Data Visualization using Stata

Iowa Social Research Center (ISRC) Workshop

Desmond D. Wallace

Department of Political Science
The University of Iowa
Iowa City, IA

November 10, 2017

Why Visualize Data?

- One can communicate information clearly and effectively via graphics
- Effective data visuals helps users analyze and reason with data
- Make complex data accessible, understandable and usable
- Display patterns and/or relationships in one's dataset
- One can visualize patterns and/or relationships with respect to discrete and/or continuous variables

`graph` *`type`* – Available Types

- Bar Graphs
- Box Plots
- Distribution Graphs
  - Histograms
  - Kernel Density Estimation Plots
- Dot Charts (Not Covered)
- Pie Charts (Not Covered)

Introduction

- Constructs bars used to visualize the distribution of a categorical variable
- Similar to a histogram
- Default is to construct a bar for each variable level

Syntax

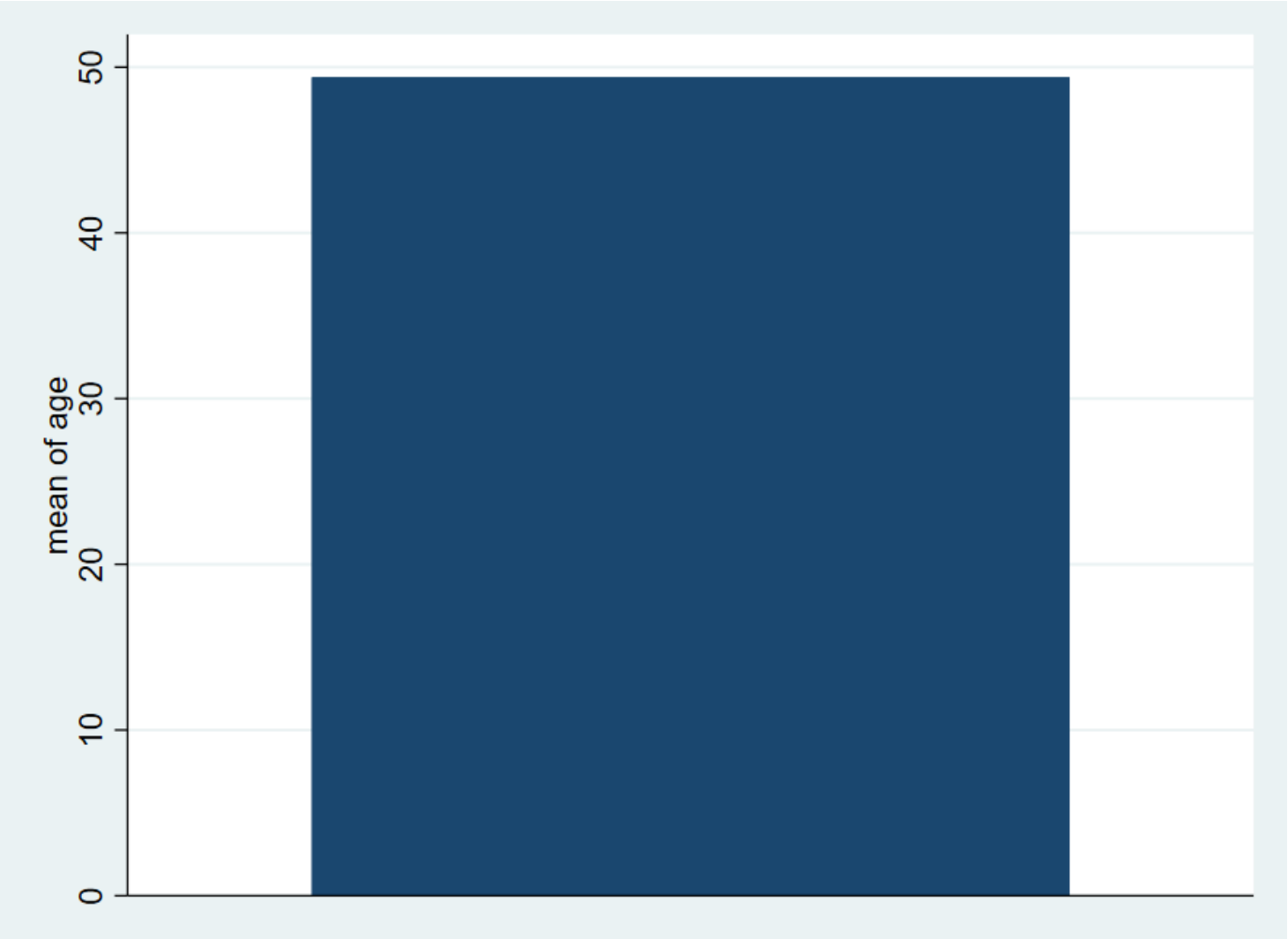
- Basic: `graph bar (stat) yvars`, where *yvars* is a variable list
- Displays specified summary statistic for variable(s); default is the mean
- Other statistics include the median, count, various percentiles, etc.
- Can specify multiple (*stat*) *yvars*
- Can display summary statistic of specified variable based on levels of a categorical variable via the `over(varname)` option
- Advanced: `graph bar (stat) yvars, over(varname)`

