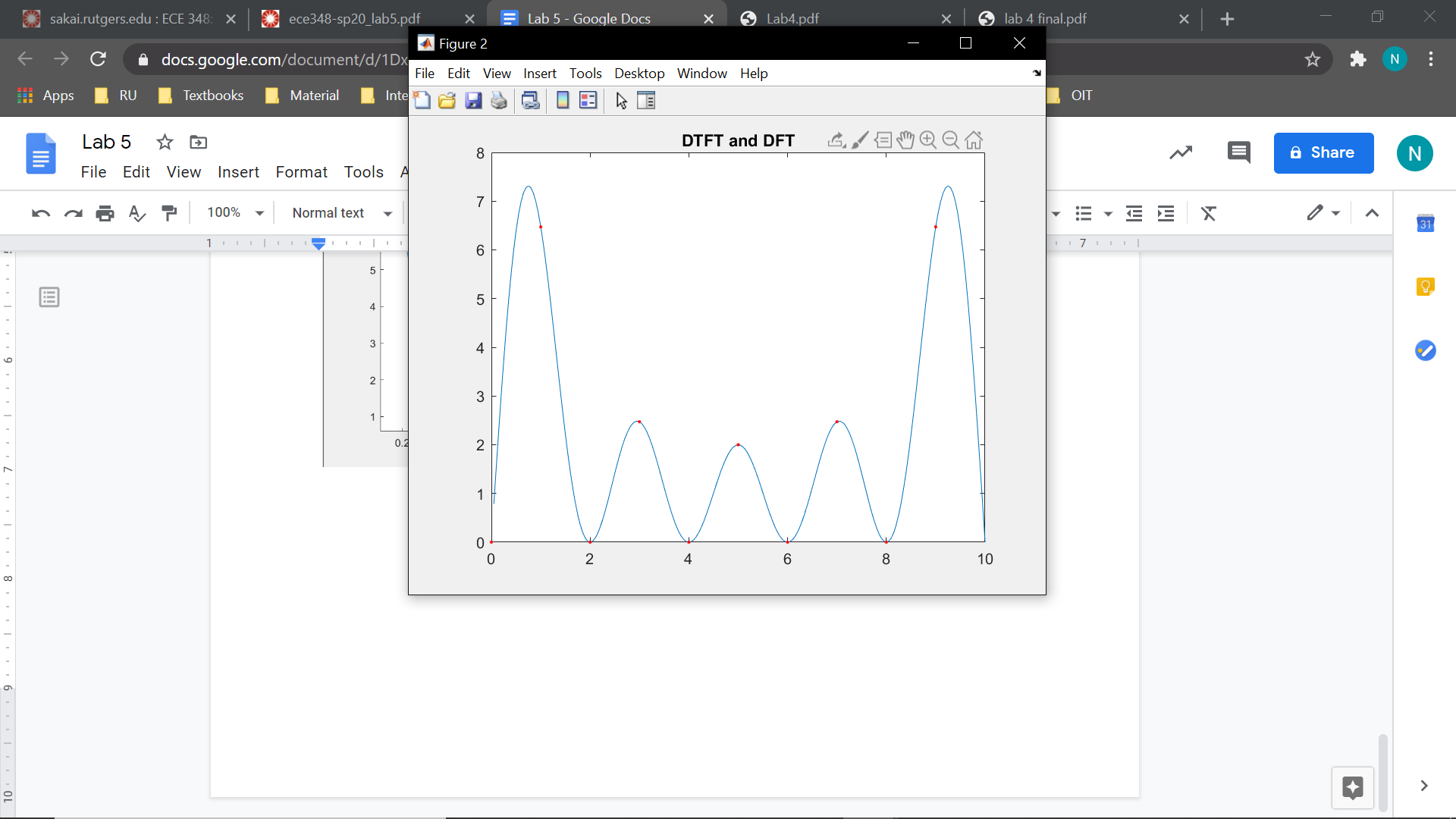
xlim([0 10]);

ylim([0 8]);

title('DTFT and DFT');

hold off;





The values at the even omegas are zero while the values at the odd omegas are nonzero which is why the odd omegas exist while the even omegas do not exist.

Problem 1.5:

s =@(n) (0.4/(sin(w1/2)))\*sin(w1.\*n+w1/2) + (0.4/(sin(w3/2)))\*sin(w3.\*n+w3/2) + 0.2\*cos(w5.\*n);

n = 0:9;

Sw =@(w) (1-exp(-5\*1i.\*w)).^2./(1-exp(-1i.\*w));

w = linspace(0, 2\*pi, 201);

Sk =@(k) 2\*(1-(-1).^k)./(1-exp(-1i\*pi.\*k./5));

k = 1:9;

Swfreqz = freqz(s(n), 1, w);

dtft = Sw(w);

dtft(1) = 0;

error = norm(Swfreqz - dtft);

disp(error);

dtffft = fft(s(n), 10);

dft = [0, Sk(k)];

error = norm(dft - dtffft);

disp(error);

Error =

7.8257e-14

Error =

3.4848e-15

Problem 2:

The purpose of this problem is to do spectral analysis using DFT. The real-valued signal consisting of the sum of two sinusoids of unknown frequencies and unknown amplitudes and phases was sampled at a rate of 10 kHz and 128 samples were collected. Next, 128-point DFT of the collected samples was computed and saved (as a complex-valued array) in the attached MAT file, X.mat. Using this data we compared estimated vs actual signal.

