Problem Set 1

Biological Physics

Due Thurs., Feb. 4, 2016

A) A 1-Dimensional Random Walk

For this exercise, we are going to simulate a 1-dimensional random walk.

1. Produce a `8x1000` array of `steps`, where each step is either –1 (left) or +1 (right).

    - *Tip:* The `rand` function will produce an array of numbers between 0 and 1. I suggest combining this with the `round` function and some simple arithmetic to transform it from a distribution *between* 0 and 1 to a distribution of *just* –1 and +1.

2. From the `steps` array, produce an array of `locations`. You will want to use the `cumsum` function. Pay careful attention to the second argument of the `cumsum` function; it tells you whether you are cumulatively summing over columns or over rows. We want to have 1000 tracks of 8 steps each.

3. **[Output]** Make a plot of the location of the first 6 tracks.

    - *Example code:* `plot(locations(:, 1:6));`

4. **[Output]** Use `xlabel` and `ylabel` to label the axes of your graphs appropriately.

5. **[Output]** In the figure window, go to `File` → `Save As...` and **save your figure as an EPS file**. Note that the default, `Matlab Figure (*.fig)`, is only openable in Matlab, but an EPS file is useable in many places. You can also use the `saveas` command.

B) Statistics

Let's find the distribution of final locations of each of these tracks.

1. Use *indexing* to get a `1x1000` array of the *final* location of each track.

2. **[Output]** Calculate the mean and variance of these final locations. Also calculate the expected theoretical value.

3. Let's make a histogram and plot $P(x)$, where x is the location of a particle after 8 steps. Using your results from 1, use the `histogram` command to make a plot.

   (a) Create your `bins`, i.e. an array of points *between which* you want to count the number of trajectories. For example, if you wanted to count all the points at 0, 10, and 20, an array $[-5, 5, 15, 25]$ would be an appropriate array for the bins.

   (b) **[Output]** Use the `histogram` function to plot the values above, normalized as a probability density.

4. Plot the binomial distribution on top of your histogram.

   (a) Use the `hold all` command before you plot to make sure that the histogram does not disappear. (If something goes wrong and you need to clear the figure and start over, you can close the window or use `clf`.)

   (b) Calculate the centers of your bins.

   (c) Calculate the binomial distribution at each of the bin centers.

   - You may want to use the `factorial` function.

   (d) **[Output]** Use `plot` to overlay the theoretical result on top of the simulation results, with format `'ko-'` to produce a black (`k`) line (`-`) with circular markers (`o`).

5. Plot the normal distribution on top of your histogram.

   (a) Create a new array of x values with much finer spacing than before, perhaps with `linspace`. Use at least 200 values for x.

   (b) Calculate the appropriate y values for the normal distribution.

   (c) **[Output]** Use `plot` to overlay the normal distribution on top of your other results, with format `'r-'` to produce a red (`k`) line (`-`).

6. Finish your plot and save it.

   (a) **[Output]** Use `xlabel`, `ylabel`, `title`, and `legend` to appropriately label your graph.

   (b) **[Output]** save as an EPS file, as before (5).

7. Do the same thing for 20000 tracks with 1000 steps.

   (a) Make a figure for the first 6 tracks (3), a histogram for the end results (3), and plot the normal distribution on top (5). You may skip the binomial distribution, as calculating it for $N = 1000$ may involve numbers too large for your computer (1000!).

   (b) **[Output]** label your plot appropriately and save as an EPS file, as before (5).

Submission

To submit your problem set, please save all the code, figures, and output to the same folder, compress the folder into a zip file (or `.tar` / `.tgz` if you wish), name the file *`LASTNAME-FIRSTNAME-PS1.zip`*, and upload the compressed file to the `Assignments` section on the classesv2 server. Here are the steps:

1. First, gather all your code in one place. If you already have it in a `.m` file, that's great; if not, open the Matlab Editor, copy and paste *all* the necessary commands into the editor (without the `>>`), and save it as a `.m` file. To test that it works, you can type `clear` into the main window, close all your figures, and then go to the editor and press `F5`. Your code will all run in the main window, and you should see the same output as before.

2. Next, put all your text output in a `.ps`, `.pdf`, or `.txt` file. You can do this however you would like, but the simplest way is to close Matlab, reopen it, run your code, and then go to `File` → `Print` and check the box `Print to File`, and then hit `Print`. You can also use the `diary` command, or Cut and Paste into a `.txt` file. **Please do not turn in a .doc**; if you want to use Word, export it as a PDF.

3. You should also save any figure you have created to this same folder.

4. Finally, you should put all of this in a single compressed file. There are instructions for Windows here and for Macs here. You can also use the `Current Folder` pane to the left of Matlab, select all the files you want to compress using `Ctl`-Click, and then right-click and select `Create Zip File`. You can also use the `zip` or `tar` functions; see the Matlab documentation. *Name your file* *`LASTNAME-FIRSTNAME-PS1.zip`* *before you upload it.* Make sure to get the extension right if you made a `tar` file. **Please do not turn in a rar file.**

5. Upload to the classesv2 **Assignments** section.