Syntax

- *yvars* is optional when `over(`*varname*`)` is specified (Stata 14 only)
- Acceptable syntax: `graph bar, over(`*varname*`)`
- Percentage is now treated as default statistic, calculated based on levels of *varname*
- Can change the statistic to `count`, which reports the frequency totals for each level of *varname*
- Replace `bar` with `hbar` to produce horizontal bar graph.
- See `help graph bar` for additional information

PDF

PNG



Introduction

- Displays a box and "whiskers" that visualizes the distribution of a continuous variable
- Box
  - Bordered at the 25th and 75th percentiles (Q1 and Q3)
  - An additional *median* line at the 50th percentile
- "Whiskers"
  - Lower Adjacent Value (LAV) – Smallest observation greater than or equal to the lower inner fence (LIF), which is $Q1 - 1.5 \times IQR$, where $IQR = Q3 - Q1$
  - Upper Adjacent Value (UAV) – Largest observation smaller than or equal to the upper inner fence (UIF), which is $Q3 + 1.5 \times IQR$
- Any observation falling smaller (larger) than the adjacent values appears as dots

Syntax

- Basic: `graph box` *yvars*, where *yvars* is a variable list
- Can display box plots of specified variable(s) based on levels of a categorical variable via the `over(`*varname*`)` option
- Advanced: `graph box` *yvars*`, over(`*varname*`)`
- Replace `box` with `hbox` to produce horizontal box plot(s).
- See `help graph box` for additional information

Histograms

- A graph that shows the distribution of a variable that takes on many values (Acock 2014).
- Syntax: `histogram` *`varname`*`,` *`options`*
- Can be used for both discrete and continuous variables
- Use the command `help histogram` for more information

Kernel Density Estimation Plots

- Non-parametric method for estimating the PDF (PMF) of a random variable.
- Syntax: `kdensity` *varname*`,` *options*
- Can be used for both discrete and continuous variables
- Use the command `help kdensity` for more information

Introduction

- Used to display relationships between two numeric-type variables
- Represents over 30 different types of graphs, which can be grouped into multiple categories
- Easy to overlay `twoway`-type plots
  - Enclose graph type and variables in parentheses ()
  - Separate graphs via double vertical bars ||

Available Types Not Covered

- Area Plots
- Bar Plots
- Range Plots*
- Regression Fits and Confidence Intervals*
- Functions*
- Contour Plots

Available Types Covered

- Scatterplots
- Line Plots
- Distribution Plots
  - Histogram
  - Kernel Density Plot

