DFT Octave Codes (0B)

Young Won Lim
4/15/17

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Based on
M.J. Roberts, Fundamentals of Signals and Systems
S.K. Mitra, Digital Signal Processing : a computer-based approach 2^{nd} ed
S.D. Stearns, Digital Signal Processing with Examples in MATLAB

B.D Storey, Computing Fourier Series and Power Spectrum with MATLAB
B Ninness, Spectral Analysis using the FFT
U of Rhode Island, ELE 436, FFT Tutorial

fft(x)

fft (**x**)

- Compute the discrete Fourier transform of **x** using a Fast Fourier Transform (FFT) algorithm.
- The FFT is calculated along the first non-singleton dimension of the array.
- if **x** is a matrix, fft (**x**) computes the FFT for each column of **x**.

U of Rhode Island, ELE 436, FFT Tutorial

fft(x, n)

fft (**x**, n)

- If called with two arguments,
  n is expected to be an integer
  specifying the number of elements of **x** to use,
  or an empty matrix to specify
  that its value should be ignored.

- If n is larger than the dimension (the number of data)
  along which the FFT is calculated,
  then **x** is resized and padded with zeros.

- If n is smaller than the dimension (the number of data)
  along which the FFT is calculated,
  then **x** is truncated.

U of Rhode Island, ELE 436, FFT Tutorial

fft(x, n, dim)

fft (**x**, n, dim)

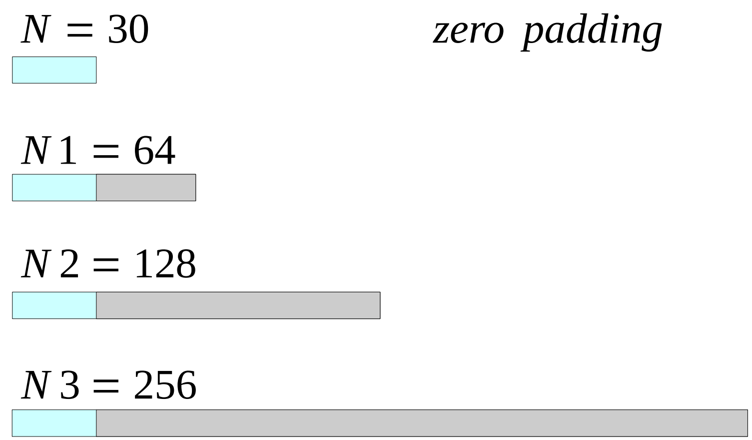
- If called with three arguments, dim is an integer specifying the dimension of the matrix along which the FFT is performed

U of Rhode Island, ELE 436, FFT Tutorial

FFT of a cosine (N=64, 128, 256)

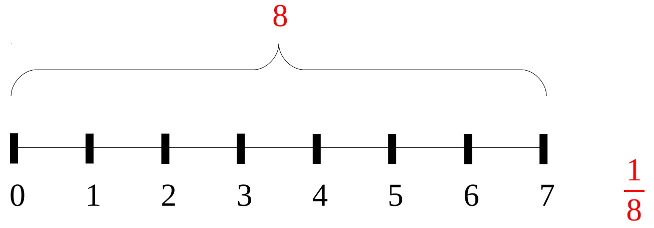
```
n= [0:29];
x= cos(2*pi*n/10);

N1= 64;
N2= 128;
N3= 256;

X1= abs(fft(x,N1));
X2= abs(fft(x,N2));
X3= abs(fft(x,N3));


F1= [0: N1-1]/N1;
F2= [0: N2-1]/N2;
F3= [0: N3-1]/N3;
```

$$\omega_0 = 2\pi f_0 = \frac{2\pi}{T_0} \qquad f_0 = 0.1 \qquad T_0 = 10$$



U of Rhode Island, ELE 436, FFT Tutorial

Linearly Spaced Elements




```
F1= [0: (N1-1)]/N1;
F2= [0: (N2-1)]/N2;
F3= [0: (N3-1)]/N3;

F1= 0 : 1/N1 : (N1-1)/N1;
F2= 0 : 1/N2 : (N2-1)/N2;
F3= 0 : 1/N3 : (N3-1)/N3;

F1= linspace(0, (N1-1)/N1, N1);
F2= linspace(0, (N2-1)/N2, N2);
F3= linspace(0, (N3-1)/N3, N3);
```

