Introduction to Computing for Sociologists
Neustadtl

Inputting Data

There are many ways to create or use a Stata data file. The following briefly describes some, but not all, of the ways data can be read into Stata from ASCII files and spreadsheets. You can use the interactive help system within Stata to read about the details of each data entry method. For example, to learn more about the Stata command `insheet` you would enter "`help insheet`" in the command window. The complete set of documentation is available in PDF format as well. To look at this documentation follow this path on the Stata toolbar: Help►PDF Documentation.

`input`—(`help input`)

- Enter data from interactively from the keyboard.
- Enter small easily described datasets from ASCII files.
- useful for small datasets

`insheet` (`help insheet`)

- `insheet` reads text (ASCII) files created by a spreadsheet or database program.
- The data must be tab separated or comma separated, but not both simultaneously, and cannot be space separated.
- An observation must be on only one line.
- The first line of the file can optionally contain the names of the variables.

`infile` (free format)—`infile` without a dictionary (`help infile1`)

- The data can be space separated, tab separated, or comma separated.
- Strings with embedded spaces or commas must be enclosed in quotes (even if tab- or comma separated).
- An observation can be on more than one line, or there can even be multiple observations per line.

`infix` (fixed format) (`help infix`)

- The data must be in fixed-column format.
- An observation can be on more than one line.
- infix has simpler syntax than infile (fixed format).

`infile` (fixed format)—`infile` with a dictionary (`help infile2`)

- The data may be in fixed-column format.
- An observation can be on more than one line.
- infile (fixed format) has the most capabilities for reading data.

`import excel (help import excel)`

- This command loads an Excel file, also known as a workbook, into Stata.
- Excel 1997/2003 (.xls) files and Excel 2007/2010 (.xlsx) files can be imported.
- There is a file size limit of 50 MB for Excel 2007/2010 (.xlsx) files.
- There are several options to point to the correct worksheet and location within a worksheet.

The following table shows information about the 2006 U.S. Senate elections and provides a small data set that you will convert into a Stata data file.

There are nine columns of data—each column represents a variable and must have a name. Variable names may be 1 to 32 characters long and must start with a-z, A-Z, or _, and the remaining characters may be a-z, A-Z, _, or 0-9 (see `help varname`).

For this assignment you will use the following names: `party`, `up`, `elected`, `notup`, `y2004`, `y2006`, `plusminus`, `vote`, and `percent`. Note that Stata respects case so `party`, `Party`, `PARTY`, and `PaRtY` are interpreted as distinct variables.

You also need to distinguish between string data and numeric data (see `help data types`). The variable party in this table can be entered as string data. While all of the other variables could be entered as strings it makes more sense to enter them as numeric variables since they have numeric interpretations. String variables that do not contain spaces can be input easily. If the string has spaces the easiest way to handle them is to enclose the string in double quotation marks. For example, compare "`Democratic Party`" versus `Democratic Party`. Most statistical programs, Stata included would interpret the second example as two variables delimited by a space.

Any variable that contain non-numeric information are string not numeric variables. For example, Stata would read 53.91% as a string variable because it contains the non-numeric character %. Input as 53.91, Stata would interpret this information as numeric.

Summary of the November 7, 2006 United States Senate election results

Party	Breakdown			Seats			Popular Vote	
	Up	Elected	Not Up	2004	2006	+/−	Vote	%
Democratic Party	17	22	27	44	**49**	+5	33,929,202	53.91%
Republican Party	15	9	40	55	**49**	−6	26,674,169	42.38%
Independents	1	2	0	1	**2**	+1	879,032	1.40%
Libertarian Party	0	0	0	0	0	0	614,629	0.98%
Green Party	0	0	0	0	0	0	414,660	0.66%
Constitution Party	0	0	0	0	0	0	132,155	0.21%
Peace and Freedom Party	0	0	0	0	0	0	117,764	0.19%
Write-in	0	0	0	0	0	0	13,567	0.02%
Socialist Workers Party	0	0	0	0	0	0	10,463	0.02%
Personal Choice Party	0	0	0	0	0	0	9,089	0.01%
Socialist Party USA	0	0	0	0	0	0	2,490	0.00%
Others	0	0	0	0	0	0	141,074	0.22%
Total	**33**	**33**	**67**	**100**	**100**	**0**	**62,938,294**	**100%**
Voter turnout: 29.7 %								

