Quick Stata Tips

Version 2.1

Todd R. Jones[1]

May 12, 2025

[1]Mississippi State University, IZA, and CESifo. toddrjones.com. Twitter page. Bluesky page.

To Ari, Charlotte, Annie, and Ned

Preface

This book compiles a number of my "Quick Stata Tips." I am basing this book on Stata version 18 (and sometimes 17) on a Mac. However, much—though not all—of the content in this book applies to (recent) prior versions of Stata. Many—though not all—of the tips build on concepts introduced in earlier tips.

I have picked up these tips over the years from many different sources. For example, StackOverflow, StataList, Google, my own coding, and other people. For a book like this, it's hard to know how to acknowledge everyone, so I will keep it very general and say that I would like to thank various people and websites from whom I learned about some of these tips, as well as others who have developed the packages that I refer to.

I've companion Stata .do file with code for most of the tips. You can download it here.

In this updated version, I have added 43 new tips, starting at number 98.

I hope you find it helpful!

Contents

1 fre

If you want to look at a one-way frequency table of a variable, *fre* displays both values and labels at the same time, while *tabulate* does not. To use *fre*:

```
ssc install fre
sysuse auto2, clear
tab foreign
fre foreign
```

```
. tab foreign
```

Car origin	Freq.	Percent	Cum.
Domestic	52	70.27	70.27
Foreign	22	29.73	100.00
Total	74	100.00	

```
. fre foreign

foreign ── Car origin
```

		Freq.	Percent	Valid	Cum.
Valid	0 Domestic	52	70.27	70.27	70.27
	1 Foreign	22	29.73	29.73	100.00
	Total	74	100.00	100.00	

2 mdesc

To quickly see how many observations of each variable are missing, use *mdesc*:

```
ssc install mdesc
sysuse lifeexp, clear
mdesc
```

```
. mdesc
```

Variable	Missing	Total	Percent Missing
region	0	68	0.00
country	0	68	0.00
popgrowth	0	68	0.00
lexp	0	68	0.00
gnppc	5	68	7.35
safewater	28	68	41.18

3 inlist and inrange

```
keep if inlist(state, "AL", "AK", "AZ")
```

is the same as:

```
keep if state=="AL" | state=="AK" | state=="AZ"
```

And:

```
keep if inrange(distance, 10, 91)
```

is the same as:

```
keep if distance>=10 & distance<=91
```

4 Multicursor Mode

Stata supports multi cursor mode. With a Mac, hold down Option and drag the cursor. In Windows, hold down Alt.

```
bys month year: egen monday_avg
bys month year: egen tuesday_avg
bys month year: egen wednesday_avg
bys month year: egen thursday_avg
bys month year: egen friday_avg
```

```
bys month day: egen monday_avg
bys month day: egen tuesday_avg
bys month day: egen wednesday_avg
bys month day: egen thursday_avg
bys month day: egen friday_avg
```

You can also place multiple cursors. On a Mac, hold down Command and click. On Windows, hold down Control and click.

```
bys make (displaceme):
bys gear_ratio (displaceme):
bys turn (displaceme):
bys weight (displaceme):
```

```
bys make (weight):
bys gear_ratio (weight):
bys turn (weight):
bys weight (weight):
```

5 compare

To quickly compare two variables to see how often the values of the first are higher, lower, and equal to the second, use *compare*:

```
sysuse auto2, clear
compare mpg trunk
```

```
. compare mpg trunk
```

		Difference		
	Count	Minimum	Average	Maximum
mpg<trunk	**17**	**-10**	**-3.529412**	**-1**
mpg=trunk	**1**			
mpg>trunk	**56**	**1**	**11.03571**	**27**
Jointly defined	**74**	**-10**	**7.540541**	**27**
Total	**74**			

6 Scroll Buffer Size

To have the ability to scroll up as far as possible in your Results window, run this code:

```
set scrollbufsize 2048000
```

7 ereplace

It is not possible to replace when using *egen* commands. You can instead use *ereplace*.

Here is the old, workaround way to do it:

```
sysuse auto2, clear
*This next line doesn't work:
*replace mpg = max(mpg)
*So you instead have to do something like:
egen mpg2 = max(mpg)
drop mpg
rename mpg2 mpg
```

Here is the easier way:

```
ssc install ereplace
sysuse auto2, clear
ereplace mpg = max(mpg)
```

8 r(table)

After a regression, you can use *r(table)* to directly get the 95% confidence interval, t-statistic, p-value, standard error, beta, etc.

```
sysuse auto2, clear
reg trunk weight
matrix list r(table)
local weight_lower_95ci = r(table)[5,1]
di "`weight_lower_95ci'"
```

```
. matrix list r(table)

r(table)[9,2]
           weight      _cons
     b  .00369959  2.5860023
    se  .00048021  1.4966048
     t  7.7041179  1.7279126
pvalue  5.464e-11  .08829094
    ll  .00274231  -.3974248
    ul  .00465687  5.5694295
    df         72         72
  crit  1.9934636  1.9934636
 eform          0          0

. local weight_lower_95ci = r(table)[5,1]

. di "`weight_lower_95ci'"
.0027423081490285
```

9 ssc hot

Use *ssc hot* to see the most popular user-contributed SSC Stata packages:

```
ssc hot, n(100)
```

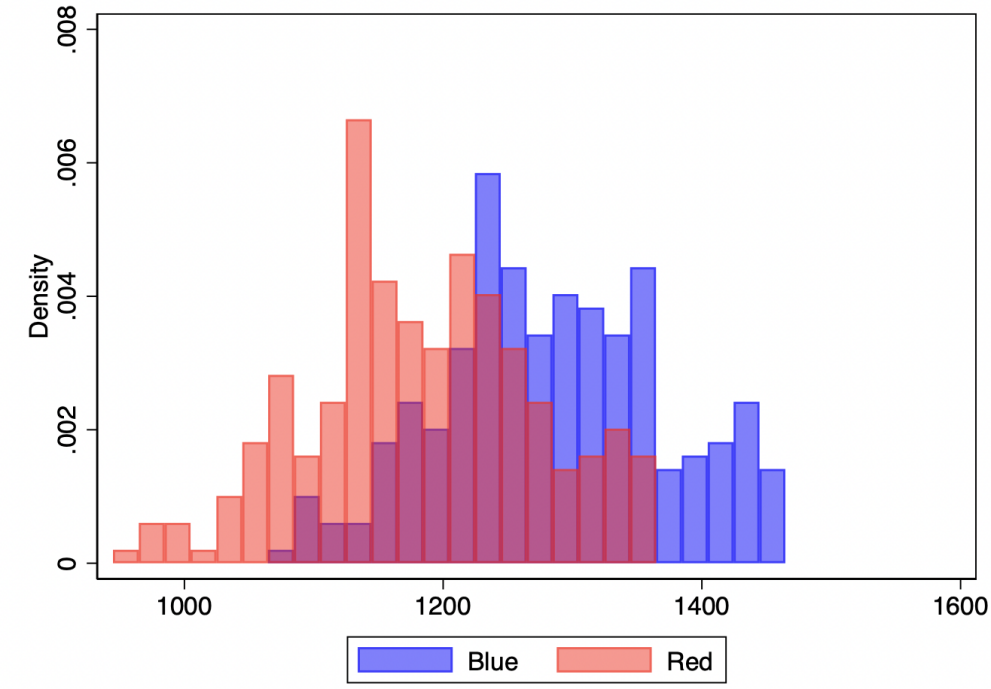
Top 100 packages at SSC

Rank	Oct 2023 # hits	Package	Author(s)
1	123406.3	xtcd2	Jan Ditzen
2	116683.0	distinct	Nicholas J. Cox, Gary Longton
3	103664.3	geodist	Robert Picard
4	103469.2	labutil	Nicholas J. Cox
5	101308.3	asgen	Attaullah Shah
6	55787.6	outreg2	Roy Wada
7	53809.7	estout	Ben Jann
8	40429.7	filelist	Robert Picard
9	36184.3	winsor2	Yujun Lian
10	35766.2	xtdcce2	Jan Ditzen
11	29871.7	reghdfe	Sergio Correia
12	28110.0	asdoc	Attaullah Shah
13	22864.3	ftools	Sergio Correia
14	15280.3	coefplot	Ben Jann
15	13431.3	logout	Roy Wada
16	13155.8	ivreg2	Steven Stillman, Christopher F Baum, Mark E Schaffer
17	13128.1	ivreg29	Mark E Schaffer, Christopher F Baum, Steven Stillman
18	13102.7	ivreg210	Christopher F Baum, Steven Stillman, Mark E Schaffer
19	13047.0	ivreg28	Mark E Schaffer, Steven Stillman, Christopher F Baum
20	12244.4	psmatch2	Barbara Sianesi, Edwin Leuven
21	11419.7	winsor	Nicholas J. Cox
22	11241.3	fre	Ben Jann
23	10925.3	egenmore	Nicholas J. Cox
24	10385.3	unique	Tony Brady
25	8837.3	bcuse	Christopher F Baum
26	8546.0	ranktest	Mark E Schaffer, Frank Kleibergen, Frank Windmeijer
27	8218.0	mdesc	Dan Blanchette, Rose Anne Medeiros
28	7996.0	sum2docx	Chuntao Li, Yuan Xue
29	6857.0	fs	Nicholas J. Cox
30	6740.5	rangestat	Nicholas J. Cox, Roberto Ferrer, Robert Picard
31	6431.8	spmap	Maurizio Pisati
32	6275.3	binscatter	Michael Stepner
33	6227.0	confirmdir	Dan Blanchette
34	5851.0	carryforward	David Kantor

10 opacity

To add opacity to your graph, use %*X* after the color, where 0≤X≤100:

```
sysuse sp500, clear
replace high = high+80
twoway (hist high, width(20) color(blue%50)) \\\
   (hist low, width(20) color(red%50)), \\\
   scheme(s1mono) legend(order(1 "Blue" 2 "Red"))
```



11 Previous line of code

To retrieve the previous line(s) of code in the Command Prompt:
fn key + up arrow (Mac)
Page Up (PC)
Control + R (Mac or PC)

And to do the reverse:
fn key + down arrow (Mac)
Page Down (PC)

12 Sort with, but don't group by

When using *bysort*, say you have a variable you want to sort with, but *not* group by. Put this variable in parentheses:

```
sysuse census, clear
*keep the least-populous state within each region, so sort by pop
* within each region:
bys region (pop): keep if _n==1
```

13 Open multiple instances of Stata

On a Mac, to open up multiple instances of Stata, type the following into Terminal:

```
open -n /Applications/Stata/StataMP.app
```

(You may have to modify *StataMP* if you have a different version.

14 isid

To check if you data are unique at a certain level, use *isid*. An error means that it is not unique, while no error means that it is unique.

```
sysuse auto2, clear
isid foreign
isid foreign make
```

```
. . sysuse auto2, clear
(1978 automobile data)

. isid foreign
variable foreign does not uniquely identify the observations
r(459);

. isid foreign make
```

15 texdoc

estout and *outreg2* are good to create .tex files, but they can't always do what I want. I've switched to the fully customizable *texdoc*. This version loops over both the right hand side variables and the different regression specifications.

```
capture ssc install texdoc
sysuse auto2, clear
global spec1 "if foreign"
global spec2 "if !foreign"
global spec3 "if !foreign & _n>5"
local vars mpg headroom trunk weight length turn
foreach spec in 1 2 3 {
reg price `vars' ${spec`spec'}
foreach i in `vars' {
local b `i' `spec': di %6.2fc b[`i'] local se `i' `spec': di %6.2fc se[`i']
qui test `i'=0
local `i' p `spec': di %12.2fc r(p)
local `i' star `spec'=cond(``i' p `spec''<.01,"***",cond(``i' p `spec''<.05,"**"
,cond(``i' p `spec''<.1,"*","")))
}
local N=e(N)
local N `spec': di %12.0fc `N' scalar r2=e(r2)
local r2 `spec': di %6.2fc r2
sum price if e(sample)
local ymean `spec': di %12.2fc r(mean)



}
foreach i in `vars' {
local b `i' "" local se `i' ""
local lab: variable label `i' local tex b ` i ' "`lab'"
local tex se `i' ""
}
local tex Y Mean "Y Mean"
local tex Observations "N"
local tex R Squared "R-squared" local tex Sample "Sample"
foreach spec in 1 2 3 { foreach i in `vars' {
local tex b `i' = "`tex b `i '' & `b `i' `spec ''`` i' star `spec''"
local tex se `i' = "`tex se `i '' & (`se `i' `spec'')"
}
local tex Y Mean = "`tex Y Mean' & `ymean `spec''"
 local N=e(N)
local N `spec': di %12.0fc `N' scalar r2=e(r2)
local r2 `spec': di %6.2fc r2
sum price if e(sample)
local ymean `spec': di %12.2fc r(mean)

local tex Observations = "`tex Observations' & `N `spec''"
local tex R Squared = "`tex R Squared' & `r2 `spec''"
}
texdoc init " test loc . tex ", replace force foreach i in `vars' {
tex `texb `i'' \\
tex `texse `i'' \\\addlinespace }
tex `tex Y Mean' \\
tex `tex Observations' \\
tex `tex R Squared' \\
tex & Foreign & Not Foreign & Not foreign, rows 6+ texdoc close
```

```
Mileage (mpg) & -14.03 & 110.96 & -13.16 \\
& ( 66.66) & (142.91) & (152.91) \\ \addlinespace
Headroom (in.) & -12.65 & -513.29 & -443.88 \\
& (682.46) & (459.76) & (472.97) \\ \addlinespace
Trunk space (cu. ft.) & 119.25 & 128.95 & 149.47 \\
& (108.04) & (140.09) & (141.23) \\ \addlinespace
Weight (lbs.) &  4.44** &  6.78*** &  7.66*** \\
& (  1.86) & (  1.27) & (  1.33) \\ \addlinespace
Length (in.) &  24.54 & -120.14** & -187.21*** \\
& ( 57.52) & ( 52.75) & ( 60.94) \\ \addlinespace
Turn circle (ft.) & -60.56 & -39.74 & -49.98 \\
& (304.59) & (144.85) & (148.31) \\ \addlinespace
Y Mean &  6,384.68 &  6,072.42 &  6,180.34 \\
N &  22 &  52 &  47 \\
R-squared &  0.80 &  0.57 &  0.61 \\
& Foreign & Not Foreign & Not foreign, rows 6+

```

16 Check if variable is constant within group

To check if a variable is constant within group, you can do:

```
bys group (var): gen a = var[1]==var[_N]
tab a
```

If all values of *a* are 1, the variable is constant within group. If some values of a group are missing, that group is considered not constant.