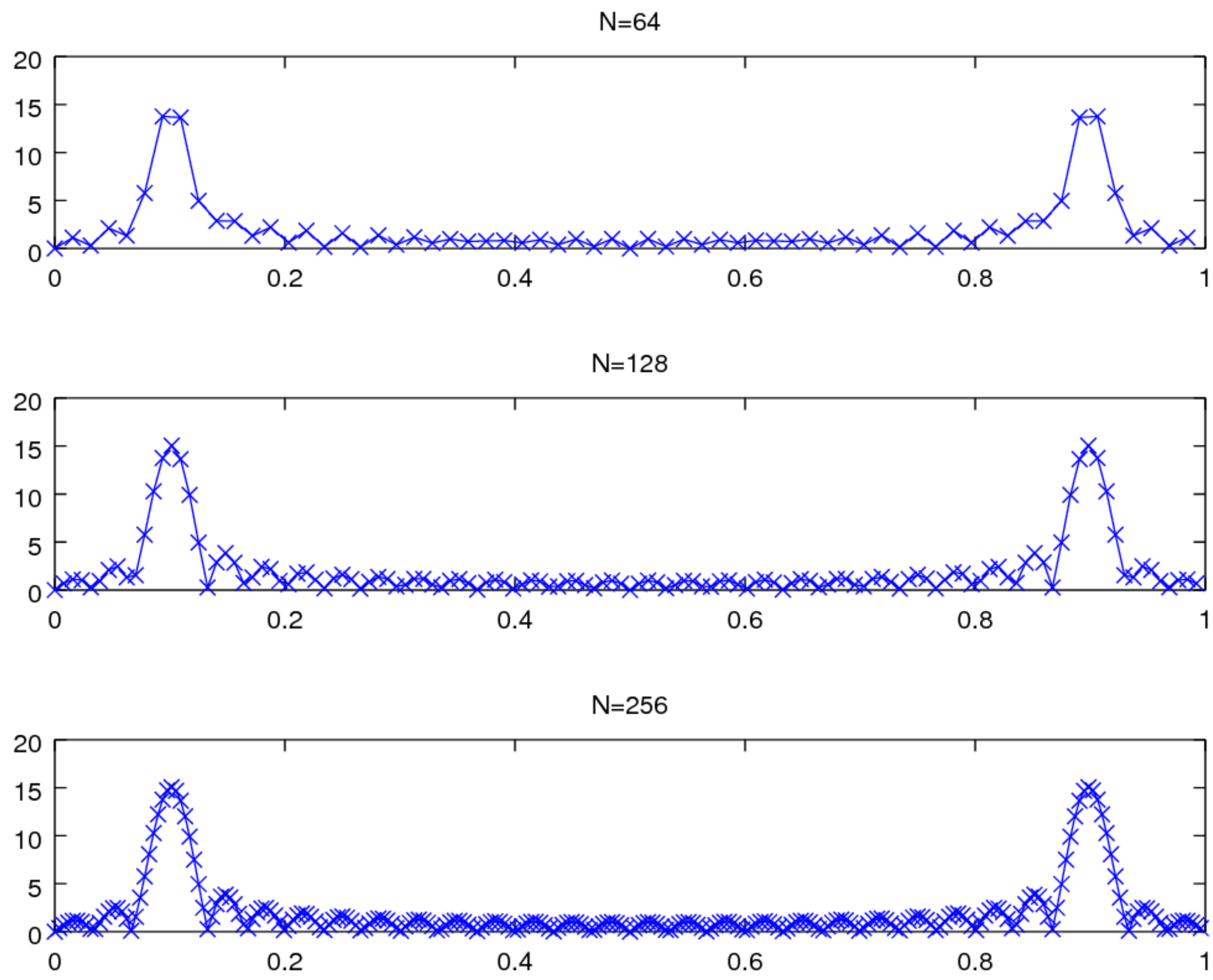
U of Rhode Island, ELE 436, FFT Tutorial

FFT of a cosine (N=64, 128, 256) – plot

```
subplot(3,1,1);
plot(F1, X1,'-x'), title('N=64'),  axis([0 1 0 20]);
subplot(3,1,2);
plot(F2, X2,'-x'), title('N=128'), axis([0 1 0 20]);
subplot(3,1,3);
plot(F3, X3,'-x'), title('N=256'), axis([0 1 0 20]);
```

U of Rhode Island, ELE 436, FFT Tutorial

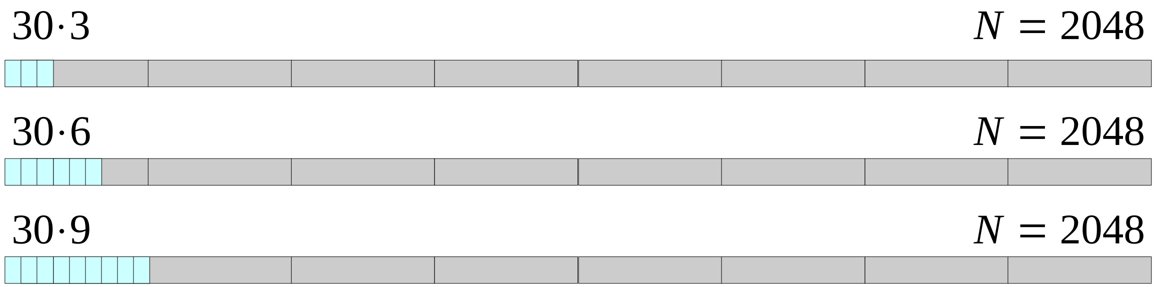
FFT of a cosine (N=64, 128, 256)– results



U of Rhode Island, ELE 436, FFT Tutorial

FFT of a cosine (3, 6, 9 periods)

```
n =  [0:29];
x1 = cos(2*pi*n/10);     % 3 periods
x2 = [x1 x1];            % 6 periods
x3 = [x1 x1 x1];         % 9 periods

N = 2048;

X1 = abs(fft(x1,N));
X2 = abs(fft(x2,N));
X3 = abs(fft(x3,N));

F = [0:N-1]/N;
```