Sources: Dave Leip's Atlas of U.S. Elections, United States Elections Project at George Mason University

Using `input`

Required commands: `input`, `format` `compress`, `describe`, and `list`.

1. Use the interactive input command to enter the first three rows of data (Democratic Party through Independents) for all nine variables.
2. When you are done with that put these commands into a "do" file and enter the entire dataset.
3. Use the command `format` to 1) left justify `party`, 2) display commas for `vote`, and display `percent` with two decimal places (see `help format`).
4. Use the command `compress` which adjusts the storage type of your variables to use the computer memory efficiently (see `help compress` and `help data types`).
5. Use the commands `describe` and `list` (`help describe` and `help list`) to describe the data you have read into memory as well as display them on the screen. Your output should look similar to the following:

variable name	storage type	display format
party	str23	%-23s
up	byte	%8.0g
elected	byte	%8.0g
notup	byte	%8.0g
y2004	byte	%8.0g
y2006	byte	%8.0g
plusminus	byte	%8.0g
vote	float	%10.0fc
percent	float	%9.2f

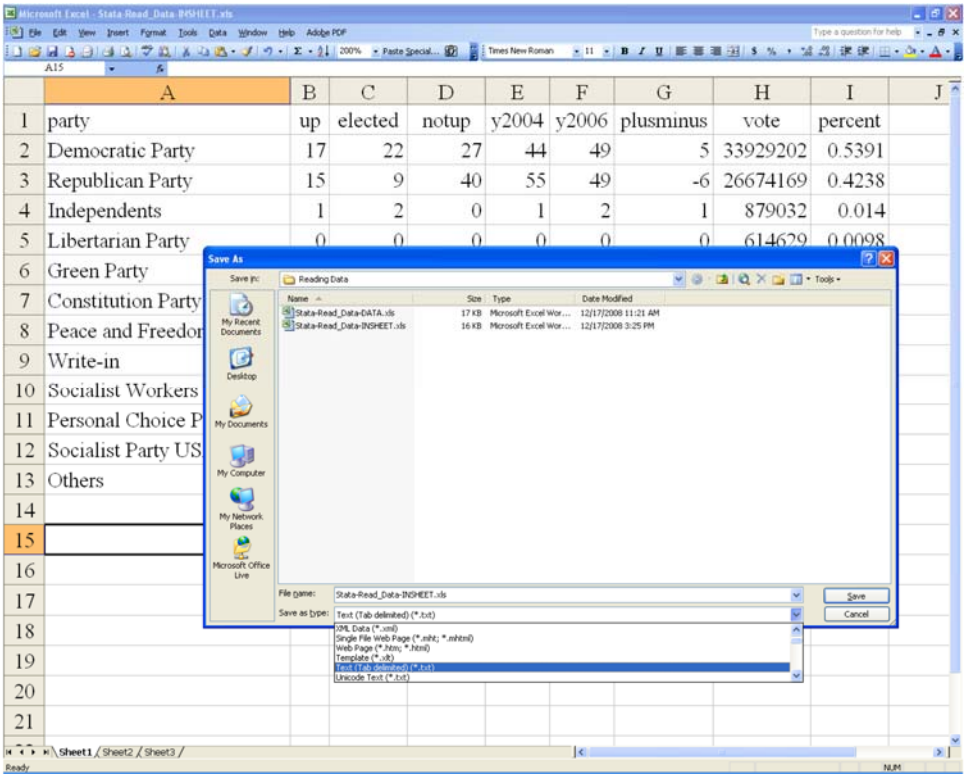
party	up	elected	notup	y2004	y2006	plusmi~s	vote	percent
Democratic Party	17	22	27	44	49	5	33,929,200	53.91
Republican Party	15	9	40	55	49	-6	26,674,168	42.38
Independents	1	2	0	1	2	1	879,032	1.40
Libertarian Party	0	0	0	0	0	0	614,629	0.98
Green Party	0	0	0	0	0	0	414,660	0.66
Constitution Party	0	0	0	0	0	0	132,155	0.21
Peace and Freedom Party	0	0	0	0	0	0	117,764	0.19
Write-in	0	0	0	0	0	0	13,567	0.02
Socialist Workers Party	0	0	0	0	0	0	10,463	0.02
Personal Choice Party	0	0	0	0	0	0	9,089	0.01
Socialist Party USA	0	0	0	0	0	0	2,490	0.00
Others	0	0	0	0	0	0	141,074	0.22

