# Magnitude Estimation of the Müller-Lyer Illusion

## Introduction

Our first lab exercise will be to perform a magnitude estimation experiment. Your mission, therefore, is to study some aspect of the Müller-Lyer illusion using the Magnitude estimation technique. We have provided a fairly generic experimental program. The program has lots of parameters that will allow you to customize your experiment and optimize the design.

## Setup

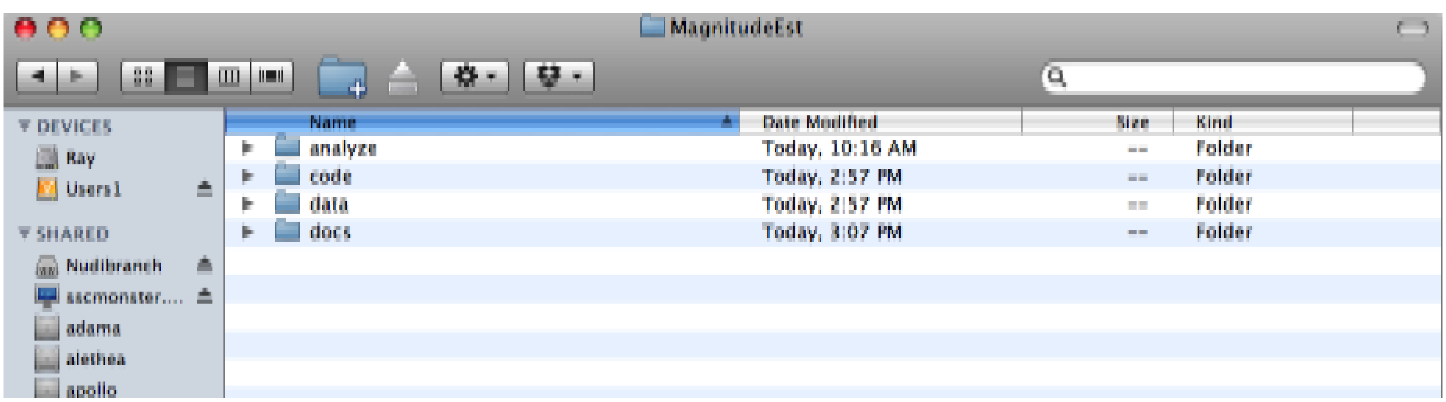
*Course Computers/Rigs*

Everyone in the course will have accounts on the course rigs in Room 308 C. There are five rigs. Each consists of an iMac plus a second monitor. (There is also a server in this room, but you won't need to log into that.) We have set up the systems so that the five rigs should be close to interchangeable. When you log into any of the rigs, your home folder will be synchronized with the server, so everything should look the same to you on all of the rigs. When you log out, there is a second synchronization. In addition to making the rigs interchangeable, this procedure provides backup of your account.

When you first log in, your account will be in the default configuration that Apple provides. There will be a lot of stuff on your dock that you don't need for the course. The application you need most, Matlab, won't be there. If you navigate to the Applications folder on the machine, you can drag Matlab to the dock and it will be esay to launch in the future. You may also want to mess with some of the system preferences. For example, the default mouse speed is very slow and you can speed it up. And you can adjust the Finder preferences to make it easier to navigate around the machine, in particular so that the hard disk icon shows up on the desktop. If you have questions about basic MacOS setup, please ask me.