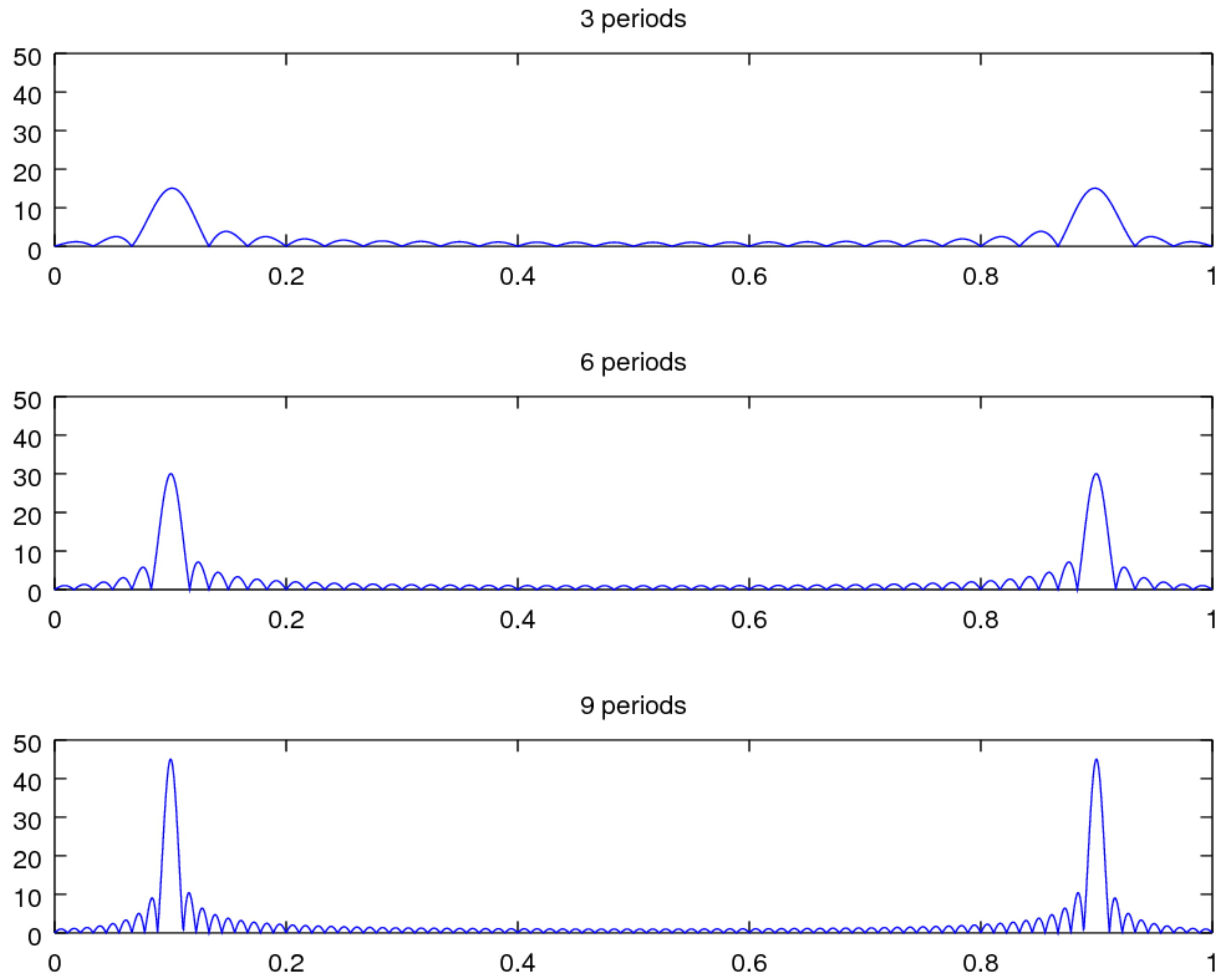
U of Rhode Island, ELE 436, FFT Tutorial

FFT of a cosine (3, 6, 9 periods) – plot

```
subplot(3,1,1);
plot(F, X1), title('3 periods'), axis([0 1 0 50]);
subplot(3,1,2);
plot(F, X2), title('6 periods'), axis([0 1 0 50]);
subplot(3,1,3);
plot(F, X3), title('9 periods'), axis([0 1 0 50]);
```

U of Rhode Island, ELE 436, FFT Tutorial

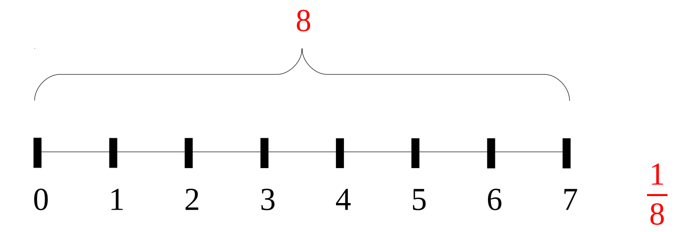
FFT of a cosine (3, 6, 9 periods) – results

