DSP_Lab_Specgram_Harmonic_Lines_Chirp_Aliasing

Student Name: Abdul Malik

Student Number: 2211011098

Matlab Synthesis of Chirp signals:

In many chirp examples with parameters $f0 = fzero$ and slope μ, the instantaneous frequency is taken to be

$$f(t) = f0 + \mu\, t, f(t)$$

so the phase is the integral of that frequency,

$$\psi(t) = 2\pi \int 0t[f0 + \mu\, \tau]\; d\tau + \phi.$$

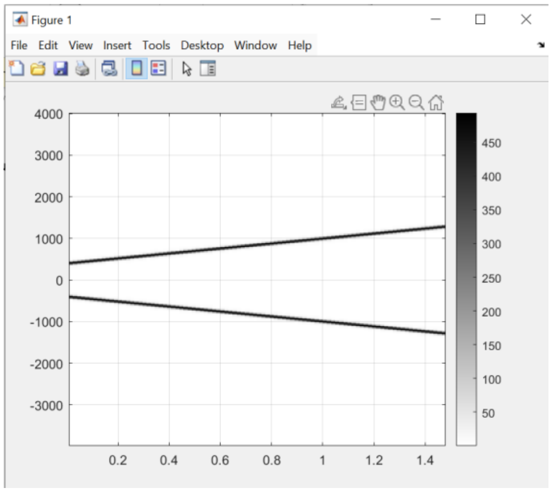
And hence the psi equation in MATLAB will be:
psi = 2*pi*(fzero*tt + 0.5*mu*tt.^2) + phi;

In the next code we will synthesize a linear-FM Chirp.

Code:

```
fSamp = 8000; %-Number of time samples per second
dt = 1/fSamp;
tStart = 0;
tStop = 1.5;
tt = tStart:dt:tStop;
mu = 600;
fzero = 400;
phi = 2*pi*rand; %-- random phase
%
psi = 2*pi*(fzero*tt + 0.5*mu*tt.^2) + phi;

%
cc = real( 7.7*exp(1i*psi) );
% soundsc( cc, fSamp ); %-- uncomment to hear the sound
plotspec( cc+1i*1e-12, fSamp, 256 ), colorbar, grid on %-- with neg
```

Output:



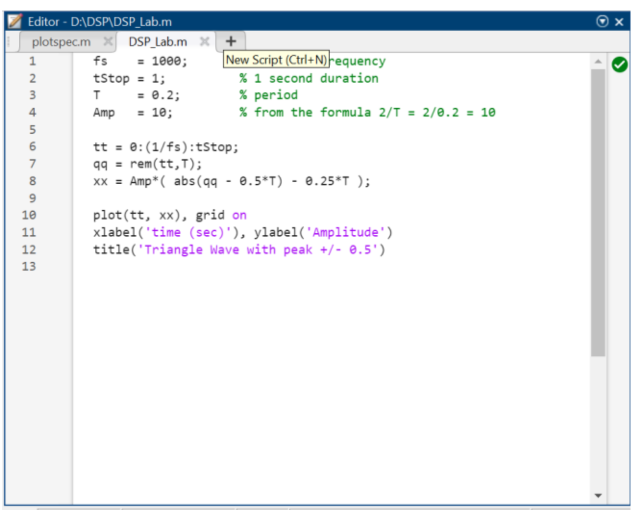
Triangle Wave:

As qq goes from 0 to T, the expression |qq-0.5T| goes from 0.5T down to 0 (at qq=0.5T) and back up to 0.5T0 (at qq=T). When you multiply by Amp, the resulting amplitude is

$$(peak\ value) \ = \ Amp \ \times 0.25\,T.$$

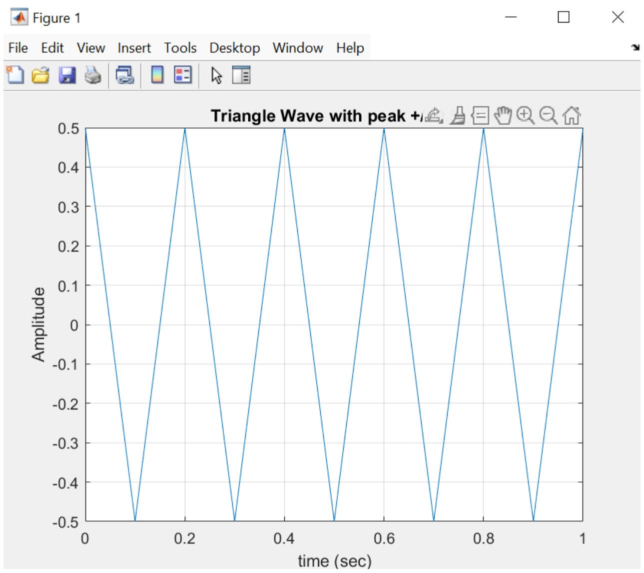
Now we want to find the am for a peak of +-0.5:

So we have the MATLAB Implementation as:



```
Editor - D:\DSP\DSP_Lab.m
plotspec.m   DSP_Lab.m   +

fs    = 1000;       New Script (Ctrl+N) requency
tStop = 1;          % 1 second duration
T     = 0.2;        % period
Amp   = 10;         % from the formula 2/T = 2/0.2 = 10

tt = 0:(1/fs):tStop;
qq = rem(tt,T);
xx = Amp*( abs(qq - 0.5*T) - 0.25*T );

plot(tt, xx), grid on
xlabel('time (sec)'), ylabel('Amplitude')
title('Triangle Wave with peak +/- 0.5')
```

Output:



Part1:

dB=20log10(A).

20log10(0.1) =20× (−1) =−20dB

20log10(1) =0dB

20log10(2) ≈20×0.3010≈6.02dB

20log10(5) ≈20×0.6990≈13.98dB

20log10(10) =20×1=20dB

20log10(100) =20×2=40dB

Part2:

A=10^(dB/20)

A=10^ (6/20) =100.3≈2

A=10^ (60/20) =103=1000

A=10^ (80/20) =104=10000

Lab procedures:

Determine Parameters for a Chirp from 1000 Hz to 11000 Hz in 4s:

A common linear-FM (chirp) signal can be defined as

$$x(t) \;=\; Acos\,(\,2\pi\,[f0\,t + \mu/2\,t^2] + \phi)$$

Where:

- f0 = starting frequency (in Hz) at t=0.
- μ = “sweep rate” or slope in Hz/s.
- A = amplitude (choose as needed, e.g. 1).
- φ = initial phase.

$$f(t) = f0 + \mu t \;=\; f(4) = f0 + \mu \cdot 4 = 11000.$$

$$\mu = f(4) - f0/4 \;=\; (11000 - 1000)/4 = \; 10000/4 = 2500\;Hz/s.$$

$$f(t) = 1000 + 2500t(0 \le t \le 4).$$

$$\psi(t) = 2\pi(f0t + 1/2\mu t^2) + \phi = 2\pi(1000t + 1250t^2) + \phi.$$

Generating the Chirp in MATLAB and making a spectrum:

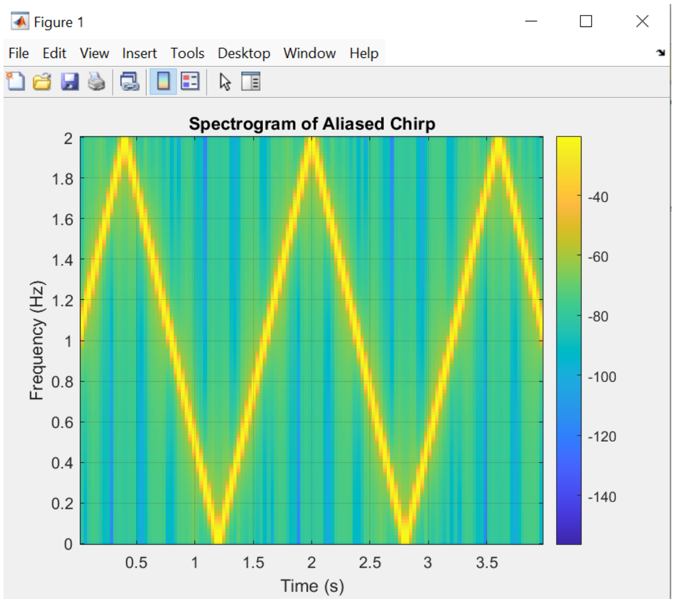
Code:

Editor - D:\DSP\DSP_Lab.m

plotspec.m | DSP_Lab.m

```
fs = 4000;                      % Sampling frequency
t = 0:1/fs:4;                   % Time vector from 0 to 4 seconds
f0 = 1000;                      % Start freq
mu = 2500;                      % Sweep rate

% Create the chirp (use amplitude 1 for simplicity)
x = cos( 2*pi*( f0*t + 0.5*mu*t.^2 ) );

% -- Make spectrogram with short section length:
LSECT = 256;                    % example short section size
TSECT = LSECT / fs;             % section duration in seconds

% spectrogram function or MATLAB’s built-in:
spectrogram(x, LSECT, [], [], fs, 'yaxis');
colorbar; grid on;
title('Spectrogram of Aliased Chirp');
xlabel('Time (s)'); ylabel('Frequency (Hz)');
```

Output:



Why Does the Frequency "Go Up and Down" Instead of Reaching 11000 Hz?

1) **Nyquist Limit (Aliasing):**
   Because the sampling rate is fs=4000 Hz, the highest frequency we can represent (without aliasing) is the Nyquist frequency fs/2=2000 Hz. Any energy above 2000 Hz is indistinguishable from energy in the range 0–2000 Hz once sampled. Thus, as the *true* (analog) instantaneous frequency continues climbing above 2000 Hz, the *sampled* signal frequency content "folds" (aliases) back into the 0–2000 Hz band.

2) **Spectrogram Display (0 to fs/2):**
   Most spectrogram displays show frequencies from 0 up to fs/2. Once our true chirp frequency passes 2000 Hz, it reappears in the spectrogram at a lower frequency. Then, as the chirp keeps increasing above even 4000 Hz, it folds (wraps around) multiple times. Ultimately, we see a zigzag or repeating up-down pattern in the spectrogram.

   MATLAB Script for a triangle periodic wave:

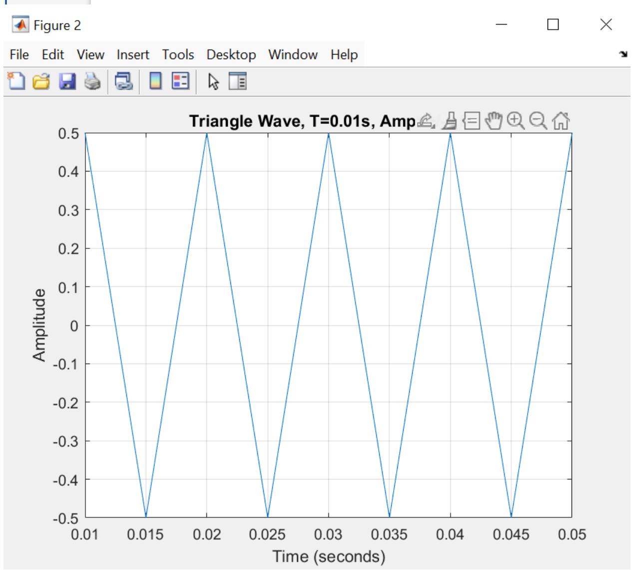
   plotspec.m | DSP_Lab.m

```
tt = 0:(1/fs):tStop;
qq = rem(tt, T);
xx = Amp * ( abs( qq - (0.5*T) ) - 0.25*T );
```