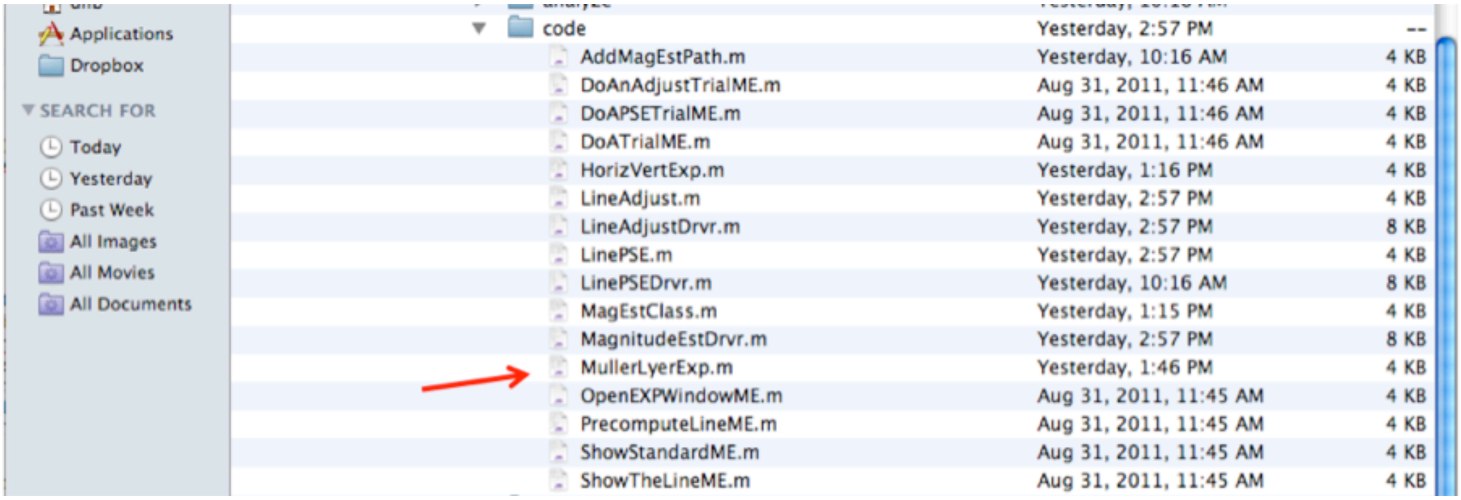
*Setting up for the lab*

We will show you how to obtain the lab materials using SVN/Versions. When you open the folder, you should see the following directory structure.



The experimental code lives in the code subfolder. The data end up in the data subfolder. A sample analysis program is available in the analyze folder (and you can put your data analysis program there when you write it.) And this document is in the docs subfolder. It's called MullerLyer.doc/pdf.

In the code subfolder, there are a bunch of Matlab files. There is only one you need to know about for this lab, although you are of course welcome to look at all of them. The relevant file is MullerLyerExp.m.




Launch Matlab, open the editor (type edit in the command window if it isn't there), and open MullerLyerExp.m in the editor (you can drag it into the editor as a quick way to open). You're now set to go.

## Changing the parameters

The organization of this laboratory program is like most of those we shall use this semester, so let's look carefully at how things work. Most programs that we will use involve a set of parameters that define the conditions you wish to test. For example, in MullerLyerExp.m you have a list of stimulus lengths (the shaft of the arrow), stimulus angles (the direction away from the shaft the "feathers" point) and arrow lengths (the lengths of the "feathers"). Each combination of shaft length, arrow length and arrow angle is run once, in random order, resulting in a single block of trials. Finally, that group of trials is repeated until the number of blocks you specify (parameter *nBlocks*) have been run. Thus, each individual condition will be shown *nBlocks* times.

For the Muller-Lyer experiment, each parameter is one field of a parameter structure (called params), and is set by a line in the Matlab program file MullerLyerExp.m. When you open up the MullerLyerExp.m, you will see a section like the one on the next page. You may edit any of the parameters. The meaning of each parameter is indicated briefly by a comment to its right and the parameters are described in more detail below.