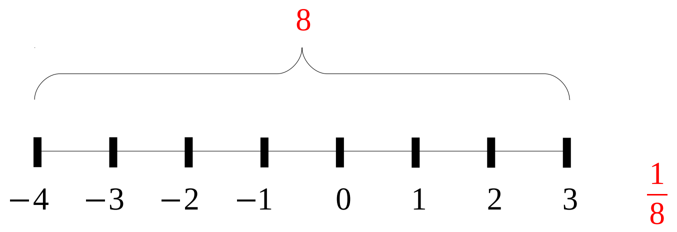


U of Rhode Island, ELE 436, FFT Tutorial

FFT Spectrum Analysis

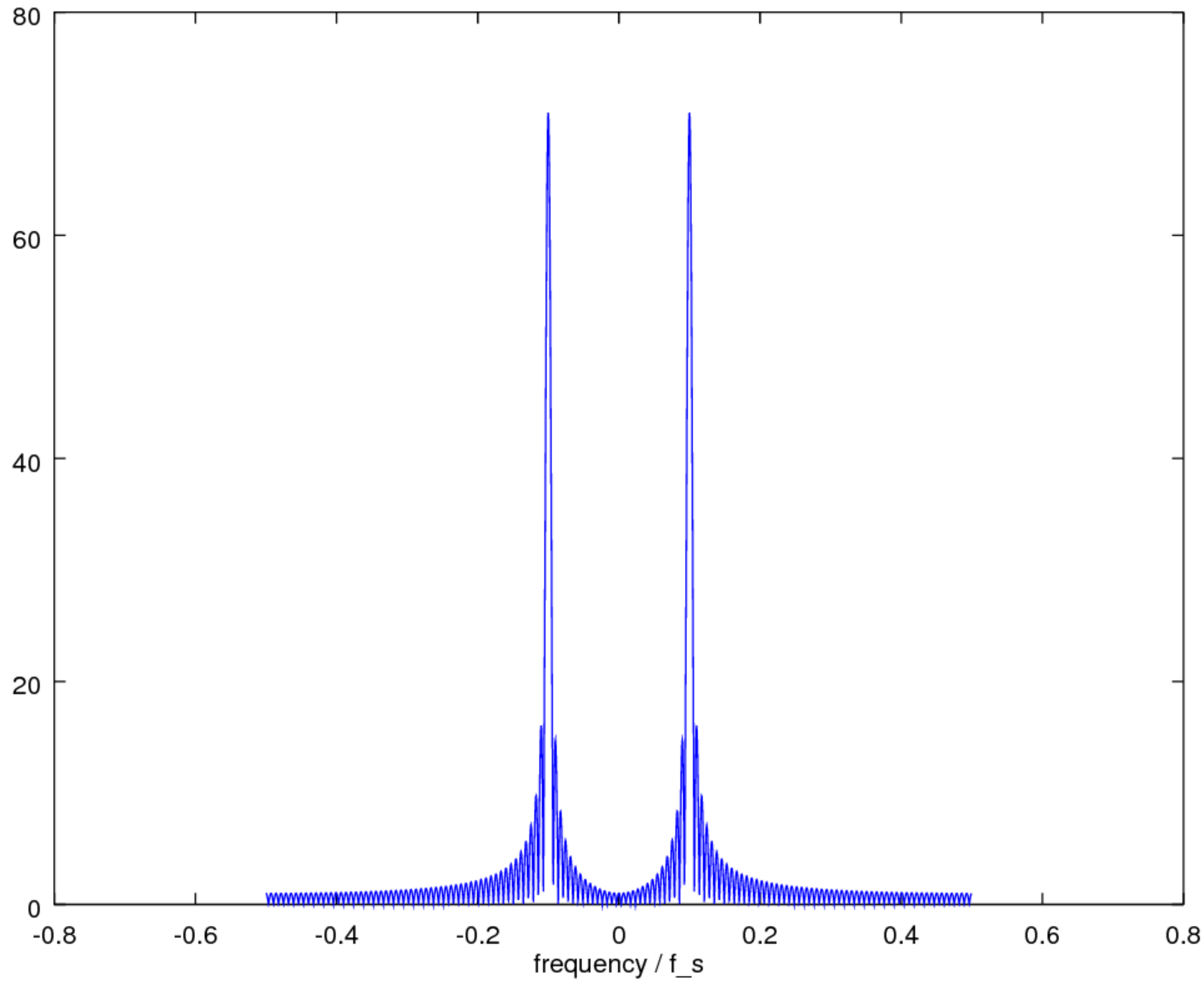
```
n = [0: 140];
x1= cos(2*pi*n/10);

N= 2048;

X = abs(fft(x1,N));
X = fftshift(X);

F = [-N/2:N/2-1]/N;

plot(F, X);
xlabel('frequency / f_s');
```