Problem 2.1:

load X;

Xabs = abs(X);

figure;

stem(Xabs);

[Xabs, k] = sort(Xabs, 'descend');

k1 = k(1)-1;

k2 = k(2)-1;

k3 = k(3)-1;

k4 = k(4)-1;

display(k1);

display(k2);

display(k3);

display(k4);

n = 128;

fs = 10000;

f1 = k1\*fs/n;

f2 = (k2-n)\*fs/n;

f3 = k3\*fs/n;

f4 = (k4-n)\*fs/n;

display(f1);

display(f2);

display(f3);

display(f4);

k1 =

13

k2 =

115

k3 =

26

k4 =

102

f1 =

1.0156e+03

f2 =

-1.0156e+03

f3 =

2.0313e+03

f4 =

-2.0313e+03

The frequencies in the second half are the opposite of those in the first half making them negative.

Problem 2.2:

load X;

x = ifft(X);

n = 0:127;

estimate1 =@(N) (1/128).\*(X(k(1)).\*exp(1i\*2\*pi\*k1.\*N./128)+X(k(2)).\*exp(1i\*2\*pi\*k2.\*N./128));

estimate2 =@(N) (1/128).\*(X(k(3)).\*exp(1i\*2\*pi\*k3.\*N./128)+X(k(4)).\*exp(1i\*2\*pi\*k4.\*N./128));

estimate3 =@(N) real(estimate1(N) + estimate2(N));

figure;

plot(n, estimate3(n), n, x);

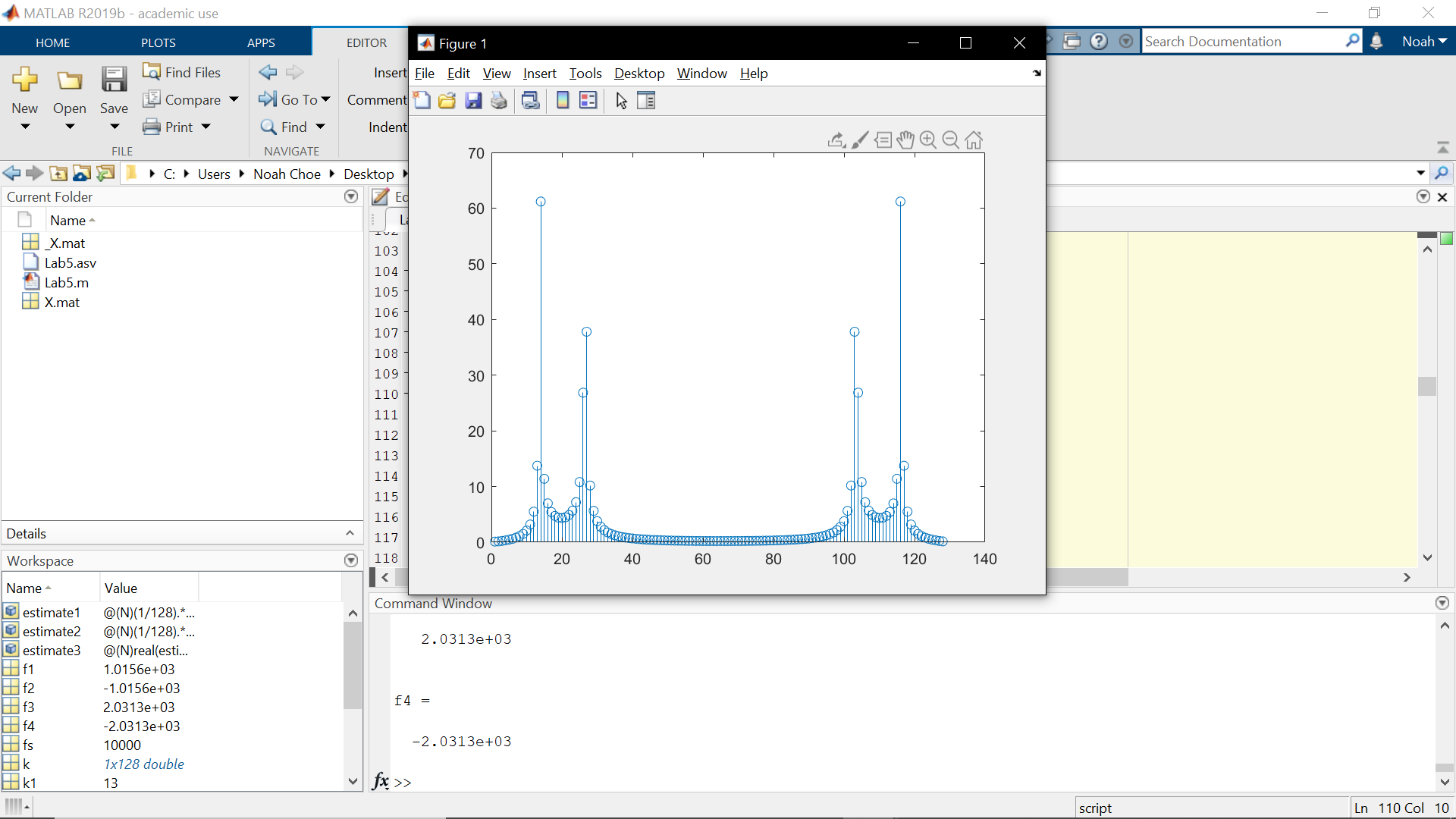
hold on;

