

Econometric Softwares: Stata

Jérémy Do Nascimento Miguel

Class 4 Spring 2024

Session 4: Outline

- Basics of visualizations in Stata
- Formatting techniques
- Types of graphs
- Combining and exporting graphs

Basics I

Common graph types:

- Scatter plots
- Line graphs
- Bar charts
- Fitted Lines
- Maps
- many more

Basic principles

- Distinctive colors ("warm" vs "cold" colors, e.g., red vs blue)
- Clean labels on axis

Basics II

- Scaled to make patterns visible
- Texts are readable but not too large
- Explanatory notes: should be readable alone

Personal preferences:

- Background should be white
- Modify font
- Specified graph settings in global

Basics

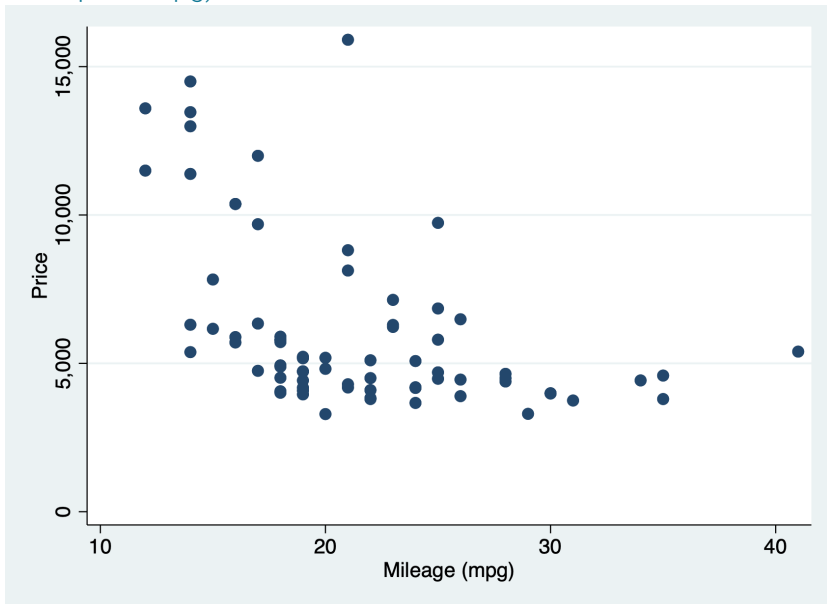
Most graphs are `twoway` plots: variables for **y-axis** and **x-axis** should be specified

- `twoway (plottype yvar xvar, plotoptions),`
- `legend (setting)`
- `xlabel (numlist, xlabelsettings)`
- `ylabel (numlist, ylabelsettings)`
- `xtitle ("text", xtitlesettings)`
- `ytitle ("text", ytitlesettings)`
- `title ("text", titlesettings)`
- `subtitle ("text", subtitlesettings)`
- `scale (#)`
- `xsize (#) ysize`
- `scheme()`

check out the help file

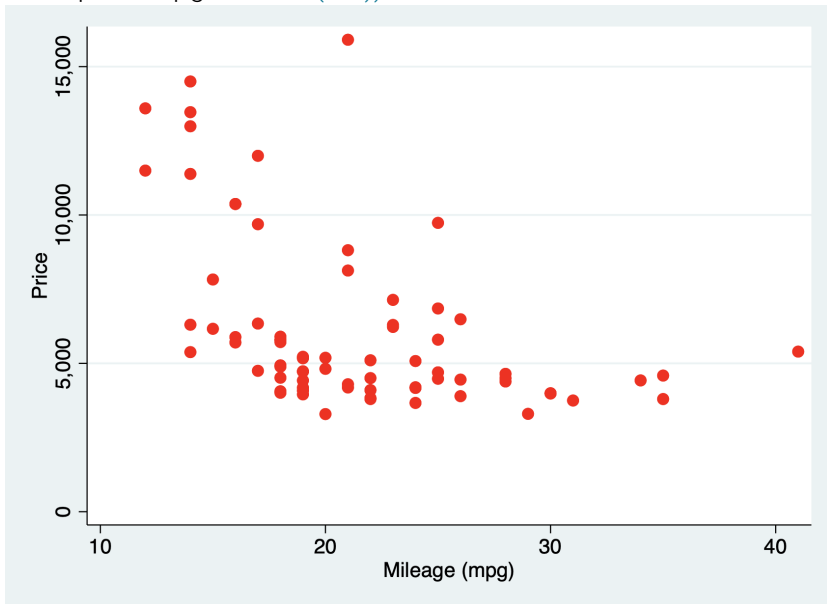
Basic *twoway* plot

twoway (scatter price mpg)





Changing colors

twoway (scatter price mpg, mcolor(red))