U of Rhode Island, ELE 436, FFT Tutorial

FFT Spectrum Analysis

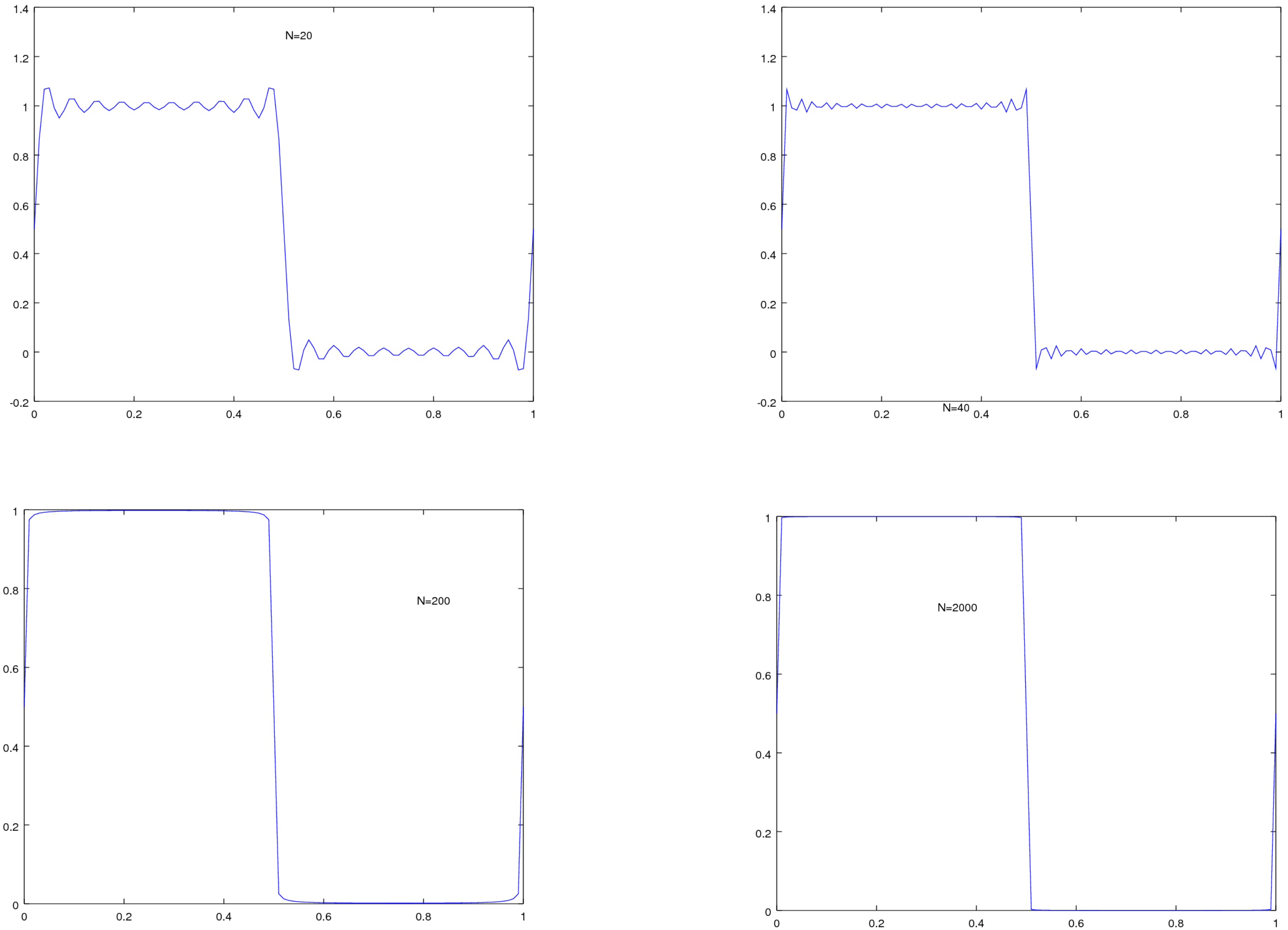


U of Rhode Island, ELE 436, FFT Tutorial

Normalized ω_s and ω_0

```
N = 200;
x = [0:100]/100;
f = ones(1,101)*1/2;
for i = 1:2:N
  a = 2/pi/i;
  f = f + a*sin(2*pi*i*x);
end

plot(x, f);
```

Normalized ω_s and ω_0



Normalized ω_s and ω_0

```
N = 8;
t = [0:N-1]'/N;
f = sin(2*pi*t);
p = abs(fft(f))/(N/2);
p = p(1:N/2).^2
```

Power Spectrum

```
N = 10000;
T = 3.4;
t = [0:N-1]/N;
t = t*T;

f = sin(2*pi*10*t);

p = abs(fft(f))/(N/2);

q = p(1:N/2).^2;
freq = [0:N/2-1]/T;

semilogy(freq,q);
axis([0 20 0 1]);
```

Normalized ω_s and ω_0

```
Fs = 44100;
y = wavrecord(5*Fs, Fs);
wavply(y, Fs);
```

DTFT Computation Example

```
k = input('the number of frequency points =');
num = input('the numerator coefficients =');
den = input('the denominator coefficients =');

w = 0 : pi/k : pi;
h = freqz(num, den, w);



% plot(w/pi, real(h));
% plot(w/pi, imag(h));
% plot(w/pi, abs(h));
% plot(w/pi, angle(h));
```

Mitra, Digital Signal Processing 2nd ed

DTFT Computation Example – plot real & imag

```
subplot(2,2,1)
plot(w/pi, real(h)); grid
title('real part');
xlabel('normalized angular frequency');
ylabel('Amplitude');

subplot(2,2,2)
plot(w/pi, imag(h)); grid
title('imaginary part');
xlabel('normalized angular frequency');
ylabel('Amplitude');
```

Mitra, Digital Signal Processing 2nd ed

DTFT Computation Example – plot mag & phase

```
subplot(2,2,3)
plot(w/pi, abs(h)); grid
title(‘magnitude spectrum’);
xlabel(‘normalized angular frequency’);
ylabel(‘Magnitude’);

subplot(2,2,4)
plot(w/pi, angle(h)); grid
title(‘phase spectrum’);
xlabel(‘normalized angular frequency’);
ylabel(‘phase, radians’);
```

Mitra, Digital Signal Processing 2nd ed

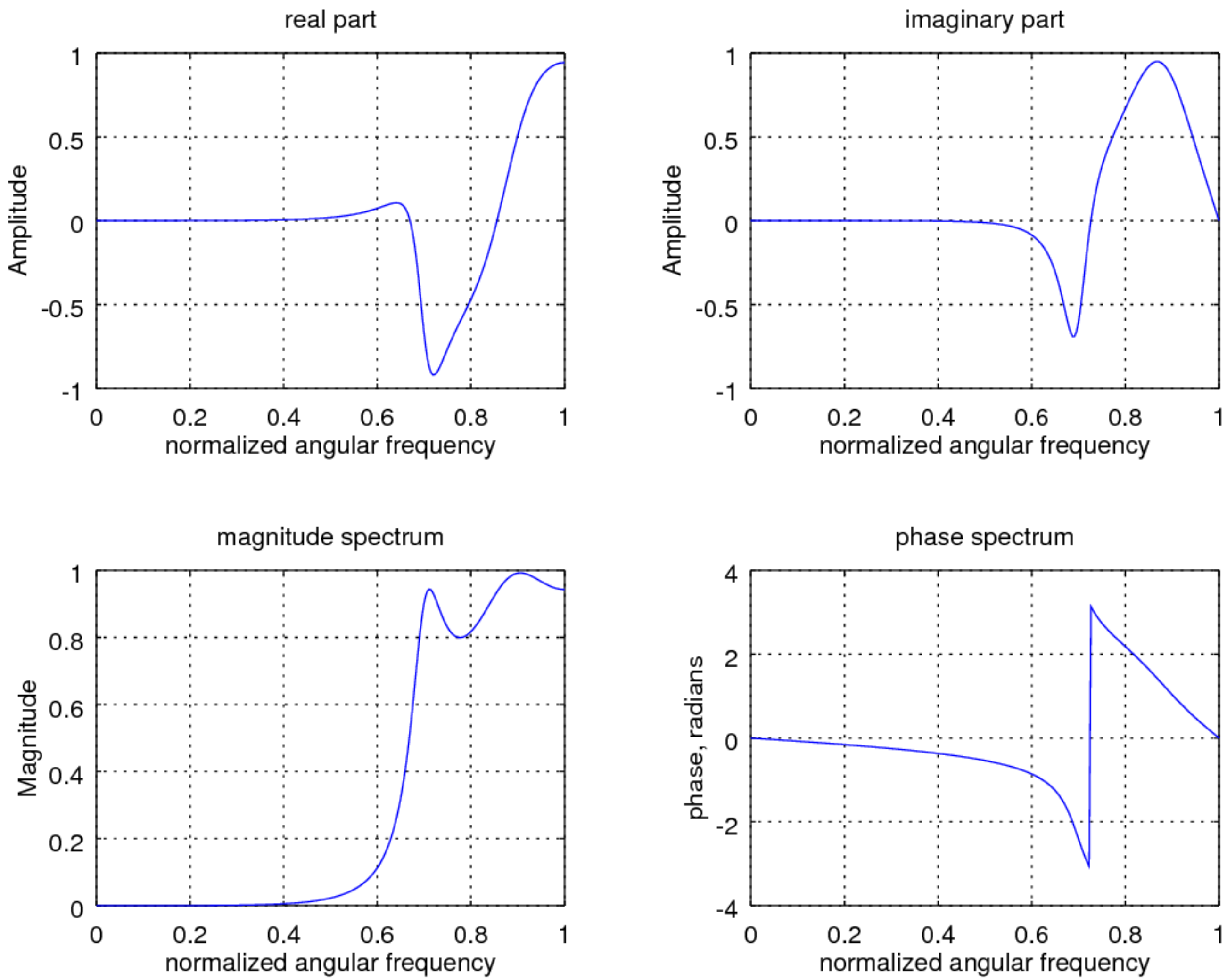
DTFT Computation Example – input data

```
k =256
Num = [0.008 -0.033 0.05 -0.033 0.008]
Den = [1 2.37 2.7 1.6 0.41]
```