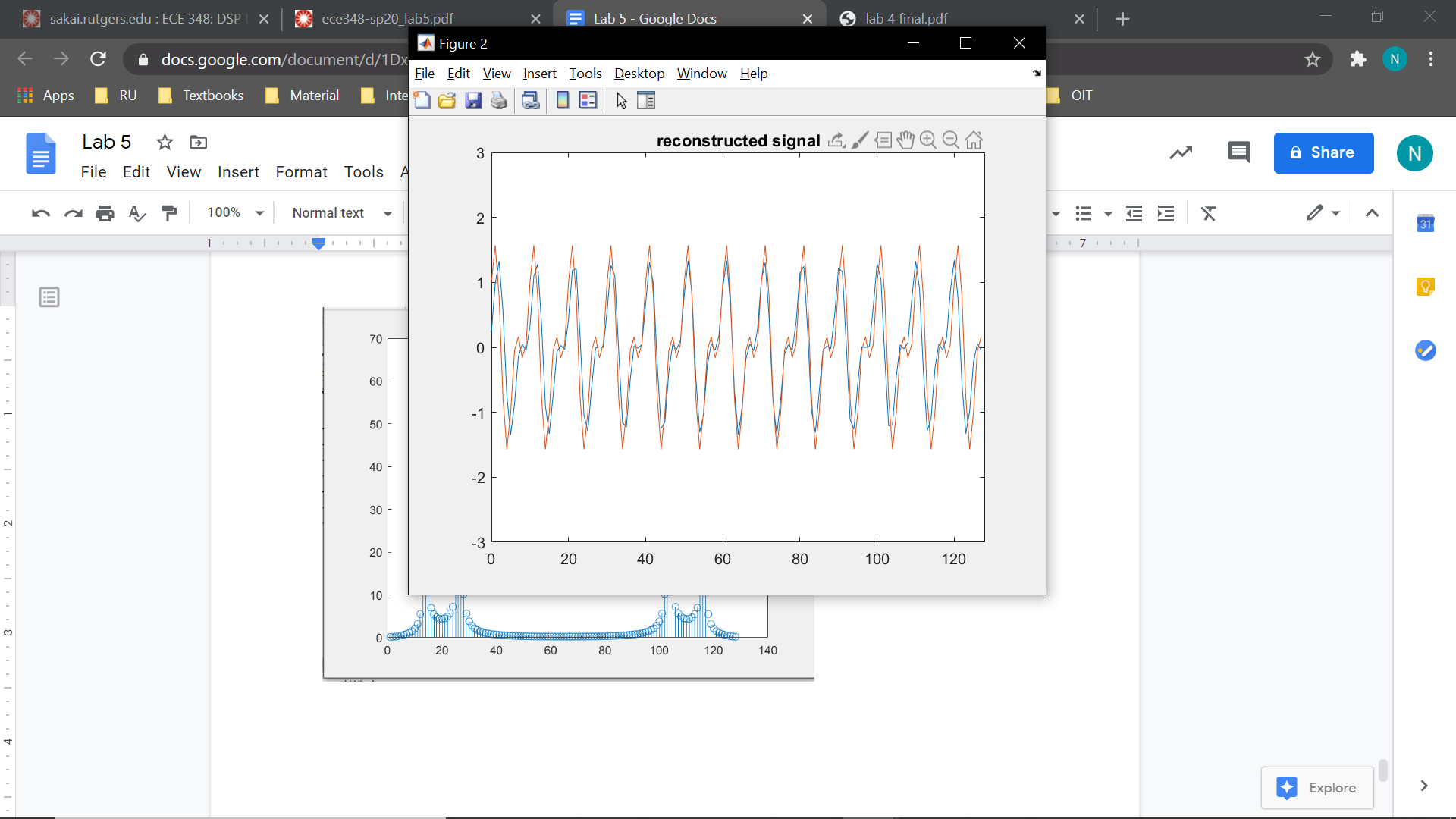
xlim([0 128]);

ylim([-3 3]);

title('reconstructed signal');

hold off;





Problem 3:

The purpose of this problem is to compare the computation speed of FFT and DFT. The FFT was faster than the DFT and the error between the two was small.

