Statistical Practices in Excel and Octave

Siddhi Krishna

Professor Tim Hickey, CS177

April 2011

1 Introduction

Using statistical packages is routine in nearly every field, whether Psychology or Economics, Biology or Politics. Being easily able to calculate averages and standard deviations, in addition to calculating t-tests, chi-square tests, or distributions is vital to any applied field.

One of the most commonly used tools for data analysis is Microsoft Excel, which is used by individuals, students, business-people, and academics alike. Included on most computer purchases, it it easy to use with it's user friendly interface, with a visual GUI and readily accessible help and FAQ section.

To contrast, Octave is a free download that works to serve as a Matlab substitute. While it can operate with much of the same functionality as Matlab, Octave has a few commands and packages that are Octave specific. Additionally, some features and functions are only available to those who own Matlab. One of the main differences between these two programs is that Matlab comes with a GUI, while Octave is used solely through a CommandPrompt-esque window. Nevertheless, Octave is able to provide the same statistical analysis that Matlab offers.

Therefore, for general statistical needs, both Excel and Octave would provide the necessary tools for functionality. While both relatively easy to use, the needs of the user would dictate which is the better choice for use.

To serve as a case-study to compare their functionality and uses, I will use data that I have gathered and perform the statistical analyses on both Excel and Octave.

2 The Research

Humans frequently exhibit altruism, or the giving of resources of expecting repayment. The standard method in experimental economics for testing altruism is the Dictator Game,

in which a participant, the Dictator, is given an endowment of money which they can split between themselves and someone else, the Receiver. There is no explicit relationship defined between the Dictator and the Receiver. The results of over a hundred experiments find that people give, on average, 20-30% of the original endowment. These results violate the common economic assumption that people will act to maximize their own payoffs. To try to explain this behavior, we have applied a theory from social psychology, Alan Fiske's Relational Models Theory, to better understand altruism. Relational Models Theory proposes that people form fundamentally different types of relationships including:

- Communal relationships based on mutual care
- Exchange relationships based on trade
- Authority relationships based on power

In a series of experiments conducted online we subtly manipulated the relationship context of the Dictator Game by varying the task description while holding the monetary incentives constant. We have found that relationship context strongly affects altruism, and Relational Models Theory correctly predicts these effects.

3 Experimental Procedure

The Standard Dictator Game essentially establishes a "context-less" relationship between the Proposer and Receiver:

> In the scenario, there is a Proposer and a Receiver. The Proposer has 10 economic units. The Proposer can decide to transfer 0 to 10 of these economic units to the Receiver. To make the Proposers decision more realistic, each economic unit will be worth $0.10 in Mturk bonus payments. You are taking the role of the Proposer.

To serve as an example for this tutorial, we will use the Communal Sharing model and it's associated scenario. The Communal Sharing model dictates that people share their resources based on need. To test this scenario, we have our scenario constructed around a brother and a sister, one who is hungry and one who isn't. The first scenario encourages the Brother to give away food to the Sister who is hungry (the actual text from the scenario is presented below):

> There is a Brother and a Sister. The Brother has 10 ounces of steak. The Sister has 0 ounces of steak. They both enjoy steak and there is no other food available. The Brother is not too hungry at the moment. The Sister is very hungry.
>
> The Brother can decide to give 0 to 10 ounces of steak to the Sister. (The Sister will not make a decision in this interaction.)

To contrast this scenario, we also provide a contrasting scenario to determine how altruism changes when the Sister is not hungry but the Brother is:

> There is a Brother and a Sister. The Brother has 10 ounces of steak. The Sister has 0 ounces of steak. They both enjoy steak and there is no other food available. The Brother is very hungry at the moment. The Sister is not too hungry.
>
> The Brother can decide to give 0 to 10 ounces of steak to the Sister. (The Sister will not make a decision in this interaction.)