$$H\left(e^{-j\omega}\right) = \frac{0.008 - 0.033\,e^{-j\omega} + 0.05\,e^{-j2\omega} - 0.033\,e^{-j3\omega}\,0.008\,e^{-j4\omega}}{1 + 2.37\,e^{-j\omega} + 2.7\,e^{-j2\omega} + 1.6\,e^{-j3\omega} + 0.41\,e^{-j4\omega}}$$

Mitra, Digital Signal Processing 2nd ed

DTFT Computation Example – resulting plots



Mitra, Digital Signal Processing 2nd ed

Rect FFT

```
N = input('the length of the sequence = ');
M = input('the length of the DFT = ');

u = [ones(1,N)];
U = fft(u, M);

% t = 0:1:N-1;
% k = 0:1:M-1;
```

Mitra, Digital Signal Processing 2nd ed

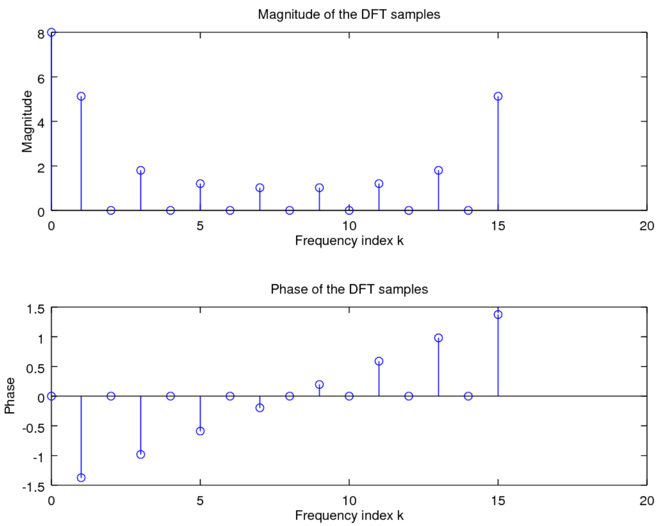
Rect FFT – plot

```
t = 0:1:N-1;
stem(t, u);
title('Original time domain sequence');
xlabel('Time index n'); ylabel('Amplitude');
Pause

subplot(2,1,1)
k = 0:1:M-1;
stem(k, abs(U));
title('Magnitude of the DFT samples');
xlabel('Frequency index k'); ylabel('Magnitude');

subplot(2,1,2)
k = 0:1:M-1;
stem(k, angle(U));
title('Phase of the DFT samples');
xlabel('Frequency index k'); ylabel('Phase');
```