Problem 3.1:

dftmat = @(N) exp(-2\*pi\*1i\*(0:N-1)'\*(0:N-1)/N);

Problem 3.2:

%512

x = randn(1, 512);

tic

A = dftmat(512);

Tmat0 = toc;

tic

X = A\*x';

Tdft0 = toc;

tic

dft = fft(x, 512);

Tfft0 = toc;

Error0 = norm(abs(X')-abs(dft));

%1024

x = randn(1, 1024);

tic

A = dftmat(1024);

Tmat1 = toc;

tic

X = A\*x';

Tdft1 = toc;

tic

dft = fft(x, 1024);

Tfft1 = toc;

Error1 = norm(abs(X')-abs(dft));

%2048

x = randn(1, 2048);

tic

A = dftmat(2048);

Tmat2 = toc;

tic

X = A\*x';

Tdft2 = toc;

tic

dft = fft(x, 2048);

Tfft2 = toc;

Error2 = norm(abs(X')-abs(dft));

%4096

x = randn(1, 4096);

tic

A = dftmat(4096);

Tmat3 = toc;

tic

X = A\*x';

Tdft3 = toc;

tic

dft = fft(x, 4096);

Tfft3 = toc;

Error3 = norm(abs(X')-abs(dft));

N = [512, 1024, 2048, 4096];

Tmat = [Tmat0 Tmat1 Tmat2 Tmat3];

Tdft = [Tdft0 Tdft1 Tdft2 Tdft3];

Tfft = [Tfft0 Tfft1 Tfft2 Tfft3];

error = [Error0 Error1 Error2 Error3];

table = [N; Tmat; Tdft; Tfft; error];

fprintf(' N Tmat Tdft Tfft Error\n');

fprintf('-------------------------------------------\n');

fprintf('%4i %6.6f %6.6f %6.6f %6.4e\n', table);

N Tmat Tdft Tfft Error

-------------------------------------------

512 0.048992 0.001140 0.000866 2.2508e-11

1024 0.146872 0.001328 0.000352 8.8162e-11

2048 0.215856 0.003721 0.000278 3.5023e-10

4096 0.809806 0.014529 0.000263 1.4056e-09

Problem 4:

The purpose of this problem is to test filtering of signals using DFT. We did so by: 1) computing FFT of one input period

2) evaluating filter at DFT frequencies

3) element-wise multiplication of FFT and filter

4) computing inverse FFT of step 3

Problem 4.1:

s = [3, 6, 3];

Hz = @(z) (2+z.^-1)./(1+0.5\*z.^-3);

b = [2, 1];

a = [1, 0, 0, .5];

k = 0:2;

periodic\_output = @(b, a, s) ifft((fft(s, 3)).\*Hz(exp(1i.\*(2\*pi.\*k/3))), 3);

s\_out = periodic\_output(b, a, s);

display(s\_out);

s\_out =

6.0000 + 0.0000i 10.0000 - 0.0000i 8.0000 + 0.0000i

Problem 4.2:

s = [1, 1, 1, 1, -1, -1, -1, -1];

H = @(z) (.3).\*((1+z.^-1+z.^-2)./(1+(.5\*z.^-4)));

b = [2, 1];

a = [1, 0, 0, .5];

k = 0:7;

periodic\_output = @(b, a, s) ifft((fft(s, 8)).\*H(exp(1i.\*(2\*pi.\*k/8))), 8);

s\_out = periodic\_output(b, a, s);

display(s\_out);

s\_out =

Columns 1 through 6

-0.6000 - 0.0000i 0.6000 + 0.0000i 1.8000 + 0.0000i 1.8000 + 0.0000i

0.6000 + 0.0000i -0.6000 - 0.0000i

Columns 7 through 8

-1.8000 - 0.0000i -1.8000 - 0.0000i

Problem 4.3:

s = [1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1];

b = 0.3.\*[1, 1, 1];

a = [1, 0, 0, 0, 0.5];

y = filter(b,a,s);

n = 0:(length(y)-1);

figure;

stem(n, s);

hold on;

xlim([0 30]);

ylim([-2 2]);

title('input');

hold off;

figure;

stem(n, y);

hold on;

xlim([0 40]);

ylim([-2 2]);

title('output');

hold off;

