Eng. 100: Music Signal Processing F15 J. Fessler

Project 1: Tone synthesizer/transcriber

1 Abstract

This project teaches you the `Matlab` Graphical User Interface (GUI) tools that you will use in subsequent projects. It also demonstrates some issues you will face during the projects. The goals of this project are: (1) to write a `Matlab` program for a tone synthesizer with an on-screen mouse-activated keypad; (2) to write a `Matlab` program for a tone transcriber that produces a `Matlab` stem plot similar to musical staff notation from a pure tonal signal. The second project will do the same thing for telephone touch-tones; the third project will do the same thing for synthesized musical instruments. This project will help you do both.

2 Background

In Lab 2 you learned the frequencies used in a pure tonal version of "The Victors." In this project you will apply that knowledge to build a tone synthesizer and transcriber. You need to learn some musical notation and nomenclature, and some graphical `Matlab` commands. These topics are covered next.

3 Basics of musical staff notation

3.1 Results from Lab 2

In Lab 2 you learned that the musical tones used in "The Victors" had the following frequencies: 392, 440, 494, 523, 587, 659 Hz (rounded to the nearest integer). You also inferred that there were "missing" frequencies: 415, 466, 554, 622 Hz. You also deduced that these notes are related to each other not by common differences, but by a common ratio of 1.06 (actually 1.0595).

In fact 1.0595 is the 12th root of 2, *i.e.*, $2^{1/12}$. That factor suggests that there is a basic set of 12 notes that "repeat" with their frequencies multiplied by integer powers (positive and negative) of two. This is indeed the case. This is the modern *12-tone* or *chromatic* musical *scale*; most Western music is based on it, but several non-Western musical genres use other scales. (For example, a modern Arab tone system divides an octave into 24 equal ratios.)

The repetitions of the 12 frequencies are called *octaves*, because for any 8 notes that make up a *major scale*, the highest note has twice the frequency of the lowest note. An 8-note major scale consists of two half steps and 5 whole steps so the ratio of the highest to lower note frequency is $(2^{1/12})^2(2^{2/12})^5 = 2^{12/12} = 2$. So 880 Hz is one octave above 440 Hz, and 220 Hz is one octave below 440 Hz. We will focus on the octave spanned by the frequencies in "The Victors," because many musical compositions use this octave.

The 88 keys of a full-sized piano keyboard have frequencies that span more than 7 octaves:

- $(2^{1/12})^{88-1} = 2^{7.25}$
- Rightmost piano key: 4186 Hz is 3 octaves above 523 Hz ($4186/2^3 \approx 523$)
- Leftmost piano key: 27.5 Hz is 4 octaves below 440 Hz ($440/2^4 \approx 27.5$).

3.2 MIDI frequency notation

The information above is enough to create one kind of notation for most Western music. The Musical Instrument Digital Interface (MIDI) notation represents musical frequencies using integers from 0 to 127 based the formula

$$\text{MIDI} = 69 + 12\log_2(F/440), \tag{1}$$

where F denotes frequency in Hertz. In MIDI notation, frequencies of "The Victors" are represented as the following sequence of integers:

$$71, 67, 69, 71, 67, 69, 71, 72, 69, 71, 72, 69, 71, 72, 74, 76, 71, 71, 72, 67, 69, 71, 74, 71, 69, 67.$$

MIDI data also includes information about amplitude, duration, and some other things. This "notation" is well-suited for communicating music between digital devices, but is not very visually intuitive for humans.

3.3 Musical staff notation

Musical staff notation (the musical notation you usually see) is more complicated. The reason is that Western music uses the frequencies used in "The Victors" more commonly than the "missing" frequencies from Lab 2 (that is why they were missing).