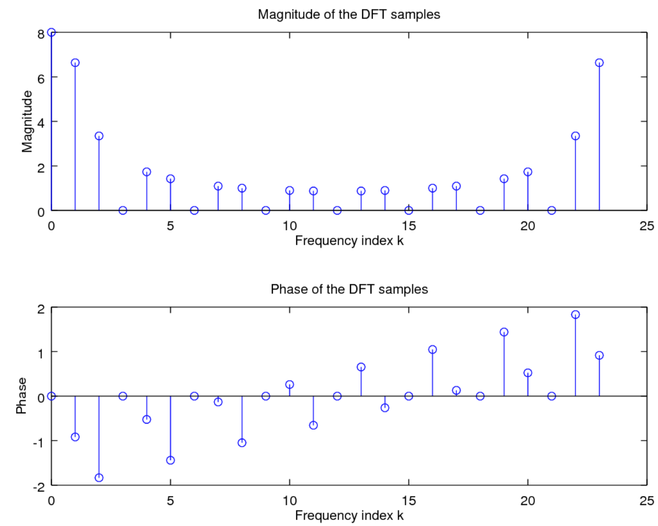
Mitra, Digital Signal Processing 2nd ed

Rect FFT – resulting plots

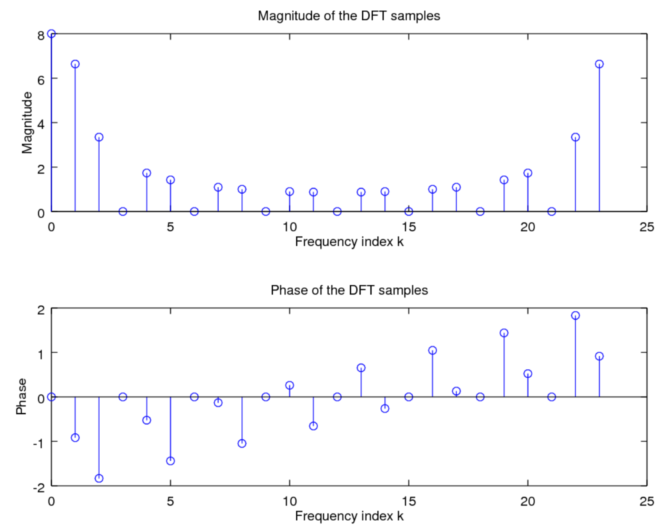
N=8
M=16



N=8
M=24



N=8
M=32



Mitra, Digital Signal Processing 2nd ed

Ramp IDFT

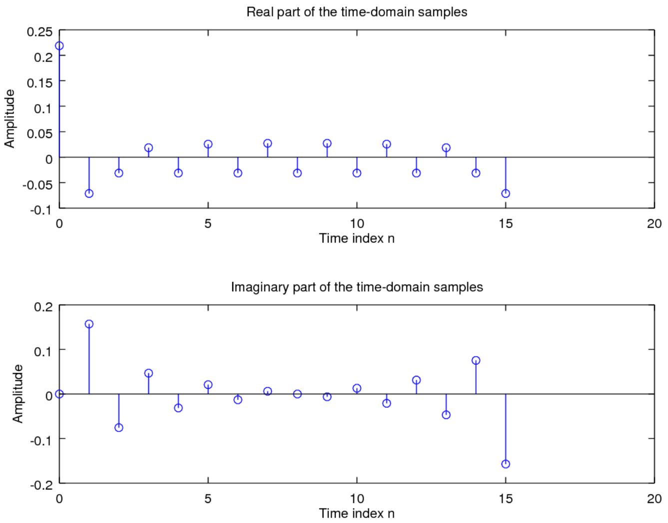
```
clear
clf

K= input('the length of the DFT = ');
N= input('the length of the IDFT = ');

k= 1:K;
U= (k-1)/K;

u= ifft(U, N);
```

Ramp IDFT – plot

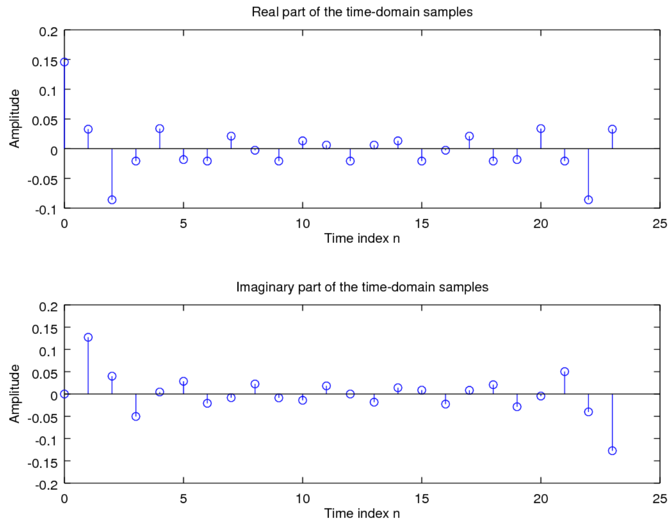
```
stem(k-1, U);
xlabel('Frequency index k');
ylabel('Amplitude');
title('Original DFT samples');
pause

subplot(2,1,1);
n= 0:1:N-1;
stem(n, real(u));
title('Real part of the time-domain samples');
xlabel('Time index n');
ylabel('Amplitude');

subplot(2,1,2);
n= 0:1:N-1;
stem(n, imag(u));
title('Imaginary part of the time-domain samples');
xlabel('Time index n');
ylabel('Amplitude');
```

