BIOL 708L/NACS 728B Lab #2
Fall 2011

Graphing sinusoidal signals in MATLAB

The main goal of this lab is to examine periodic sinusoidal signals. The secondary goal is to graph multiple signals simultaneously in Matlab (and save the figures for use in a presentation, poster, paper, or for further editing). There are many ways to graph several signals at the same time.

All signals in this lab should be defined for N points, and the individual points, n, that we use, go from 0 to N-1. In this case we will use $N = 8$. In Matlab, `N = 8` and `n = [0:N-1].'` (it's important that `n` be a column).

1) *Graphing a single signal with the correct index values*. Consider the periodic sinusoidal signal $x_{c1} = \cos\left(2\pi \frac{1}{N} n\right)$ (note that the last part of the signal name is a "one" not an "ell"). Define it in Matlab, calling it `xc1`, using `n` and `N` defined above.
   a) What is its angular frequency ω? What is its frequency f? What is its fundamental period?
   b) Open a new figure window and graph `xc1` using `bar()`. Make sure you use the two input form of `bar()` (that is use both `n` and `xc1`) or the horizontal axis will not have the correct values.

2) *Graphing two signals using subplots*. Consider also the periodic sinusoidal signal $x_{c2} = \cos\left(2\pi \frac{2}{N} n\right)$. Define it in Matlab, calling it `xc2` using `n` and `N` defined above.
   a) What is its angular frequency ω? What is its frequency f? What is its fundamental period? Is f simply related to the fundamental period? If so, how?
   b) Open a new figure window. We will graph `xc1` and `xc2` in the same figure window, using the Matlab command `subplot()`. `subplot()` makes a *visual matrix* of graphs in the window (e.g. puts four graphs in a 2 x 2 format, or six graphs in a 3 x 2 format, or six graphs in a 2 x 3 format). `subplot()` takes 3 arguments: the first two are the size of the matrix graph (e.g. 2 and 2, or 3 and 2, or 2 and 3 for the above examples); the third is which element of the matrix to use for the next graph. It is best seen by example. In this case, graph `xc1` above `xc2`, e.g. in a 2 x1 matrix:

```
figure
subplot(2,1,1)
bar(n,xc1)
subplot(2,1,2)
bar(n,xc2)
```

3) *Graphing two signals using subplots revisited*.
   a) Repeat (2), **including part (a)**, but for the two signals $x_{c4} = \cos\left(2\pi \frac{4}{N} n\right)$ and $x_{s4} = \sin\left(2\pi \frac{4}{N} n\right)$, and this time graph them side by side instead of one above the other.
   b) Examining the formula for x_{c4} in more detail, how should it behave? Does the graph behave like it should? The answer may be no.

c) Examining the formula for x_{s4} in more detail, how should it behave? Does the graph behave like it should? The answer may be no.

Make sure you are confident in your answers to (b) and (c) before proceeding.

d) You can fix the axes on your graphs by hand by using the `axis()` command. `axis()` takes a single argument which is a four component array. The first two components are the left and right ends of the horizontal ("x") axis. The second two components are the bottom and top ends of the vertical ("y") axis. Try using left and right ends of -1/2 and *N*-1/2, and bottom and top ends of -1 and 1. You'll need to reuse `axis()` after every `bar()`. Does this help with your answers to (b) and (c)?

e) Bonus: use subplots to graph all the signals of the form x_{cM} and x_{sM} where *M* goes from 0 to *N*-1. Plot all the cosine signals down the left and all the sine signals down the right. We will see this figure in lecture.

4) *Graphing two signals using hold*. Create the two signals $x_{c3} = \cos\left(2\pi \frac{3}{N} n\right)$ and $x_{s3} = \sin\left(2\pi \frac{3}{N} n\right)$ in Matlab. We will graph x_{c3} and x_{s3} in the same graph, on top of each other, using the Matlab command `hold`. Normally when you use `bar()` twice in a row, Matlab erases the first bar graph before drawing the second. The command `hold on` instructs Matlab to "hold" the graph instead of erasing it. Since the bar graphs are now overlaid, we need to tell Matlab to draw each in a different color. The first one will be blue (`'b'`) and the second red (`'r'`).

a) Draw the graphs using `hold`:

```
figure
bar(n,xc3,'b')
hold on
bar(n,xs3,'r')
hold off
```

b) Fix the axis as in (3d). You only need to do it once at the end, not after each `bar()`, since they share the same axes.

c) For each signal, what is its angular frequency ω? What is its frequency *f*? What is its fundamental period? Is *f* simply related to the fundamental period? If so, how?