Using `insheet`

Required commands: `insheet`, `format` `compress`, `describe`, and `list`.

Many datasets are distributed as spreadsheets. The U.S. Senate data have been entered in an MS Excel spreadsheet (Stata-Read_Data-DATA.xls). One way to move the data from a spreadsheet format to Stata is to use Excel to write out the data in a tab-delimited format and then use the command `insheet` to convert the data to Stata format. Note that the spreadsheet below is slightly different than Stata-Read_Data-DATA.xls—the data have been formatted to be easily output to Stata (good variable names, no non-numeric characters in numeric measures, etc.).

The data are written using the Excel "Save as" dialogue shown below. Note that the file is saved as text, or more specifically "Tab delimited". This file is named "Stata-Read_Data-INSHEET.txt".



	A	B	C	D	E	F	G	H	I	J
1	party	up	elected	notup	y2004	y2006	plusminus	vote	percent	
2	Democratic Party	17	22	27	44	49	5	33929202	0.5391	
3	Republican Party	15	9	40	55	49	-6	26674169	0.4238	
4	Independents	1	2	0	1	2	1	879032	0.014	
5	Libertarian Party	0	0	0	0	0	0	614629	0.0098	
6	Green Party									
7	Constitution Party									
8	Peace and Freedor									
9	Write-in									
10	Socialist Workers									
11	Personal Choice P									
12	Socialist Party US									
13	Others									
14										
15										
16										
17										
18										
19										
20										
21										

1. Write a program or "do" file that uses `insheet` to read the data from the tab delimited ASCII file into Stata.
2. Use the command `format` to 1) left justify `party`, 2) display commas for `vote`, and display `percent` with two decimal places (see `help format`).
3. Use the command `compress` which adjusts the storage type of your variables to use the computer memory efficiently (see `help compress` and `help data types`).
4. Use the commands `describe` and `list` (`help describe` and `help list`) to describe the data you have read into memory as well as display them on the screen.

Using `infile`

Required commands: `infile`, `format` `compress`, `describe`, and `list`. Within `infile` you will also need `_column(#)`, `_firstlineoffile(#),` and `_line(#)`.

The `infile` command read ASCII (text) data in fixed format with a dictionary and has the most flexibility of all the data input methods. Two ASCII files that contain the Senate data are provided and named "Stata-Read_Data-INFILE1.raw" and "Stata-Read_Data-INFILE2.raw". For each file:

1. Write a program or "do" file that uses `infile` to read the data from the ASCII file into Stata.
2. Declare the data type in your dictionary file (so you don't need `compress` later).
3. Use the command `format` to 1) left justify `party`, 2) display commas for `vote`, and display `percent` with two decimal places (see `help format`).
4. Use the commands `describe` and `list` (`help describe` and `help list`) to describe the data you have read into memory as well as display them on the screen.