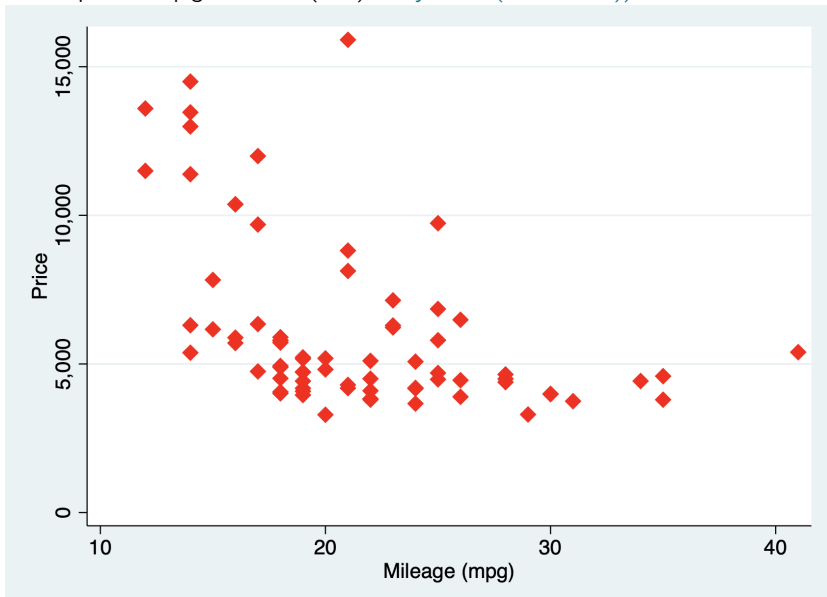


Colors in Stata

1		dkgreen	6		sienna	11		lime
2		orange_red	7		orange	12		brown
3		navy	8		magenta	13		purple
4		maroon	9		cyan	14		olive_tea
5		teal	10		red	15		ltblue


Changing marker shape

`twoway (scatter price mpg, mcolor(red) msymbol(diamond))`



Marker shapes in Stata


1  circle

6  plus


11  smcircle


2  diamond

7  circle_hollow

12  smdiamond


3  square

8  diamond_hollow


13  smsquare

4  triangle

9  square_hollow

14  smtriangle

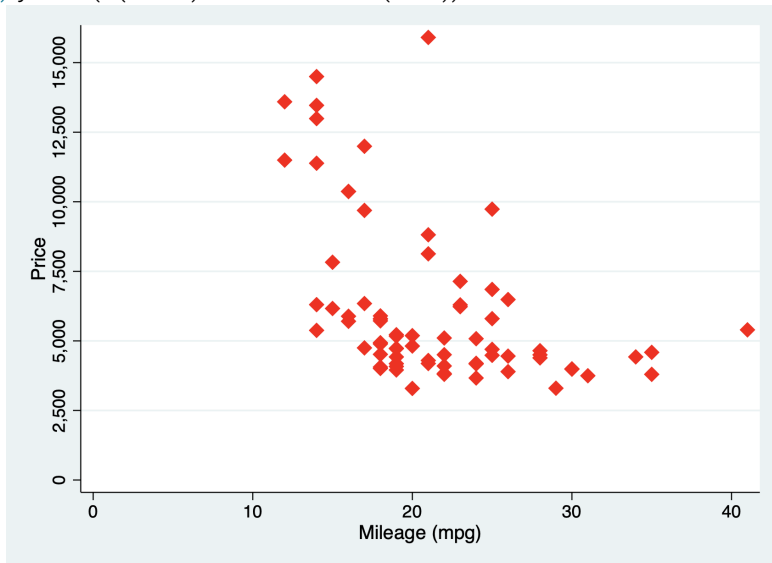
5  x

10  triangle_hollow

15  smx

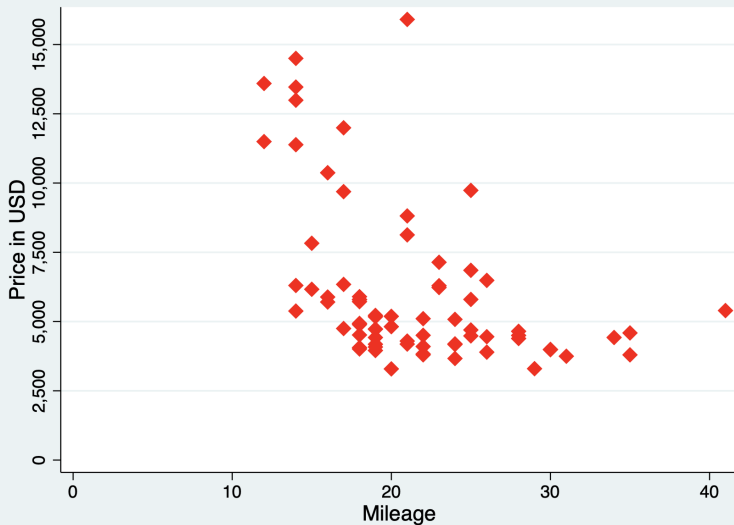
Changing axis titles

```
twoway (scatter price mpg, mcolor(red) msymbol(diamond)), xlabel(0(10)40,  
labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9))
```