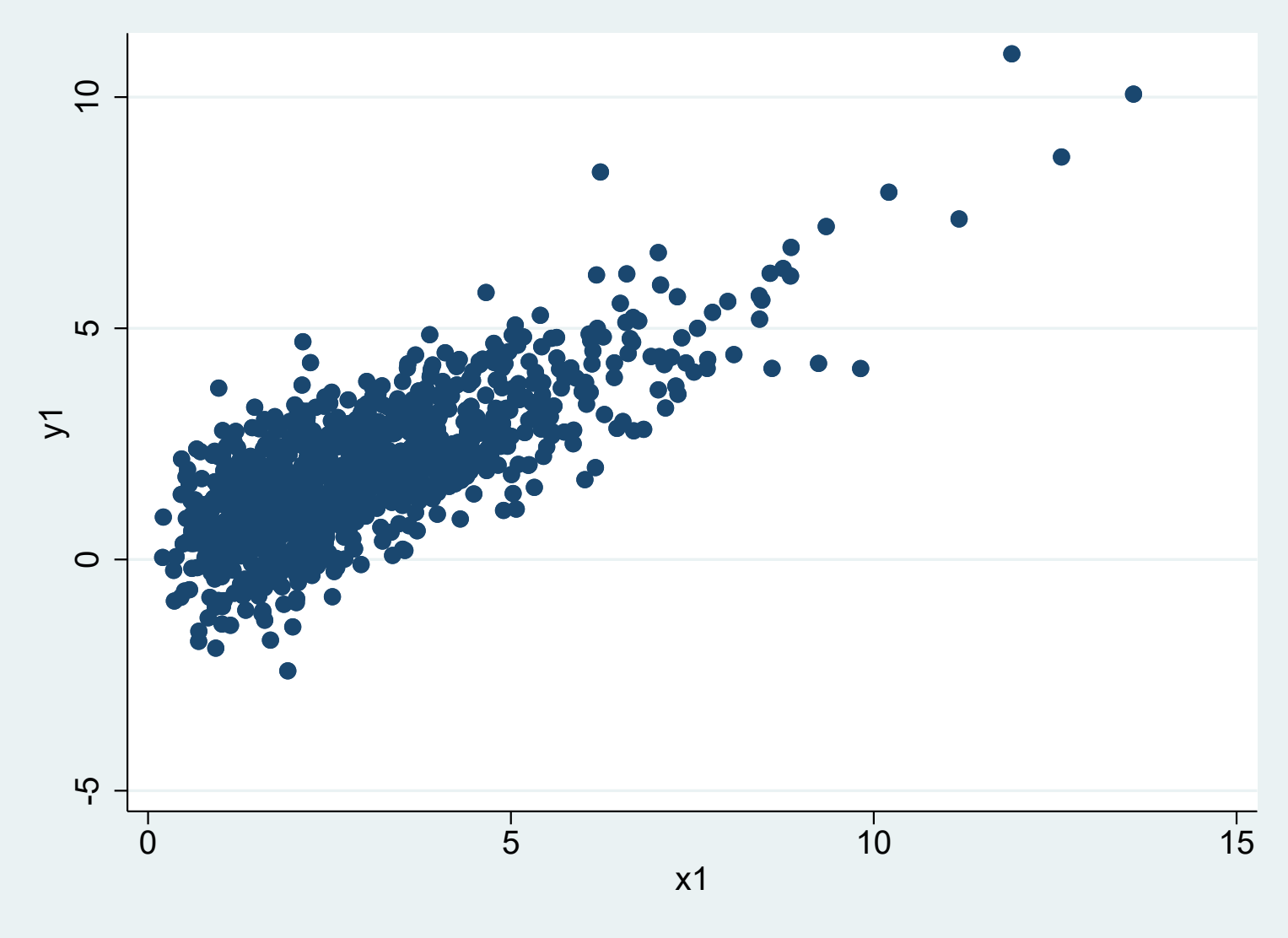
Introduction

- Utilize horizontal and vertical axes to plot data
- Communicates how much one variable is affected by another
- Visual representation of the correlation between two variables

