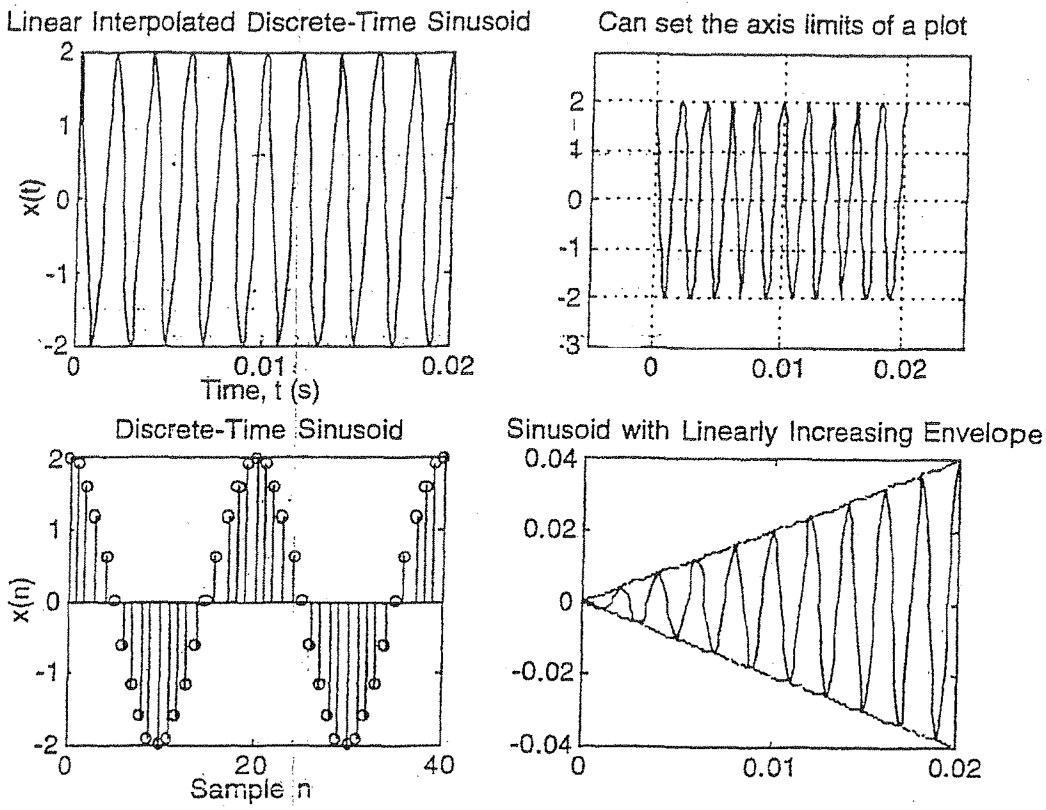
Matlab Examples

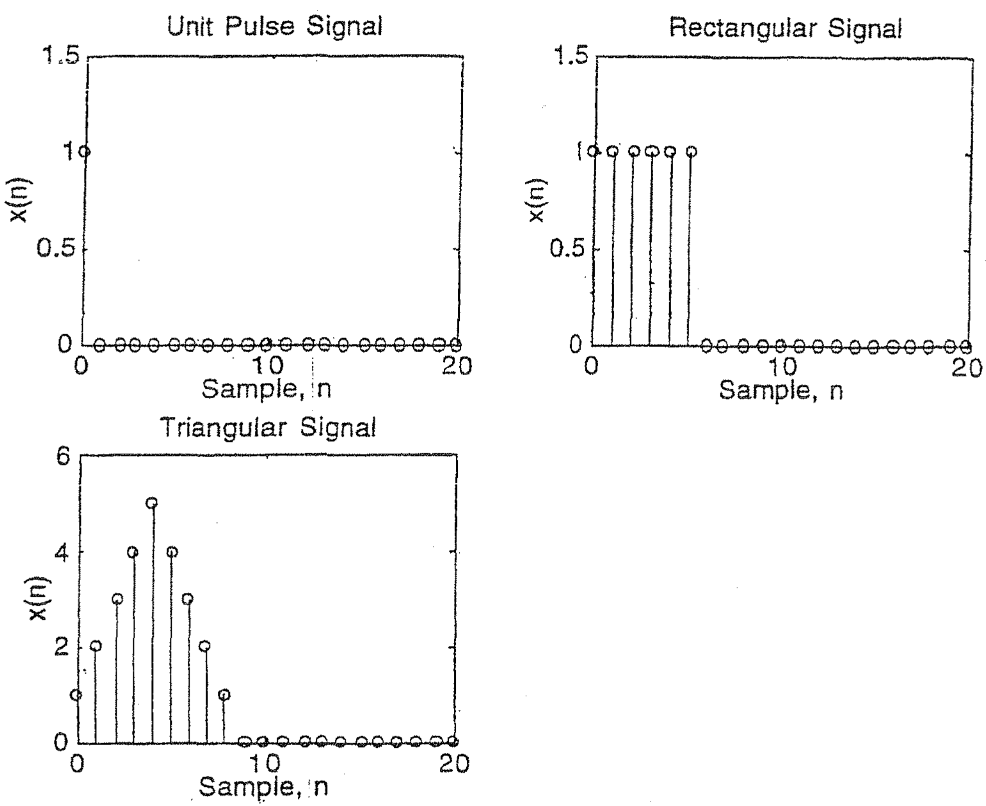
```
% EE253                                                          Prof. Essam Marouf
% DSP I
% The Discrete Fourier Transform                                 Matlab Handout #1
% ------------------------------------------------------------------------------
%
% For help on any of the basic function calls below, type "help function_name"
% when in Matlab, or invoke the other Matlab help facilities.
%
% Script File: Basics_Ex1.m
% ---------------------------
% Generates a simple sinusoidal signal and demonstrates vector operations
%
clear, close all             % closes all windows and clears all variables in memory
F=500;                       % Assigns scalar value to F;
                             % F is the analog frequency in Hz.
A=2;                         % Amplitude of a sinusoid
Fs=10000;                    % Sampling frequency (Hz);
                             % We will plot below an oversampled sinusoid.
T=1/Fs;                      % Sampling interval (s).
n=[0:1:200];                 % Generates an index vector n, 0 <= n <= 200, step = 1;
                             % first element n(1) starts from index 1.
Omega = 2*pi*F;              % Some universal constants are predefined in Matlab,
                             % e.g., pi; Omega is the analog frequency (rad/s).
t= T*n;                      % Each element of the vector n is multiplied by T;
                             % creates a time vector t = [0:T:200*T] (s).
x= A*cos(Omega*t);           % Can call functions with vector (or matrix) arguments;
                             % functions can return vector (or matrix) results.
subplot(2,2,1),plot(t,x,'-g'),  % plot the sinusoidal signal
xlabel('Time, t (s)'),
ylabel('x(t)'),
title('Linear Interpolated Discrete-Time Sinusoid')
%
  subplot(2,2,2),plot(t,x,'-g'),grid         % To add a grid to the plot.
axis([-0.005 .025 -3 3]);                    % To specify the x- and y- plot ranges.
title('Can set the axis limits of a plot');
axis;                                        % To reset, or float, axis limits.
%
  subplot(2,2,3),stem(n(1:41),x(1:41),'-r')  % discrete-time plot of x(n) using the "stem"
                                             % function; a subset of vector elements can
                                             % be chosen using (n_start:n_step:n_end);