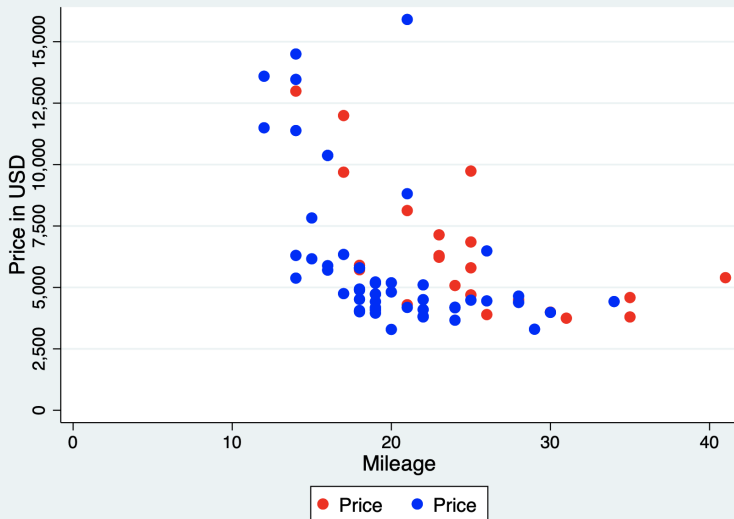
Plot by groups - overlaying plots

twoway (scatter price mpg, mcolor(red) msymbol(diamond)), xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9)) xtitle("Mileage", size(*1.1)) ytitle("Price in USD", size(*1.1))



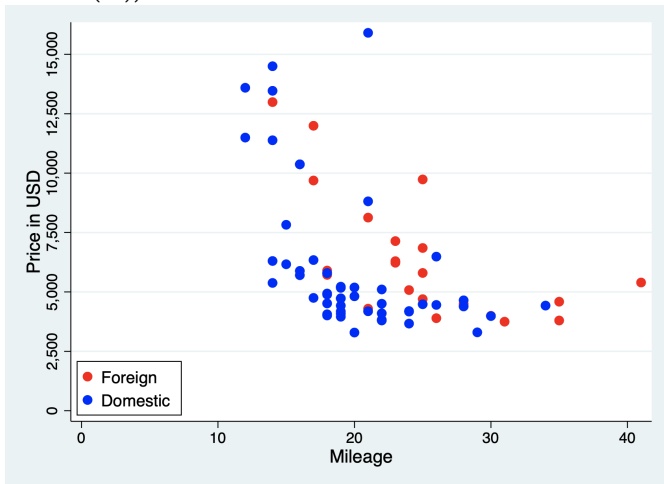
Formatting legend

```
twoway (scatter price mpg if foreign==1, mcolor(red)) (scatter price mpg if foreign==0,  
mcolor(blue)), xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9))  
xtitle("Mileage", size(*1)) ytitle("Price in USD", size(*1))
```



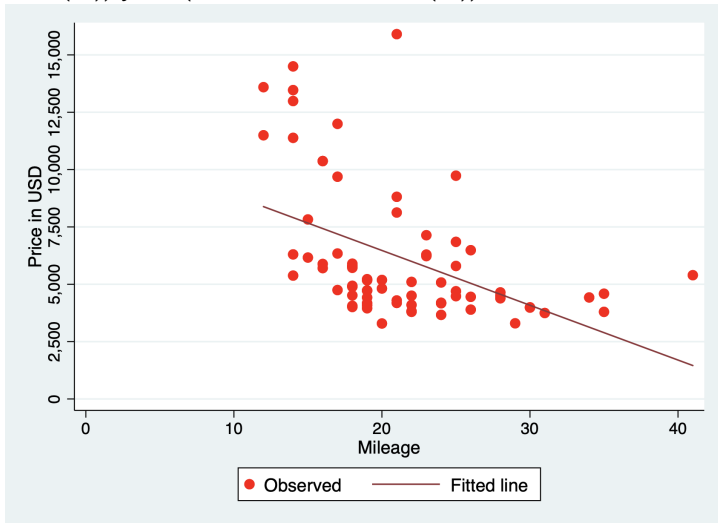
Adding linear fitted line

```
twoway (scatter price mpg if foreign==1, mcolor(red)) (scatter price mpg if foreign==0,  
mcolor(blue)), legend(order(1 "Foreign" 2 "Domestic") pos(8) ring(0) rows(2))  
xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9)) xtitle("Mileage", size(*1))  
ytitle("Price in USD", size(*1))
```



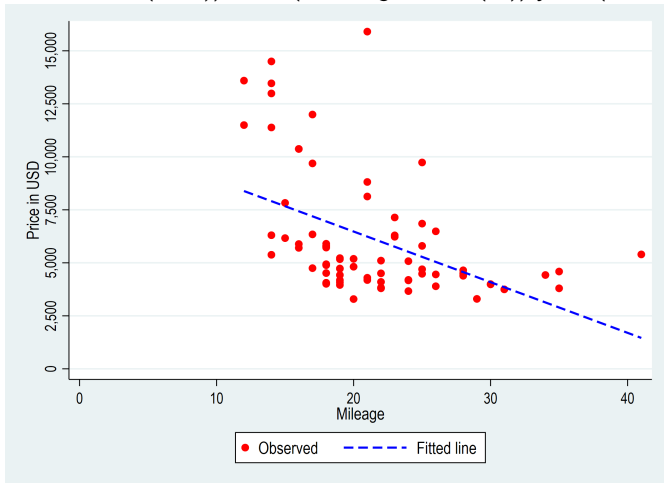
Adding linear fitted line

```
twoway (scatter price mpg, mcolor(red)) (lfit price mpg), legend(order(1 "Observed" 2  
"Fitted line")) xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9))  
xtitle("Mileage", size(*1)) ytitle("Price in USD", size(*1))
```