   Determining the Amp:

$$Amp \times 0.0025 = 0.5 = Amp = 0.00250.5 = 200$$

```matlab
fs = 10000;         % sampling rate
T = 0.01;           % period = 10 ms
tStop = 3;          % total duration of 3 sec
Amp = 200;          % to get +/- 0.5 peak amplitude

tt = 0 : 1/fs : tStop;
qq = rem(tt, T);
xx = Amp * ( abs(qq - 0.5*T) - 0.25*T );

% Plot a short section of 3-5 periods, e.g. from t=0.01 to t=0.05
figure;
plot(tt, xx);
xlim([0.01 0.05]);   % This region contains about 4 periods
grid on;
xlabel('Time (seconds)');
ylabel('Amplitude');
title('Triangle Wave, T=0.01s, Amp=+/-0.5');
```



Harmonic Line Spectrum in the Spectrogram

A **purely periodic** signal has a line spectrum at integer multiples of its fundamental frequency:

1. **Fundamental frequency**: f0 = 1/T=1/0.01 s=100 Hz
2. **Harmonics** appear at 2f0=200 Hz,3f0=300 Hz,4f0=400 Hz,…

In a **spectrogram** display (frequency on the vertical axis, time on the horizontal axis), these harmonics show up as **horizontal lines** at 100, 200, 300,… Hz (depending on how high we zoom).

Fundamental Frequency

From the period T=0.01 s, the fundamental frequency is:

$f0=1/T=100$ Hz

Measure Harmonic Amplitudes & Compute Ratio

Using our spectrogram plot (or plotspec figure), we can typically use the **Data Cursor** tool in MATLAB to hover over each harmonic line. It will show us the frequency (e.g., ≈100 Hz, 300 Hz, etc.) and the corresponding amplitude (in dB or linear magnitude, depending on how the spectrogram was computed).

1. **Locate the first harmonic** (fundamental) near 100 Hz. Record its amplitude (either in linear scale or in dB).
2. **Locate the third harmonic** near 300 Hz. Record its amplitude.

If we have **linear scale** amplitudes A1and A3:

ratio = A3/A1.

If the spectrogram is in **dB scale**, and we read dB1 for harmonic 1 and dB3 for harmonic 3, convert each back to linear magnitude:

$A1=10^{(dB1/20)}$, $A3=10^{(dB3/20)}$, ratio=A3/A1.

Why a Factor of Two Is ~6 dB

The decibel (dB) measure for amplitude A is given by

$dB(A)=20\log10(A)$.

A factor-of-2 difference in amplitude means

$B2/B1=2 = dB(B2) - dB(B1) = 20\log10(B2) - 20\log10(B1) = 20\log10(B2/B1) = 20\log10(2) \approx 6.02$ dB.

Hence, we commonly say “6 dB is (approximately) a factor of two.”

dB Difference Between a3 and a1for a Triangle Wave

For a **zero-mean** triangular wave of amplitude ±0.5, the nonzero Fourier series coefficients are (for odd k):

$ak = 2/\pi^{\wedge}2. k^{\wedge}2.$

In decibels, that becomes

$20\log10(19) = 20[\log10(1) - \log10(9)] = -20\log10(9) \approx -19.08$ dB.

So a3 is about **19 dB below** a1.

How Far Below a1 Is a15 in dB?