We want to compare the number of discrete units that participants give in the three different scenarios: the standard Dictator Game (which is the control group), the Original Communal Sharing Scenario (where the brother is not hungry but the sister is), and the

Flipped Communal Sharing Scenario (where the brother is hungry and the sister is not). While each of these scenarios is essentially the same as the standard dictator game, the context changes. To compare the three scenarios, we need the following information about each:

- Mean (μ)
- Standard Deviation (σ)
- Mode
- T-Statistic and P-Value between:
  - Original and Flipped Communal Sharing Scenarios
  - Standard Dictator Game and the Original Communal Sharing Scenario
  - Standard Dictator Game and the Flipped Communal Sharing Scenario
- Bar Graphs comparing distributions

We will examine getting these statistics in both Excel and Octave.

4 Using Excel to obtain vital statistics

Data is collected from Amazon Mechanical Turk (also known as Mturk), a crowd-sourcing website. Mturk exports the data collected as a .csv file to use in Excel. The Data, in it's raw form, is a list of the amount of units each participant in the "Proposer" role would like to give; that is, it is a list of (in this case) 40 values ranging from 0 to 10. In order to get the mean, standard deviation, and mode, a cell to store the data is selected. Then, the user simply enters the following accordingly:

```
=MEAN(<data selection>)
=STDEV(<data selection>)
=MODE(<data selection>)
```

The user selects the data they wish to analyze by simply highlighting it. Excel, in general, takes a relatively visual approach to functions, ensuring that the user does not have to be skilled at using packages to make effective use of their program. Similarly, performing a T-Test is not difficult. Since T-Tests involving comparing the distributions of two different sets of data, there are more parameters required for execution. Besides the data for each of the two data sets, it also requires making a decision as to whether to perform a *one sided tail test* or a *two sided tail test*. This factor determines the "tail" factor when entering the T-Test command. Additionally, the user needs to decide whether to perform a *paired or dependent* t-test (where the samples are from the same group of people), or a *unpaired or independent* t-test (where the two samples are from two distinct groups of people). Excel takes in all of this information through the following command:

```
=TTEST(array1, array2, tails, type of test)
```

The result of the TTEST command is the P-Value associated with the likelihood that the null hypothesis, H_0 is correct. Another useful command is the TINV command, which returns the t-value, which is essentially the standard deviation that is associated with the P-Value. This is helpful when trying to analyze results in comparison to the 68-95-99.7 Rule in statistics, which states that what percentage of the normal distribution is covered within the first 3 standard deviations of the mean of a normal distribution. Understanding where the data's t-value is with respect to the standard normal distribution is a helpful conceptual tool. In addition to needing the P-Value, calculating the t-statistic also requires the *degrees*

of freedom, where $dof = n_1 + n_2 - 2$, and n_1 and n_2 are the respective number of observations in each sample. To capture this t-statistic, the user simply has to calculate:

```
=TINV(P-value, Degrees of Freedom)
```

All of these calculations are relatively simple to perform. However, more than just having pure statistics, having graphs are extremely useful, not just for the user but also for anyone who needs to quickly understand the data presented.

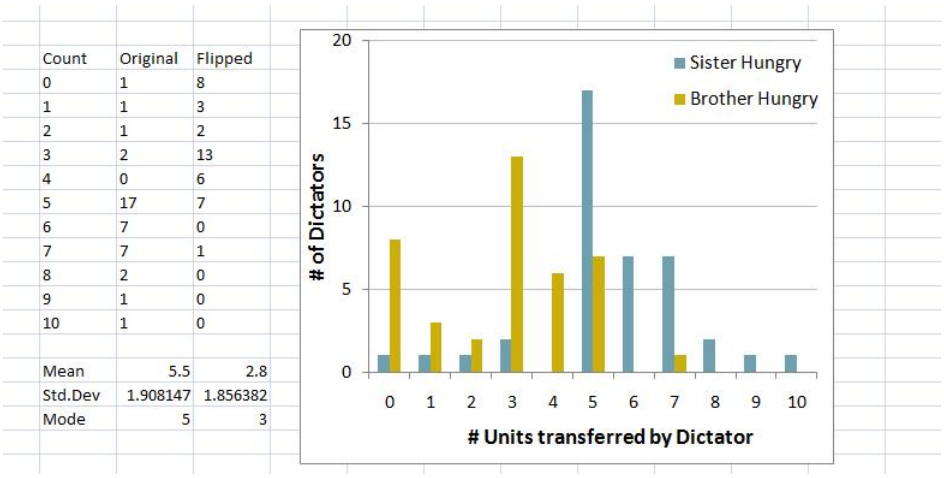
Constructing bar graphs in Excel is also relatively simple. For the purposes of this particular experiment, rather than using the pure lists of "transferred units", we need a count of each number of units transferred; that is, 1 person gave 0 units, 1 person gave 1 unit, etc. This can be achieved by using Excel's COUNT function, such that by first sorting the data, the user can then select a new cell, and enter:

```
=COUNT(<selection of array>)
```