- The 7 tones in "The Victors" are called *naturals* and are designated with letters: A, B, C, D, E, F, G.
- The 5 "missing" tones in a single octave are called *accidentals* or *sharps* and *flats*, and they are designated by "♯" and "♭" respectively.
- Some sharps and flats are equivalent: A♯ =B♭, C♯ =D♭, D♯ =E♭, F♯ =G♭, G♯ =A♭
  These are pronounced "A-sharp" and "B-flat," etc.
- B♭ is below B (between A and B) and C♯ is above C (between C and D); A♯ is the same as B♭, etc.
- A 12-note *chromatic scale* can be written as either of the following lists:
  {A, A♯, B, C, C♯, D, D♯, E, F, F♯, G, G♯ } or {A, B♭, B, C, D♭, D, E♭, E, F, G♭, G, A♭ }
  These chromatic scales include the naturals and all the *accidentals*.
- On a standard piano keyboard, the naturals are the white keys and the accidentals (sharps and flats) are the black keys.

Certain combinations of notes occur together more frequently in music traditionally. Why? Because they sound "more harmonious" because the ratios between their frequencies are very close to ratios of small integers. For example, A is 440 Hz and E is 659 Hz, almost exactly a 3:2 ratio. Indeed, early Western music was based on using these ratios of small integers, rather than the equally-spaced logarithms of frequencies used today. Proposals to use the $2^{1/12}$ ratio date back to the 16th century, but the more widespread change occurred around when J.S. Bach composed "The Well-Tempered Clavier" in 1722. These two books had compositions in all 12 keys, so an "equal tempered" scale (like we use today) would sound equally good (but perhaps not perfect) for all the pieces. In contrast, other tuning methods (based on small integer ratios) might sound good for some keys but not for others ("the ill-tempered clavier"?).

The following table summarizes the similarities of the integer ratios and the powers of $2^{1/12}$.

Notes	A	A♯	B	C	C♯	D	D♯	E	F	F♯	G	G♯
$\frac{\text{FREQ}}{440\,\text{Hz}}$	$2^{0/12}$	$2^{1/12}$	$2^{2/12}$	$2^{3/12}$	$2^{4/12}$	$2^{5/12}$	$2^{6/12}$	$2^{7/12}$	$2^{8/12}$	$2^{9/12}$	$2^{10/12}$	$2^{11/12}$
Hertz	440	466.2	493.9	523.3	554.4	587.3	622.3	659.3	698.5	740.0	784.0	830.6
Ratio	1:1	none	9:8	6:5	5:4	4:3	none	3:2	8:5	5:3	16:9	15:8
	440	?	495	528	550	$586.\bar{6}$	?	660	704	$733.\bar{3}$	$782.\bar{2}$	825

One need not always start with A. Starting with G, C and A and looking mostly at naturals, we have

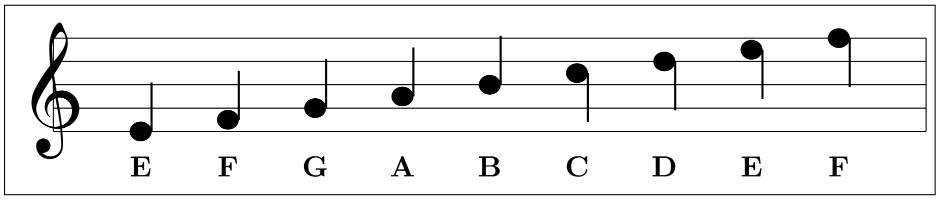
G	A	B	C	D	E	F♯	G
1:1	9:8	5:4	4:3	3:2	5:3	15:8	2:1
DO	RE	MI	FA	SO	LA	TI	DO

Natural	C Major:	C,D,E,F,G,A,B,C
Notes	Interval:	1,1, $\frac{1}{2}$,1,1,1, $\frac{1}{2}$
Only	A Minor:	A,B,C,D,E,F,G,A

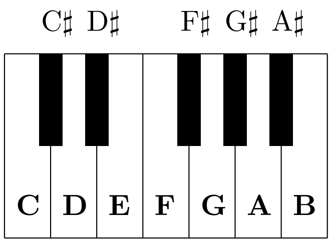
DO-RE-MI-FA-SO-LA-TI-DO, a notation called solfège, will be familiar to anyone who has seen the movie or musical *The Sound of Music*.