There are three kinds of parameters. A parameter set to be a single number takes on that value for the entire experiment. A description of a color is represented as a list of three numbers in square brackets representing, in order, the amount of red, green and blue added together to make the color, where each value can range from 0 to 255. For example, [0 0 0] is black, [0 255 0] is green, [255 255 0] is yellow, and [255 255 255] is white. Other parameters that are defined using square brackets are lists. You may have multiple items in these lists, or only one. Elements without brackets may have just one value.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% EXPERIMENTAL PARAMETERS
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
params.standardLength = 100;                      % Standard line length
params.stimLengths = [70 80 90 100 110 120 130];  % Test lengths (pixels)
params.stimAngles = [45];                         % Test angles (degrees re. horizontal)
params.arrowLengths = [30];                       % Arrowhead lengths (pixels)
params.arrowAngles = [45 135];                    % Arrowhead angles (degrees re. inward)
params.nBlocks = 4;                               % Number of blocks
params.penSize = 4;                               % Line thickness (pixels)
params.trialDuration = 500;                       % Trial duration (milliseconds) ...
                                                  % trialDuration = -1 means mouse click ends trial
params.blankDuration = 10;                        % Time before response box goes up (millisecs);
params.ITI = 100;                                 % Intertrial interval (millisecs)
params.bgColor = [0 0 0];                         % Color of background (RGB, each between 0 and 1)
params.lineColor = [1 1 1];                       % Color of arrows (RGB, each between 0 and 1)
params.arrowColor = [1 1 1];                      % Color of arrowheads (RGB, each between 0 and 1)

params.jitterHPosition = 0;                       % Vary horizontal line position trial to trial?
params.jitterVPosition = 0;                       % Vary vertical line position trial to trial?
                                                  % 0 means no., nonzero gives
                                                  % maximum number of pixels of shift
                                                  % (left or right, up or down)

params.standardAlways = 0;                        % Show standard on every trial?
params.standardDuration = 750;                    % If standard shown, duration
params.standardITI = 200;                         % If standard shown, time to wait before line.

params.experimenter = 'dummy';                    % Experimenter
params.experimentName = 'MullerLyer';             % Root name of the experiment and data file.
params.subject = 'subject1';                      % Name of the subject.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

## Parameter field descriptions

standardLength -- This determines the length of the line that gets called 100 at the start of the experiment. The length is specified in units of pixels, which are somewhat arbitrary.

stimLengths -- This list determines what stimulus lengths are presented. You can set up multiple line lengths.

stimAngles -- This list determines the angles at which the lines are presented, in degrees relative to horizontal. You can set up multiple angles

arrowLengths -- This list determines the lengths of the arrows that will be drawn for the Müller-Lyer illusion. You can set up multiple arrow lengths.

arrowAngles -- This list determines how steep the arrows are relative to the line. An angle of 0 means no arrow at all. You can set up multiple arrow angles.

nBlocks -- This determines the number of trials presented for each (stimLength,stimAngle, arrowLength,arrowAngle) combination.

penSize -- This determines the thickness of the lines drawn.

trialDuration -- This determines how long in milliseconds the trial is presented before the response box goes up. If you set this to -1, it means that the trial stays up until you click the mouse. This option may be useful if you wish to measure the line lengths on the screen.

blankDuration -- Time in milliseconds between when the trial goes off and the response box comes on.

ITI -- Time in milliseconds after response and before next trial (inter-trial interval).

bgColor -- RGB settings for the background

lineColor -- RGB settings for the lines.

arrowColor -- RGB settings for the arrows.

jitterHPosition -- Specifies size of random left/right position jitter on every trial. Set to 0 to have lines centered.

jitterVPosition -- Specifies size of random up/down position jitter on every trial. Set to 0 to have lines centered

standardAlways –- Set to 1 to show standard on every trial, 0 otherwise.

standardDuration – Time in milliseconds that standard will be shown, if it is shown.

standardITI –- Time in milliseconds before trial, after standard is shown.

experimenter – This string determines the top level name of the directory tree where the data files are stored. See extended comment under subject field below.

experimentName – This string determines the mid level name of the directory tree where the data files are stored. See extended comment under subject field below.

subject – This string determines the bottom level name of the directory where the data files are stored. All in all, then, the data appear in a directory called data/experimenter/experimentName/subject. The individual data files are named as experimentName-N.csv, where N represents the Nth repeat of the experiment. If you change parameters, you'll want to keep a note of which data files go with which parameters. Indeed, the data files for this experiment do not contain all of the parameter values, so you will want to record these in some way in any case. Note that you will set the three strings (experiment, experimentName, and subject) to match in your data analysis program.