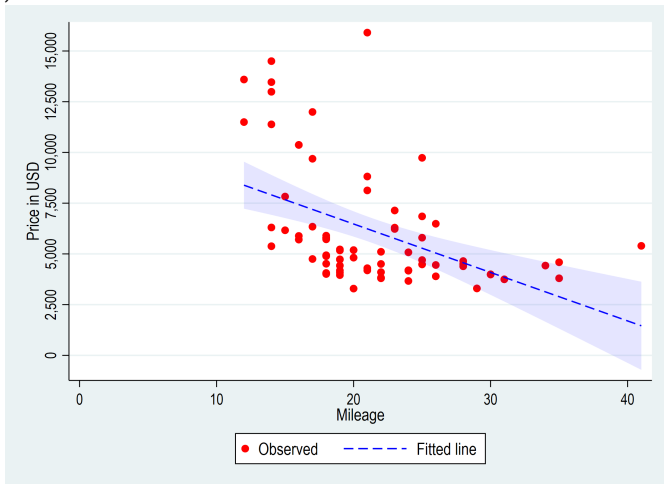
Adding linear fitted line with CI

```
twoway (scatter price mpg, mcolor(red)) (lfit price mpg, lcolor(blue) lpattern(dash)
lwidth(*1.5)), legend(order(1 "Observed" 2 "Fitted line")) xlabel(0(10)40, labsize(*0.9))
ylabel(0(2500)15000, labsize(*0.9)) xtitle("Mileage", size(*1)) ytitle("Price in USD", size(*1))
```



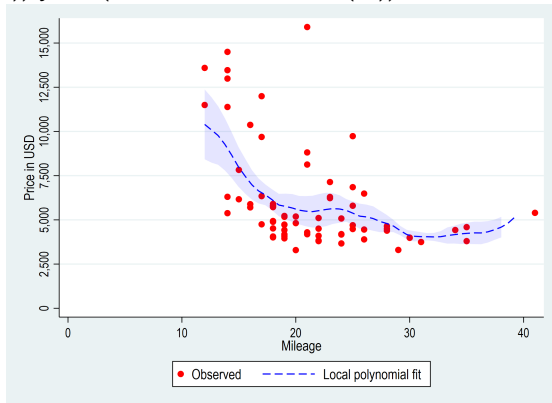
Adding linear fitted line with CI

```
twoway (scatter price mpg, mcolor(red)) (lfitci price mpg, clcolor(blue) clpattern(dash)
fcolor(blue%10) alwidth(none)), legend(order(1 "Observed" 3 "Fitted line")) xlabel(0(10)40,
labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9)) xtitle("Mileage", size(*1.1)) ytitle("Price
in USD", size(*1.1))
```



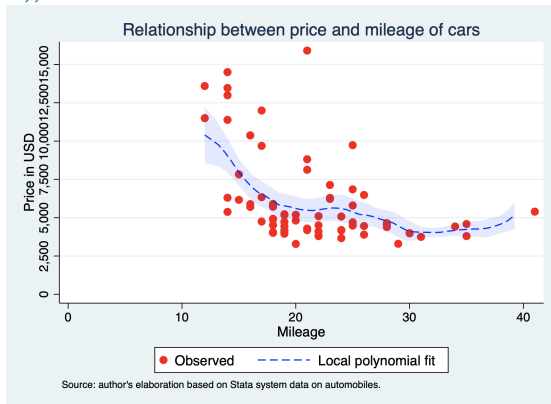
Adding local polynomial fit with CI

```
twoway (scatter price mpg, mcolor(red)) (lpolyci price mpg, bwidth(2) clcolor(blue)
clpattern(dash) fcolor(blue%10) alwidth(none)), legend(order(1 "Observed" 3 "Local
polynomial fit")) xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9))
xtitle("Mileage", size(*1)) ytitle("Price in USD", size(*1))
```



Adding title/subtitle and notes

```
twoway (scatter price mpg, mcolor(red)) (lpolyci price mpg, bwidth(2) clcolor(blue)
clpattern(dash) fcolor(bluelegend(order(1 "Observed" 3 "Local polynomial fit")))
xlabel(0(10)40, labsize(*0.9)) ylabel(0(2500)15000, labsize(*0.9)) xtitle("Mileage", size(*1))
ytitle("Price in USD", size(*1)) title("Relationship between price and mileage of cars",
size(*0.9)) notes("Source: author's elaboration based on Stata system data on
automobiles.", size(*0.9))
```



Other cool packages

You can go beyond the Stata basic graphics and have a look at some user-based packages. Few I like you can find through [ssc install](#) or online:

- [scatterfit](#)
- [catcibar](#)
- [A lot more on Asjad Naqvi website](#)