                                             % n_step = 1 if not specified.

xlabel('Sample n'),
ylabel('x(n)'),
title('Discrete-Time Sinusoid'),
%
x1=t.*x;                                     % .*, ./, and .^ denote element-by-element
                                             % operations; x1 is a sinusoudal signal with
                                             % linearly increasing amplitude
%
subplot(2,2,4),plot(t,x1,'-g',    t,A*t,'--r',    t,-A*t,'--r'),
                                             % Multiple plots on the same graph
title('Sinusoid with Linearly Increasing Envelope')
subplot                                      % get out of subplot mode
```

```
pause                                   % allows you time to copy a plot into the
                                        % clipboard, print to a file or to the printer
```

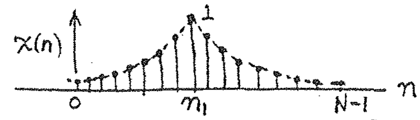



```
% Script File: Basics_Ex2.m
% ------------------------------
% Generates Some Basic Signals
%
clear, close all
%
% a. The Unit Pulse (Impulse) Signal
% ------------------------------------
M=20;
n=[0:M];                % starts from n(1) and ends at n(M+1)
x=[1 zeros(1,M)];       % zeros(N,M) generates (NxM) matrix of zeros
                        % x(1) = 1, and x(n) = 0, 2 <= n <= M+1

% or
% x=zeros(size(n));     % zeros vector of the same size as n;
% x(1) =1;              % can assign value to any given matrix element
subplot(3,1,1), stem(n,x);
axis([0 20 0 1.5]),
xlabel('Sample, n'), ylabel('x(n)'),
title('Unit Pulse Signal')
%
% b. Shifted Rectangle Signal (x(n) =1, 0 <= n <= M; x(n) = 0 otherwise)
% ------------------------------
%
M=5; N=20;                              % Matlab allows multiple statements
```

```
                                                     % on the same line
n=[0:N];                                             % n(1) to n(N+1)
x=[ones(1,M+1) zeros(1,N-M)];                        % specify rows and columns, or
% or
% x=[ones(n(1:M+1)) zeros(n(M+2:N+1))];              % specify a vector of the same size
%
subplot3,1,2), stem(n,x);
axis([0 20 0 1.5]),
xlabel('Sample, n'), ylabel('x(n)'),
title('Rectangular Signal')
pause
subplot

% c- Triangle Signal
% --------------------
M=5; N=20;
n=[0:N];
x=[[1:1:M] [M-1:-1:0] zeros(1,N-2*M+1)];             % or, can use
% x = [n(2:M+1)  n(M:-1:1)  zeros(n(2*M+1:N+1))];
subplot(3,1,3), stem(n,x);
axis([0 20 0 6]),
xlabel('Sample, n'), ylabel('x(n)'),
title('Triangular Signal')
axis;
subplot
```



