

Psych 254, Fall 2007
Exercise #2. Due Oct 17

Please submit your programs by emailing them to me at poldrack@ucla.edu. Please be sure to put your name and the exercise number in the comment section at the top of each program.

Problem 1. Your job in this problem is to alter the `guessing_game.m` program used in class.

Prior to working on this problem, you should save the `guessing_game.m` program from the Psych 254 web site into a local work directory so that you can alter it.

After doing this, you should modify the `guessing_game.m` program do to the following (saving it to the filename “`guessing_game_<name>.m`”, where you replace `<name>` with your last name, and also putting a comment with your name at the top of the program):

- Perform the game 20 times in a row (using a new random number each time). Keep track of the outcome on each trial.
- Change the program to use magic numbers from 1 to 8.
- After the 20 guesses, determine whether the subject was above chance at guessing. Because you can treat each trial as an identical independent Bernoulli event, you can model the number of successful guesses using a binomial distribution, where the probability of correct guesses by chance is $1/8$. You can obtain the cumulative distribution function for the appropriate binomial distribution using the following function:

```
binomial_p = 1 - binocdf(x-1,n,p)
```

where

x: the observed frequency of the event (number, not probability)

n: the number of observations

p: the theoretical probability (not frequency) of the event (in this case, $1/8$ for guessing)

The CDF is subtracted from one because the function returns the cumulative density up to X, whereas we want the probability of a value larger than X, which is one minus the cumulative density to that point. It is necessary to use `x-1` instead of `x` because `binocdf()` computes the $\Pr\{X \leq x\}$, such that `1-binocdf()` gives $\Pr\{X > x\}$. However, we really want $\Pr\{X \geq x\}$, which we can get by subtracting 1 from x.

Note: The functionality of `binocdf()` is duplicated by the `spm_Icdf()` function from the SPM99 package, which is an analysis package for fMRI data. If you are working on a machine that does not have the MATLAB statistics toolbox, you can install the `spm` files (available on the course web page) and use that function instead of `binocdf()`.

TIP: First test the program using a smaller number of guesses to make sure that it runs all the way through, then run it with 20 guesses. Otherwise, if there are bugs at the end of the program you will waste time going through all 20 guesses each time only to have it crash. Implementing a short version of the program first often helps in debugging.

d. Print a brief summary reporting the number of correct guesses and the probability of this number (or more) correct guesses under the null hypothesis of chance behavior. You should print this summary to the screen, and also print it to a text file, using `fprintf` in both cases.

2. In this exercise you will read in a dataset and perform a number of analyses on it. These are simulated data from a study in which subjects made lexical decisions about words from two different conditions (high versus low frequency) over 400 trials; in each of those conditions, half of the stimuli were words, and half were nonwords, leading to four possible conditions: HF words, HF nonwords, LF words, and LF nonwords, with 100 items in each. On each trial, the subject pressed either the W (word) or N (nonword) key. Please name your program “lab2_ex2_<name>.m” replacing <name> with your last name.

You will first need to save the data file for this exercise, called `ex2rt.txt`, from the course web site into your local work directory.

a. Read in the data from `ex2rt.txt`. There are four columns in the data file.

Condition	RT	response	Word/Nonword
h	781	w	w
l	812	n	n
...			

b. Determine for each trial whether it was correct or not, and compute mean accuracy for each condition. TIP: Computing mean accuracy (and RT in step C) will be easiest if you first create a variable for each condition that contains an index of all the correct trials in that condition. This should be done using the `find()` command.

c. Compute the mean reaction time for correct responses only in each condition, along with the standard deviation of those RTs (you can use the `std()` function to compute the standard deviation). Print a nicely formatted report of the RT and accuracy data to the screen.

d. Create a histogram for the response time distribution of correct responses, separately for each condition. Normalize the histograms so that they have a maximum (y-axis) value of one, so that all conditions will plot on the same scale. Plot these histograms together in the same figure as line graphs, with the different conditions in different colors. (TIP: MATLAB has a built-in function for creating a histogram. You will need to use the

version of this function that returns both the histogram values and the bin values in order to plot the data the different conditions.) Include a legend and axis labels in the figure.