Change font: LMRoman10-Regular

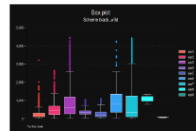
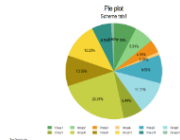
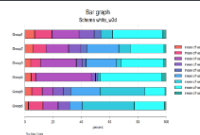
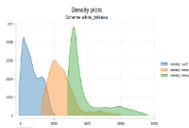
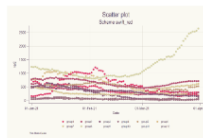
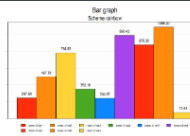
1. Download font here <https://www.fontsquirrel.com/fonts/Latin-Modern-Roman>
2. Convert the .odf file to .ttf
3. Install it in the font folder
4. Change fonts in your dofile `graph set window fontface "LMRoman10-Regular"`

Change scheme: see scheme pack

schemepack

A package with
ready-to-use Stata
schemes

ssc install schemepack, replace
GitHub:
<https://github.com/anjochaz/stata-schemes>



Settings I am using

I used to define global with all my graph settings:

```
*Graph option
    global plotregion plotregion(margin(b=0 t=2) color(white) fcolor(white)
lcolor(white) icolor(white) ifcolor(white) ilcolor(white))
    global graphregion graphregion(margin(l=0 r=4 b=-2 t=-1) color(white) fcolor(
white) lcolor(white) icolor(white) ifcolor(white) ilcolor(white))
    global blabel blabel(bar, format(%4.3f))
    global ylabel ylabel(, nogrid ang(hor) labsiz(e=small))
    global scheme scheme(white_tableau)
```

Exercise 4.1

1. Open the datafile hh.dta. This data is a household-level expenditure survey data and contains basic information on household demographics, consumption and poverty.
2. Generate a scatter plot with log per capita consumption on y-axis and area of land owned on x-axis.
3. Restrict your sample in the graph to land area above 0 and below 200.
4. Format axis labels, axis titles, colors (depending on your preferences) and sizes of text if necessary.
5. Add local polynomial fit with confidence intervals to your scatter plot. Keep the same sample restriction as in your scatter plot. Use bandwidth of 7 in the options.
6. Format colors and line of polynomial fit so that it is distinctive from scatter plot.
7. Format legends so that it is clear what is plotted in the graph.
8. Add title explaining what this graph is about and source of data as a note. Econometric

Bar Charts

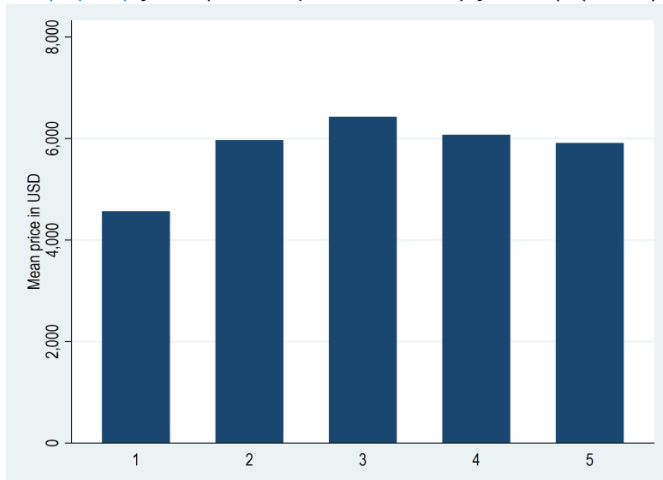
There are two ways you can generate bar charts in Stata:

1. `graph bar yvars, over(groupvar)`
2. `twoway (bar xvar yvar)`

For horizontal bar charts: `graph hbar yvars, over(groupvar)`

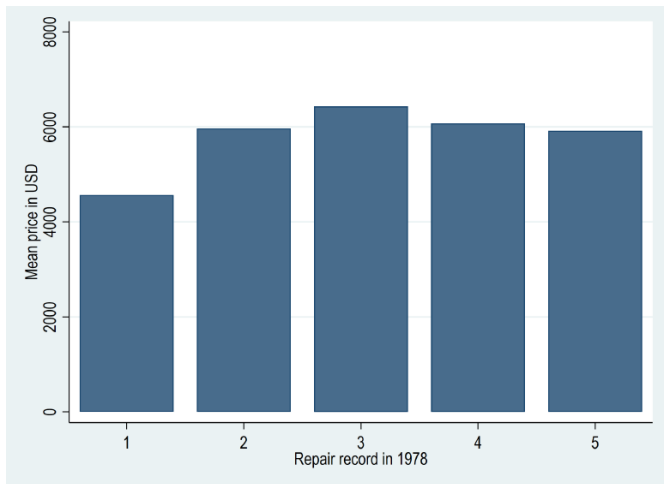
Bar charts: graph bar

graph bar price, over(rep78) ytitle("Mean price in USD") ylabel(0(2000)8000)