The array can be selected by highlighting the appropriate data. By doing this for the integer values from 0 to 10, the following table, which provides counts for the Original and Flipped Scenario for the Communal Sharing Model, can be produced:

Count	Original	Flipped
0	1	8
1	1	3
2	1	2
3	2	13
4	0	6
5	17	7
6	7	0
7	7	1
8	2	0
9	1	0
10	1	0

Once in this form, it is easy to create a bar graph by following the following steps: **INSERT** $\Rightarrow$ **CHARTS** $\Rightarrow$ **COLUMN** Then, the user selects the style of chart that they would like to use. Selection of the chart type creates an empty chart plot box in the Excel spreadsheet. The user selects the data to be plotting by right-clicking on the chart area, which makes a *Select Data Source* Window pop up. By selecting **ADD** and then highlighting the appropriate data values as desired. The Excel Interface allows the user to plot as many different data sets on the same graph as desired. Color themes for the data can also be changed by accessing **CHART TOOLS** $\Rightarrow$ **DESIGN**. Chart titles and axes are modified by simply clicking on the various fields and editing them accordingly. The result is the following:



Count	Original	Flipped
0	1	8
1	1	3
2	1	2
3	2	13
4	0	6
5	17	7
6	7	0
7	7	1
8	2	0
9	1	0
10	1	0
Mean	5.5	2.8
Std.Dev	1.908147	1.856382
Mode	5	3

Figure 1: Image of final output using Excel

Excel makes strong use of it's GUI and the completion function, making data analysis and plotting in Excel very easy. We can now compare this to similar procedures in Octave.

5 Using Octave to Obtaining Vital Statistics

Octave is not GUI driven, which means that users need to be a bit more experienced with using commands and being procedural to execute the appropriate functions. This may seem a bit intimidating to first time users; however, it is not difficult to execute the same statistical practices in Octave. The first step is to download the Octave Statistics package, which is readily available on the internet.

The first step for data analysis is to load in or enter the data. If the data is being loaded in, it is best that it is in either .csv or .dat form (I will be using .csv). To load the files into Octave, the user simply need enter **load** $< NAME >$**.csv**. The data is now loaded into Octave. By typing the name of the file, Octave will print the data in the file. If the user wishes to rename the data file for convenience, the following command allows them to do so:

Example: **myData** = $< filename >$

The first step is to find the mean, standard deviation, and mode of our data. Since each scenario (the original and the flipped) are in two different columns, each needs to be accessed individually, and then the appropriate functions need to be called with these individual columns as parameters.

Since each column will need to be called on multiple occasions separately, it makes sense to give each column a label. This is accomplished by doing the following[1]:

```
original = myData(:,1);
flipped = myData(:,2);
```

This therefore assigns all of the original scenario's data to **original** and the flipped scenario's data to **flipped**. To find the mean, standard deviation, and mode of the respective data sets is now easy by using the **MEAN, STD** and **MODE** commands respectively. To perform an unpaired T-Test, we use the T_TEST_2 command, and save the results into a 1x3 array, which stores the P-Value, T-Statistic, and Degrees of Freedom accordingly. To save these values, they should be assigned to appropriately named variables. Therefore, we get the following:

```
meanOri = mean(original);
meanFlip = mean(flipped);


stdOri = std(original);
stdFlip = std(flipped);
```