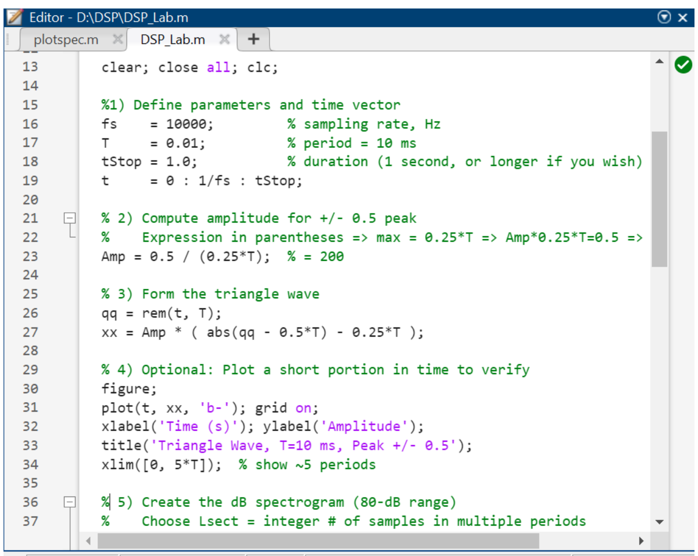
Similarly,

a15=2/π^2 . (15) ^2 = 1/225 .

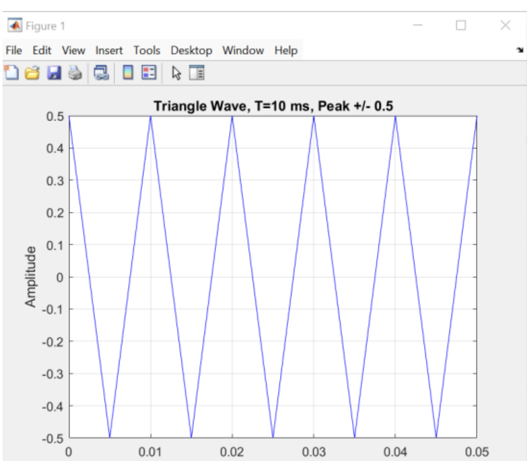
Convert to decibels:

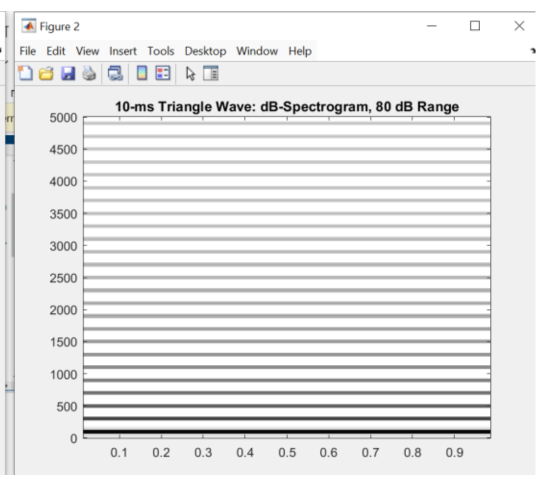
20log10 (1225)=−20 log10(225)≈−20 (2.352)=−47.0 dB.

So a15 is about **47 dB below** the fundamental a1.



```
clear; close all; clc;

%1) Define parameters and time vector
fs    = 10000;          % sampling rate, Hz
T     = 0.01;           % period = 10 ms
tStop = 1.0;            % duration (1 second, or longer if you wish)
t     = 0 : 1/fs : tStop;

% 2) Compute amplitude for +/- 0.5 peak
%    Expression in parentheses => max = 0.25*T => Amp*0.25*T=0.5 =>
Amp = 0.5 / (0.25*T);  % = 200

% 3) Form the triangle wave
qq = rem(t, T);
xx = Amp * ( abs(qq - 0.5*T) - 0.25*T );

% 4) Optional: Plot a short portion in time to verify
figure;
plot(t, xx, 'b-'); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Triangle Wave, T=10 ms, Peak +/- 0.5');
xlim([0, 5*T]);  % show ~5 periods

% 5) Create the dB spectrogram (80-dB range)
%    Choose Lsect = integer # of samples in multiple periods
```





Harmonic Frequencies for a Zero-DC Triangle Wave

For a standard "symmetric" triangle wave (with zero DC), all **even** harmonics vanish, and only the **odd** harmonics remain. The **fundamental** is

f0=1/T=100 Hz.