A *musical staff* consists of 5 parallel horizontal lines with circles (notes) on it. Horizontal position denotes time (read left to right) and vertical position denotes note frequency. Both the lines themselves, and the spaces between lines, are used to represent *natural notes* only; accidentals (sharps and flats) have a "♯" or "♭" in front of the note, respectively. This may seem messier than MIDI notation, but because accidental notes occur less often, it can simplify reading the notation while playing an instrument (called *sight-reading*).

Here is musical staff representation of natural notes.

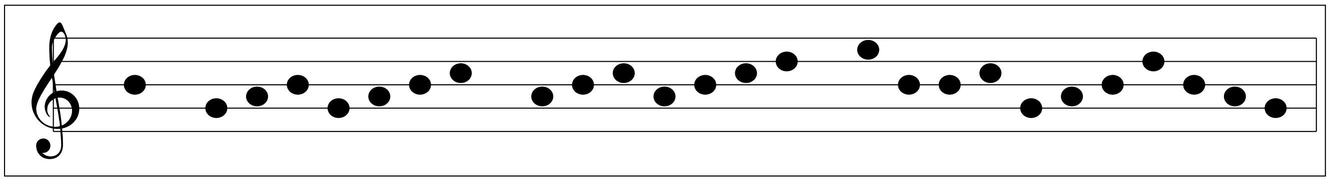


Here are notes on a piano keyboard.



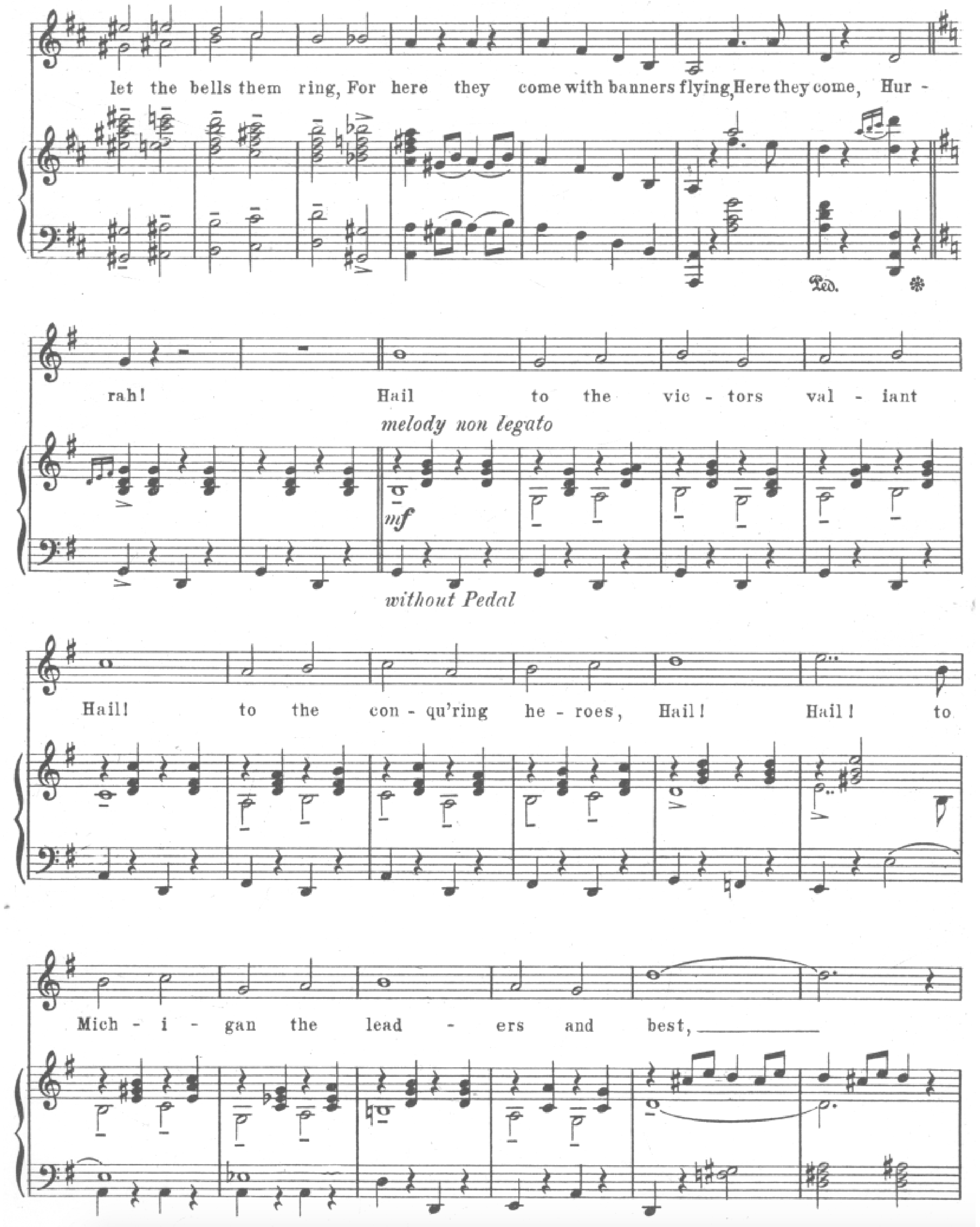
The symbol that looks like a script G in front is called a *treble clef* sign. The small circular lower end encircles the line representing note G. For the treble clef, G is the second line from the bottom, as shown above. Lines that look like flag staffs and flags extend from the circles to represent the duration of that note. We will ignore bass clefs and the rest (including rests) in this project.