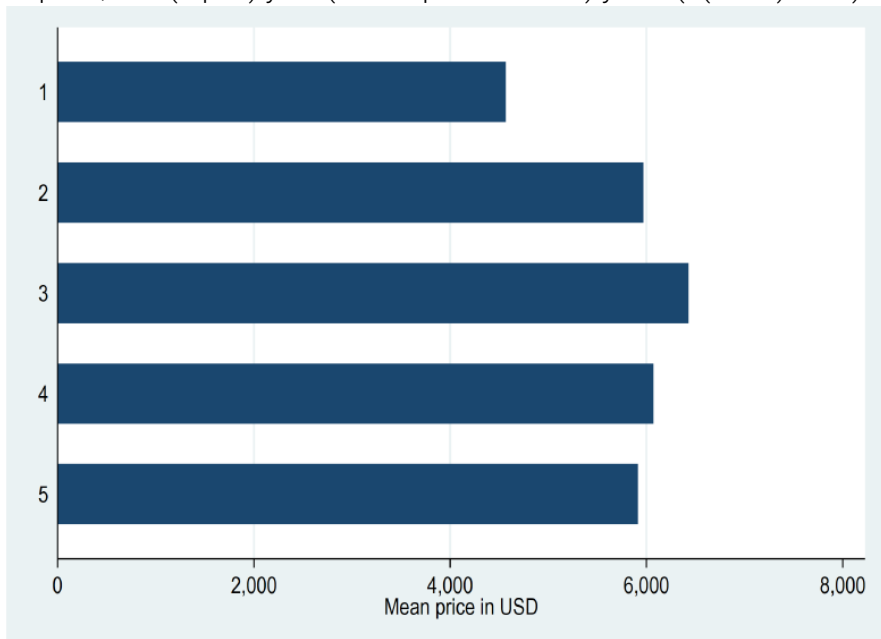
Bar charts: twoway bar

- `bys rep78: egen meanpricerep78 = mean(price)`
- `twoway (bar meanpricerep78 rep78, barwidth(0.8)), xtitle("Repair record 1978")
ytitle("Mean price in USD") ylabel(0(2000)8000)`



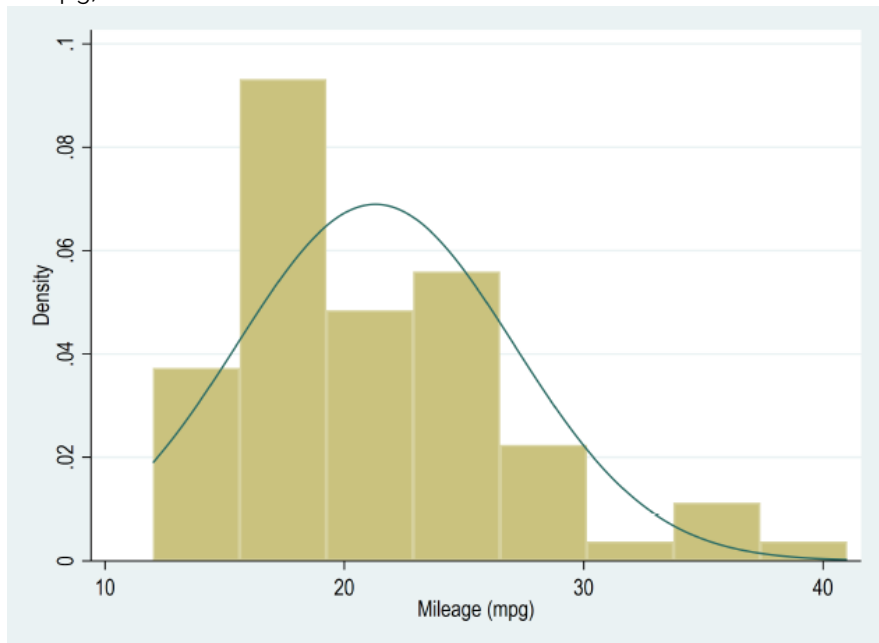
Bar charts: graph hbar

```
graph hbar price, over(rep78) ytitle("Mean price in USD") ylabel(0(2000)8000)
```



