**% Tune-Up #7**

% Working part of Exercise 1.1 of Mini-Project #2.  Mini-Project #2 assignment,

% hints, and code are available on the [homework page](http://users.ece.utexas.edu/~bevans/courses/signals/homework/index.html).

% The exercises for Mini-Project #2 are from [Chapter 10](https://utexas.instructure.com/files/62726675/download?download_frd=1) of the book

% Computer-Based Exercises for Signal Processing in Matlab, 1994.

**% Wireless Localization**

% The project involves estimating distance to an object in the environment.

% using wireless signals.  This problem simulates a radar system that sends a

% signal and then listens for the return signal that bounced off the object.

% Using the round-trip time *Td* and speed of propagation in the environment *c*,

% the distance can be determined as *d* = (1/2) *Td* *c*.

% Methods for finding the direction (angle) to the object include

% (a) Directional beams

% (b) Triangulation

% For an overview of radar signals, please see

% <https://en.wikipedia.org/wiki/Radar_signal_characteristics>

**% Complex-Valued Chirp Signals**

% In a chirp signal, the principal frequency increases or decreases over time.

% When chirp signals propagate in an environment, they are resistant to

% frequency distortion and thermal noise.  You'll evaluate this in Mini-Project #2.

% **Part (a)** **Write Matlab code to generate and plot a complex-valued chirp**

% pulse that sweeps frequencies from -W/2 to W/2:

% s(t) = exp(j pi (W t^2) / T) for -T/2 <= t <= T/2.

% Parameters from Table 10.1 in the excerpt of Chapter 10:

%      T  pulse length   25 us

%      W swept bandwidth  2 MHz

%      fs sampling frequency 20 MHz

%      TW time-bandwidth product  50  [dimensionless]

% The oversampling factor is p where fs = p W.

T = 25E-6;

W = 2E6;

fs = 20E6;

Ts = 1/fs;

t = (-T/2) : Ts : (T/2);

s = exp(j\*pi\*W\*(t.^2)/T);

% Time-domain plot.  We have to be careful at plotting s(t)

% because it is complex-valued.  We'll plot the real part.

figure;

plot(t, real(s));

xlabel( 't' );

**% Question: Describe the chirp signal.**

**% *Answer:* The chirp signal is a finite-length signal that last from -T/2 to T/2 seconds.**

**% The principal frequency decreases from -T/2 to 0 seconds and increases**

**% from 0 to T/2 seconds.**

**% (b) Write a Matlab function to generate a discrete-time version** of the

% complex-valued chirp following the Mini-Project #2 guidance:

% s[n] = exp(j 2 pi alpha (n - N/2)2) for for 0 ≤ n ≤ N-1

% We'll need to connect s(t) for -T/2 ≤ t ≤ T/2 via s[n] = s(n Ts) where

% Ts is the sampling time.  N samples would correspond to T seconds of

% continuous time i.e. T = N Ts.  The Matlab function will take two parameters

%     TW time-bandwidth product

%     p oversampling factor

% The sampling rate fs is p W.  Using the hints for Mini-Project #2, we can

% express the parameters alpha and N in terms of p and TW:

%     N = p TW

%     alpha = TW / (2 N2)

**% Question: Verify the formulas for *N* and *alpha* using the code from part (a).**

**% *Answer:* The formulas for *N* and *alpha* can be obtained by equating the first**

**% value of the complex-valued chirp *s*(-*T*/2) and the first sample of the**

**% discrete-time chirp *s*[0].**

Diagram, histogram

Description automatically generated**% For this Tune-Up, we’ll plot *s*(*t*) and**

**% *s*[*n*] to see if *s*[*n*] are samples of *s*(*t*).**

%% Plot real part of s(t)

T = 25E-6;

W = 2E6;

fs = 20E6;

Ts = 1/fs;

t = (-T/2) : Ts : (T/2);

soft = exp(j\*pi\*W\*(t.^2)/T);

figure;

plot(t, real(soft));

xlabel( 't' );

Histogram

Description automatically generated with low confidencexlim( [-T/2 T/2] );

%% Plot real part of s[n]

TW = T\*W;

p = fs / W;

N = p \* TW;

alpha = TW / (2\*N^2);

n = 0 : N-1;

sofn = exp(j\*2\*pi\*alpha\*(n - N/2).^2);

figure;

plot(n, real(sofn));

xlabel( 'n' );

% Yes, as seen on the right, s[n] are samples of s(t).

**% Question: Write a MATLAB function to generate the discrete-time chirp**

% called dchirp and place it in a file called dchirp.m.

% ***Answer***: See the code below.

function s = dchirp( TW, p )

% DCHIRP   generate a sampled chirp signal

%   usage  s = dchirp( TW, p )

%          s : samples of a discrete-time "chirp" signal

%              exp(j pi (W/T) t^2 )   for -T/2 <= t <= T/2

%          TW : time-bandwidth product

%          p : sample at p times the Nyquist rate (W)

N = p\*TW;

alpha = TW / (2\*N^2);

n = 0 : N-1;

s = exp(j\*2\*pi\*alpha\*(n - N/2).^2);