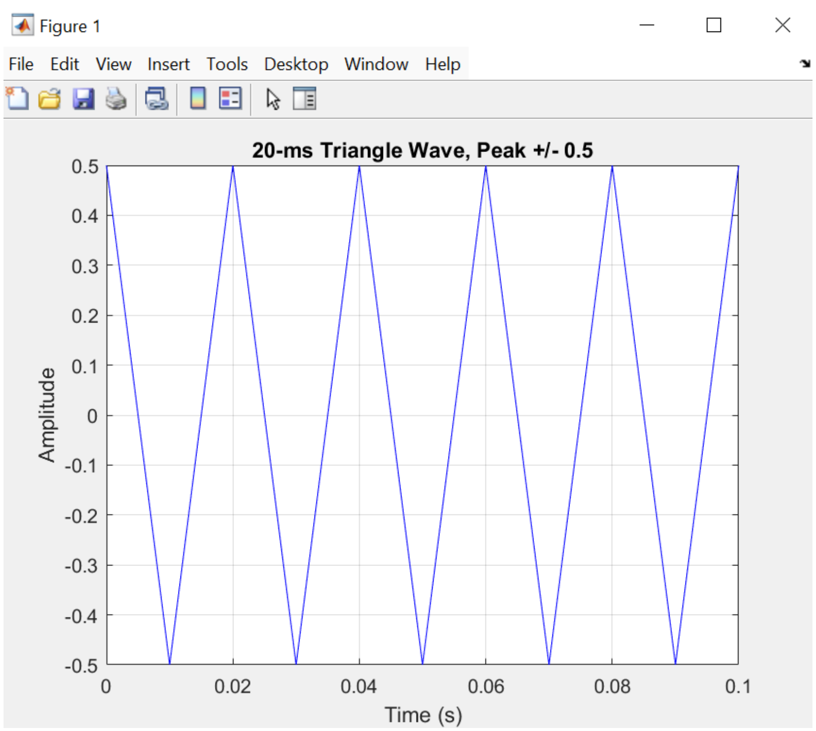
Hence, the nonzero harmonic lines appear at

fk=(2k−1) f0 for k=1,2, 3… Numerically, that's:

100 Hz,300 Hz,500 Hz,700 Hz, continuing (in principle) all the way up toward the Nyquist frequency (fs/2=5000 Hz).

When you use **dB scaling**, even the very small higher harmonics (e.g., 9th, 11th, etc.) become visible as **faint horizontal lines** in the spectrogram, whereas on a linear scale they might be too small to show up.

```
----------------------------------------------------
Fundamental frequency f0 = 1/T = 50 Hz
For odd harmonics: freq_k = (2k-1)*50 Hz
The largest odd harmonic below 5000 Hz is k=50 => freq=4950 Hz.
That is the 50th odd harmonic line.
```