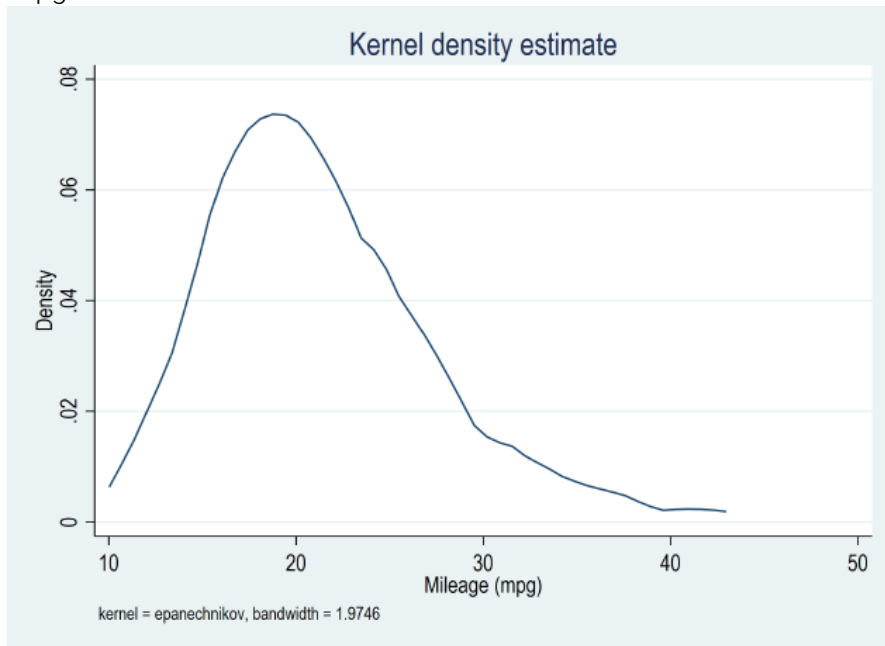
Distribution plots: histogram

histogram mpg, normal



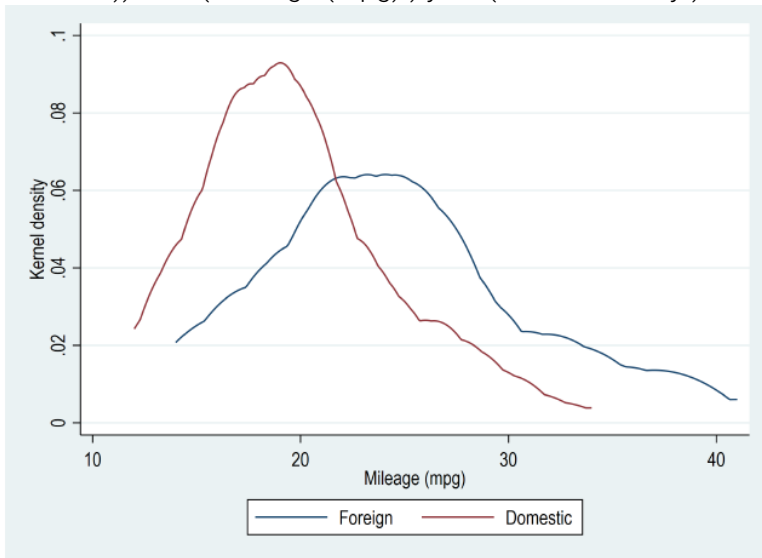
Distribution plots: kernel density

kdensity mpg



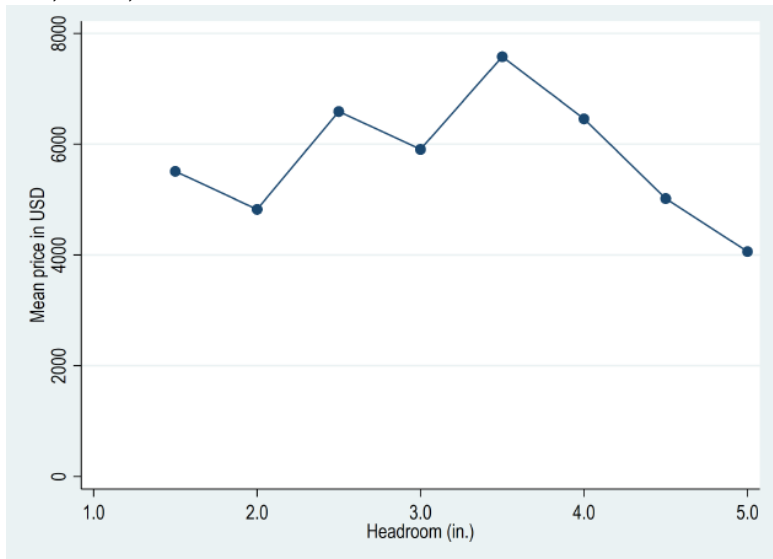
Distribution plots: kernel density by groups

```
twoway (kdensity mpg if foreign==1) (kdensity mpg if foreign==0), legend(order(1  
"Foreign" 2 "Domestic")) xtitle("Mileage (mpg)") ytitle("Kernel density")
```



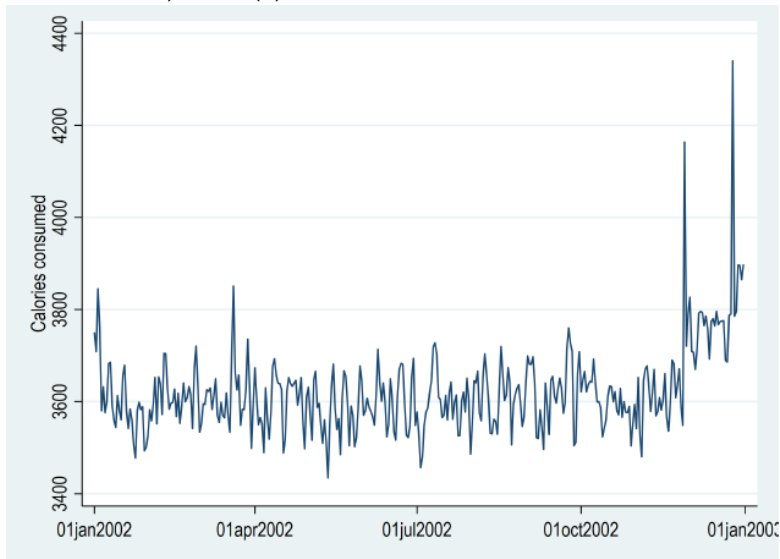
Connected Lines

- `bys headroom: egen meanpricehead = mean(price)`
- `twoway (connected meanpricehead headroom), ytitle("Mean price in USD")
ylabel(0(2000)8000)`