The Discrete Fourier Transform: Computational Examples

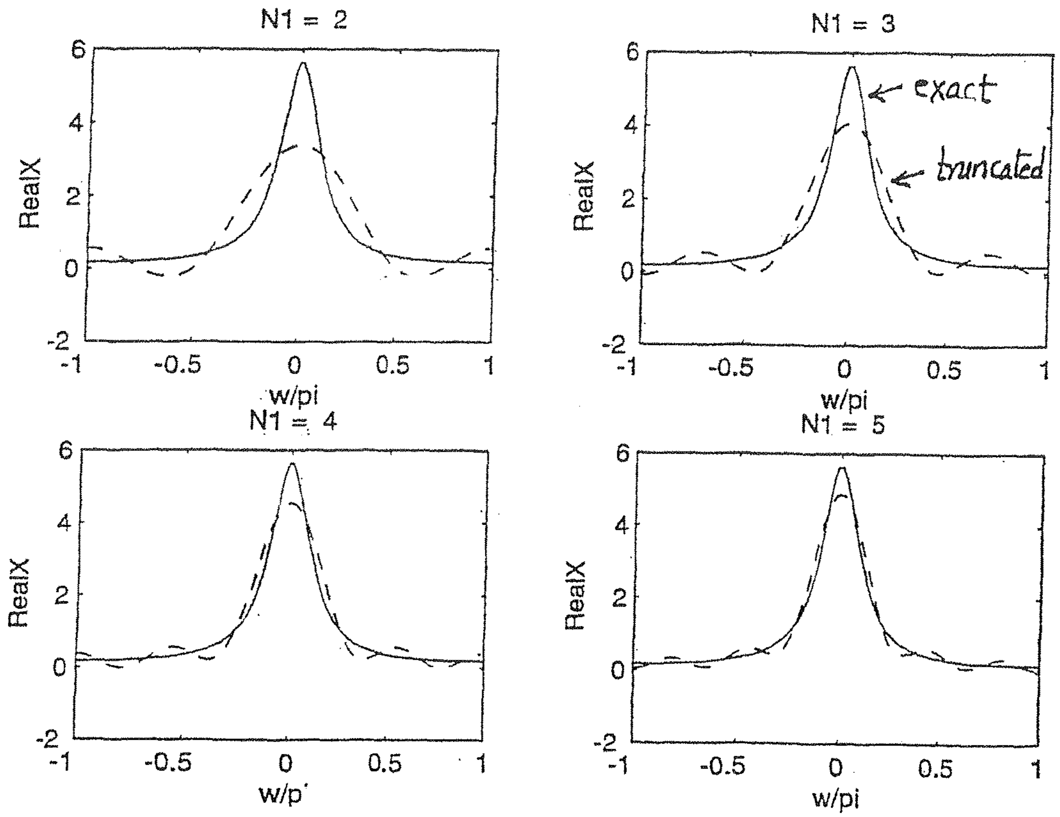


$$x(n) = a^{|n-n_1|} \xleftrightarrow{DTFX} X(e^{j\omega}) = \frac{1-a^2}{1-a^2-2a\cos(\omega)} e^{-j2\pi n_1 \omega}$$