Editor - D:\DSP\DSP_Lab.m

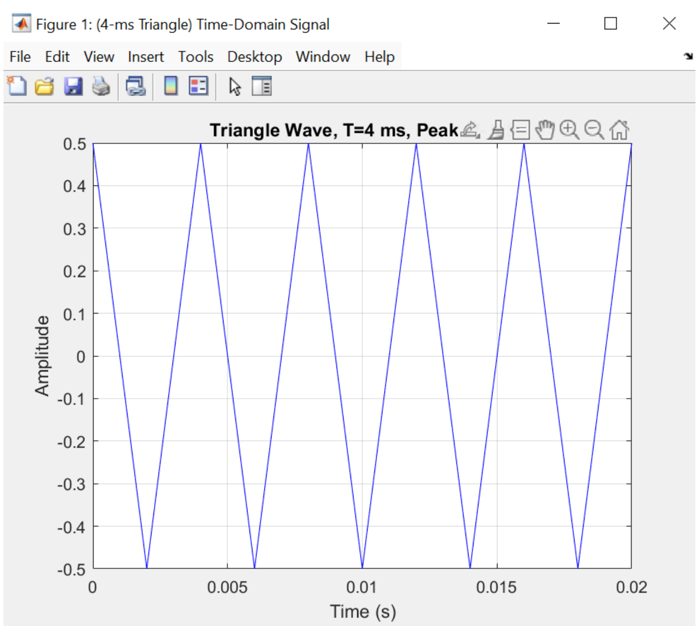
plotspec.m | DSP_Lab.m

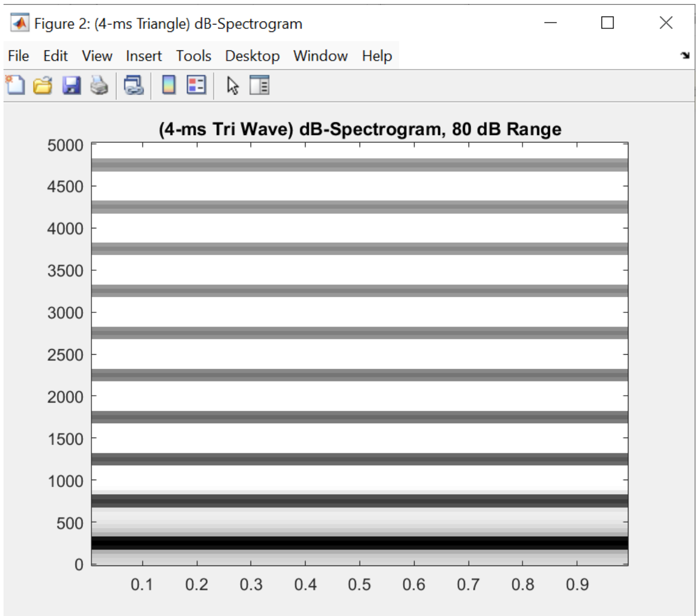
```matlab
clear; close all; clc;

%% 1) Define parameters
fs    = 10000;     % sampling rate (Hz)
T     = 0.004;     % 4 ms period
f0    = 1/T;       % 250 Hz fundamental
tStop = 1.0;       % 1 second duration
t     = 0 : 1/fs : tStop;

%% 2) Compute amplitude for +/- 0.5 peaks
% Inside parentheses, max = 0.25*T = 0.001
% We want Amp * 0.001 = 0.5 => Amp = 500
Amp   = 0.5 / (0.25*T);  % same formula used before

%% 3) Form the 4-ms triangle wave
qq    = rem(t, T);
xx4   = Amp * ( abs(qq - 0.5*T) - 0.25*T );

%% 4) Optional: Plot a short section in time to confirm
figure('Name','(4-ms Triangle) Time-Domain Signal');
plot(t, xx4, 'b-'); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Triangle Wave, T=4 ms, Peak +/- 0.5');
xlim([0, 5*T]);  % show about 5 periods

%% 5) Create the dB spectrogram with an 80-dB range
```

Editor - D:\DSP\DSP_Lab.m

plotspec.m | DSP_Lab.m

```matlab
%% 5) Create the dB spectrogram with an 80-dB range
% We must pick Lsect as an integer number of periods in samples:
%   T=0.004 s => 0.004 * 10000 = 40 samples/period
% Let's use 5 periods => 5*40 = 200 samples
Lsect   = 5 * round(T*fs);  % 5 periods
DBrange = 80;               % -80 dB floor

figure('Name','(4-ms Triangle) dB-Spectrogram');
plotspecDB(xx4, fs, Lsect, DBrange);
title('(4-ms Tri Wave) dB-Spectrogram, 80 dB Range');

%% 6) Comments on frequency spacing
disp('----------------------------------------------');
disp(['Fundamental frequency f0 = ', num2str(f0), ' Hz (i.e., 250 Hz).']);
disp('Because the period is shorter (4 ms vs. 10 or 20 ms),');
disp('the harmonic lines are more widely spaced in frequency.');
disp('For a zero-DC triangle wave, only odd harmonics exist:');
disp('   f_k = (2k - 1)*250 Hz, k=1,2,3,...');
disp('----------------------------------------------');
```