For a one-line solution:

```
bysort group: assert var==var[1]
```

17 trim

To get rid of the leading and trailing space of a string variable, use *trim*:

```
clear all
input str12 str
"String A "
" String B "
" String C"
end

replace str = trim(str)
```

18 group

To create a variable identifying groups, use *group*:

```
sysuse xtline1, clear
egen grp = group(day)
*check that it worked
sort day
```

	person	day	calories	grp
1	Sam	01jan2002	4000	1
2	Arnold	01jan2002	4500	1
3	Tess	01jan2002	3700	1
4	Tess	02jan2002	3700	2
5	Sam	02jan2002	4000	2
6	Arnold	02jan2002	4500	2
7	Tess	03jan2002	3718	3
8	Sam	03jan2002	3985.6	3
9	Arnold	03jan2002	4490.2	3
10	Arnold	04jan2002	4499.2	4
11	Sam	04jan2002	3986.8	4
12	Tess	04jan2002	3702.4	4
13	Tess	05jan2002	3646.4	5
14	Sam	05jan2002	3779.8	5
15	Arnold	05jan2002	4009	5
16	Tess	06jan2002	3655.8	6
17	Arnold	06jan2002	4028.2	6
18	Sam	06jan2002	3780	6
19	Tess	07jan2002	3644.8	7
20	Arnold	07jan2002	4076.8	7
21	Sam	07jan2002	3807	7
22	Sam	08jan2002	3804.6	8
23	Tess	08jan2002	3649.2	8

19 tempfiles

You can use tempfiles to save and then repeatedly load data. They are deleted when you quit Stata.

```
sysuse auto2, clear
tempfile a
save `a'
```

Later:

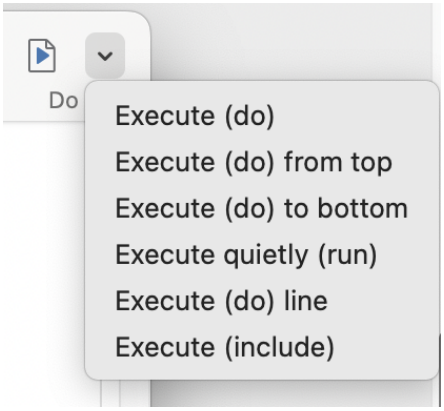
```
use `a', clear
```

Or:

```
merge m:1 id using `a'
```

20 Execute (include)

Usually, you cannot access in the Command Prompt the locals and tempfiles you created in the Do-File Editor, and vice versa. But you can if you run your code with *Execute (include)* (as opposed to the default *Execute (do)*). My coding preference is to be constantly shifting between the Command Prompt and Do-file Editor, and this tip allows me to do so.



In addition, I have programmed a keyboard shortcut (command+return) for *Execute (include)*. In Mac, I did this with:

System Preferences - Keyboard - Shortcuts - App Shortcuts - Stata - eg. Execute (include).

If you do prefer to run your code the traditional way, you can use keyboard shortcuts to run/*Execute (do)* your code. On Windows: Control+Shift+D. On Mac: Command+Shift+D.

21 Control placement of newly-created variables

When you *generate* a new variable, use *before* to place it before another variable, and *after* to place it after a variable.

```
sysuse auto2, clear
*create version of price in units of thousands
gen price_k = price/1000, before(price)

*create lower case version of make and place after make
gen make_lower = lower(make), after(make)
```

	make	make_lower	price_k	price
1	AMC Concord	amc concord	4.099	4,099
2	AMC Pacer	amc pacer	4.749	4,749
3	AMC Spirit	amc spirit	3.799	3,799
4	Buick Century	buick century	4.816	4,816
5	Buick Electra	buick electra	7.827	7,827
6	Buick LeSabre	buick lesabre	5.788	5,788
7	Buick Opel	buick opel	4.453	4,453
8	Buick Regal	buick regal	5.189	5,189

22 substr

substr is one way to extract characters from a string. The format is *substr(var, X, Y)*, where *X* indexes the first position, and *Y* is the number of characters you want to extract.

```
sysuse auto2, clear
*get the first two letters of the string
gen first = substr(make, 1, 2), after(make)
```

make	first
AMC Concord	AM
AMC Pacer	AM
AMC Spirit	AM
Buick Century	Bu
Buick Electra	Bu
Buick LeSabre	Bu
Buick Opel	Bu

23 browse if

To browse only a subset of your data, use *browse if*:

```
sysuse auto2, clear
*browse only the cars that begin with "A"
br if substr(make,1,1)=="A"
```

	make	price	mpg	rep78	headroom	trunk	weight	length	turn	displace…	gear_ratio	foreign
1	AMC Concord	4,099	22	Average	2.5	11	2,930	186	40	121	3.58	Domestic
2	AMC Pacer	4,749	17	Average	3.0	11	3,350	173	40	258	2.53	Domestic
3	AMC Spirit	3,799	22	.	3.0	12	2,640	168	35	121	3.08	Domestic
53	Audi 5000	9,690	17	Excellent	3.0	15	2,830	189	37	131	3.20	Foreign
54	Audi Fox	6,295	23	Average	2.5	11	2,070	174	36	97	3.70	Foreign

```
*go back to browsing all data
br
```

24 sysuse

sysuse loads toy datasets that are already in Stata. *sysuse dir* lists all such datasets. These can be useful when, for example, you want to use toy data to ask a question on StackOverflow or StataList.

```
sysuse dir
```

```
. sysuse dir
  auto.dta         citytemp.dta     lifeexp.dta      surface.dta
  auto16.dta       citytemp4.dta    network1.dta     tsline1.dta
  auto2.dta        educ99gdp.dta    network1a.dta    tsline2.dta
  autornd.dta      gitget.dta       nlsw88.dta       uslifeexp.dta
  bplong.dta       github.dta       nlswide1.dta     uslifeexp2.dta
  bpwide.dta       githubfiles.dta  pop2000.dta      voter.dta
  cancer.dta       gnp96.dta        sandstone.dta    xtline1.dta
  census.dta       kountry.dta      sp500.dta
```

```
*load the citytemp dataset
sysuse citytemp, clear
```

25 bysort

Use *bysort*.

26 _n and _N

Use *_n* to get the current observation number. Use *_N* to get the maximum observation number.

```
sysuse auto2, clear
keep rep78
sort rep78
*obs #:
gen n = _n
*obs # w/i group:
bys rep78: gen group_n = _n
*max obs #:
gen N = _N
*max obs # w/i group:
bys rep78: gen group_N = _N
```

	rep78	n	group_n	N	group_N
1	Poor	1	1	74	2
2	Poor	2	2	74	2
3	Fair	3	1	74	8
4	Fair	4	2	74	8
5	Fair	5	3	74	8
6	Fair	6	4	74	8
7	Fair	7	5	74	8
8	Fair	8	6	74	8
9	Fair	9	7	74	8
10	Fair	10	8	74	8
11	Average	11	1	74	30
12	Average	12	2	74	30
13	Average	13	3	74	30
14	Average	14	4	74	30
15	Average	15	5	74	30
16	Average	16	6	74	30

27 preserve/restore

You can save a copy of your data with *preserve*. You can then change the data and restore the saved version with *restore*. (You can also accomplish this goal using tempfiles.)

```
sysuse auto2, clear
*preserve the data to be used later
preserve
*change the data
keep if _n<5
scatter price mpg
*restore the data
restore
```

You can cancel the *preserve* with restore, not. Then you can *preserve* the data again.

```
*preserve again
preserve
keep if _n<10
*cancel the prior preserve
restore, not
*preserve again
preserve
```

28 capture

Use capture to allow Stata to keep running the code even if there is an error in the line. In this example, we want to *preserve* the data but are not sure if the data has been preserved before. If it has, then we will get an error. So we want to cancel the preserve. However, if we cancel the preserve and it hasn't been preserved, it will also give an error. So we will use *capture* to cancel the preserve if it exists; if it doesn't, the code will continue to run.

```
sysuse auto2, clear
capture restore, not
preserve
```

29 tab1

To make one-way frequency tables for multiple variables, use *tab1*:

```
sysuse auto2, clear
*tabulate (separately) make, price, and mpg:
tab1 make price mpg
*tabulate all variables:
tab1 *
```

With that said, it's probably better to instead use *fre*:

```
fre make price mpg
```

30 statastates

Use *statastates* for a crosswalk between two-digit US state abbreviation, state name, and state FIPS code:

```
capture ssc install statastates
sysuse census, clear
keep state2
statastates, abbreviation(state2) nogen
replace state_name = strproper(state_name)
```

	state2	state_name	state_fips
1	AK	Alaska	2
2	AL	Alabama	1
3	AR	Arkansas	5
4	AZ	Arizona	4
5	CA	California	6
6	CO	Colorado	8
7	CT	Connecticut	9
8	DE	Delaware	10
9	FL	Florida	12
10	GA	Georgia	13
11	HI	Hawaii	15
12	IA	Iowa	19
13	ID	Idaho	16
14	IL	Illinois	17

31 compress

If you have a long string variable, and then shorten its values, use ”compress” to shorten the variable. This makes the Data Editor easier to look at.

```
sysuse auto2, clear
replace make = "This is a long string...." in 1
replace make = substr(make, 1, 6)
```

	make
1	This i
2	AMC Pa
3	AMC Sp
4	Buick
5	Buick
6	Buick
7	Buick
8	Buick
9	Buick
10	Buick
11	Cad. D
12	Cad. E
13	Cad. S
14	Chev.

```
compress
```

	make
1	This i
2	AMC Pa
3	AMC Sp
4	Buick
5	Buick
6	Buick
7	Buick
8	Buick
9	Buick
10	Buick
11	Cad. D
12	Cad. E
13	Cad. S
14	Chev.

32 quietly

Place *quietly* in front of a command like *reg* to suppress the output.

```
sysuse auto2, clear
quietly reg trunk weight
```

33 set graphics off

Use *set graphics off* to not display graphs when they are created. This can be useful if you are making a lot of graphs and want to save them, but don't want the graphs to incessantly pop up on your screen.

```
set graphics off
```

To again display the graphs:

```
set graphics on
```

34 Undocumented and previously documented commands

To see undocumented commands:

```
help undocumented
```

To see commands that were previously documented, but are not documented anymore:

```
help prdocumented
```

35 Pull variables from the ACS

Use *getcensus* to pull variables from the American Community Survey (ACS). To be able to submit more than 500 requests per day, you need have to get a Census Key at `https://api.census.gov/data/key_signup.html`; you should then activate it.

```
ssc install getcensus
*replace XYZ w/ your key
global censuskey XYZ
*get population by county
getcensus B01003, year(2015) sample(5) geography(county) clear
```

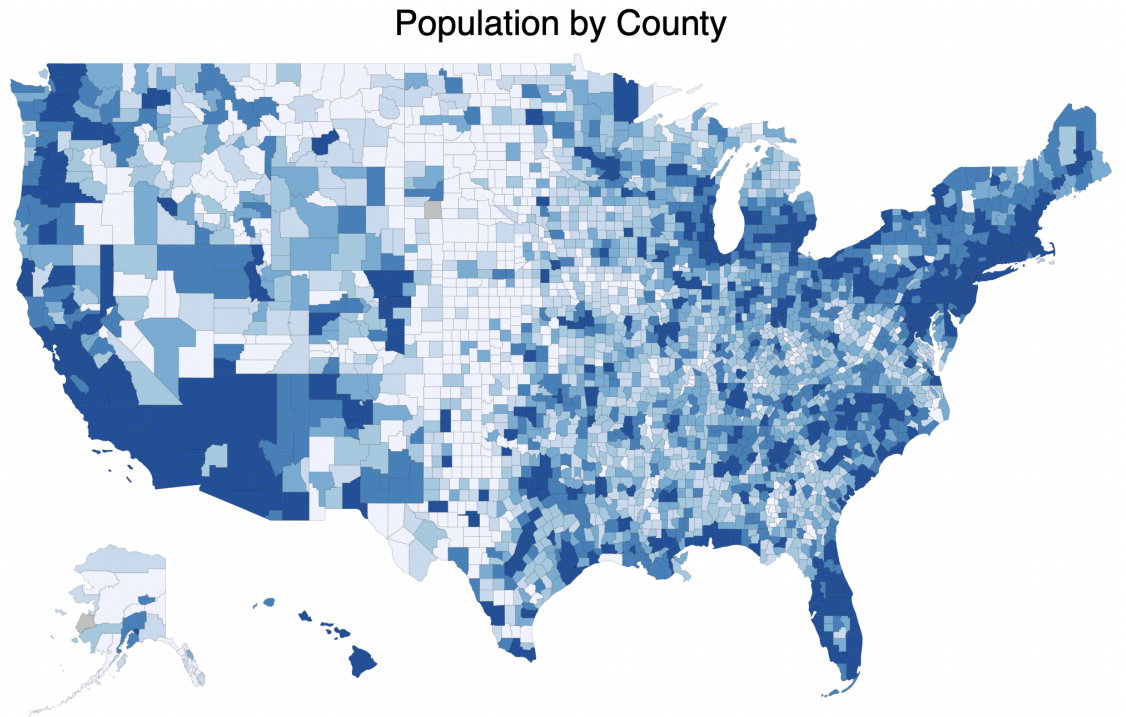
	year	state	county	geo_id	name	b01003_0…
1	2015	01	001	0500000US01001	Autauga County, Alabama	55221
2	2015	01	003	0500000US01003	Baldwin County, Alabama	195121
3	2015	01	005	0500000US01005	Barbour County, Alabama	26932
4	2015	01	007	0500000US01007	Bibb County, Alabama	22604
5	2015	01	009	0500000US01009	Blount County, Alabama	57710
6	2015	01	011	0500000US01011	Bullock County, Alabama	10678
7	2015	01	013	0500000US01013	Butler County, Alabama	20354
8	2015	01	015	0500000US01015	Calhoun County, Alabama	116648
9	2015	01	017	0500000US01017	Chambers County, Alabama	34079
10	2015	01	019	0500000US01019	Cherokee County, Alabama	26008
11	2015	01	021	0500000US01021	Chilton County, Alabama	43819
12	2015	01	023	0500000US01023	Choctaw County, Alabama	13395
13	2015	01	025	0500000US01025	Clarke County, Alabama	25070
14	2015	01	027	0500000US01027	Clay County, Alabama	13537

36 Graph at county level

Use *maptile* to create choropleth maps at the county level. Other geographies, such as state, are also supported.