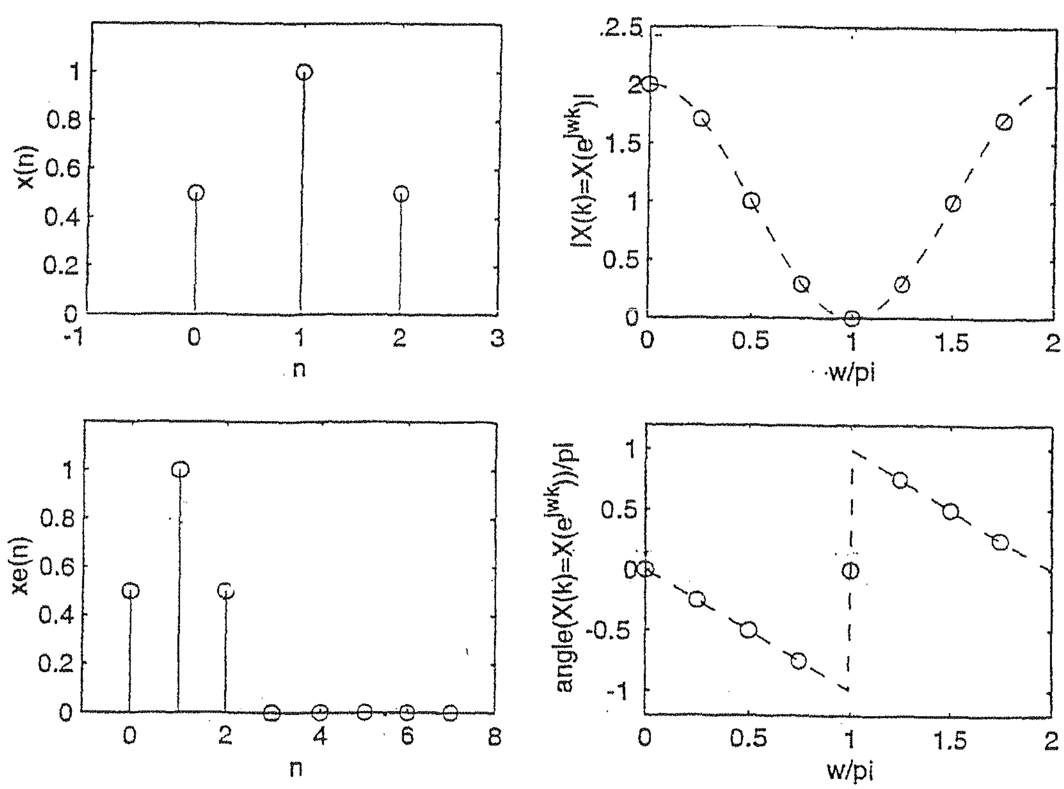
```
% Script File: DFT_Ex1.m
% ---------------------------
% Example to illustrate the change in the DTFT when a signal x(n) is truncated.
% ----------------------------------------------------------------------------
% DTFT of x(n) = a^|n| , |a|<1, is X(e^jw) = (1-a^2)/(1-a^2-2acos(w), -pi ≤ w < pi
% Truncated x(n) = a^|n|, -N1≤ n≤ N1
%
a=0.7;
w=linspace(-pi, pi,100);                      % w vector; type help linspace
Xw= (1-a^2) ./  (1+a^2-2*a*cos(w));           % Xw =DTFT[x(n)]; analytical solution
%
L=0;  ↙ n1
for N1=[2  3  4  5];                          % Exmine differnt truncation length
        N=2*N1+1;                             % total trncated signal length
        n=[0:N-1];                            % sample index of shifted signal
        x=a.^abs(n-N1);                       % truncated and shifted x(n)
        N2=128;                               % DTFT[x(n)] ~ DFT[x(n)] if N2 is large
        X=fft(x,N2);        ↙ N2 is even      % Calculating DTFT using DFT
        X=fftshift(X);                        % place negative freqs head of positive freqs
        wk=[-N2/2: (N2/2)-1]*2*pi/N2;         % correponding wk in radian
        X=X.*exp(j*wk*N1);                    % compenstate for the time-shift of x(n)
        L=L+1;
        subplot(2,2,L), plot(w/pi,Xw,'-r', wk/pi,real(X),'--b'),
        xlabel('w/pi'), ylabel('Real{X}')
        title(sprintf('N1 = %2.0f', N1))      % type help sprintf
        axis([-1 1 -2 6]);      ↑ variable
end
subplot
```

```
% Script File: DFT_Ex2.m
% ---------------------------
% Example to illustare effect of sampling of DTFT{x(n)}
% ----------------------------------------------------------
% In class we showed that DTFT{x(n) = [1/2  2  1/2]} = (1+cos(w))*exp(-jw)
%
n=[0  1  2];
x=[1/2  1  1/2];                              % signal x(n)
Nx=length(n);
subplot(2,2,1), stem(n,x)
xlabel('n'), ylabel('x(n)')
axis([-1 3 0 1.2])
%
w=linspace(0, 2*pi,100);
Xw=(1+cos(w)).*exp(-j*w);                     % DTFT{x(n)}; analytical solution
%
N= 8;                                         % Number of samples in frequency domain
k=[0:N-1];                                    % frequency bins
wk=2*pi*k/N;                                  % correponding frequencies in rad
Xk=(1+cos(wk)).*exp(-j*wk);                   % samples X(k), 0 ≤ k ≤ N-1
xe=real(ifft(Xk, N));                         % extended time sequence xe(n)
```