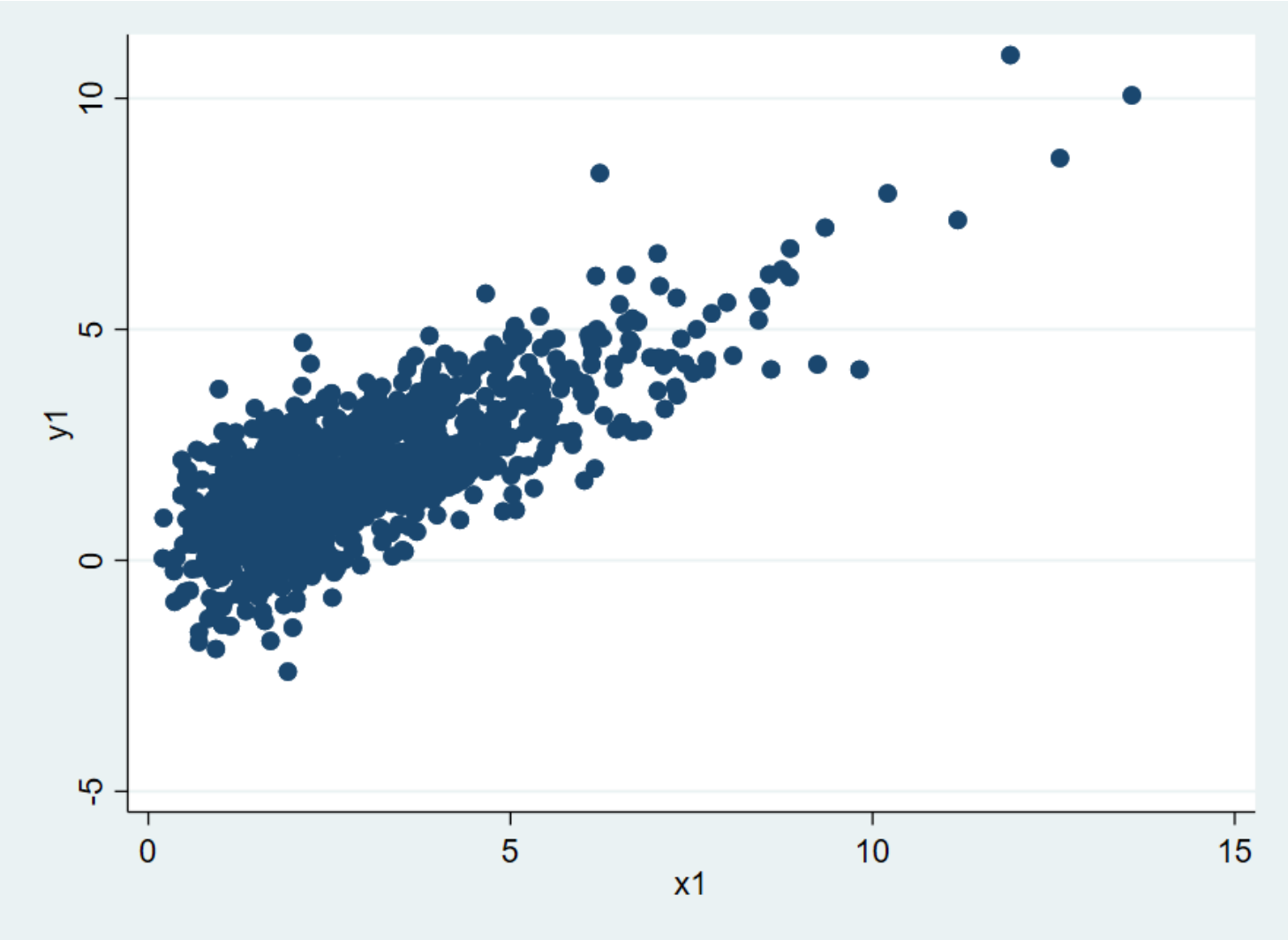
Syntax

- Basic: `scatter` *`varlist`*, where *varlist* is a variable list
- At least two variables need to be specified; last variable specified is treated as "independent" variable (located on the x-axis).
- Can generate scatterplots based on levels of a categorical variable via the `by(`*`varname`*`)` option
- Advanced: `scatter` *`varlist`*`, by(`*`varname`*`)`
- See `help scatter` for additional information