Ramp IDFT – resulting plots

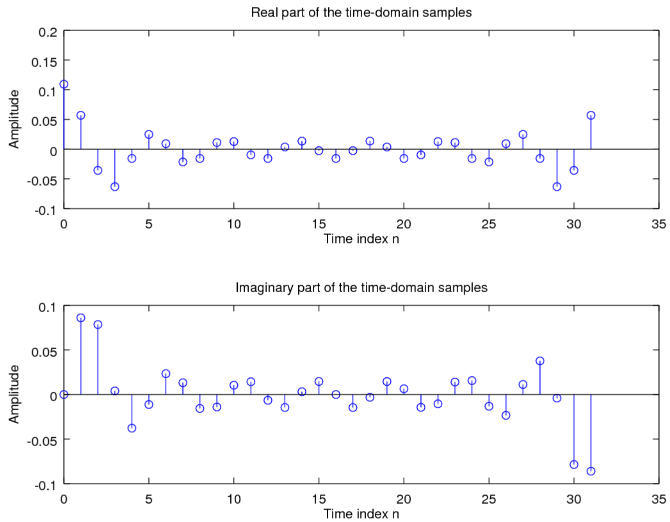
K=8
N=16



K=8
N=24



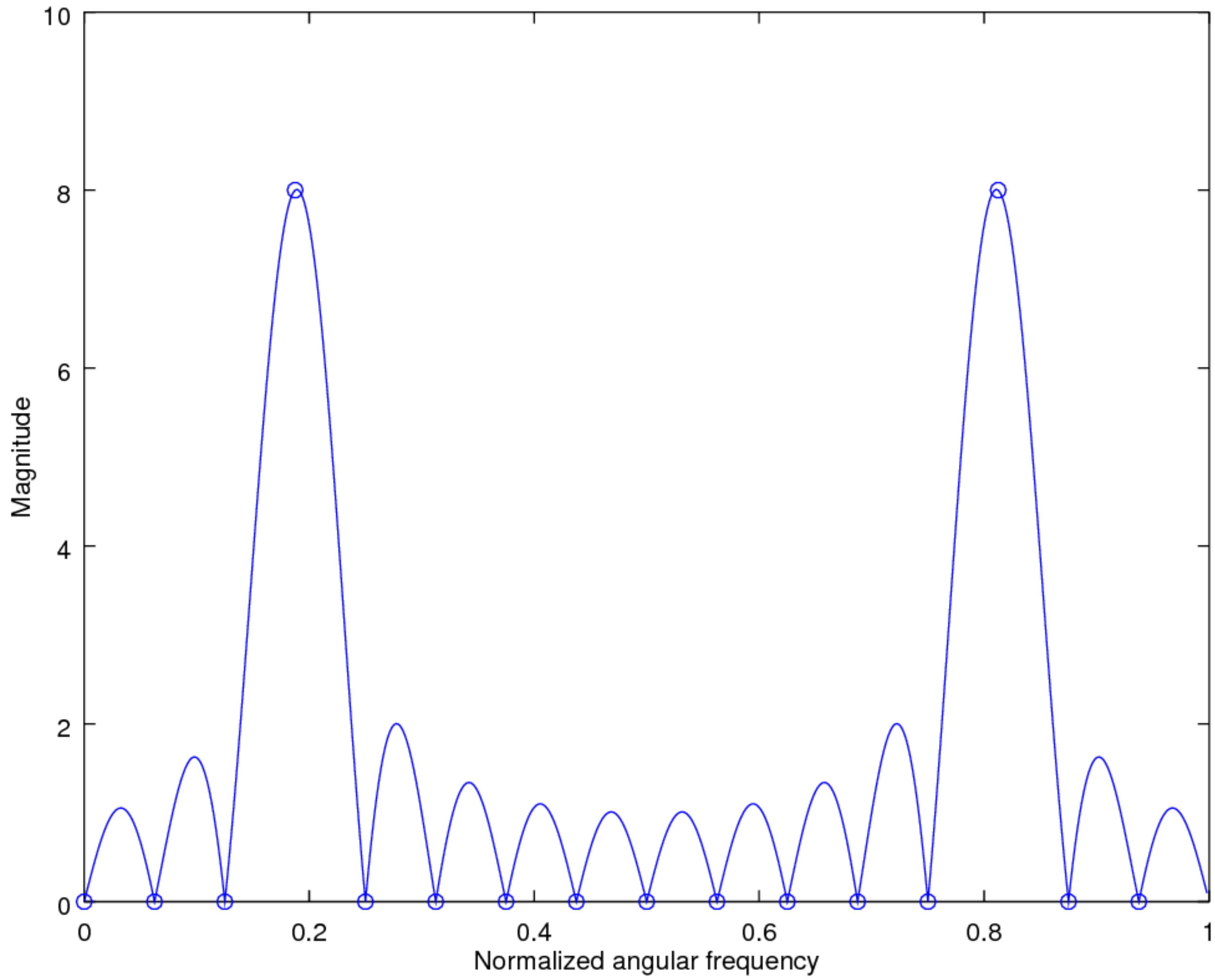
K=8
N=24



Numerical Computation of DTFT

```
clear
clf

k = 0:15;
x = cos(2*pi*k*3/16);

X = fft(x);
XE = fft(x, 512);

L = 0:511;
plot(L/512, abs(XE));
hold

plot(k/16, abs(X), 'o');
xlabel('Normalized angular frequency');
ylabel('Magnitude');
```

Numerical Computation of DTFT – results



Correlation

```
Correlation

n = 0:127;

x = [ones(1,25), -ones(1,25), zeros(1,78)];
y = [0:24, 25:-1:1, zeros(1,78)]/25;
yy = [y, y];

for m=0:127
  phi(m+1) = (2/128)*x*yy(m+1:m+128);
end
```

```
DFT

N = 50;
n = [0:N-1];
m = [0:N-1];

x = [zeros(1,28), ones(1,12), zeros(1, N-40)];

X = x*exp(-j*2*pi*m'*n/N);



flops(0);
fft(x,N);
f(N)=flops;
```

DFT

```
n = 0:99;
x = sin(2*pi*(n-50.5)/5)./(n-50.5);
X = fftshift(fft(x));
amplitude = abs(X);
phase = unwrap(angle(X));
```

```
N1 = 32; N2 = 128;
x = [ones(1,N1/2), -ones1,N1/2);
X = fft(x, N1);
Y = fft(x, N2);
```

```
X = fft(x);
Y = [X(1:(N+1)/2), zeros(1,K*N), X((N+!)/2+1:N)];
Y = (K+!)*ifft(Y);
```

Numerical Computation of DTFT

A

Numerical Computation of DTFT

A

References

[1] http://en.wikipedia.org/
[2] J.H. McClellan, et al., Signal Processing First, Pearson Prentice Hall, 2003
[3] M.J. Roberts, Fundamentals of Signals and Systems
[4] S.J. Orfanidis, Introduction to Signal Processing
[5] K. Shin, et al., Fundamentals of Signal Processing for Sound and Vibration Engineerings

[6] A “graphical interpretation” of the DFT and FFT, by Steve Mann