Output of DFT_Ex1



Output of DFT_Ex2



```
ne=k;
subplot(2,2,3), stem(ne,xe)
xlabel('n'), ylabel('xe(n)')
axis([-1 8 0 1.2])
%
subplot(2,2,2), plot(wk/pi, abs(Xk),'o', w/pi,abs(Xw),'--')
xlabel('w/pi'), ylabel('|X(k)=X(e^{jwk})|')
axis([0 2 0 2.5])
%
subplot(2,2,4), plot(wk/pi, angle(Xk)/pi,'o',w/pi,angle(Xw)/pi,'--')
xlabel('w/pi'), ylabel('angle(X(k)=X(e^{jwk}))/pi')
axis([0 2 -1.2 1.2])
subplot
```

```
% Script File: DFT_Ex6
% ----------------------
% Example to illustrate that spectra of real signals have conjugate (Hermitian) symmetry
% ----------------------
for N=[7  8];   ← odd & even cases
  n=[0:N-1];
  k=n;
  xn=[ones(1,5) zeros(1,N-5)];                           % real x(n) ==> Hermitian spectrum
  Xk=fft(xn,N);                                          % |X(k)| = |X(N-k)|; <X(k)= - <X(N-k)
A=[k; real(Xk); imag(Xk); abs(Xk); angle(Xk)*180/pi];    % matrix of results
                                                         % Display results (on the screen)
  disp(sprintf('N = %2.0f',N))
  disp(      'k=        real(Xk)    imag(Xk)       |Xk(k)}|   <Xk(k)')
  disp(sprintf('%2.0f       %7.2f       %7.2f        %7.2f    %7.2f\n', A))
  disp(' ')
end                    ↑ Formatted output
%
```

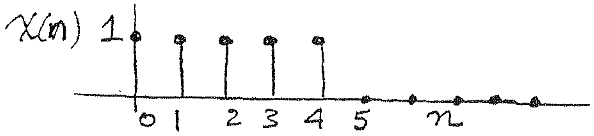
Output of DFT_Ex6

$$X(k) = X^*(N-k) \quad , 1 \leq k \leq N-1$$

N = 7 (odd)

k=		real (Xk)	imag (Xk)	|Xk(k)}|	<Xk(k)
0	real	5.00 ← DC	0.00 ← DC	5.00	0.00
1		-0.40	-1.76	1.80	-102.86
2		1.12	-0.54	1.25	-25.71
3		0.28	0.35	0.45	51.43
4		0.28	-0.35	0.45	-51.43
5		1.12	0.54	1.25	25.71
6		-0.40	1.76	1.80	102.86

X(0) is always real (DC)



N = 8 (even)

k=	real (Xk)	imag (Xk)	|Xk(k)}|	<Xk(k)
0	real 5.00 ← DC	0.00 ← DC real	→ 5.00	0.00 ←
1	-0.00	-2.41	2.41	-90.00
2	1.00	0.00	1.00	0.00
3	0.00	-0.41	0.41	-90.00
N/2 → 4	real 1.00 ←	0.00 ←	→ 1.00 real :	0.00 ←
5	0.00	0.41	0.41	90.00
6	1.00	-0.00	1.00	-0.00
7	-0.00	2.41	2.41	90.00