In this example, we'll color counties according to population, which we pull from the ACS:

```
ssc install getcensus
*replace XYZ w/ your key
global censuskey XYZ
*get population by county
getcensus B01003, year(2015) sample(5) geography(county) clear

ssc inst maptile
ssc inst spmap
maptile_install using
   "http://files.michaelstepner.com/geo_county2014.zip"
drop county
gen county = substr(g,10,5)
destring county, replace
maptile b01003_001e, geo(county2014) twopt(title(Population by County)
   legend(off)) fcolor(Blues)
```



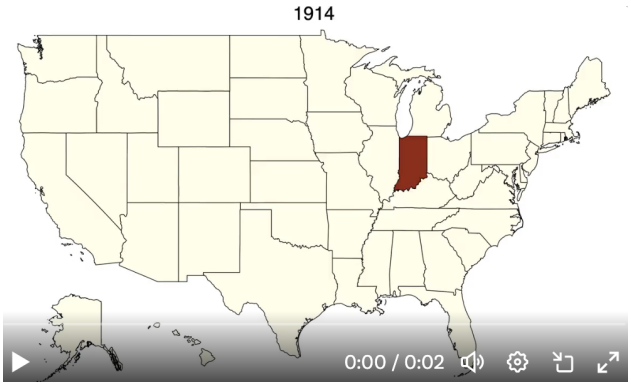
37 Animated maps

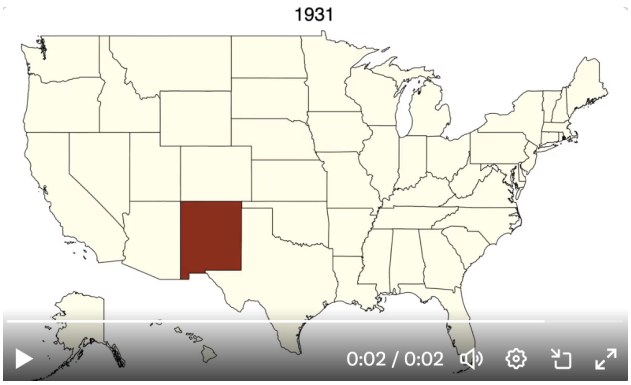
Maybe you want to show your variation over time and space. While it's much easier to make an animated map with R's *gganimate*, you can do it in Stata as in this toy example:

```
ssc install maptile
ssc install spmap
maptile_install using "http://files.michaelstepner.com/geo_state.zip"
sysuse census, clear
rename state q
rename state2 state
gen year = _n+1900
fillin state year

bys state: replace medage = 0 if medage==.
forvalues i = 1914/1928 {
  maptile medage if year==`i', geo(state) twopt(title(`i') legend(off))
  graph export `i'.png, replace
}
```

You then need to stitch them together. On a Mac, you can go to the Terminal, change the directory to where the files are stored, then:

```
convert *.png a.gif
```





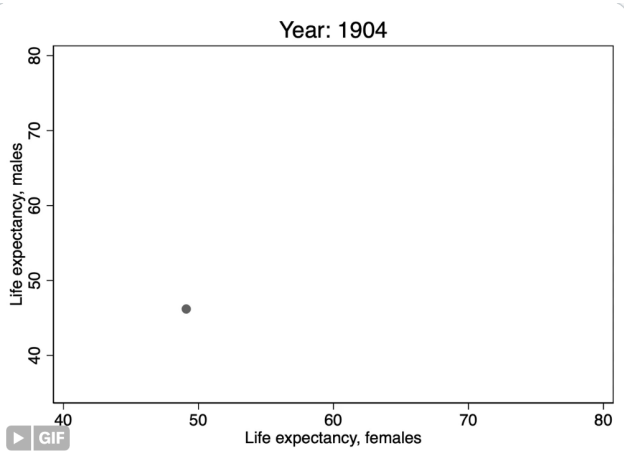
38 Animated graphs

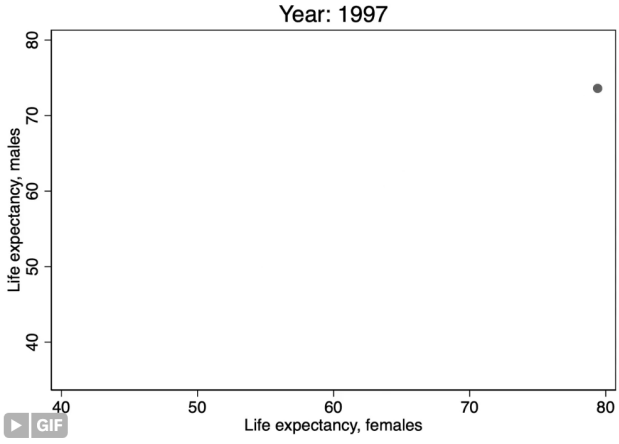
Using the same general approach as the tip directly above, you can create animated graphs:

```
sysuse uslifeexp, clear
forvalues i = 1900/1999 {
  scatter le_male le_female if year==`i', title(`i') scheme(s1mono)
     yscale(r(35 80)) xscale(r(40 80)) ylabel(40(10)80)
     xlabel(40(10)80)
  gr export `i'.png, replace
}
```

In Terminal (on a Mac), navigate to the directory, then:

```
convert *.png a.gif
```

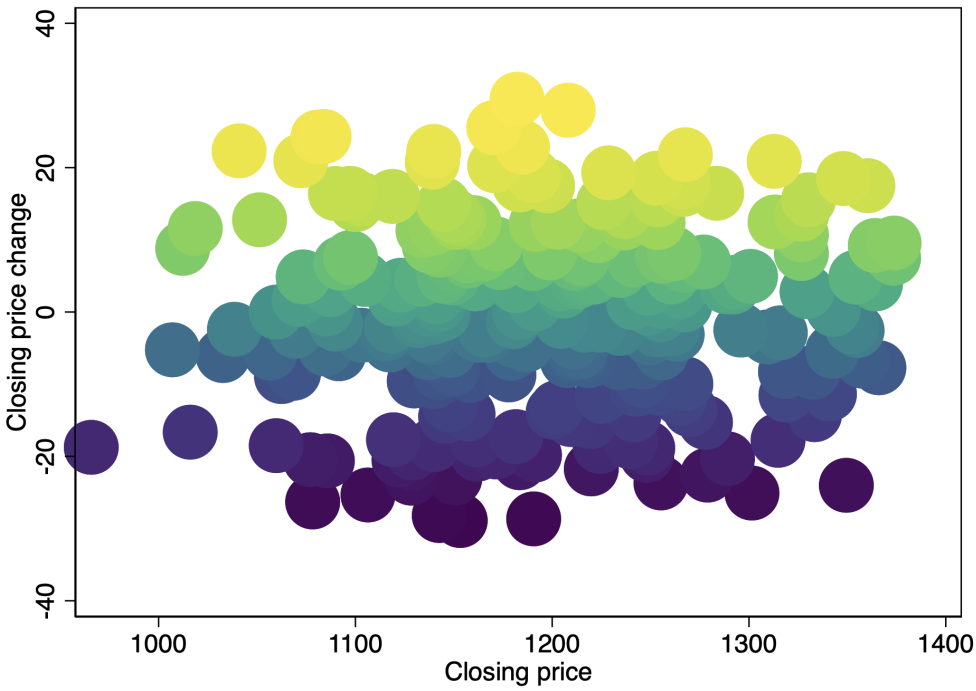




39 mscatter

Use *mscatter* to create scatter plots with color gradients.

```
capture ssc inst mscatter
capture ssc inst palettes
sysuse sp500, clear
mscatter change close if inrange(change, -30, 30), msymbol(O) msize(7)
   sch(s1mono) over(change) colorpalette(viridis)
```



40 duplicates

To keep one observation per group:

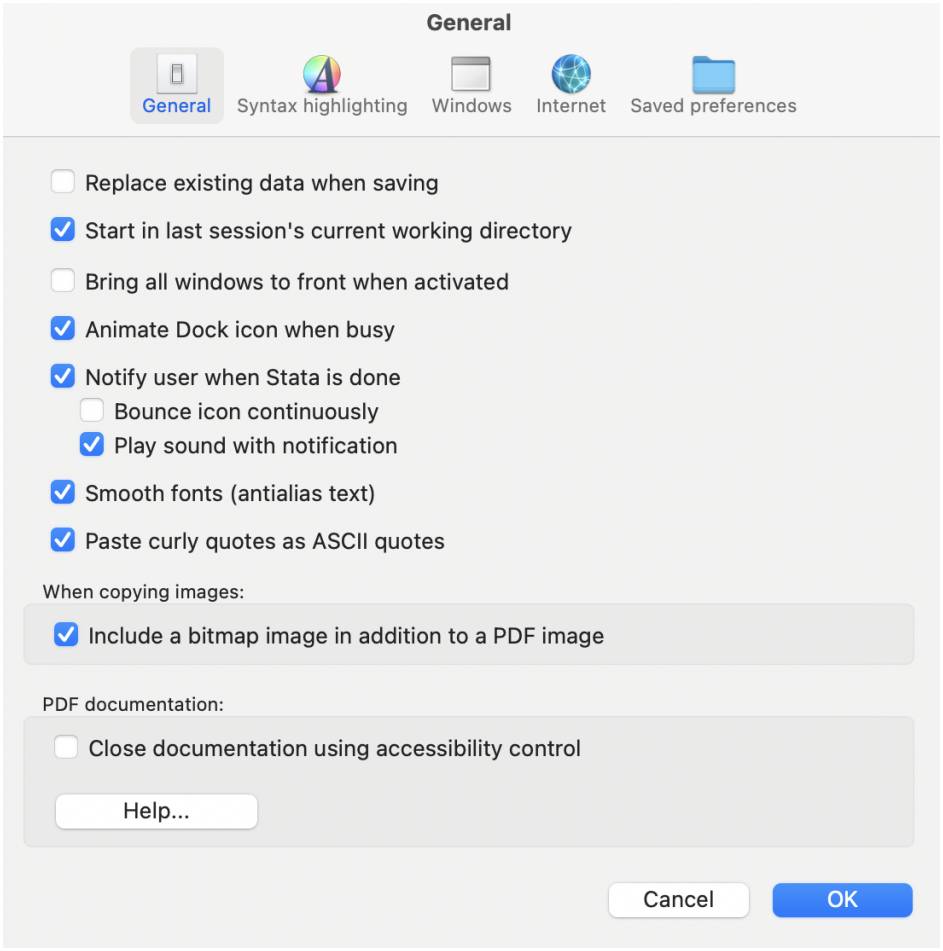
```
sysuse auto2, clear
keep if _n<15
bys turn: keep if _n==1
```

You can also use *duplicates*. To check if observations are unique within group, you can use *duplicates report*. To drop duplicates, you can use *duplicates drop*. The observations that are kept will not necessarily be the same as the bysort approach.

```
sysuse auto2, clear
keep if _n<15
*not unique:
duplicates r turn
*unique:
duplicates r gear_ratio turn
*drop dups
duplicates drop turn, force
```

41 Notify when code is finished running

To have Stata play a sound when it's done running your code, select the *Play sound with notification* option.



And you use the *beep* to make Stata beep. This could be useful to run when a loop is finished.

```
beep
```

You can also use *statapush* to have Stata message your phone when it's done running.

```
ssc install statapush
help statapush
```

42 Display loop progress

When you're doing a loop, you can show the progress using the following approach, which displays a message every 100th iteration:

```
local iterations = 1000
forvalues i=1/`iterations' {
  if mod(`i'/100, 1)==0 di "Iteration `i' of `iterations'"
}
```

```
Iteration 100 of 1000
Iteration 200 of 1000
Iteration 300 of 1000
Iteration 400 of 1000
Iteration 500 of 1000
Iteration 600 of 1000
Iteration 700 of 1000
Iteration 800 of 1000
Iteration 900 of 1000
Iteration 1000 of 1000
```

43 Timers

To time how long your code takes, you can use *etime*:

```
capture ssc install etime
etime, start
forvalues i=1/1000000 {
  quietly di "`i'"
}
etime
```

```
. forvalues i=1/1000000 {
  2.         quietly di "`i'"
  3. }

. etime
Elapsed time is 6 seconds
```

Or, you can use *timer*:

```
timer clear
timer on 1
forvalues i=1/1000000 {
  di "`i'"
}
timer off 1
timer list
```

```
. forvalues i=1/1000000 {
  2.         quietly di "`i'"
  3. }

. timer off 1

. timer list
   1:      6.37 /        1 =       6.3660
```

Or, you can use *rmsg*. (Turn it off with *set rmsg off*.)

```
set rmsg on
*to do this permanently
*set rmsg on, permanently
forvalues i=1/1000000 {
  quietly di "`i'"
}
```

```
. forvalues i=1/1000000 {
  2.         quietly di "`i'"
  3. }
r; t=6.55 22:06:26

. r; t=6.56 22:06:26
```

44 Loop over all variables

To loop over all variables, use *varlist _all*:

```
sysuse census, clear
  foreach var of varlist _all {
  rename `var' `var'_42
}
```

	state_42	state2_42	region_42	pop_42	poplt5_42	pop5_17_42	pop18p_42	pop65p_42	popurban_42	medage_
1	Alabama	AL	South	3,893,888	296,412	865,836	2,731,640	440,015	2,337,713	29
2	Alaska	AK	West	401,851	38,949	91,796	271,106	11,547	258,567	26
3	Arizona	AZ	West	2,718,215	213,883	577,604	1,926,728	307,362	2,278,728	29
4	Arkansas	AR	South	2,286,435	175,592	495,782	1,615,061	312,477	1,179,556	30
5	California	CA	West	23,667,902	1,708,400	4,680,558	17,278,944	2,414,250	21,607,606	29

45 Calculate total

If you want to calculate the total of a variable, use *egen total*. Don't use *gen sum* as this calculates the running/cumulative total.

```
sysuse auto2, clear
gen one = 1
egen total = total(one)
```

46 xtile

Use *xtile* to create a new variable that groups the values of another variable into quantile bins.

```
sysuse auto2, clear
keep turn
*quartiles
xtile turn_quartile = turn, nq(4)
*deciles
xtile turn_decile = turn, nq(10)
```

	turn	turn_qu…	turn_de…
1	40	2	5
2	40	2	5
3	35	1	2
4	40	2	5
5	43	3	7
6	43	3	7
7	34	1	1
8	42	3	6

47 Remove elements from a local

Here's how to create a local with all variables except one:

```
sysuse auto2, clear
ds
local vars `r(varlist)'
di "`vars'"
local remove_vars "make"
local vars_new: list vars - remove_vars
di "`vars_new'"
```

```
. di "`vars'"
make price mpg rep78 headroom trunk weight length turn displacement gear_ratio foreign

. local remove_vars "make"

. local vars_new: list vars - remove_vars

. di "`vars_new'"
price mpg rep78 headroom trunk weight length turn displacement gear_ratio foreign
```

48 Add elements to a macro

Here's how to add elements to a local:

```
local loc a b c
di "`loc'"
*add d & e
local loc `loc' d e
di "`loc'"
```

```
. local loc a b c

. di "`loc'"
a b c

. *add d & e
. local loc `loc' d e

. di "`loc'"
a b c d e
```

Here's how to add elements to a global:

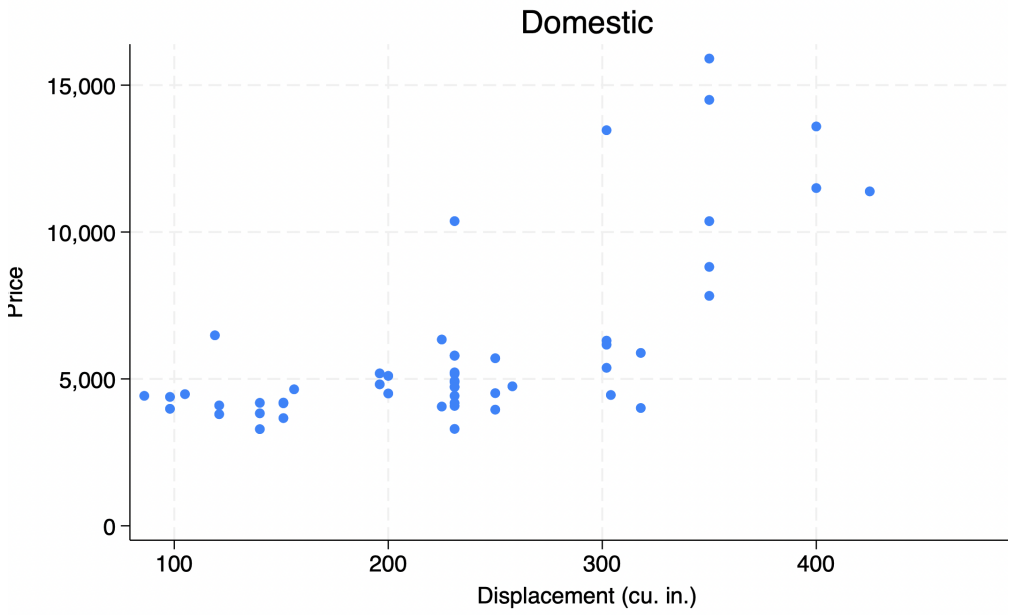
```
global glo v w x
di "$glo"
*add y & z
global glo $glo y z
di "$glo"
```

49 Save value label to macro

To save a value label to a local, use this format: *local loc: label (var) X*, where *X* is the value.

```
sysuse auto2, clear
fre foreign
local foreign_0: label (foreign) 0
local foreign_1: label (foreign) 1
di "`foreign_0'"
di "`foreign_1'"
scatter price displacement if foreign==0, title(`foreign_0')
```

```
. fre foreign

foreign — Car origin

                          Freq.    Percent      Valid        Cum.
-------------------------------------------------------------------
Valid    0 Domestic          52      70.27      70.27       70.27
         1 Foreign           22      29.73      29.73      100.00
         Total               74     100.00     100.00
-------------------------------------------------------------------

. local foreign_0: label (foreign) 0

. local foreign_1: label (foreign) 1

. di "`foreign_0'"
Domestic

. di "`foreign_1'"
Foreign
```



50 Access stored results and other parameters

You can use *return list, all* and *ereturn list, all* (and *sreturn list, all*) to show all stored results. *creturn list* shows other parameters.

```
sysuse auto2, clear
reg weight gear_ratio
ereturn list, all
return list, all
matrix list r(table)
creturn list
di "`c(pi)'"
```

51 Go between numeric and string with existing variable

destring converts from string to numeric, and *tostring* converts from numeric to string:

```
sysuse auto2, clear
tostring turn, replace
destring turn, replace
```

52 Go between numeric and string when creating variable

real() converts from string to numeric, and *string()* converts from numeric to string:

```
sysuse pop2000, clear
keep if _n>2
gen age = real(substr(agest, 1, 2)), after(agestr)
gen age_string = string(age), after(age)
```

	agestr	age	age_string
1	10 to 14	10	10
2	15 to 19	15	15
3	20 to 24	20	20
4	25 to 29	25	25
5	30 to 34	30	30
6	35 to 39	35	35

53 Version control

A rudimentary version of version control: at the beginning of your file, create a global for the file path of your output, which you can then change every day:

```
global output "/Users/me/Research/projectName/output/12-25-2023/"
...
graph export "${output}/a.png"
```

54 asgen

Use *asgen* to create a weighted average:

```
capture ssc install asgen
sysuse census, clear
bys region: asgen weighted_medage = medage, weight(pop)
```

55 collapse

Use *collapse* to perform operations within groups to create new variables. Here, we will compute the mean of *medage* within region, as well as the number of observations within region.

```
sysuse census, clear
gen N = 1
collapse (mean) medage (sum) N, by(region)
```

	region	medage	N
1	NE	31.23	9
2	N Cntrl	29.52	12
3	South	29.62	16
4	West	28.28	13

We can also weight. This gives the same numbers as the *asgen* example above.

```
sysuse census, clear
collapse (mean) medage [aweight=pop], by(region)
```

56 seq

Use *seq* to repeat sequences of numbers:

```
ssc install seq
sysuse auto2, clear
*repeat 1 2 3
seq rep1, from(1) to(3)
*repeat 1 1 2 2 3 3
seq rep2, from(1) to(3) block(2)
```

foreign	rep1	rep2
omestic	1	1
omestic	2	1
omestic	3	2
omestic	1	2
omestic	2	3
omestic	3	3
omestic	1	1
omestic	2	1
omestic	3	2
omestic	1	2
omestic	2	3

57 Add leading zero to number

here's how to add a leading zero to a number, which is useful when working with state FIPS codes. Here, we will add a zero to a one-digit numbers:

```
clear all
set obs 15
gen fips = _n
tostring fips, gen(fips2)
replace fips2 = "0" + fips2 if strlen(fips2)==1
```

	fips	fips2		
1	1	01		
2	2	02		
3	3	03		
4	4	04		
5	5	05		
6	6	06		

58 Main effects and interactions in regressions

Here's how to think about main effects and interactions in regressions.

By default, main effects are continuous. The i. prefix makes them categorical.

By default, variables in interaction terms are categorical. The c. prefix makes them continuous.

Assume the interaction term has two variables. To include only the interaction term, use # between the variables.

To include both the main effects and the interaction term, use ## between the variables.

```
sysuse auto2, clear
gen b = _n<10
gen b_weight = b*weight

*equivalent: (note the i. is redundant b/c b is binary):
reg price i.b weight b_weight
reg price i.b weight b#c.weight
reg price b##c.weight
```

59 keepusing

When merging using data to the main data, use the ”keepusing” option to keep only select variables from the using data:

```
webuse nhanes2, clear
tempfile nh
save 'nh'

use sampl houssiz using 'nh', clear
merge 1:1 sampl using 'nh', keepusing(height weight)
```

	sampl	houssiz	height	weight	_merge
1	1400	4	174.598	62.48	Matched (3)
2	1401	6	152.297	48.76	Matched (3)
3	1402	6	164.098	67.25	Matched (3)
4	1404	9	162.598	94.46	Matched (3)
5	1405	3	163.098	74.28	Matched (3)
6	1406	1	147.098	66	Matched (3)
7	1407	2	153.898	54.55	Matched (3)
8	1408	1	160	58.97	Matched (3)
9	1410	3	164	68.95	Matched (3)

60 geodist

geodist calculates the "as the crow flies" distance between points:

```
ssc install geodist
clear
input double lat lon
34.043026 -118.26694
39.74915 -105.00740
end
geodist 42.366570 -71.06186 lat lon, gen(dist) miles
```

	lat	lon	dist
1	34.043026	-118.2669	2592.1708
2	39.74915	-105.0074	1765.9633

61 geonear

Use *geonear* to find the nearest point in B to each point in A:

```
ssc inst geonear
clear
set ob 20
set se 1
g n2=_n
g la2=39+5*rt(5)
g lo2=-99+9*rt(5)
tempfile a
save `a'
ren n2 n
g la=la2+3*rt(3)
g lo=lo2+4*rt(4)
drop *2
geonear n la lo using `a', n(n2 la2 lo2)
```

```
Unique base locations    = 20            Unique neighbor locations = 20
Bases * Neighbors        = 400           Number of regions         = 1
Computed distances       = 400           Total run time (seconds)  = .053
```

Data Editor (Browse) — S_00721.00000d

n[1] 1

	n	la	lo	nid	km_to_nid
1	1	28.47509	-106.5063	1	212.58241
2	2	47.46821	-95.7141	10	152.70921
3	3	32.293	-93.91636	7	173.84768
4	4	37.47478	-154.9184	4	398.7779

62 georoute

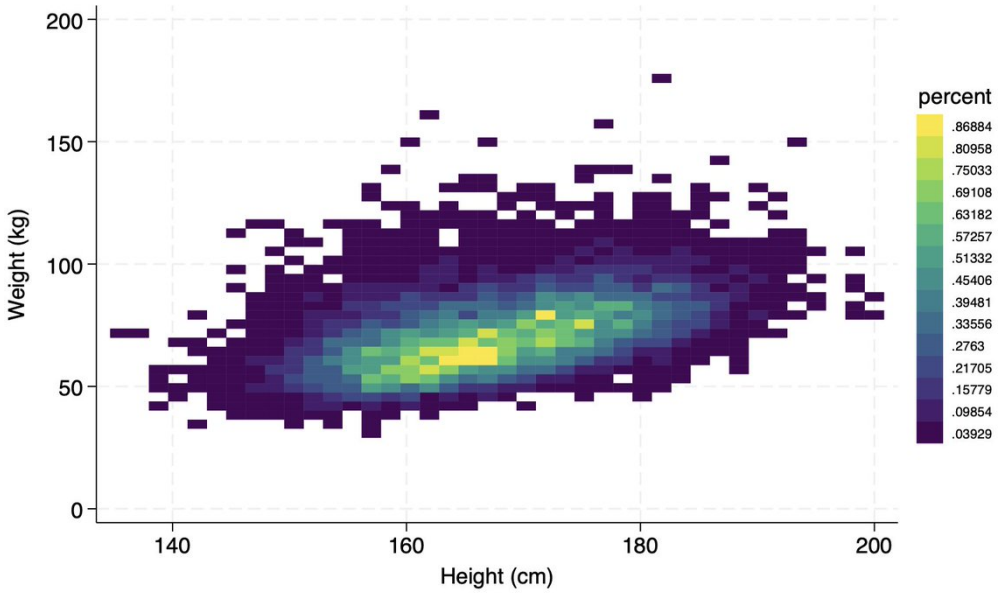
Use *georoute* to calculate travel times between addresses or coordinates. You can choose car, bike, walk, and public transit.

Note you have to register an account to use the here .com API, and the free version limits the number of requests.

63 heatplot

use *heatplot* to make a heatmap:

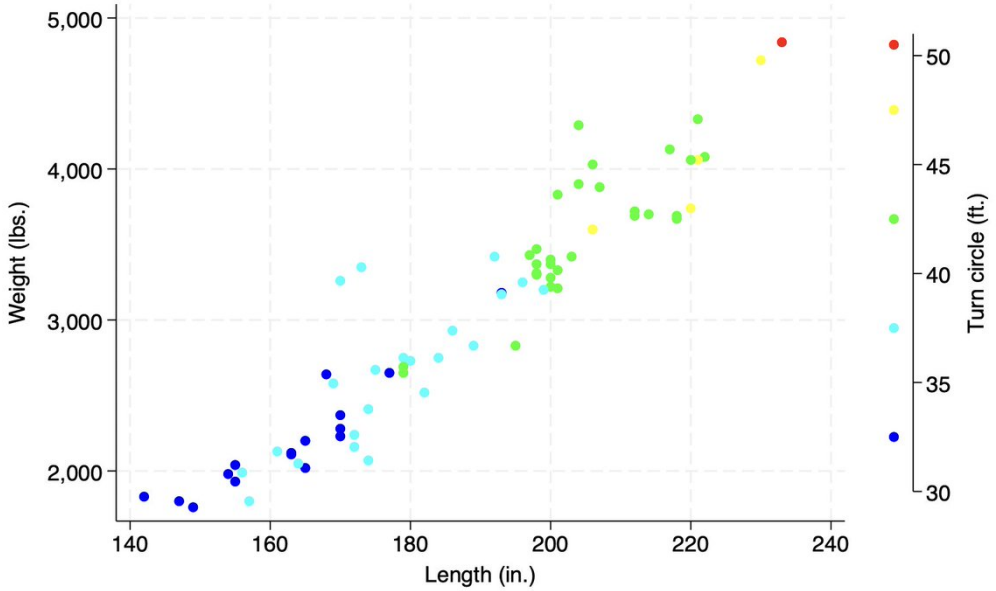
```
capture ssc install heatplot
webuse nhanes2, clear
heatplot weight height
```



64 Color by a third variable

Use *colorvar* to color a scatterplot by a third variable:

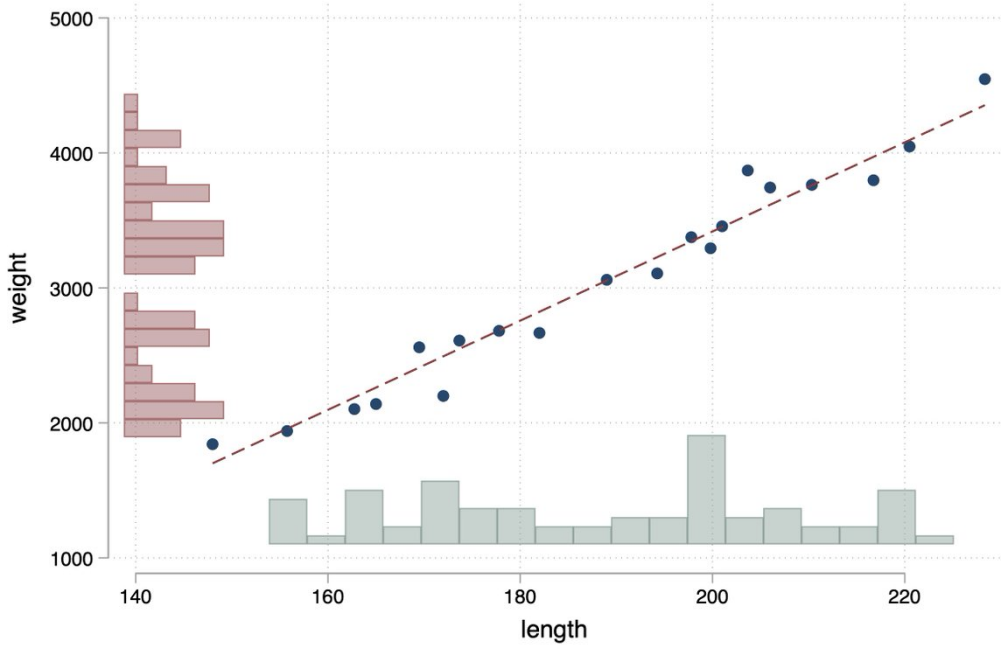
```
sysuse auto2, clear
scatter weight length, colorvar(turn)
```



65 binscatterhist

Use *binscatterhist* to show a binned scatterplot along with histograms of the variables.