## Your mission

Your mission is to make measurements of the Müller-Lyer illusion. You should (in conjunction with your lab partners) consider some of the following issues, as well as anything else you can think of that will make your experiment insanely great.

a) There are many stimulus attributes that might affect the size of the illusion. These include the length of the lines, the length of the arrows, and the angle of the arrows. You probably will not have time to study all of these, so the first step is to pick your question. For example, you may wish to know how the illusion varies with the angle of the arrows, for several different line lengths. Alternatively, you might try to measure how the illusion varies with the length of the arrows and whether this interacts with the horizontal vertical illusion. For this lab, I recommend picking a relatively simple question.

b) What parameters do you want to use? This will depend on the question you try to answer. Note that the total number of trials per session is the product of the number of stimulus lengths, stimulus angles, arrow lengths, arrow angles, and number of blocks. You want to have a fair number of blocks to get an accurate measurement. But you don't want an individual session to last too long. (It's hard to concentrate if the session goes on and on and on.) One strategy is to repeat the same experiment a few times for the same subject. For example, you might prefer to run three sessions of two blocks each rather than one session of six blocks. It is fairly easy to aggregate the data using an Excel spreadsheet.

c) How many subjects will you run? Will you average data across subjects or use individual subject data? Will you repeat sessions for each subject, or run each subject through only once? If you repeat, will you use the same design in each session?

d) If the subject knows something about the experimental parameters, especially the range or exact values of line lengths, she might find it difficult to avoid being influenced by that knowledge when making magnitude estimations. Think about how you might try to prevent the subject from answering on the basis of knowing the parameters. Here are a couple of suggestions:

1) when setting the experimental parameters, use values that are close together (e.g.: 97, 99, 101, etc.)
2) use "oddball" values (e.g.: 83.3692)

## Who Should You Use As Subjects?

There are important guidelines about the ethical use of human subjects in research, and under some circumstances the use of human subjects must be reviewed and approved by an Instutitonal Review Board (IRB). Because this is a class exercise, the formal regulations do not apply and you can recruit each other or friends to serve as subjects for this initial lab.

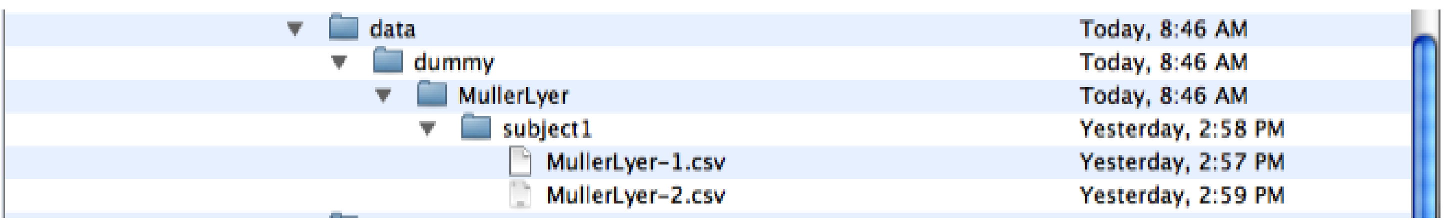
The one principle to keep in mind even at this stage is that participation should be voluntary. For purposes of this lab, that means don't pay anyone to be a subject, and don't coerce them in any way. If they decide to quit halfway through, smile and thank them. We will talk about human subjects regulations in more detail later in the course.

In terms of the substance, you may want to think about whether you want to run naïve subjects (i.e. your friends who don't know about the experiment, yourselves (i.e. you and your lab partner who know the details of your experimental design), or perhaps your classmates (who know generally what you are up to but perhaps not the specifics of your design.) How much the subjects know can matter in an experiment, and if nothing else you should document who your subjects were and what they knew about the design.

Often it is a good idea to have written experimental instructions, and to actually read the instructions to the subjects. (Note, for example, that Stevens explicitly quotes from his instructions, and from those used in an experiment whose data he reanalyzed.)

## Data Analysis

You should use Matlab to analyze your data. Thus the first step will be to get your experimental data into Matlab. The experimental program writes the data into a file structure that helps keep track of how it was collected.