PDF



PNG



Modes

- Data Editor (Edit) – Allows one to view a dataset, and make changes
- Data Editor (Browse) – Allows one to view the dataset, but not make any changes
- Can switch between edit mode and browse mode
- NOTE: When switching from browse mode to edit mode, a warning will appear, whether the user is sure about switching from browse mode to edit mode

Colors

When viewing a dataset in the Data Editor/Browser, different types of data is represented by different colors

- Black – Data used for various descriptive and analytic tasks
- Red – Observations that contain strings, or textual data; can perform descriptive tasks, but not analytic tasks
- Blue – Same as data appearing in black, except the blue represents value labels; can perform both descriptive and analytic tasks
- NOTE: Default is for the Data Editor/Browser to display the value labels, if any for the variables

Full Command Syntax

[**by** *varlist*:] *command* [*varlist*] [=*exp*] [**if** *exp*] [**in** *range*] [*weight*] [**using** *filename*] [,*options*]

command

- Only required element of command statement
- Case-sensitive
- Commands can be abbreviated
- Example: To display summary statistics of a variable, or variables:
  - `summerize`
  - `sum`
  - The underlined portion of `summerize` represents the abbreviation

varlist

- Represents one variable, or at least two variables
- Case-sensitive
- Variables can be abbreviated to minimum number of letters that makes variable unique
- To refer to several variables at the same time:
  - Use the * symbol
  - Use a name range

=*exp*

- Used to generate new variables
- Can include variables in expression statements
- Usually an arithmetic expression
  - Can include the four basic operation symbols ($+$, $-$, $*$, $/$)
  - Can use ^ for an exponentiation statement
  - Can include other functions, such as *abs* and *log*
  - Can include parentheses to manage order of operations

if *exp* and in *range*

- Used to restrict dataset to a subsample of interest
- Represented as a logical statement that is either true or false
- Relation operators are $<$, $<=$, $==$, $>=$, $>$, and $!=$
- Can also specify a range of observations
- Example: `in 1/10` refers to the first ten observations of a dataset

weights

- Used to weigh the observations
- Example: survey data typically uses weights in order to make the sample representative of the population
- Used in conjunction with many commands

using *filename*

- Introduces a file into the command
- File can be on the computer, on a network, or on the internet

options

- Most commands have additional options that the user can specify
- Look at the help file for the command to list its options

by *varlist*

- Used to execute a command for groups of observations defined by distinct values of the variable(s) specified
- Command in question has to be "byable"
- Data must be sorted by the grouping variable
- If data is not pre-sorted, use `bysort`

What are Do-files?

- Typically, the Stata command line only allows the user to run individual commands, not collectively.
- A Do-file is a file that allows the user to run a number of commands at once.
- Do-files allow the user to keep a record of their analysis.

Commands

- Do-files can consist of commands that require either a single line, or multiple lines.
- Commands that span a single line are the same as typing the command into the command line on the main Stata window.
- Commands that span multiple lines requires a delimiter (i.e. a character, or group of characters, Stata recognizes signifying the end of the command).
- The default delimiter is a carriage return (CR)
- Can treat carriage return as a comment using three forward slashes ///.
- Commands that run multiple lines cannot be executed in the main Stata command line.

Comments

- Do-files allow the user to insert any necessary comments with respect to the Do-file.
- Ways to include comments:
  - An asterisk – *
  - Enclosed with – /* */
- Lines that are commented are not executed by Stata.