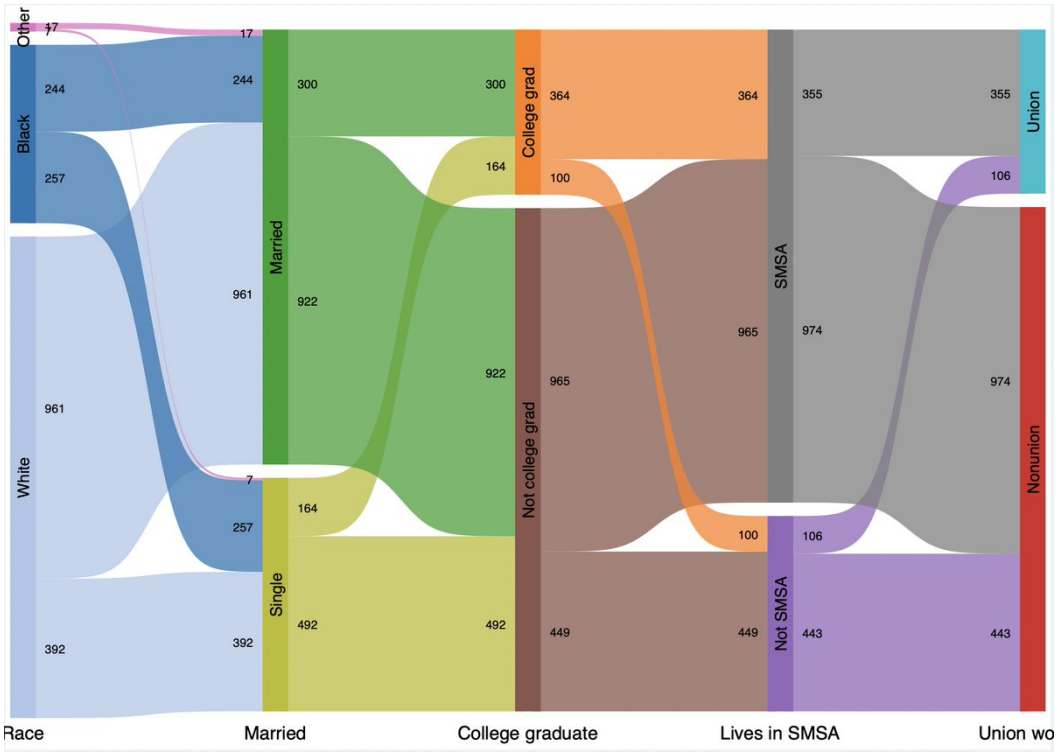
```
capture ssc install binscatterhist
sysuse auto2, clear
binscatterhist weight length, hist(weight length) ymin(1100)
   xhistbarheight(30) yhistbarheight(13)
```



66 alluvial

Use *alluvial* to make alluvial plots in Stata:

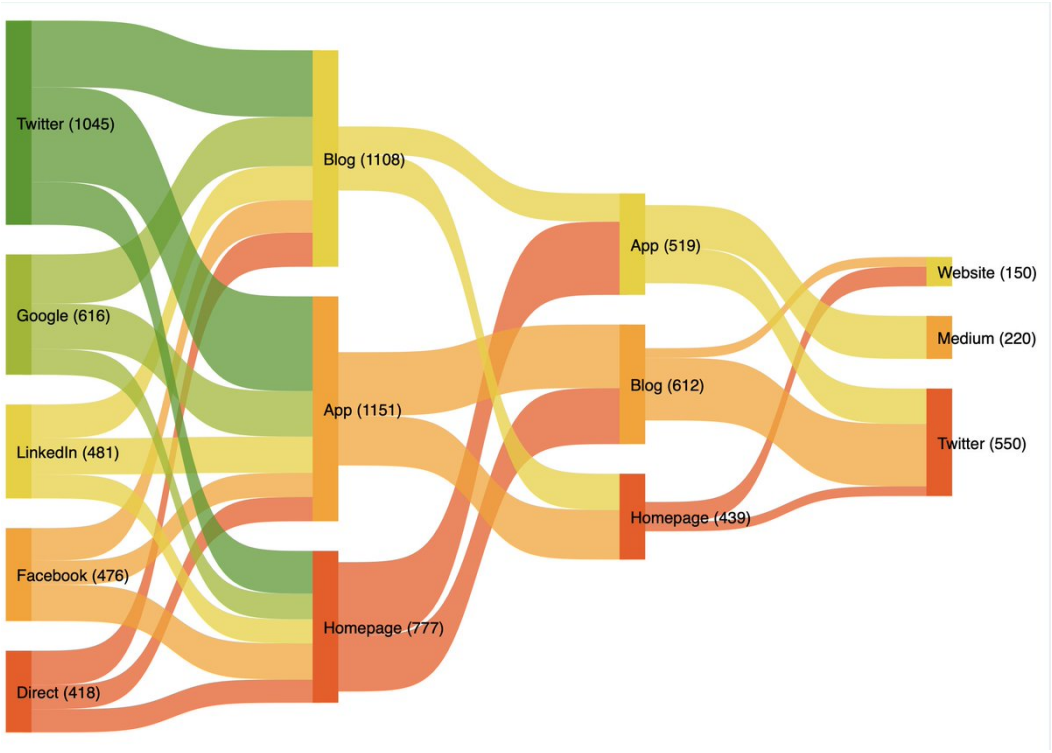
```
ssc install alluvial
*example from package tutorial
sysuse nlsw88.dta, clear
alluvial race married collgrad smsa union
```



67 sankey

Use *sankey* to make Sankey plots:

```
ssc install sankey
*example from package tutorial
use
   "https://github.com/asjadnaqvi/stata-sankey/blob/main/data/sankey2.dta?raw=true'
   clear
sankey value, from(source) to(destination) by(layer) noval showtot
   palette(CET C6) laba(0) labpos(3) labg(-1) offset(10)
```



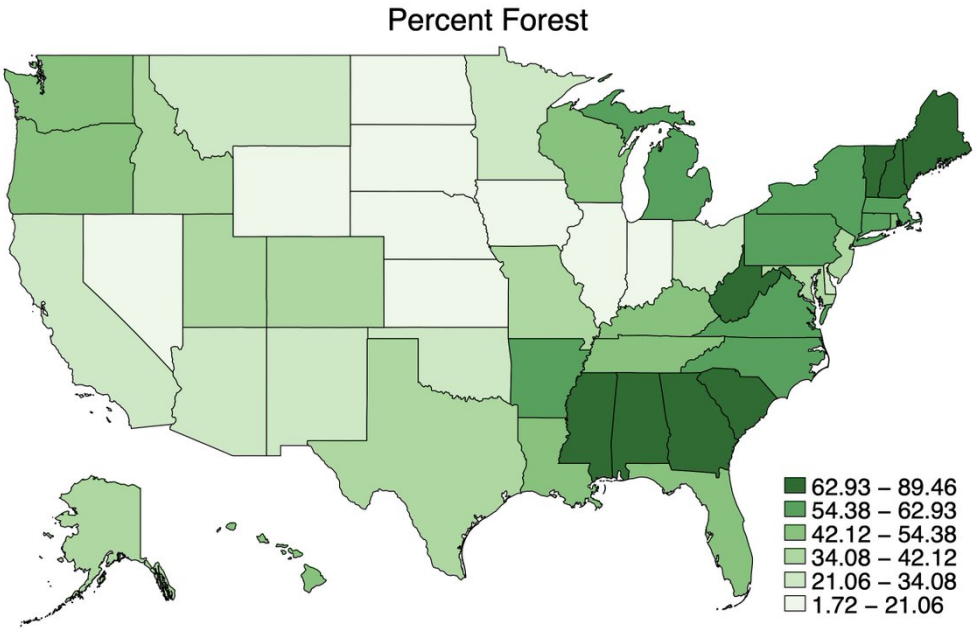
68 Scrape webpages with readhtml

Use *readhtml* to scrape (parts of certain) web pages. Below is the percent of a state that's covered by forest.

```
capture net install readhtml, from(https://ssc.wisc.edu/sscc/stata/)
capture ssc install statastates
capture ssc install maptile
capture ssc install spmap
capture maptile_install using
   "http://files.michaelstepner.com/geo_state.zip"
```

```
readhtmltable https://en.wikipedia.org/wiki/Forest_cover_by_state_and_territory_in_the_United_States,
    varnames
```

```
gen st=substr(S,7,30)
drop if inlist(_n,3,4,16,18,24,40,57)
gen forest=substr(Percent_forest_2,1,length(Percent_forest_2)-2)
destring forest, replace
keep st forest
statastates, name(st)
rename state_abbrev state
maptile forest, geo(state) fcolor(Greens) twopt(title("Percent
   Forest"))
```



69 Choose which variables and observations to load

When loading data with *use*, you can load only select vars with a varlist, use *in* to select observations by observation number, and/or use *if* to select observations with logic:

```
sysuse auto2, clear
tempfile a
save `a'
use make turn using `a' in 2/9
use `a' if length>200
```

70 Use datasets from internet

You can load toy datasets from the internet. The links from *help dta_manuals* give you a lot to choose from.

```
help dta_manuals

help q_base

use https://www.stata-press.com/data/r18/apple.dta
```

Stata 18 manual datasets

Clicking on a Stata 18 manual listed below provides a list of datasets used within that manual. The links to the datasets require web access.

For a list of datasets installed with Stata, see dta_examples.

Base Reference Manual [R]

Bayesian Analysis Reference Manual [BAYES]

Bayesian Model Averaging Reference Manual [BMA]

Causal Inference and Treatment-Effects Estimation Reference Manual [CAUSAL]

Choice Models Reference Manual [CM]

Customizable Tables and Collected Results Reference Manual [TABLES]

Data Management Reference Manual [D]

Dynamic Stochastic General Equilibrium Models Reference Manual [DSGE]

Extended Regression Models Reference Manual [ERM]

Finite Mixture Models Reference Manual [FMM]

Graphics Reference Manual [G]

Item Response Theory Reference Manual [IRT]

Lasso Reference Manual [LASSO]

Longitudinal-Data/Panel-Data Reference Manual [XT]

Meta-Analysis Reference Manual [META]

Multilevel Mixed-Effects Reference Manual [ME]

Multiple-Imputation Reference Manual [MI]

Multivariate Statistics Reference Manual [MV]

Power, Precision, and Sample-Size Reference Manual [PSS]

Programming Reference Manual [P]

Reporting Reference Manual [RPT]

Spatial Autoregressive Models Reference Manual [SP]

Structural Equation Modeling Reference Manual [SEM]

Survey Data Reference Manual [SVY]

Survival Analysis Reference Manual [ST]

Time-Series Reference Manual [TS]

User's Guide [U]

71 Choose which category to omit in a regression

To choose which category to omit in a regression, use *bX*, where *X* is the value you wish to omit:

```
sysuse auto2, clear
*view values of rep78
fre rep78
*default:
reg mpg i.rep78
*omit category 3 (average)
reg mpg ib3.rep78
```

```
. reg mpg ib3.rep78
```

Source	SS	df	MS	Number of obs	=	69
				F(4, 64)	=	4.91
Model	549.415777	4	137.353944	Prob > F	=	0.0016
Residual	1790.78712	64	27.9810488	R-squared	=	0.2348
				Adj R-squared	=	0.1869
Total	2340.2029	68	34.4147485	Root MSE	=	5.2897

mpg	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
rep78						
Poor	1.566667	3.863059	0.41	0.686	-6.150681	9.284014
Fair	-.3083333	2.104836	-0.15	0.884	-4.513226	3.896559
Good	2.233333	1.577087	1.42	0.162	-.9172607	5.383927
Excellent	7.930303	1.86452	4.25	0.000	4.205497	11.65511
_cons	19.43333	.9657648	20.12	0.000	17.504	21.36267

72 fillin

Use *fillin* to fill in dataset so that all combinations of the variables are present:

```
clear all
input year str1 state value
1 "A" 6
2 "A" 3
4 "A" 5
3 "B" 1
end
fillin year state
```

	year	state	value	_fillin
1	1	A	6	0
2	1	B	.	1
3	2	A	3	0
4	2	B	.	1
5	3	A	.	1
6	3	B	1	0
7	4	A	5	0
8	4	B	.	1

73 levelsof

To get all values of a variable, use *levelsof* (and the *local()* option to put them into a local)

```
sysuse auto2, clear
levelsof mpg
levelsof trunk
*put it into a local
levelsof trunk, local(trunklevels)
di "`trunklevels'"
```

```
. levelsof mpg
12 14 15 16 17 18 19 20 21 22 23 24 25 26 28 29 30 31 34 35 41

. levelsof trunk
5 6 7 8 9 10 11 12 13 14 15 16 17 18 20 21 22 23

. *put it into a local
. levelsof trunk, local(trunklevels)
5 6 7 8 9 10 11 12 13 14 15 16 17 18 20 21 22 23

. di "`trunklevels'"
5 6 7 8 9 10 11 12 13 14 15 16 17 18 20 21 22 23
```

You can do this and then loop over the values:

```
sysuse auto2, clear
levelsof rep78, missing local(rep78_levels)
foreach i of local rep78_levels {
  sum price
}
```

For strings:

```
levelsof make
levelsof make, clean
```

74 Increment local

Use ++ to increment a local:

```
local b = 1
local b = `b' + 1
di "`b'"
*Equivalent:
local a = 1
local ++a
di "`a'"
```

```
. local b = 1

. local b = `b' + 1

. di "`b'"
2


.
. local a = 1

. local ++a

. di "`a'"
2
```

75 labelbook

To show all value labels, use *labelbook*:

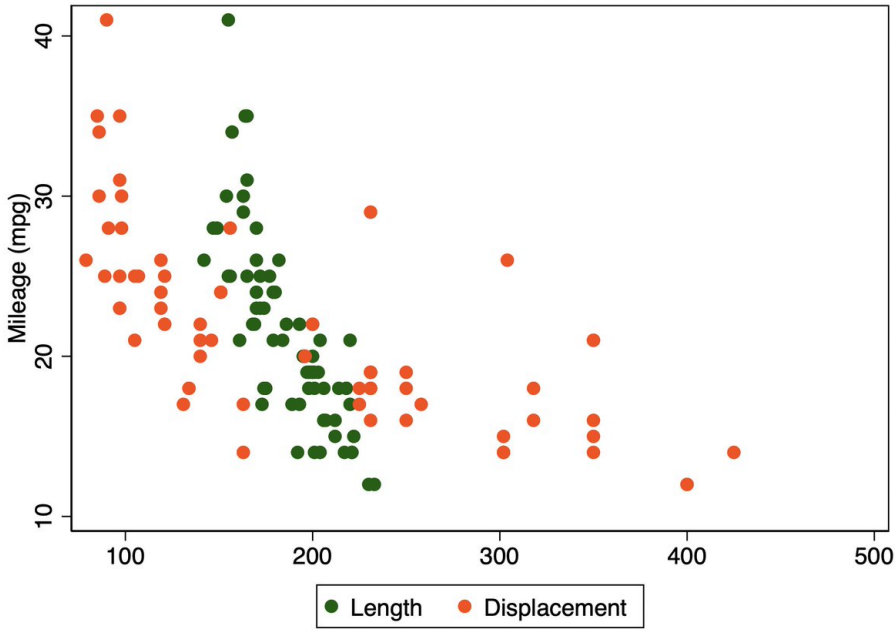
```
sysuse auto2, clear
labelbook
```

```
. labelbook

-------------------------------------------------------------------------------
Value label origin
-------------------------------------------------------------------------------

     Values                                  Labels
      Range:  [0,1]                  String length:  [7,8]
          N:  2                    Unique at full length:  yes
       Gaps:  no                     Unique at length 12:  yes
 Missing .*:  0                              Null string:  no
                                Leading/trailing blanks:  no
                                     Numeric -> numeric:  no
 Definition
          0   Domestic
          1   Foreign

  Variables:  foreign


-------------------------------------------------------------------------------
Value label repair
-------------------------------------------------------------------------------

     Values                                  Labels
      Range:  [1,5]                  String length:  [4,9]
          N:  5                    Unique at full length:  yes
       Gaps:  no                     Unique at length 12:  yes
 Missing .*:  0                              Null string:  no
                                Leading/trailing blanks:  no
                                     Numeric -> numeric:  no
 Definition
          1   Poor
          2   Fair
          3   Average
          4   Good
          5   Excellent

  Variables:  rep78
```

76 twoway

To graph multiple things at once, use *twoway*:

```
sysuse auto2, clear
twoway scatter mpg length || ///
     scatter mpg displacement, ///
     scheme(s1color) legend(label(1 "Length") label(2 "Displacement"))
```



77 rowtotal

Use *rowtotal* to sum across columns. Note that rowtotal treats missings as 0s.

```
sysuse auto2, clear
gen sum1 = price + mpg + rep78
egen sum2 = rowtotal(price mpg rep78)
egen sum3 = rowtotal(price-rep78)
```

price	mpg	rep78	sum1	sum2	sum3
4,099	22	Average	4124	4124	4124
4,749	17	Average	4769	4769	4769
3,799	22	.	.	3821	3821
4,816	20	Average	4839	4839	4839
7,827	15	Good	7846	7846	7846
5,788	18	Average	5809	5809	5809
4,453	26	.	.	4479	4479
5,189	20	Average	5212	5212	5212
10,372	16	Average	10391	10391	10391
4,082	19	Average	4104	4104	4104
11,385	14	Average	11402	11402	11402

78 Random variables

To create a uniform random variable, use *runiform()*.

```
sysuse auto, clear
gen r=runiform()
```

displace…	gear_ratio	foreign	r
121	3.58	Domestic	.8162583
258	2.53	Domestic	.787128
121	3.08	Domestic	.6520915
196	2.93	Domestic	.7938479
350	2.41	Domestic	.4025587
231	2.73	Domestic	.9366145
304	2.87	Domestic	.3477634
196	2.93	Domestic	.2994616
231	2.93	Domestic	.7381327
231	3.08	Domestic	.5938806
425	2.28	Domestic	.3526423
350	2.19	Domestic	.1386851

There are other functions, such as *rexponential*. To see them, type in *help runiform* and then click on the *View complete PDF manual entry* link.

79 set seed

To ensure reproducibility, use *set seed*. This makes it so that the generate random numbers will be the same each time you run it.

```
sysuse auto, clear
set seed 42
gen r=runiform()

*this will produce exactly the same numbers:
sysuse auto, clear
set seed 42
gen r=runiform()
```

	displacement	gear_ratio	foreign	r
	121	3.58	Domestic	.7551555
	258	2.53	Domestic	.6390314
	121	3.08	Domestic	.7521452
	196	2.93	Domestic	.1362727
	350	2.41	Domestic	.903269
	231	2.73	Domestic	.0940683
	304	2.87	Domestic	.5745703
	196	2.93	Domestic	.3728877
	231	2.93	Domestic	.2738741
	231	3.08	Domestic	.3902709

80 Length of string

To count the number of words in a string, use *word count*. To count the number of characters in a string, use *strlen*.

```
local loc "How long is this?"
local loc_words : word count `loc'
di "`loc_words'"

local loc_chars = strlen("`loc'")
di "`loc_chars'"
```

```
. local loc "How long is this?"

. local loc_words : word count `loc'

. di "`loc_words'"
4


.
. local loc_chars = strlen("`loc'")

. di "`loc_chars'"
17
```

81 clonevar

To make an exact copy of another variable, use ”clonevar”:

```
sysuse auto2, clear
*this does not create an exact copy:
gen foreign2 = foreign
*this does:
clonevar foreign3 = foreign
```

foreign	foreign2	foreign3
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic
Domestic	0	Domestic

82 Label values based on value labels of another variable

To label the values of var 1 using the value labels of var 2, use ”describe” on var 2 to find the name of the value label, then apply to var 1.

```
sysuse auto2, clear
set seed 1
gen new = round(runiform()*4+1)
describe rep78
label values new repair
ssc install fre
fre new
```

```
. fre new

new
```

		Freq.	Percent	Valid	Cum.
Valid	1 Poor	14	18.92	18.92	18.92
	2 Fair	19	25.68	25.68	44.59
	3 Average	20	27.03	27.03	71.62
	4 Good	13	17.57	17.57	89.19
	5 Excellent	8	10.81	10.81	100.00
	Total	74	100.00	100.00	

83 Locate .do files

Find the .do files in a given directory on your computer that contain a particular word or phrase:

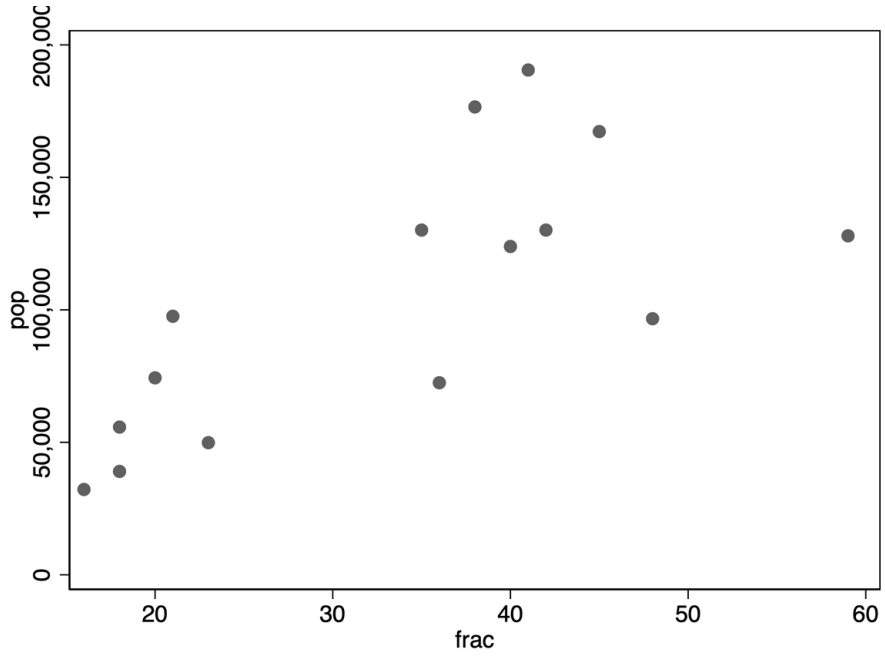
```
ssc install find
ssc install rcd
rcd "/Users/Todd/Google Drive": find *.do, match(sysuse auto2) show
```

```
. rcd "/Users/Todd/Google Drive": find *.do , match(sysuse auto2) show
1 matches found in build_v3.do
line 323: sysuse auto2, clear

3 matches found in test.do
line 22: sysuse auto2, clear
line 74: sysuse auto2, clear
line 117: sysuse auto2, clear

1 matches found in interactions binary continuous.do
line 3: sysuse auto2, clear

14 matches found in stata_book_code.do
line 3: sysuse auto2, clear
line 6: sysuse auto2, clear
line 9: sysuse auto2, clear
line 18: sysuse auto2, clear
```

84 Include commas in large numbers

To make it so that large numbers are displayed with commas, use *format* with a *c*. This can be useful in graph labeling.

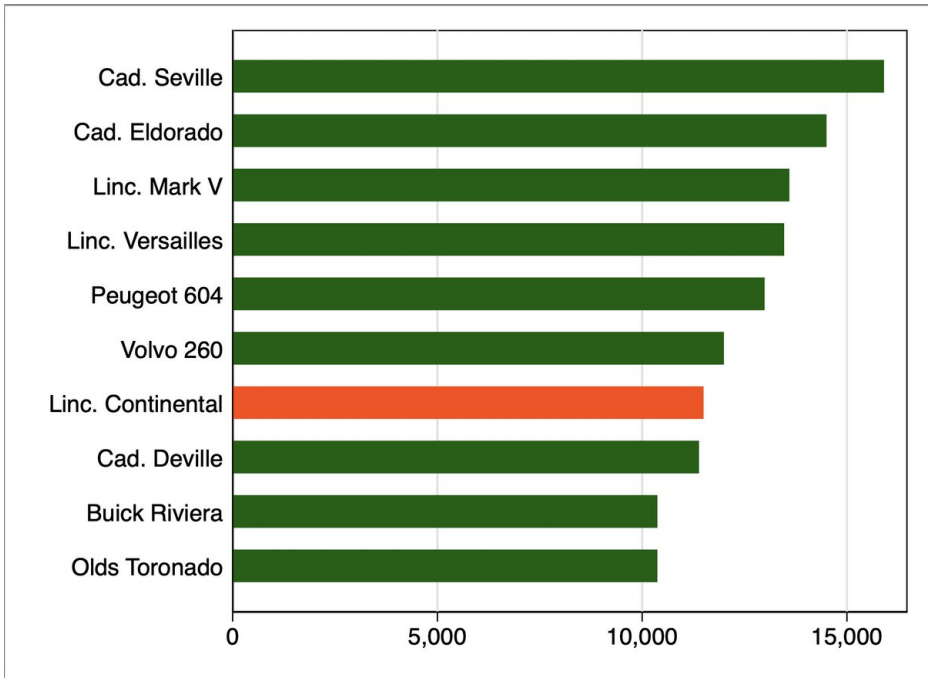
```
sysuse voter, clear
format pop %15.0fc
scatter pop frac, scheme(s1mono)
```



85 Highlight selected bars in bar chart

To highlight selected bars in a bar chart, use ”separate”

```
sysuse auto2, clear
keep if price>=10000

separate price, by(make=="Linc. Continental")

graph hbar (asis) price0 price1, nofill over(make, sort(price) desc)
   legend(off) scheme(s1color)
```



86 keeporder

Use *keeporder* to keep and order variables in one line:

```
capture ssc install keeporder

*old way
sysuse auto2, clear
keep foreign rep78 make
order foreign rep78 make

*new way
sysuse auto2, clear
keeporder foreign rep78 make
```

	foreign	rep78	make
1	Domestic	Average	AMC Concord
2	Domestic	Average	AMC Pacer
3	Domestic	.	AMC Spirit
4	Domestic	Average	Buick Century
5	Domestic	Good	Buick Electra
6	Domestic	Average	Buick LeSabre
7	Domestic	.	Buick Opel
8	Domestic	Average	Buick Regal

87 Create your own function/program

In this example, we'll create our own function (called a "program") that creates a new variable that is the sum of two other variables.

```
capture program drop s_pr
program s_pr, rclass
args x y
*access args as locals
gen sum = `x' + `y', after(`y')
end
```

Run the program. We will enter the arguments *length* (for *x*) and *turn* for *y*).

```
sysuse auto2, clear
s_pr length turn
```

length	turn	sum
186	40	226
173	40	213
168	35	203
196	40	236
222	43	265
218	43	261
170	34	204
200	42	242

88 gsort

Use *gsort* to sort in descending order:

```
sysuse auto2, clear
*sort ascending
gsort mpg
*sort descending
gsort -mpg
```

89 Sort descending when using bysort

You can't directly sort in descending order when using "bysort". Here's a workaround:

```
sysuse auto2, clear
*doesn't work:
bys foreign (-turn): gen n=_n
*instead:
gsort foreign -turn
by foreign: gen n = _n
```

If the sorting variable is non-string, you can do:

```
sysuse auto2, clear
gen turn_rev = -turn
bys foreign (turn_rev): gen n=_n
drop turn_rev
```

Note, though, that exact ties might be handled differently depending on what you do.

90 moreobs

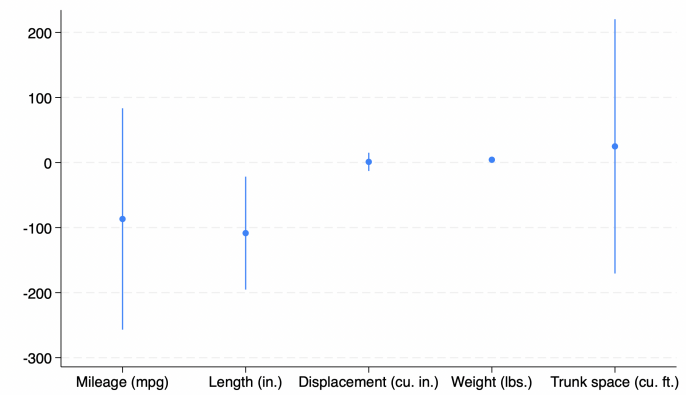
Use *moreobs* to add additional observations to your data:

```
ssc install moreobs
sysuse auto2, clear
moreobs 10
sort make
```

91 coefplot

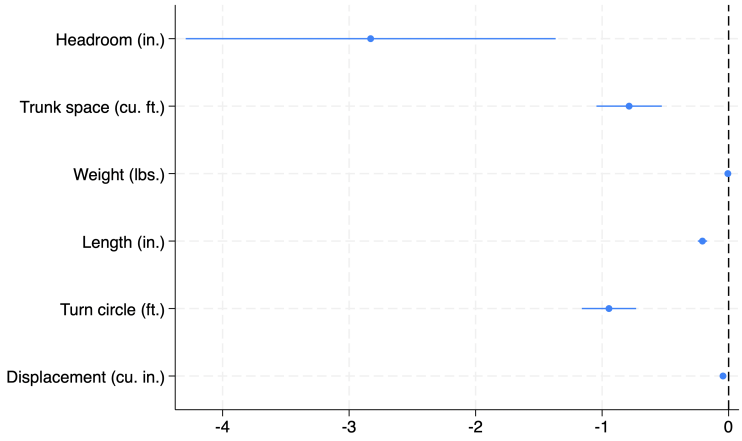
Use *coefplot* to quickly plot coefficients and confidence intervals.

```
sysuse auto2, clear
reg price mpg length displacement weight trunk
coefplot, drop(_cons) vertical
```