Time series plot

- `sysuse tsline2, clear`
- `tsset day`
- `twoway (tsline calories), xtitle("")`



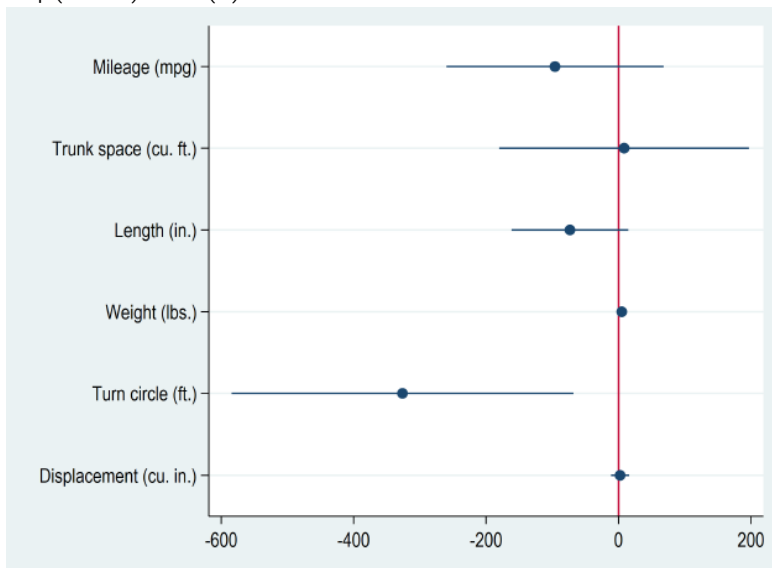
Coefficient plot: `coefplot`

Sometimes we want to plot our estimated coefficients with its confidence intervals

- Command to use: `coefplot`
- To install (if not yet installed): `ssc install coefplot`
- A lot of examples [here](#)

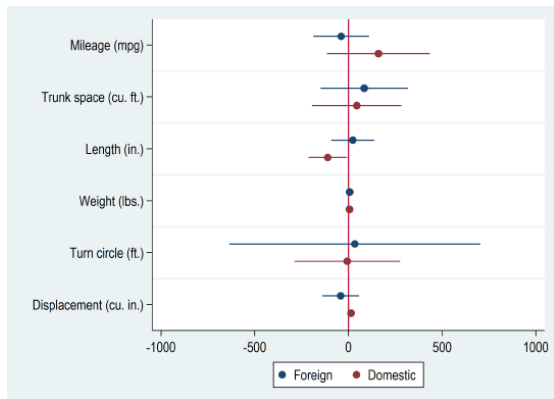
Coefficient plots: coefplot

- regress price mpg trunk length weight turn displacement
- coefplot, drop(cons) xline(0)



Coefficient plots: coefplot

- regress price mpg trunk length weight turn displacement if foreign==1
- est store foreign1 ← store your estimation results
- regress price mpg trunk length weight turn displacement if foreign==0
- est store foreign0
- coefplot (foreign1, label("Foreign")) (foreign0, label("Domestic")), drop(cons) xline(0)



Combining graph

To combine graph:

1. Run the code that generates your graphs: save them in disk using `saving()` option or give name using `name()` option
2. To combine: `graph combine plotname1 plotname2, cols(2) rows(1)`
3. Number of columns and rows in a combined graph is specified using options `cols(#)` and `rows(#)`
4. You can rescale content of your all plots in a combined graph using options `altshrink` (shrinks text, line width in all plots) or `iscale()` to rescale the content

Exporting graph

To export graphs:

1. Run the code that generates your graphs
2. To export the graph that is displayed in graph editor window: `graph export filename, as(formatname) replace`
3. Or you can directly use the name of graph you want to export (if you have given the name): `graph export filename, as(formatname) name(plotname) replace`
4. Supported vector formats to export: ps (PostScript), eps (Encapsulated PostScript), pdf, svg (Scalable Vector Graphics) ...
5. Supported bitmap formats to export: png, jpg, tif, gif ...

Exercise 4.2

1. Generate a bar graph showing poverty rate and extreme poverty by urban/rural location (hint: use command `graph bar`)
2. Generate a kernel density plot depicting density of employment ratio (`empratio`) by poor and non-poor households (using `poor` variable). Format line colors, line patterns and legends so that it is clear what is depicted in the graph.
3. Add titles/notes to both graphs, combine them (pay attention to scales!) and export them as a single figure (with two plots horizontally placed).
4. Run a regression of poverty dummy (`poor`) on following variables (independent variables) `hhsz`, `nchild`, `nmigrant`, `empratio`, `land`, `urban`, `hhh female`.
5. Plot all coefficients (excluding constant term) using `coefplot`.