

%Exercise 6.1

function [X, W] = dtft(x, w)

DTFT at frequencies that are equally spaced

usage: X = dtft(x, w)

x: finite-length input vector, whose length is L

W: number of frequencies for evaluation over $[-\pi, \pi)$

==> constraint: $W \geq L$

Xbox: DTFT values (complex)

W: (2nd output) vector of freqs where DTFT is computed

L = length(h);

if(W < L)

error('DTFT: # data samples cannot exceed # freq samples')

end

W = (2*pi/W) * [0:(W-1)];

W = fftshift(W);

W(1:(N/2)) = W(1:(N/2)) - 2*pi; <--- $[-\pi, \pi)$

X = fftshift(fft(h, W)); <--- move negative freq components

```
%Exercise 6.2
```

```
function fplot( xa, dt )
```

```
FPLOT
```

```
fplot( xa, dt )
```

xa: the "ANALOG" signal

dt: the sampling interval for

the simulation of xa(t)

```
L = length(xa);
```

```
Nfft = round( 2 .^ round(log2(5*L)) ); <-- next power of 2
```

```
range = 0:(Nfft/4);
```

```
ff = range/Nfft/dt;
```

```
plot( ff/1000, abs( Xa(1+range) ) )
```

```
title('CONT-TIME FOURIER TRANSFORM (MAG)')
```

```
xlabel('FREQUENCY (kHz)'), grid
```

```
pause
```

%Exercise 6.3

```
function [y,code] = projsig(digits,scale)
```

PROJSIG

generate "mystery" signal for ee 4078 project

(summer 1989)

usage: [Y,C] = projsig (D,S)

D are digits of a 5-element code

S is a scale factor that multiplies the

interference signal

Y is output signal

C is output code

```
load qwert
```

```
if ~exist('yint')
```

```
error (' problem loading interference')
```

```
end
```

```
tones=[ 0.025+0.05*[0:9]' ];
```

```
if margin == 1
```

```
scale = 1.0; add 100 of interference
```

```
end
```

```
if length(digits) < 5
```

```
digits = mod(fix(clock),10); digits=digits(2:6);
```

```
end
```

```
code = mod(fix(digits(1:5)),10); just 5 digits, must be integers
```

```
tones(code+1); create the tones
```

```
rand('uniform')
```

```
LL = 50*rand(7,1)-25; variation in lengths
```

```
LL = fix(LL) + [55;175*ones(5,1);95];
```

```
Ltot = sum(LL);
```

```
if Ltot > length(yint)
```

```
LL = fix(LL*Ltot/length(yint));
```

```
end
```

```
tft = [0.5*rand(1);tones(code+1);0.5*rand(1)];
```

```
for j = 1:7;
```

```
f = [ f; tft(j)*ones(LL(j),1) ];
```

```
end
```

```
N = length(f); Nm1 = N-1; nn = [0:Nm1]';
```

```
-----
```

```
tau = 0.8;
```

```
[tft,fi] = filter(1-tau,[1 -tau],f(1)*ones(99,1)); set init conds.
```

```
f = filter(1-tau,[1 -tau],f,0.9*fi);
```

```
rand('normal')
```

```
z = cos(2*pi*f.*nn);
```

```
z = z + scale*zint(1:N);
```