To plot coefficients from different models:

```
sysuse auto2, clear
est clear
local i = 1
foreach var in headroom trunk weight length turn displacement {
  reg mpg `var'
  est store est`i'
  local i = `i'+1
}
coefplot (est1\est2\est3\est4\est5\est6), drop(_cons) xline(0)
```



92 expand

To make *n* copies of each observation, use *expand(n)*.

```
sysuse auto2, clear
expand 2
sort make
```

	make	price	mpg	rep78	headroom	trunk	weight	length	turn	displace…	gear_ratio	foreign
1	AMC Concord	4,099	22	Average	2.5	11	2,930	186	40	121	3.58	Domestic
2	AMC Concord	4,099	22	Average	2.5	11	2,930	186	40	121	3.58	Domestic
3	AMC Pacer	4,749	17	Average	3.0	11	3,350	173	40	258	2.53	Domestic
4	AMC Pacer	4,749	17	Average	3.0	11	3,350	173	40	258	2.53	Domestic
5	AMC Spirit	3,799	22	.	3.0	12	2,640	168	35	121	3.08	Domestic
6	AMC Spirit	3,799	22	.	3.0	12	2,640	168	35	121	3.08	Domestic
7	Audi 5000	9,690	17	Excellent	3.0	15	2,830	189	37	131	3.20	Foreign
8	Audi 5000	9,690	17	Excellent	3.0	15	2,830	189	37	131	3.20	Foreign

93 nvals

To create a variable with the number of unique values of another variable, use *nvals*:

```
sysuse auto2, clear
egen headroom_unique_values = nvals(headroom)
sum headroom_unique_values
```

94 regsave

To save the coefficients and standard errors from your regression, use *regsave*:

```
capture ssc install regsave
tempfile coefficients
sysuse auto2, clear
reg price mpg headroom turn length gear_ratio
regsave using 'coefficients', replace

use 'coefficients', clear
```

	var	coef	stderr	N	r2
1	mpg	-185.4687	88.38206	74	.2700311
2	headroom	-556.5785	422.6933	74	.2700311
3	turn	-214.5423	141.7701	74	.2700311
4	length	63.33879	33.42999	74	.2700311
5	gear_ratio	-214.952	961.9933	74	.2700311
6	_cons	9032.163	7875.368	74	.2700311

95 Access certain rows of a variable

If you are creating a new variable and want to assign it the value of another observation *x* values apart use *[_n+x]* or *[_n-x]*.

```
sysuse auto2, clear
*lagged one observation
gen lag_length = length[_n-1], after(length)
*lead one observation
gen lead_length = length[_n+1], after(lag_length)
```

length	lag_length	lead_length
186	.	173
173	186	168
168	173	196
196	168	222
222	196	218
218	222	170
170	218	200

96 Refer to observations by row number

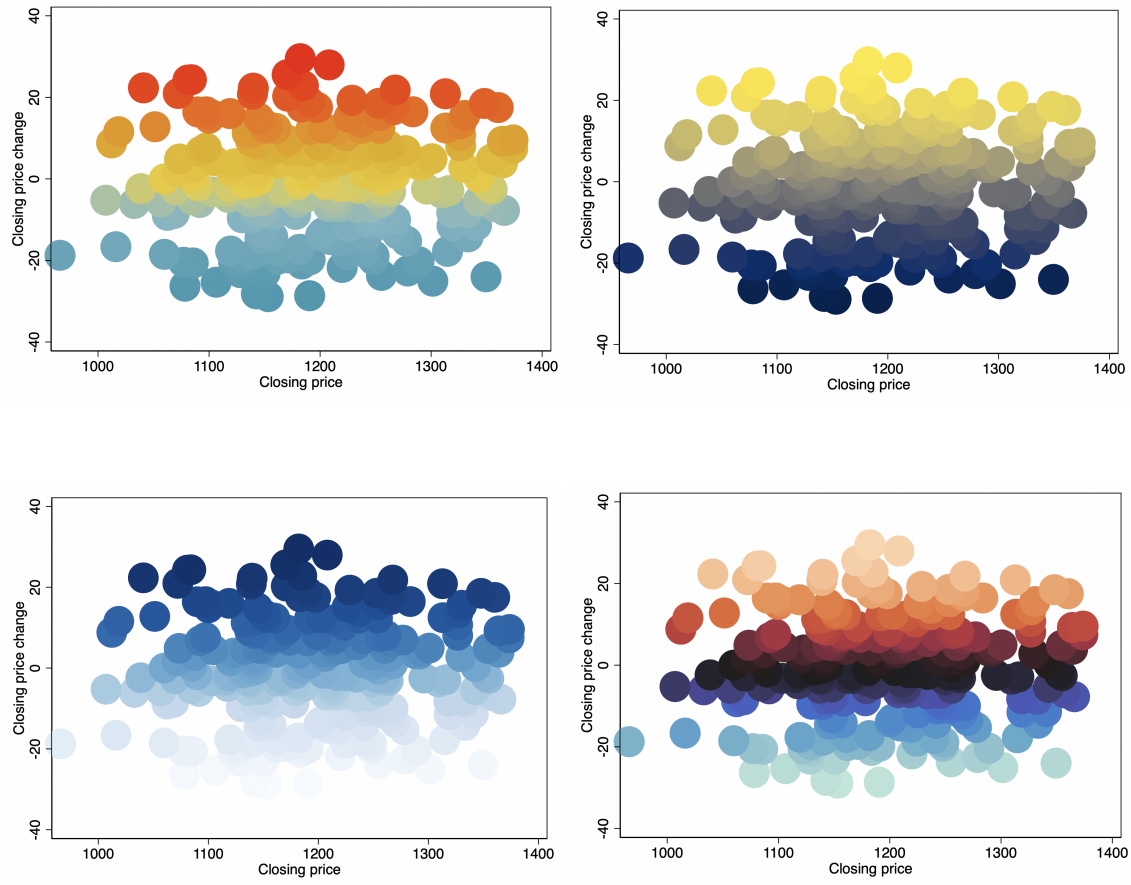
You can refer to observations by their row number with *in*

```
sysuse auto2, clear
*a single observation
replace trunk = 1 in 1
*multiple observations
replace trunk = 0 in 2/5
```

97 colorpalette

Use *colorpalette* to color your graph:

```
capture ssc inst mscatter
capture ssc inst palettes
sysuse sp500, clear
foreach i in Zissou1 cividis icefire Blues {
  mscatter change close if inrange(change, -30, 30), msymbol(O)
     msize(7) sch(s1mono) over(change) colorpalette('i')
}
```



98 Lower case variable names

To rename all variables with lower case names:

```
rename *, lower
```

99 tablbl

tablbl displays both values and value labels for one-way and two-way frequency tables, while *tabulate* does not.

```
net install tablbl,
   from(https://raw.githubusercontent.com/tgoldring/tablbl/master)
sysuse auto2, clear
tablbl rep78 foreign, m
```

Repair record 1978	Car origin 0. Domest	1. Foreig	Total
1. Poor	2	0	2
2. Fair	8	0	8
3. Average	27	3	30
4. Good	9	9	18
5. Excellent	2	9	11
.	4	1	5
Total	52	22	74

100 greshape

greshape can be much faster than "reshape":

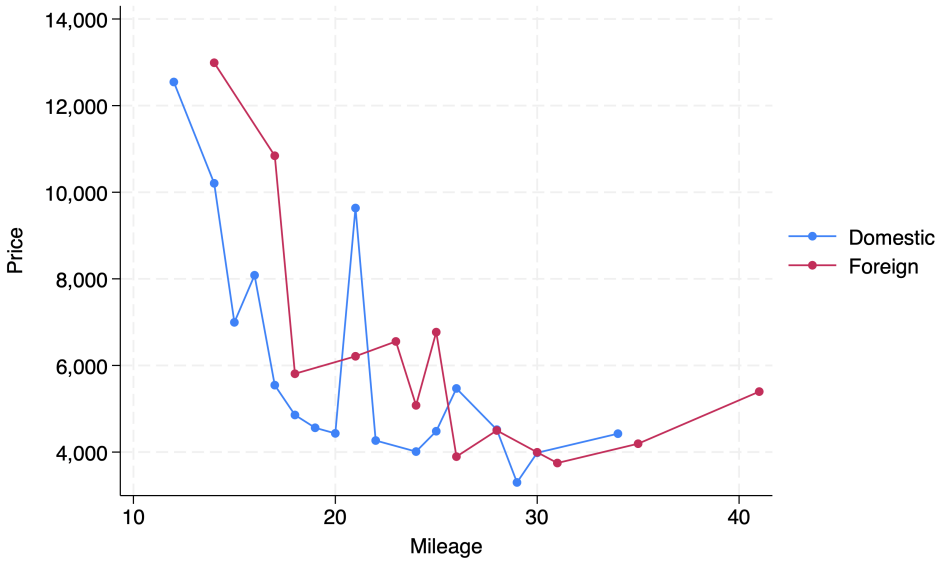
```
ssc install gtools
clear all
set obs 1000000
gen i = _n
forvalues i=1/20 {
  gen a`i'=runiform()
}

preserve
reshape long a, i(i) j(j)
restore

greshape long a, i(i) j(j)
```

```
. *reshape long a, i(i) j(j)
. timer list 1
   1:     17.25 /        1 =      17.2520

.
. *greshape long a, i(i) j(j)
. timer list 2
   2:      0.98 /        1 =       0.9840
```

101 Connected scatterplots

Use *connected* within *twoway* to make connected scatter plots:

```
sysuse auto2, clear
collapse (mean) price, by(mpg foreign)
twoway (connected price mpg if foreign==0) (connected price mpg if
   foreign==1), legend(order(1 "Domestic" 2 "Foreign")) ytitle(Price)
   xtitle(Mileage)
```



102 distinct

Use *distinct* to quickly see how many distinct values a variable has:

```
ssc install distinct
sysuse auto2, clear
distinct trunk
```

```
. distinct trunk

       |      Observations
       |     total   distinct
-------+----------------------
 trunk |        74         18
```

103 unique

Use *unique* to tell you how many distinct values a variable has. It can also tell you how many combinations of values are in multiple variables.

```
ssc install unique
sysuse auto2, clear

unique trunk
unique trunk turn
```

```
. unique trunk
Number of unique values of trunk is  18
Number of records is  74

. unique trunk turn
Number of unique values of trunk turn is  55
Number of records is  74
```

104 reghdfe

reghdfe can absorb multiple fixed effects and has some nice features that *areg* doesn't have:

```
ssc install reghdfe
webuse hdfe, clear
reghdfe y x, absorb(id FE1=a1 FE2=a2) resid(resid)
```

105 areg

If you don't need the extra features of *reghdfe*, *areg*, which can also absorb multiple fixed effects, can be faster:

```
webuse hdfe, clear

areg y x i.a1 i.a2, absorb(id)

areg y x, absorb(id a1 a2)

reghdfe y x, absorb(id a1 a2)
```

```
. *areg y x i.a1 i.a2, absorb(id)
. timer list 1
   1:    113.27 /        1 =     113.2690

.
. *areg y x, absorb(id a1 a2)
. timer list 2
   2:      0.69 /        1 =       0.6900

.
. *reghdfe y x, absorb(id a1 a2)
. timer list 3
   3:      6.72 /        1 =       6.7210
```

Note also that *reg* can also absorb multiple fixed effects using the same syntax as *areg*.

106 reghdfejl

reghdfejl can be faster than *areg* and *reghdfe* (note that the first regression you run with *reghdfejl* is slower as julia gets warmed up, but subsequent ones are faster):

```
ssc install julia
*update version number with latest version here:
   https://github.com/droodman/reghdfejl/releases
net install reghdfejl, replace
   from(https://raw.github.com/droodman/reghdfejl/v1.1.1)
webuse hdfe, clear
tempfile a
save `a'
forv i=1/9 {
  append using `a'
}

areg y x, a(id a1 a2)
reghdfe y x, a(id a1 a2)
reghdfejl y x, a(id a1 a2)
```

```
. *areg y x, a(id a1 a2)
. timer list 1
   1:      6.76 /        1 =       6.7640

.
. *reghdfe y x, a(id a1 a2)
. timer list 2
   2:     38.07 /        1 =      38.0740

.
. *reghdfejl y x, a(id a1 a2)
. timer list 3
   3:      2.84 /        1 =       2.8400
```

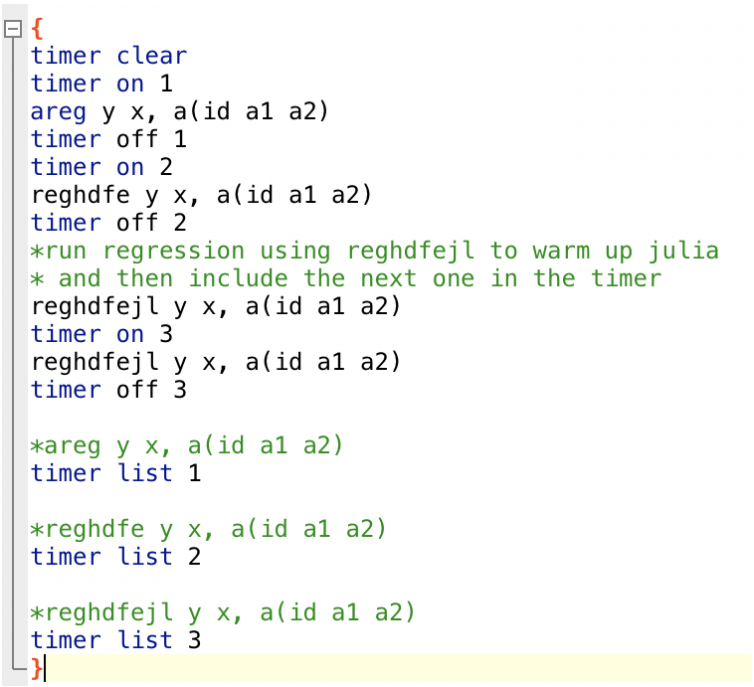
107 set timeout

if you are trying to download something (for example, a Stata update or education data using the *educationdata* package) and are getting an error, see if this fixes it:

```
set timeout1 1000, perm
```

108 braces

In the Do-file Editor, surround a section of code with braces to give you the option to collapse that section of code (which you can do by clicking on the white box next to the first brace.