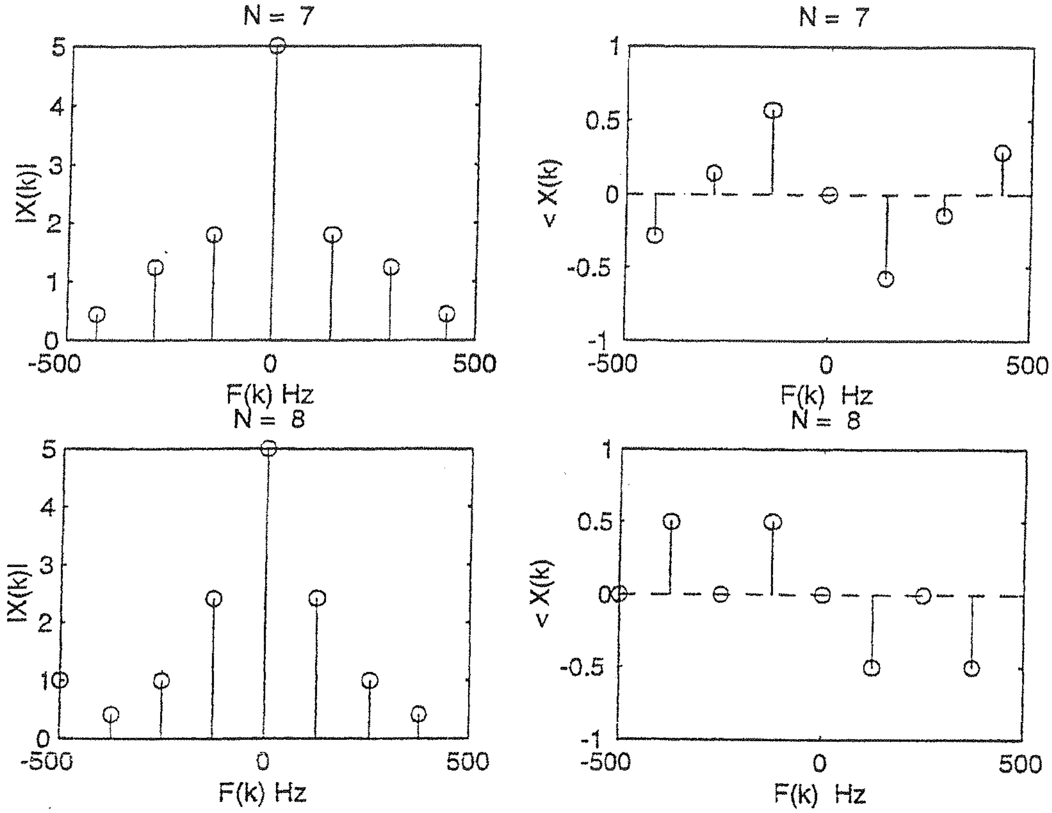
X(N/2) is a real sample

```
% Script File: DFT_Ex7
% ----------------------
% Example to illustrate bin frequencies in Hz
% --------------------------------------------

Fs=1E3;
L=1;                                            % sampling frequency in Hz
for N=[7 8];
        xn=[ones(1,5) zeros(1,N-5)];            % same signal x(n) as in DFT_Ex6
        Xk=fft(xn,N);
        Xk=fftshift(Xk);           N even       % reorder X(k)
        if (rem(N,2)==0)                        % case N is even; type help rem
                Fk=Fs*[-N/2:N/2-1]/N;           % Bin frequencies in Hz
        else
                Fk=Fs*[-(N-1)/2:(N-1)/2]/N;     % case N is odd
        end
        % Plot results            N odd
        % ----------
        subplot(2,2,L), stem(Fk,abs(Xk))
        xlabel('F(k) Hz'), ylabel('|X(k)|')
        title(sprintf('N = %2.0f', N))
        axis([-Fs/2 Fs/2  0  max(abs(Xk))]);
        %
        subplot(2,2,L+1), stem(Fk,angle(Xk)/pi), hold on
        plot([-Fs/2 Fs/2],[0  0],'--r'), hold off    % add a horizontal line
        xlabel('F(k)  Hz'), ylabel('< X(k)')
        title(sprintf('N = %2.0f', N))
        axis([-Fs/2 Fs/2 -1  1]);
        %
        L=L+2;                                  % update L for the next two plots
end
```