A musical transcription of the chorus of "The Victors," *without correct rhythms*, looks something like this:



RQ Proj1.1. What is the MIDI number of the note A♯ that is right above A 440?

Here is actual sheet music for the chorus of "The Victors" (obtained from the UM web site). This tune has a long history.

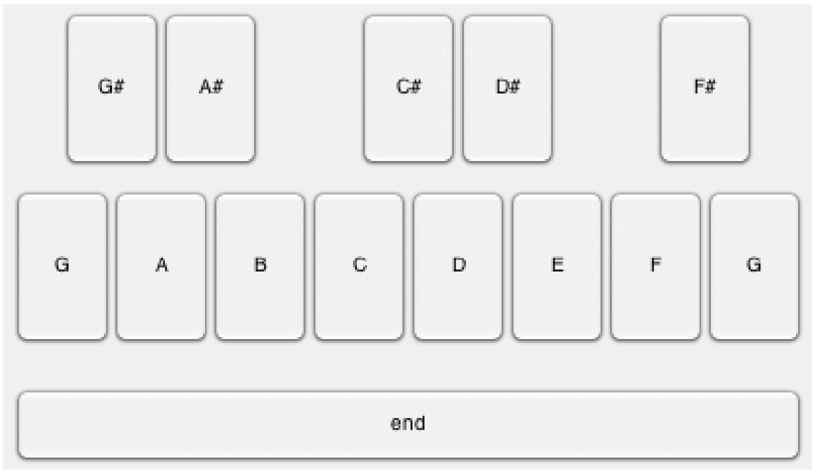

let the bells them ring, For here they come with banners flying, Here they come, Hur - rah!
Hail to the vic - tors val - iant
Hail! to the con - qu'ring he - roes, Hail! Hail! to
Mich - i - gan the lead - ers and best,
melody non legato
mf
without Pedal
Ped.

4 `Matlab` graphics commands used in this project

You will need to use new `Matlab` commands in this project as discussed below.