Using `import excel`

Required commands: `import excel`, `list`, `describe`, `clear`, and `rename`. Within `import excel` you will also need the following options: `describe`, and `cellrange([start][:end])`

The `import excel` loads an Excel file, also known as a workbook, into Stata. `import excel filename, describe` lists available sheets and ranges of an Excel file.

Excel 1997/2003 (.xls) files and Excel 2007/2010 (.xlsx) files can be imported, exported, and described using import excel, export excel, and import excel.

1. Try using the command to import the data without using any options. The `describe` and `list` commands are helpful after the data have been imported.
2. Identify the columns/rows of the spreadsheet where the data begin and end (that is, the range containing the data, not the labels) and import the data using the `cellrange` option.
3. Read the help on rename and name the variables sequentially to `party`, `up`, `elected`, `notup`, `y2004`, `y2006`, `plusminus`, `vote`, `and percent`.

Saving Data

Once a dataset is read into Stata it only exists in the computer's memory—if your computer shuts down your data are gone. As long as you have saved your program files (the "do" files) you can always recreate your work. But there are some advantages in saving the data in memory to a hard drive or flash drive. The biggest advantage is speed—reading large ASCII data files can be time consuming. The Stata formatted data files can mostly only be read by Stata and have the extension ".dta". See `help save` for more details. Note that the files you save are formatted according to the version of Stata you are using.

Stata versions 10 and 11 Version 11 have identical file formats and can read data files created in earlier versions of Stata. While Stata 11 can read earlier Stata files (e.g. Stata 9), Stata 9 cannot read Stata 11 or Stata 10 files. You can use the `saveold` command to share datasets saved in versions 11 or 10 with people who are using earlier versions of Stata.

Using save

The `save` command is quite simple—`save [filename] [, save_options]`. Typically the `save` command is followed by an optional filename and there are options to consider. Arguably the most important option is `replace`. Stata will not write over an existing data file unless you specify `replace`.

1. Change the default directory to your flash drive.
2. Save one of the files you have created as a Stata file in that directory.

Using Stata Formatted Data

If you have a Stata data file, a file that has a .dta extension you can read these data directly into Stata with the command `use`. This command loads a Stata-format dataset previously saved by `save` into memory. If filename is specified without an extension, .dta is assumed. If your filename contains embedded spaces, remember to enclose it in double quotes. See `help use` for more details.

Using use

The `use` command has two variations and some interesting options. Generally, however you will use the simplest form which is—`use` *`filename`*. As long as Stata is looking in the directory that contains your data file or you have specified the complete drive, directory, and file name in your command, this should be sufficient. However, if you already have a data file in memory this command will fail and you should use the `clear` option which specifies that it is okay to replace (i.e. destroy) the data in memory, even though the current data have not been saved to disk.

The following are typical ways to read data:

```
use "Stata-US Senate Elections-DATA"

use "c:\data\Stata-US Senate Elections-DATA"

use "Stata-US Senate Elections-DATA", clear
```

1. Change the default directory to your flash drive.
2. Read one of the Stata data files you created earlier.

Bonus Problem!

In some of the examples you have worked in this assignment you needed three Stata files to read in raw data. You had your Stata program (a "do" file), a Stata dictionary file (a "dct" file), and a file containing the raw data (e.g. a "raw" or "txt" file).

You can reduce the number of files by including the dictionary information in the raw data file. This works well for small datasets.

1. Create a dictionary header for the data file called "stata-read_data-STATE_DATA.raw". The variables are based on state level data. The following example shows the first five cases:

```
Alabama             0.114     -11.11   0.27   0.21
Alaska              0.187     -11.25   0.35   0.27
Arizona             0.141     -17.81   0.30   0.24
Arkansas            0.135      -5.19   0.26   0.21
California          0.094     -12.57   0.34   0.28
```

   In order, the variables are 1) state name, 2) number of suicides per 1,000 people, 3) best educated state index, 4) the proportion of men in the state who have never been married, and 5) the proportion of women in the state who have never been married.

2. Create a Stata do file to read these data.