The data files are stored as comma separated values (csv) and you can open them in almost any text editor if you want to look at what is in there. But, we want you to analyze the data in Matlab. To get you started, we have provided a template program that will read the data. It is in the folder analyze and is called MagEstExampleAnalysis.m. You will probably want to make a copy with a new name and then start to make it work. You'll need to set the variables experimenter, experimentName, and subject strings to match those used when you ran the experimental program. You also need to specify which sessions you want to analyze. These are given as a list of numbers that correspond to the numbers post-pended to the data files.

```
% Define parameters that describe the experiment that was run.
experimenter = 'dummy';
experimentName = 'MullerLyer';
subject = 'subject1';

% List of file numbers we want to analyze.  These are the numbers appended
% to each of the data files.
fileNumbers = [1 2];
```

The example program reads in the data from the specified sessions. The data end up in a matrix. Each row of the matrix represents a trial. The columns provide information about the stimulus and response on that trial. What's in the columns is specified by strings in the returned cell array columnHeaders. The third dimension of the matrix indexes session.

Here is what the two key variables will look like if you run the template as provided (we set up two dummy data files for it to read.)

```
>> columnHeaders'

ans =

    'Line Length'    'Line Angle'    'Arrow Length'    'Arrow Angle'    'Estimate 1'    'Estimate 2'
'Estimate 3'    'Estimate 4'

>> data

data(:,:,1) =

    70    45    30    45    40    40    40    35

    80    45    30    45    80    50    45    50

    90    45    30    45    85    80    90    80

   100    45    30    45    85    50    90    90

   110    45    30    45   100   100    70   100

   120    45    30    45   110   110   110   110

   130    45    30    45   130   110   120   120

    70    45    30   135   100   100    60    80

    80    45    30   135   110    90    55   130

    90    45    30   135   120   110    80   110

   100    45    30   135   120   120   110   120

   110    45    30   135   110   120   140   120

   120    45    30   135   150   120   150   130

   130    45    30   135   160   140   140   150

data(:,:,2) =

    70    45    30    45    35    40    42    39

    80    45    30    45    50    80    45    50

    90    45    30    45    80    86    90    83

   100    45    30    45    90    85    90    50

   110    45    30    45   100   100    70   100

   120    45    30    45   110   110   110   110

   130    45    30    45   120   130   120   110

    70    45    30   135    80   100    55   100

    80    45    30   135   130   110    55    94

    90    45    30   135   110   120    80   110

   100    45    30   135   120   120   110   120

   110    45    30   135   120   110   140   120

   120    45    30   135   130   150   150   120

   130    45    30   135   150   155   145   140

>>
```

You will have to decide how you want to analyze and plot your data, but the example should make it easy for you to get the data into Matlab. The solutions to the first homework will also provide some examples about how to plot in Matlab.

## The actual assignment

The purpose of this lab is to start you thinking about some of the basic issues that you must confront when doing experimental psychology and to get you started on using Matlab to analyze experimental data. In addition to doing the experiment, you should turn in a short lab report, as well as a printout of the program you used to analyze the data.

When you write your lab report, try to keep it simple. Your introduction need not review the literature. Just explain what you were trying to measure. Concentrate on the methods and results sections, which should be as clear as possible. The discussion can also be short, perhaps just your thoughts about this lab, what worked, what didn't, how you might improve things if you were to do it again, etc.

Magnitude estimation data are often fitted with power functions. We will be talking about function fitting soon in the course, but you need not do any fancy analysis of your data for this lab report, although you can if you know how. In addition, one might typically perform a statistical test to decide whether trends seen in the data are likely to replicate. But for this lab, you don't need to do that, although again you may if you know how and would like to. What you do need to do is produce a clear plot that shows your results, with mean and standard errors (or median values and interquartile ranges, if you prefer).

## Working together

You will do the experiment together in groups, and the design and analysis of your experiment can and should be a collaborative effort. Thus each group can turn in the same analysis program, and the plots in the lab reports can be the same within group.

But, I would like everyone to write his or her own lab report, so that the actual writing represents your own individual work.