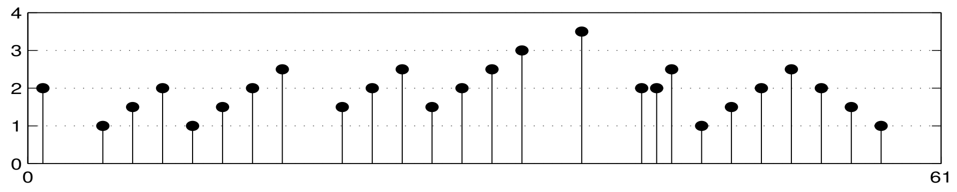
- `uicontrol('Style', 'Pushbutton', 'Position', [200 400 50 100], ...`
  `'String', 'A', 'Callback', 'sound(cos(2*pi*440*[1:3000]/8192))');`
  (You can put all of this command on one line, without the "`...`" line continuation symbol, or type as is.) This command creates an on-screen rectangular pushbutton of size 100×50 screen pixels, with lower left corner 200 pixels from the left border and 400 pixels above the bottom border, and puts an 'A' in the middle of it. So this pushbutton has corners at $\{(200,400),(250,400),(200,500),(250,500)\}$, in Cartesian coordinates where the origin is the lower-left corner of the figure window, and distance is measured in pixels. (You might need to enlarge the figure window to see it.) When the button is "pushed" by putting the mouse cursor over it and left-clicking, `Matlab` executes the command shown as the "callback" function. Here, that command sounds a 440 Hz tone for $\frac{3000}{8192}$ seconds. The role of each command argument should be clear. The callback argument can be a sequence of `Matlab` commands separated by semicolons, *e.g.*, `'load train; sound(y)'` which first loads the `Matlab` file `train.mat` (that happens to contain a variable named "`y`") and then it plays the sound stored in that variable.

  You will use a sequence of commands like this to create an on-screen keyboard, labeled with all of the various notes, and with the accidental notes (sharps and flats) in a row above the whole notes, just like an actual piano keyboard. You can even color the pushbuttons black and white, or you can label them.



- An alternative to `uicontrol` is to use the `guide` (GUI developer) command in `Matlab` that allows interactive interface design. You are welcome to use this option but you would need to learn how to use it by studying the documentation.
- In this project, you must design the command for each pushbutton so that it both sounds the corresponding tone *and* concatenates that tone with all previous ones for later playback of the entire sequence of pushed buttons. In `Matlab`, if `x` and `y` are two (row) vectors (possibly of different lengths) then we concatenate them to make a new (longer) vector `v` by typing `v = [x y]`. Similarly, if we have a vector `x` and we want to append a vector `y` to it, the command is `x = [x y]`.
  When the synthesizer begins, you will want to initialize an "empty" song, and you can do this by setting a variable, say `x`, to be an empty array by using `x = [ ];`
- Therefore, the callback commands needed here look something like:
  `z = cos(2*pi*440*[1:L]/S); x=[x z]; sound(z,S)`
  This command generates a sinusoidal signal `z` of a certain frequency, appends it to the vector `x` (for later playback) and plays the sound. This command assumes a length of `L` samples and sampling rate `S`.

- `subplot(311), stem(x,'filled'), axis([0 length(x) 0 4])`
  This command produces a stem plot of the numbers in vector `x` with axes as shown.
  To see an example stem plot, try: `stem([3 1 4 1 5 9])`
  You can fill in the circles using `stem(x,'filled')`, and can even change the stem lines.
  You may prefer to use the following command instead:
  `subplot(311), stem(x,'filled'), axis([0 length(x) 0 5])`
  This version will allow the "high G" on your keyboard appear, but it looks a bit less like a musical staff.
- `set(gca, 'ygrid', 'on', 'ytick', [0 1 2 3 4])`
  This changes the graph so that it looks somewhat like musical staff notation (close enough for us).
  For solid lines, add `'GridLineStyle', '-'` to this command. You can also add labels, titles, etc.



- `save proj1.mat x y`
  Saves the `Matlab` variables `x` and `y` to a file `proj1.mat`.
- `load proj1.mat`
  Loads all of the variables saved in `proj1.mat` into `Matlab`. Caution: if there is a variable in the file with the same name as a variable you have in your current `Matlab` session, then after the `load` command the current variable will be replaced by the one in the file instead.

