ECON 442: Quantitative Trade Models **Professor Jack Rossbach**

Tips for Problem Set 2

Problem 1

Tip 1.1: We will only be using a subset of the variables in the dataset. To keep only the variables we will use, you can run the following command:

keep iso_o iso_d gdp_o gdp_d flow distw rta contig comlang_off col_hist

This has the benefit of removing potentially distracting variables from the dataset, but other than that it doesn't change anything. Make sure not to run the above after you generate the new logged variables, since the above command will drop them.

Tip 1.2: You can write your STATA commands in any plain text editor, such as Notepad++, and then copy and paste your commands into the STATA command line to run them. This allows you to save your code, so you can easily continue your work at a different time. See instructions here for how to add STATA syntax highlighting in Notepad++: http://goo.gl/1U9iVw

Tip 1.3: If you continue learning STATA, you'll eventually use .do files. You can read more about them here: http://www.stata.com/manuals13/u16.pdf

Tip 1.4: STATA should be available to download from Georgetown: https://georgetown.onthehub.com/WebStore/Welcome.aspx

Tip 1.5: The 3 digit ISO country code for China is CHN and for the United States is USA. They are string variables, so we need quotes when trying to use those values.

Tip 1.6: One equals sign (=) is assignment. The thing on the right is computed, then stored in the variable on left. Two equals signs (==) is for equality, which is usually used to say "execute this command if thing on the right side equals thing on the left side". Example: **keep if iso_o == "CHN"** keeps observations where China is the origin country (exporter).

Tip 1.7: You can combine statements using & and |. Example: **keep if iso_o == "CHN" & iso_d == "USA"** keeps observations where both China is the origin country (exporter) and USA is the destination country (importer). Alternatively, **keep if iso_o == "CHN" | iso_d == "USA"** keeps observations where China is the origin country (exporter) and/or USA is the destination country (importer). | is an *inclusive or*.

Tip 1.8: There are some nice user-made packages for creating and exporting tables in STATA. The estout package is one option (run command **ssc install estout** then command **help estout**). For nice LaTeX tables another option is the outex package (**ssc install outtex** then help **outtex**).

Tip 1.9: You can do this problem in R using the lm() function to do regressions. You can use the factor() function to create fixed effects. You can import the foreign library and read.dta() function.

Problem 2

Tip 2.1: You can compute the welfare changes by hand, using the wages and price indices that are displayed after you get the solution. Alternatively, you can do it in R. After doing exercise 2.iii create a variable to store w/p, e.g. **welfare <- Wages/Prices** , then after doing exercise 2.iv do it again but for a different variable name: **welfare_new <- Wages/Prices**. Then the welfare change will be **welfare_new/welfare**. This is just one option, there are lots of alternative ways to do the same thing.

Tip 2.2: Most of the time, the reason the program isn't working is due to typos. If your code won't work, go through it and check to make sure you didn't accidentally type a wrong letter somewhere that you aren't missing a parenthesis or bracket. A common mistake is typing 1 (one) instead of l (letter L), or typing 0 (zero) instead of o (letter O), or vice versa.

Tip 2.3: Another common mistake is having a space before the first parenthesis of a function or index (e.g. writing **v [1]**, which is incorrect, instead of **v[1]**). Typically, having extra spaces is fine, and something I recommend doing to align your code for easy readability..

Tip 2.4: The second most common mistake is calling a variable that hasn't been defined yet. If you try to do **2*x** in R, but you haven't defined x before then, you'll get an error. I defined all the variables in the template, so you shouldn't have this problem; but in general it's a common mistake.

Tip 2.5: For most of the variables, we are storing the values in a vector or matrix. We can access a specific value by using vector and matrix indexing. Example: **pi[2,3]** returns the element in the 2nd row and 3rd column of the matrix pi. Make sure to use square brackets here.

Tip 2.6: Like STATA, there are R packages to create nice tables in LaTeX. One such package is stargazer. You can install it with the **install.packages("stargazer")** command.

Problem 3

Tip 3.1: Some of the instruction steps have already been done for you. Step 1 is done completely, the full list of 4-digit SITC Rev. 2 commodity codes for Step 3 is already listed in the tab "Steps 3&4 - Merge Codes", and the binning part of step 6 will happen automatically once you finish step 5.

Tip 3.2: Be sure to scroll down to the bottom of the "Steps 5&6 - Sort and Bin" tab, that is where we sum the columns to get number of products and trade shares.

Tip 3.3: There are tons of tutorials on youtube for learning excel functions. Here's a video that discusses Pivot Tables and another that discusses the Vlookup function.

Tip 3.4: When we use the Vlookup function in step 3, we get errors. Comtrade doesn't report trade flows of zero, so Vlookup is trying to find codes that we don't have any data for, and so it will return #N/A. You can copy and paste the Vlookup results as values then do find and replace to replace #N/A with 0.

Tip 3.5: As an alternative to tip 3.4, it is possible to combine the Vlookup function with an IF function and a ISNA function. This is what I do in the example workbook. There is a video tutorial that does a similar thing here.

Tip 3.6: You shouldn't get any errors when doing the Vlookup function in step 5, since we will have entered zeros for missing trade flows.

Tip 3.7: It is possible to do this problem in STATA or R. This data appendix for one of my papers includes code that you can use to figure out how to do it in STATA. Here is a snippet of R-code that might give you some hints how to do something similar in R. The only difference is you will need to sort by the first three years before we bin the data using just the first year. You can sort with the order() function.