EE253: The Discrete Fourier Transform

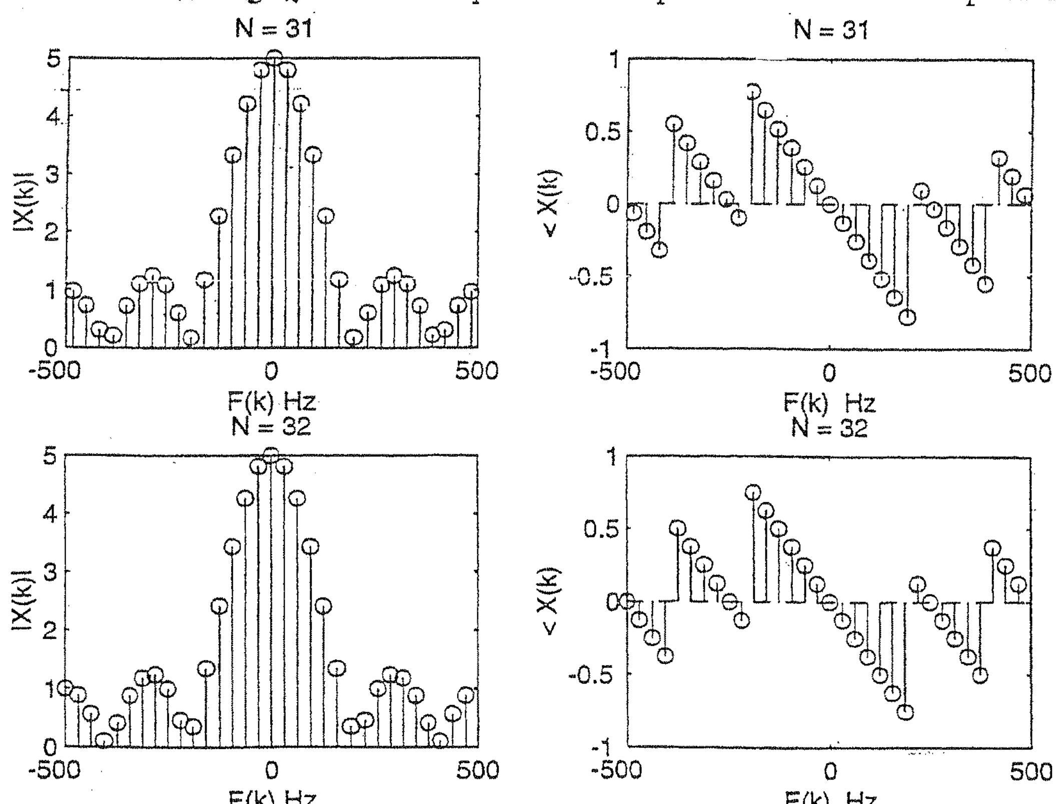
Output from DFT_Ex7: N=7 and 8

Note that when N is odd, there is no sample at -Fs/2, as when N is even.
Note also that the sample at F(k) = 0 (the DC sample) and at F(k)= -Fs/2 are always real.

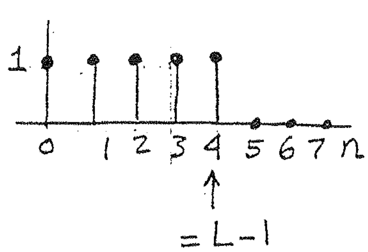


N=31 and 32

Note that [redacted] the more frequent DFT samples better define the shape of the DTFT of x(n)

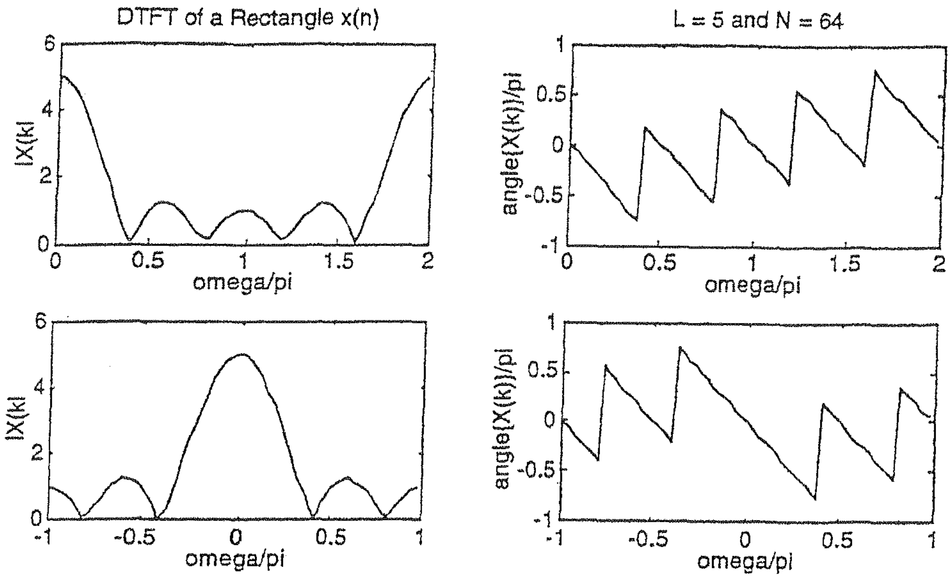


```
% Script File: Basics_Ex3.m
% ------------------------------
% Computes DTFT{x(n)} using the DFT (implemented as an FFT)
% Both the DFT and FFT will be discussed at length in future lectures
%
clear, close all
%
% Example: x(n) is a rectangle signal of length N1 = L = 5;
% DFT transform length must be chosen so that N >> L; here N = 64
% -------------------------------------------------------------------
L=5; N=64;
n=[0:N-1];                                  % time index
k=n;                                        % frequency index
x=ones(1,L);                                % rectangle signal
X=fft(x,N);                                 % Matlab's fft automatically pads x(n)
                                            % with zeros to length N
wk=(2*pi/N)*k;                              % 0 <= wk = 2*pi*k/N < 2*pi
%
subplot(2,2,1),plot(wk/pi, abs(X),'-g'),    % omega/pi is used to have clean limits
xlabel('omega/pi'),
ylabel('|X(k|'),
title('DTFT of a Rectangle x(n)')
%
subplot(2,2,2),plot(wk/pi, angle(X)/pi,'-g'),     % phase is wraped to [-pi,pi] interval
xlabel('omega/pi'),
ylabel('angle{X(k)}/pi'),                         % phase relative to pi; [-1,1] interval
title('L = 5 and N = 64')
%
% Can rearrange the results to correspond to the period -pi <= omega < pi as follows
%
index=find(wk >= pi);                       % find index of w's equal to or exceeding pi
wk(index) = wk(index) - 2*pi;               % shift w=[pi,2pi) to w=[-pi,0)
wk = fftshift(wk);                          % Type help fftshift;
                                            % fftshift(V) swaps left and right halves of vector V
X = fftshift(X);                            % Place negative frequencies bins ahead of
                                            % positive frequencies bins
subplot(2,2,3), plot(wk/pi, abs(X),'-g'),
xlabel('omega/pi'),
ylabel('|X(k|')
%
subplot(2,2,4),plot(wk/pi, angle(X)/pi,'-g'),
xlabel('omega/pi'),
ylabel('angle{X(k)}/pi')
subplot
%
% Can group the above steps into the single function dtft, if we wish. Try it.
```