5) *Graphing two signals at once*. Use the same two signals x_{c3} and x_{s3} in Matlab. We will graph x_{c3} and x_{s3} in the same graph, on top of each other in a different way.

a) What is the size of `xc3`? of `xs3`?

b) Create a new matrix called `x3` which has the same vertical dimension as both `xc3` and `xs3`, but which is twice as wide and whose first column is the same as `xc3` and the second column is the same as `xs3`. That is, concatenate `xc3` and `xs3` horizontally. Open a new figure window and use `bar()` to graph `x3` (with the correct index values). If it works, the first one will automatically be blue and the second red.

c) Fix the axis as in (3d). You only need to do it once since there is only one `bar()` command.

6) Prettifying and Using Matlab graphs.
   a) Pick the last of the figures you made. Note its figure number (e.g. 6). Type `figure(6)` (replace "6", here and below, by the number of the figure you are using, if necessary) to make it the current figure. You can also use the Matlab function `gcf` ("get current figure") instead of "6".
   b) Save the figure, so you can reload it later if you need it: `saveas(6,'myfigure')` will save the figure in the current working directory with the name `myfigure.fig` (you can print the working directory using `pwd`). You can reload the figure later using `open('myfigure.fig')`. Both saving and opening the figure can also be done with the mouse and menus.
   c) Add the title "Cosine and Sine for f = 3/8" using the command `title()`.
   d) (Bonus) Instead of (c), figure out how to used `title()` to add the that title and *simultaneously* make its `'FontName'` property `'Times'` and make its `'FontSize'` property `12`.
   e) (Bonus) In addition to adding a title to a graph, you can also change the name of the window itself by changing its `'Name'` property. (This can make the window much easier to find if you have a dozen figure windows on your screen.) Here is an example: `set(gcf,'Name','f = 3/8')`.
   f) Add an x-axis label of "time/space index" with `xlabel()`. Add a y-axis label of "signal strength" with `ylabel()`.
   g) Add a legend with `legend()` and the 2 strings: `'cosine'` and `'sine'`, and after the last string, put `0` (zero) (which indicates that Matlab should draw the legend in an empty corner). You can then drag the legend where you want it. Each string in the legend should appear in the appropriate color.
   h) Copy the figure into a Powerpoint slide. This is the easiest method to put a Matlab figure into a presentation, if it works. Check the y-axis label—that's the most likely thing to go wrong. *Warning: for many versions of Matlab, this only works for Windows, not Macintosh or Unix.*
   i) Exporting figure files in standard graphic file formats can be a life-saver when copying and pasting doesn't work, or when you want to send the figure to a colleague who doesn't use Matlab. However, not every graphic format works in all circumstances.
      i) *Vector* graphic formats. Almost all graphical information is saved and can be edited and reformatted in a graphics program—this is always the most desirable type of format:
         (1) Save the figure as an EPS file: `saveas(6,'myfigure','epsc2')`. EPS files are excellent when the file will be imported into an intelligent graphics program, such as Adobe Illustrator, Canvas, CorelDraw, and others. PowerPoint can import the EPS file but will only print it, not display it. Open the EPS file if you can.
         (2) Save the figure as a PDF file: `saveas(6,'myfigure','pdf')`. Like EPS files, PDF files are excellent when the file will be imported into some but not all graphics programs. The have the extra advantage that almost anyone can read them (but not edit them) with Acrobat Reader. Open the saved PDF file if you can.

(3) Save the figure as a Adobe Illustrator file: `saveas(6,'myfigure','ill')`. This is less universal than EPS or PDF (and less supported by Matlab) but it might help in a crisis.

ii) *Bitmap* graphic formats. Use only in emergency. Bitmap graphic formats lose information, cannot be edited, are usually large, and often turn fonts and lines into blocky messes.

(1) Save the figure as a TIFF file: `saveas(6,'myfigure','tiff')`. Of all bitmap formats, TIFF files lose the least information. Open the file, and examine what happened to the fonts and the curves.

(2) Save the figure as a PNG file: `saveas(6,'myfigure','png')`. PNG files are the successors to the older GIF files. They are not as useful as TIFF files, but they may be smaller.

(3) Aside from this exercise, you should **never** do this: Save the figure as a JPEG file: `saveas(6,'myfigure','jpeg')`. **JPEG files are the worst of all file formats for Matlab graphics**. They compress the bitmaps by spatially lowpassing (blurring) the figures and then redigitizing at a lower bit density. They will often look smeared or blurry. Open the file, and examine what happened to the fonts and the curves.

j) Print the figure (this may not work in the lab, depending on the printer):

i) From the figure window, go to the File menu and choose Page Setup.

ii) From the Page Setup window, go to the "Paper" tab, and for orientation, choose Landscape.

iii) Still in the Page Setup window, now go back to the to the "Size and Position" tab and click the "Fill page" button. Click the "OK" button at the bottom.

iv) From the figure window, click the Print icon and proceed as you can.