5 Project 1: What you must do

The results of this team project are two `.m` files, one implementing a *synthesizer* and one implementing a *transcriber*. You also have to demonstrate to your lab instructor that they work, as described below.

Remember throughout that sharps and flats complicate things; MIDI transcription and this lab would be easier without accidentals, but life is not like that. (Engineers must remember this.)

5.1 Musical tone synthesizer: On-screen keyboard

Write a `Matlab` program (and store it as an `.m` file) that:

- Creates an on-screen keyboard, using a sequence of `uicontrol` commands, that resembles the one octave of a keyboard on which the chorus of "The Victors" is played.
- Keys should be arranged to look reasonably similar to those on a piano;
- The keyboard should run from "G" to "G" and include, in a row above, the accidentals in between;
- Produces the appropriate tone when pressed by left-clicking the mouse on it;
- Includes an "end" key that plays back the notes in order, and writes the entire signal to `proj1.mat`;
- Makes each tone last $\frac{2000}{7999}$ seconds at a sampling rate of $7999\frac{\text{Sample}}{\text{Second}}$;
- Show you can play "The Victors" and "Taps" on it. You may need to practice to get the notes right;

Notes and hints:

- This program is almost entirely a sequence of `uicontrol` commands, one for each piano key.
- Using a sampling rate of $8192\frac{\text{Sample}}{\text{Second}}$ would be more standard, but that would make some values of the tonal signal zero, and then the frequency estimation algorithm would try to divide by zero! (See if you can figure out why.) Using $7999\frac{\text{Sample}}{\text{Second}}$ simplifies this project.

5.2 Musical tone transcriber

Write another `Matlab` program (and store it as an `.m` file) that:

- Loads the file `proj1.mat` generated by your synthesizer above;
- Prints out on the screen a musical transcription of the tones in simplified musical staff notation using `Matlab`'s `stem` command;
- Uses the frequency estimation algorithm used in Lab 2, assuming each tone lasts $\frac{2000}{7999}$ seconds, to determine the frequencies of each note;
- Only has to work for the octave in which the chorus of "The Victors" is played;

Notes and hints:

- Use `load('proj1.mat'); y=reshape(x,2000,length(x)/2000);`
  where `x` is your synthesizer output.
- To determine the frequencies of each tone, use a command like one of the following:
  - `F=7999/2/pi*acos(mean((y(3:2000,:)+y(1:1998,:))./y(2:1999,:)/2));`
  - `n=1000; F=7999/2/pi*acos((y(n+1,:)+y(n-1,:))./y(n,:)/2);`
- For translating the frequency estimates into notes, let `midi=round(69+12*log2(F/440))`.
  Explain why `round` is likely needed here. (Hint: try without it.)
- Consider `stem(V(midi-63))` where the vertical positions are defined in an array:
  `V=[0 .5 .75 1 1.25 1.5 1.75 2 2.5 2.75 3 3.25 3.5 4 4.25 4.5]`
  Explain why the `-63` is used here. (Hint: think about the note "A" with frequency 440 Hz.)

5.3 Project presentation

- Your team will present the results of this project in an *oral presentation* to your discussion section.
- Also address these issues in your presentation (more details from your discussion leader):
  - Should you be able to copyright any sequence of tones you can create?
  - If we can write tonal signal as a mathematical expression, can music legitimately be copyrighted?
- Upload a copy of your presentation to Canvas as a `.pdf` file, *not* as a `.ppt` file.
- Your discussion instructor will provide further details about the presentation requirements, and may also require a printed copy for grading.
- One member of your team must upload to Canvas a zip file with your two solution m-files, named `p1_`*yourteamname*`1.m` and `p1_`*yourteamname*`2.m`. (Do not include any spaces or unusual characters in *yourteamname*.) These files are due by the start of the lab section on the day you give your oral presentation.

RQ Proj1.2. What is the duration of the tone produced by the following `Matlab` command?
`sound(cos(2*pi*[1:16384]*587/8192))`