[1]Recall that in Octave, given an array A, the indecies are represented as A(rows, columns); if you want to select all the rows or columns, the ":" command accomplishes this.

```
modeOri = mode(original);
modeFlip = mode(flipped);

[Pval, tstat, dof] = t_test_2(original, flipped)
```

Now that we have the statistics for the data, we need to have the bar graph. In order to graph these values, we need to have a count function, which will count the number of instances of a particular value in the array. This is a very easy function to write ourselves. The code is included below:

```
function y=count(A,x)
% counts the number of occurances of value x in array A
% Author: Siddhi Krishna

  c=0;
  for i=1:1:length(A)
    if (A(i) == x)
      c=c+1;
    end
  end

  y=c;
```

Now that we have a count function, we need to create an array with the appropriate counts for each number from 0 to 10. We first initialize an 11 x 1 array, and then fill each one as necessary:

```
a = zeros(11,1)
a(1) = count(original,0)
a(2) = count(original,1)
...
a(10) = count(original,9)
a(11) = count(original,10)
```

This is the "brute force" way of finding the count for each value from 1 to 10. However, the easier way to do this is by using a for loop:

```
a = zeros(11,1)
for j=1:1:11
a(j) = count(original, j-1)
end
```

We can do this for both the Original and Flipped data, which results in (for example) *variable a* holding the counts for the Original, and *variable b* holding the counts for the Flipped.

Now, we need only plot our results. This can by done by compiling the original and flipped data into one array, and then graphing this new array in a bar plot, as follows:

```
% Let a be the counts for the original data
% Let b be the counts for the flipped data
% Array c will hold the combined counts
c = zeros(11,2)
c(:,1) = a;
c(:,2) = b;
```

```
figure; bar(c);
colormap <COLORMAP_NAME>
title('TITLE OF GRAPH')
xlabel('X AXIS LABEL')
ylabel('Y AXIS LABEL')
```

Following these directions graphs the Original and Flipped count data values on the same graph, with the appropriate title and labels. Like Excel, Octave provides themes (called *colormaps*) for the colorings of the bars in the graph; some include "jet", "hot" and "copper". The Octave final graph, using the colormap *summer* is below:

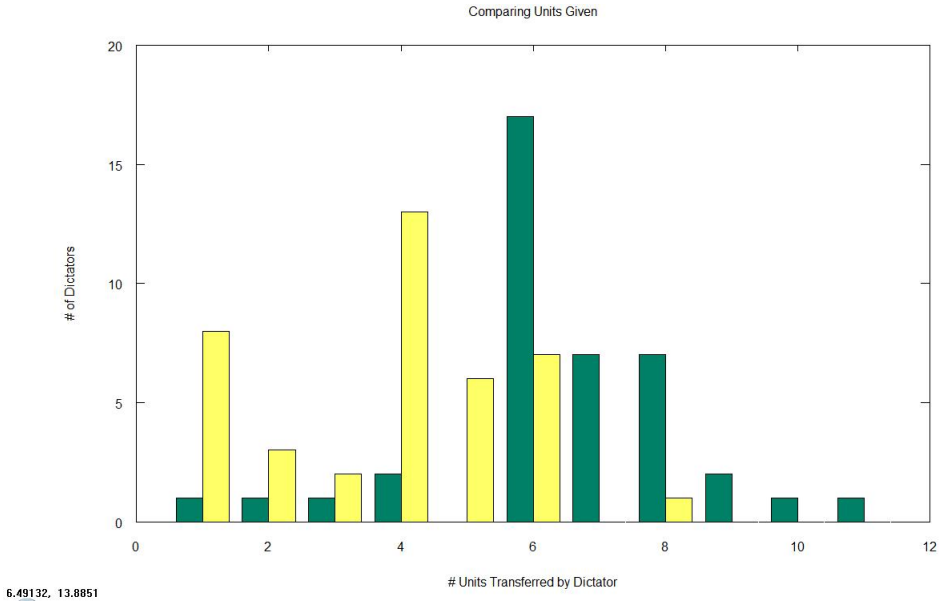


Figure 2: Image of final output using Octave

Like the Excel output, the Octave graphical output is clear and easy to read.