$x(n) =$ 

$\longleftrightarrow X(e^{j\omega}) = \dfrac{\sin(L\omega/2)}{\sin(\omega/2)}\, e^{-j\frac{1}{2}(L-1)\omega}$

aliased sinc-function;
also called Dirichlet function




```
% Script File: DFT_Ex4
% -----------------------
% Illustrates the circular-shift property
%
x1=[1  2  3  4];
N=length(x1);
X1=fft(x1,N);
%
k=[0:N-1];
W=exp(-j*2*pi/N);
m=2;                          % shift in samples
X2=X1.*W.^(m*k);;             % linear phase for a circular shift  of m samples
x2=real(ifft(X2,N))           % inverse DFT; x2(n)= x1[ (n-m) mod N]
```

Output of DFT_Ex4

```
x2 =
   3    4    1    2
```

```
% Script File: DFT_Ex5
%
% Illustrates linear and circular  convolution
% ----------------------------------------
%
x=[1  2  3]; Nx=length(x);
h=[1  0  1]; Nh=length(h);
yL=conv(x,h);                           % lineaar convolution; type help conv
NyL=length(yL);                         % note that Ny=Nx+Nh-1
nL=[0:NyL-1];
%
N=max(Nx,Nh);                           % circular convolution requires same length signals
yC=real( ifft ( fft(x,N) .* fft(h,N) ));  % frequenct domain implementation of circular conv
NyC=length(yC);                         % Ny = Nx = Nh in this case
nC=[0:NyC-1];
%
```

```
% Implementation of linear convolution using circular convolution
% ---------------------------------------------------------------
Ny=Nx+Nh-1;
N1=Ny;                                        % Nmust be chosen so that N ≥ Ny
y1=real( ifft ( fft(x,N1) .* fft(h,N1) ));    % circular convolution of zero padded x and h
n1=[0:N1-1];
%
% here is the answer for a larger N = 8; note the answer is the ame except for the zero padding
N2=8;
y2=real( ifft ( fft(x,N2) .* fft(h,N2) ));    % circular convolution of zero padded x and h
n2=[0:N2-1];
%
subplot(2,2,1), stem(nL,yL),
axis([-0.5 NyL-0.5  0  max(yL)+0.5]);
xlabel('n'), ylabel('yL(n)'),
title('Linear Convolution: y = x * h')
%
subplot(2,2,2), stem(nC,yC),
axis([-0.5 NyC-0.5  0  max(yC)+0.5]);
xlabel('n'), ylabel('yC(n)'),
title('Circular Convolution x @ h')
%
subplot(2,2,3), stem(n1,y1),
axis([-0.5  N1-0.5  0  max(y1)+0.5]);
xlabel('n'), ylabel('y1(n)'),
title(sprintf ('xe(n) @ he(n); N= %2.0f', N1))
%
subplot(2,2,4), stem(n2,y2),
axis([-0.5  N2-0.5  0  max(y2)+0.5]);
xlabel('n'), ylabel('y2(n)'),
title(sprintf ('xe(n) @ he(n); N= %2.0f', N2))
subplot
```