Execution

- There are two ways to execute commands in a Do-file.
  - Method 1: Execute the entire file.
  - Method 2: Execute the file in pieces via highlighting the specific code you want to execute.
- You can also nest Do-files within other Do-files.

Commands

- `doedit` launches the Do-file editor with a blank do-file.
- `doedit` `[`*`filename`*`]` launches the Do-file editor with the specified do-file.
- `do` *`filename`* executes all commands stored in specified do-file.

Opening the Data

- The command for opening a dataset in Stata is `use`.
- If a dataset is already open, opening a new dataset requires including the option `clear` with the `use` command.
- Examples
  - Example: `use` *`filename`* works if there is no data in Stata's memory.
  - Example: `use` *`filename`*`, clear` works if data is already in memory.

describe

describe provides basic information about a Stata dataset.
describe [*varlist*] provides basic information about specified variables.

- Number of observations and variables
- Size of file (in bytes)
- Most recent timestamp
- Summary Information
  - Variable Name
  - Storage Type
  - Display Format
  - Value Label
  - Variable Label

summarize

summarize gives summary statistics for the variables in the dataset. summarize [*varlist*] provides summary statistics for specified variables.

- Number of Observations
- Mean
- Standard Deviation
- Minimum
- Maximum

The option detail provides additional statistics.

- Skewness
- Kurtosis
- Four largest (smallest) values
- Various percentiles (1, 5, 10, 25, 50, 75, 90, 95, 99)

tabulate (One-way)

- tabulate *varname* or tab1 *varlist* produces a frequency table for a variable, or list of variables.
  - Example: tab *var1*
- However, using tab alone will not provide frequencies with respect to missing observations.
- Frequencies of missing observations requires including the option missing.
  - Example: tab *var1*, m
- Default is to produce a frequency table featuring value labels
- Creating a frequency table without value labels requires including the option nolabel
  - Example: tab1 *var1 var2 var3*, nol

tabulate (Two-way)

- tabulate *varname1 varname2* or tab2 *varlist* produces a contingency table for a pair of variables.
  - Example: tab *var1 var2*
  - Usually, dependent variable is listed first, followed by independent variable
- Can report row, column, and cell relative frequencies using row, column, and cell options.
- Can report various measures of association (e.g., Chi-Squared (χ^2), Cramer's V) (See help tabulate twoway for full list of options)

codebook

`codebook` examines the variable names, labels, and data to produce a codebook describing the dataset.
`codebook` [*varlist*] provides a codebook for the specified variables.

- Variable Name and Variable Label
- Type
- Value Label
- Range (Smallest and Largest Values)
- Unique Values
- Units
- Missing
- Tabulation (Small number of unique values)
  - Frequency
  - Numeric Value
  - Value Label

What is a Log file?

- A file that keeps a "permanent" record of the output displayed in the Results window
- When a log file is open, Stata will write the results of executed commands to both the Results window and the log file

Commands

- Basic Command: `log using` *`filename`*
  - *filename* is the name user gives to the log file
- Most Common Options: `text` and `replace`
  - `text` gives the log file a .txt file extension, which allows the file to be opened in another text editor (e.g. Notepad, Notepad++, Sublime Text, Atom)
  - `replace` tells Stata to overwrite the file if a file with the same filename already exists
- To close an open log file, use `log close`

New Features in Stata 15

Stata 15 includes a number of refinements and new features, including:

- `bayes:` prefix for estimating Bayesian regression models (Example: `bayes: regress` *depvar indvar1 indvar2*)
- Markdown and dynamic documents – Integrating Stata code into documents (e.g., results graphs)
- Including transparency features into graphs
- Number of new methods
  - Spatial Autoregressive Models
  - Bayesian Multilevel Models
  - Nonlinear Multilevel Models

Click here for the full list of new features.

Available Resources

- Stata Documentation
- Stata Press
- UCLA Institute for Digital Research and Education
- Stata Cheat Sheets
- ISRC Workshops

Any Questions?