6 Experiment Results

The average cents given by the Brother the Sister dropped from 55 cents to 28 cents. Additionally, performing a t-test shows that the distributions themselves are starkly different, resulting in a p value less than 0.001. The mean ages for the original and inverted scenarios were about 31 (standard deviation of 11) and 26 (standard deviation of 9) years respectively, and the proportion of females to males being .425 for both.

The results indicate that key notion of complementary obligation has indeed had an effect on the altruistic behavior of the Dictator. When another member of the group (the Sister) needed resources (food) more than the owner (the Brother) of the resources, the Brother fulfilled his communal obligation by giving food to the Sister, and vice versa. Even though both scenarios are exactly the same as the standard dictator game, they are not only significantly different from one another, but also have distributions significantly different from the standard dictator game (these calculations are not provided here).

7 Comparing Excel and Octave

Excel and Octave are targeted at two different audiences: the former for the everyday user needed graphing and statistical tools, and the latter for those with some computer programming experience who need to analyze large quantities of data. Accordingly, Excel and Octave each have pros and cons.

Excel makes use of an "auto-completion" function, ensuring that users do not need to remember command names in their entirety, as Excel "helps" users by providing the remainder of the name. This is a great function for novice and experienced users alike. Additionally, their gui is easy to use and allows for more visual data processing. Octave, being non-gui, forces users to be a bit more knowledgeable about their software, and the Octave language. This may be a bit off putting for beginners, yet is quite intuitive for those

with more computing experience.

One big difference between the two is that users can provide explicit labels for items in Excel, as Figure 1 demonstrated; the means, standard deviations, and modes are explicitly labeled. In Octave, these values are stored in variables; this means that Excel users need to be able to pin-point the physical location of the piece of information they desire, and conversely, Octave users need to remember variable names. Neither of these are problematic, but rather depends on preference.

Users of Excel may expect functionality to be built into the program; if there is a missing functionality, there is the ability to write a micro to create that function. Basic functionality missing from Octave is easy to program and implement (as demonstrated in this tutorial). This is a great function provided in both programs. However, it is easier to add functionality to Octave than to Excel. Finally, it is easy to perform data cleaning, analysis, and presentation all within Excel. However, Matlab is useful for data cleaning and analysis - but requires a little bit more work for presentation. While both can easily produce easy-to-read graphs, only Excel produces clean, formatted tables ready for presentation.

8 Conclusion

Both Excel and Octave provide essentially the same functions for statistical analysis. Excel, being gui driven, is more intuitive in use; Octave, being a programming language, is easier to use when needing to analyze large quantities of data. Octave's greatest asset is it's flexibility, allowing the creation of new functions to facilitate analysis, while Excel's greatest asset is it's ability to analysis and present data in one convenient program.

9 Exercises

Perform the following in both Excel and Octave.

- Create an array (called "A")of 100 random values from 1 to 10 using the *normrnd* function. Plot these in a bar graph. Provide the title "Plotting Random Values" and the colormap "cool".
- Create a function to count the number of values in A that are between 20 and 40.
- Find the mean, standard deviation, and mode of A. Store these values in the variables mean, std, and mode respectively.

10 References

Colormap. Mathworks Product Documentation. $< http://www.mathworks.com/help/techdoc/ref/colormap.html >$ Accessed April 18, 2011.

Fiske, A.P (1992). The Four Elementary Forms of Sociality: Framework for a Unified Theory of Social Relations. Psychological Review. 4:689-723.

Gilat, Amos. Matlab. John Wiley and Sons. 2008.

GNU Octave: 24.4 Tests. Eaton, John W, David Bateman, Soren Hauberg. $< http://www.network-theory.co.uk/docs/octave3/octave_233.html >$ Network Theory Limited. United Kingdom. Accessed April 20, 2011.

Statistics. Octave-Forge. $< http://octave.sourceforge.net/statistics/overview.html >$ Accessed April 15, 2011.