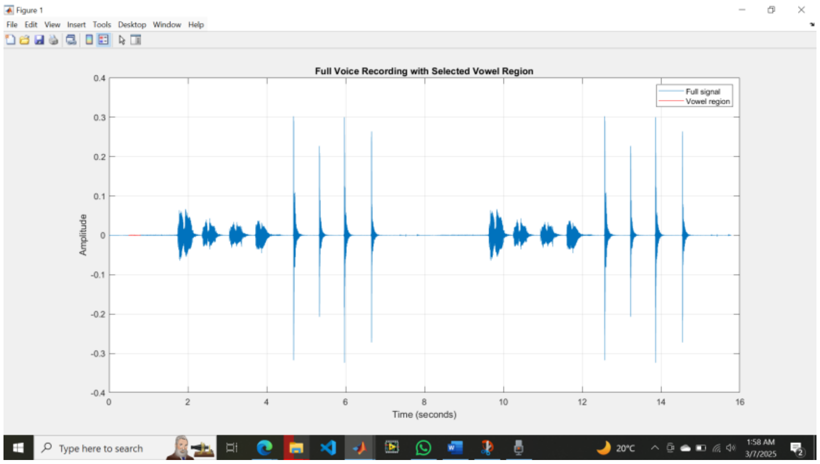

```
----------------------------------------------
Fundamental frequency f0 = 250 Hz (i.e., 250 Hz).
Because the period is shorter (4 ms vs. 10 or 20 ms),
the harmonic lines are more widely spaced in frequency.
For a zero-DC triangle wave, only odd harmonics exist:
   f_k = (2k - 1)*250 Hz, k=1,2,3,...
----------------------------------------------
```

Showing my own voice spectrum:

Editor - D:\DSP\DSP_Lab.m

plotspec.m | DSP_Lab.m

```matlab

clear; close all; clc;

%% (1) Load your recorded voice signal
% Suppose your file is "my_voice.wav" or "lab1_voice.wav"
[xx, fs] = audioread('Recording(1).m4a');  % <-- change filename as
xx = xx(:);                                % ensure column vector

% Create a time axis for reference
tt = (0:length(xx)-1)/fs;

%% (2) Select the vowel region
% Suppose from prior inspection or from Lab #1 we know the vowel is
% Adjust these times based on your data.
tStartVowel = 0.5;
tEndVowel   = 0.8;
idxVowel    = (tt >= tStartVowel) & (tt <= tEndVowel);
xxVowel     = xx(idxVowel);
ttVowel     = tt(idxVowel);

% Optional: plot wave to visually confirm region
figure;
plot(tt, xx);
hold on; grid on;
xlabel('Time (seconds)'); ylabel('Amplitude');

```



Conclusion:

In this lab, we synthesized and analyzed periodic triangle waves of several different periods (10 ms, 20 ms, and 4 ms). By creating dB spectrograms (with carefully chosen section lengths to match integer multiples of the period), we clearly observed harmonic lines at integer (or odd integer) multiples of the fundamental frequency. This demonstrated how shorter periods lead to higher fundamental frequencies and thus a wider spacing between harmonic lines.

Next, we turned to a real voice recording from Lab #1, focusing on a vowel region (which is quasi-periodic). By selecting a long section length for the spectrogram, we could more easily visualize the speech harmonics. Measuring the frequency separation between adjacent harmonics yielded the fundamental frequency f0, and hence the fundamental period T0. Comparisons with the time-domain pitch period measurement (from Lab #1) showed how the two approaches align and provided insight into minor differences due to pitch fluctuations or measurement precision. Overall, these tasks reinforced the interplay between time-domain periodicity and frequency-domain harmonic structure in both synthesized signals and real speech.