109 Create a variable based on one observation within a group

Say that you have groups and you want to create a new variable based on one observation within a group that meets some criteria. This new variable will be constant within the group. Here's one way to do this:

```
sysuse sandstone, clear
gen n=_n
preserve
keep if collection==2
bysort northing (n): keep if _n==1
rename depth depth_estimated
tempfile m
save 'm'
restore
merge m:1 northing using 'm', keepusing(depth_estimated)
```

110 set rmsg on

Use *set rmsg on* to show how long everything takes:

```
set rmsg on
webuse hdfe, clear
quietly sum a
set rmsg off
```

```
. set rmsg on
r; t=0.00 22:13:51

. webuse hdfe, clear
r; t=4.15 22:13:55

. quietly sum a
r; t=0.01 22:13:55

. set rmsg off
```

111 Time each iteration takes

If you would like to see how long each iteration of a loop takes, you can do:

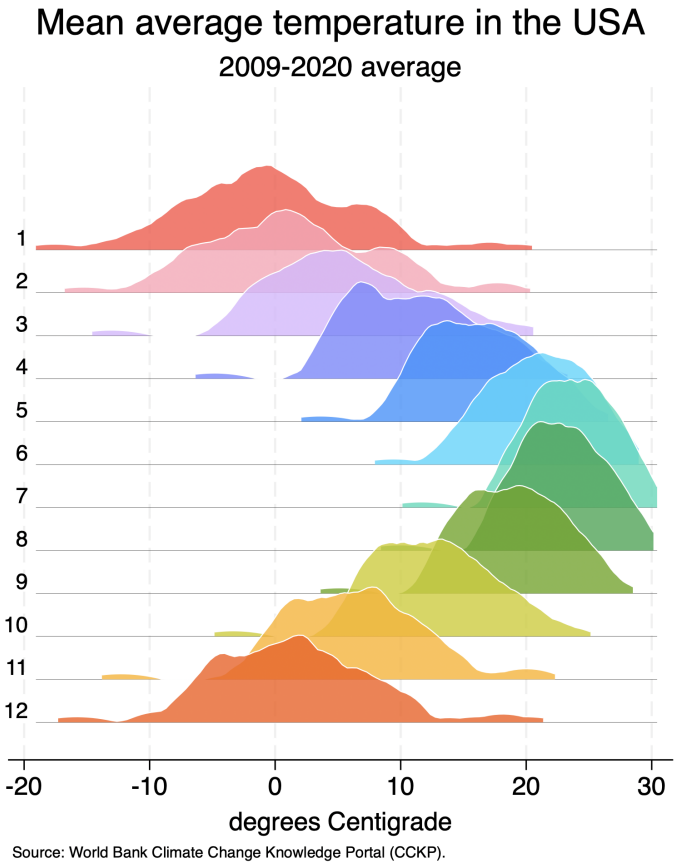
```
timer clear
foreach i of num 1/4 {
  timer on 1
  quietly webuse hdfe, clear
  timer off 1

  di "`var', *******Iteration `i'"
  timer list 1
}
```

```
, *******Iteration 1
   1:      3.95 /        1 =       3.9510
, *******Iteration 2
   1:      8.09 /        2 =       4.0425
, *******Iteration 3
   1:     11.82 /        3 =       3.9413
, *******Iteration 4
   1:     16.48 /        4 =       4.1193
```

112 Ridgeline plots

To make ridgeline plots, use the *ridgeline* package:

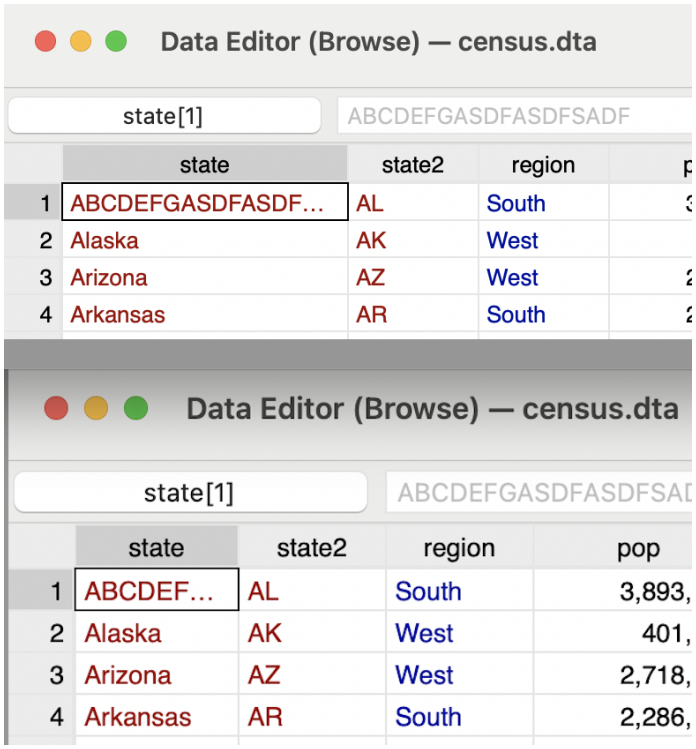
```
*Uses code from https://github.com/asjadnaqvi/stata-ridgeline
ssc install ridgeline, replace
*Get data from:
   https://github.com/asjadnaqvi/stata-ridgeline/tree/main/data
use us_meantemp.dta, clear

ridgeline meantemp, by(month) bwid(1.5) labs(3) overlap(3) yline yrev
   palette(CET C6) ///
  xlabel(-20(10)30) ///
  xtitle("degrees Centigrade") ///
  title("Mean average temperature in the USA") subtitle("2009-2020
     average") ///
  note("Source: World Bank Climate Change Knowledge Portal (CCKP).",
     size(vsmall)) ///
    xsize(4) ysize(5)
```



113 Resize column width

Use *recol* to shorten the column width of variables in the Data Editor. This is useful for really long strings.

```
ssc install recol
sysuse census, clear
replace state = "ABCDEFGASDFASDFSADF" in 1
recol state, maxwidth(7) compress
```



114 set sortseed

Use *set sortseed* to set the seed for sorting tie breakers for things like *sort*, *gsort*, and *kmatch, wor*. This can be useful when creating replicable code.

115 Python

You can use Python within Stata:

```
python
print("hello world")
end
```

And you can use the Python *pystata* package to use Stata from within Python.

116 bcuse

To access many additional datasets, use *bcuse*. The list of available datasets is available at http://fmwww.bc.edu/ec-p/data/wooldridge/datasets.list.html. To use a particular dataset, just put the name of the dataset after *bcuse*:

```
ssc install bcuse
bcuse fish, clear
```

117 ftools and gtools

Commands in *ftools* can be faster than the analogous base commands, particularly for large datasets. Here are (most of) the available commands: *fegen*, *fcollapse*, *join/fmerge*, *fisid*, *flevelsof*.

Similarly, commands in *gtools* are often faster than analogous commands (including in *ftools*. Here are (most of) the available commands: *gcollapse*, *greshape*, *gegen*, *gcontract*, *gisid*, *glevelsof*, *gduplicates*, *gquantiles* (replaces *xtile*, *pctile*, and *_pctile*), *gstats tab* (replaces *tabstat*), *gstats sum* (replaces *sum, detail*), *gstats hdfe*, *gstats winsor*, *gunique*, and *gdistinct*.

118 estout, outreg2, asdoc, and sum2docx

If you don't want to deal with creating custom .tex files for your tables as in Tip 15 (*texdoc*), there are several alternatives.

My favorite alternative for creating .tex files is the very-versatile *estout*. *outreg2* is also very popular. According to the March 2025 most-downloaded ssc package list (*ssc hot*), these are the first- and second-most downloaded ssc packages, respectively. Both of these can also write to other formats, such as Microsoft Word.

asdoc and *sum2docx* are two additional popular solutions to create tables for Microsoft Word, the latter focusing on summary statistics.

119 Conditions within for-loop

You can use conditions in a for loop as follows:

```
foreach i in first second {
  sysuse auto2, clear
  if "`i'"=="first" keep if mpg<17
  if "`i'"=="second" keep if trunk>11

  reg price mpg
}
```

120 sencode

Use *sencode* to gain control over the order in which values are encoded. With *encode*, the string variable is encoded in alphabetical order. By default, *sencode* encodes in the order in which the values appear in the data. You can also use the *gsort* option to code in alphabetical or reverse alphabetical order:

```
sysuse educ99gdp, clear
sort private
encode country, gen(country_encode)
sencode country, gen(country_sencode)
sencode country, gen(country_sencode2) gsort(-country)
```

	country	public	private	country_encode	country_sencode	country_senco…
1	Denmark	1.5	.1	Denmark	Denmark	Denmark
2	Sweden	1.5	.2	Sweden	Sweden	Sweden
3	Germany	.9	.2	Germany	Germany	Germany
4	Ireland	1.1	.3	Ireland	Ireland	Ireland
5	France	.9	.4	France	France	France
6	Netherlands	1	.4	Netherlands	Netherlands	Netherlands
7	Britain	.7	.4	Britain	Britain	Britain
8	Australia	.7	.7	Australia	Australia	Australia
9	Canada	1.5	.9	Canada	Canada	Canada
10	United States	1.1	1.2	United States	United States	United States

```
. fre country_encode

country_encode ── Country
```

			Freq.	Percent	Valid	Cum.
Valid	1	Australia	1	10.00	10.00	10.00
	2	Britain	1	10.00	10.00	20.00
	3	Canada	1	10.00	10.00	30.00
	4	Denmark	1	10.00	10.00	40.00
	5	France	1	10.00	10.00	50.00
	6	Germany	1	10.00	10.00	60.00
	7	Ireland	1	10.00	10.00	70.00
	8	Netherlands	1	10.00	10.00	80.00
	9	Sweden	1	10.00	10.00	90.00
	10	United States	1	10.00	10.00	100.00
	Total		10	100.00	100.00	

```
. fre country_sencode

country_sencode ── Country
```

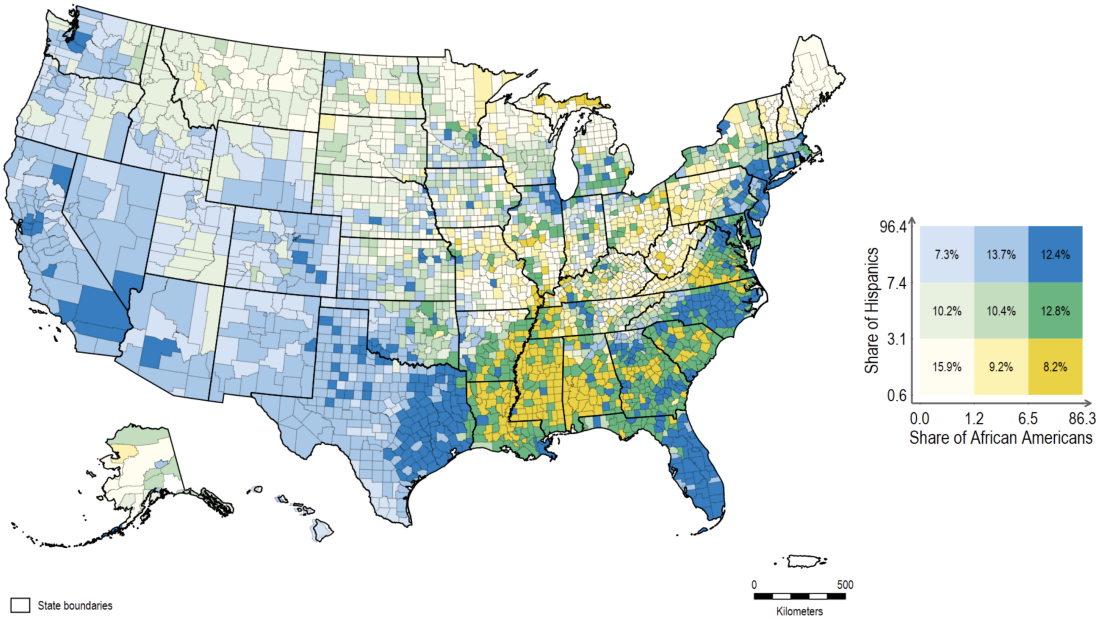
			Freq.	Percent	Valid	Cum.
Valid	1	Denmark	1	10.00	10.00	10.00
	2	Sweden	1	10.00	10.00	20.00
	3	Germany	1	10.00	10.00	30.00
	4	Ireland	1	10.00	10.00	40.00
	5	France	1	10.00	10.00	50.00
	6	Netherlands	1	10.00	10.00	60.00
	7	Britain	1	10.00	10.00	70.00
	8	Australia	1	10.00	10.00	80.00
	9	Canada	1	10.00	10.00	90.00
	10	United States	1	10.00	10.00	100.00
	Total		10	100.00	100.00	

```
. fre country_sencode2

country_sencode2 ── Country
```

			Freq.	Percent	Valid	Cum.
Valid	1	United States	1	10.00	10.00	10.00
	2	Sweden	1	10.00	10.00	20.00
	3	Netherlands	1	10.00	10.00	30.00
	4	Ireland	1	10.00	10.00	40.00
	5	Germany	1	10.00	10.00	50.00
	6	France	1	10.00	10.00	60.00
	7	Denmark	1	10.00	10.00	70.00
	8	Canada	1	10.00	10.00	80.00
	9	Britain	1	10.00	10.00	90.00
	10	Australia	1	10.00	10.00	100.00
	Total		10	100.00	100.00	

121 bimap

Acknowledging that some people have very strong opinions about bivariate maps, you can use the *bimap* package to create them. I take the figure below directly from https://github.com/asjadnaqvi/stata-bimap, where you can find the code and data needed to create it.



122 grstyle

grstyle gives you a lot of control over the appearance of your graphs.

123 parmest

Use *parmest* to put your most recent estimation output into a dataset:

```
ssc install parmest
sysuse auto2, clear
reg price mpg headroom trunk weight
parmest, norestore
```

```
. reg price mpg headroom trunk weight
```

Source	SS	df	MS	Number of obs	=	74
				F(4, 69)	=	8.21
Model	204838391	4	51209597.9	Prob > F	=	0.0000
Residual	430227005	69	6235173.98	R-squared	=	0.3225
				Adj R-squared	=	0.2833
Total	635065396	73	8699525.97	Root MSE	=	2497

price	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
mpg	-54.79153	85.91635	-0.64	0.526	-226.19	116.6069
headroom	-726.5434	462.0322	-1.57	0.120	-1648.272	195.1856
trunk	23.04248	108.3649	0.21	0.832	-193.1396	239.2246
weight	2.011936	.7036432	2.86	0.006	.6082062	3.415666
_cons	3114.94	3648.08	0.85	0.396	-4162.779	10392.66

```
. parmest, norestore
```

Data Editor (Browse)

parm[1] | mpg

	parm	estimate	stderr	dof	t	p	min95	max95
1	mpg	-54.791529	85.91635	69	-.63773111	.52575944	-226.18996	116.6069
2	headroom	-726.54336	462.03217	69	-1.5724952	.12041033	-1648.2723	195.1856
3	trunk	23.042485	108.36492	69	.21263786	.83223638	-193.13961	239.22458
4	weight	2.011936	.70364321	69	2.8593127	.00561091	.60820619	3.4156658
5	_cons	3114.9403	3648.0796	69	.85385755	.3961386	-4162.7794	10392.66

See also *regsave*.

124 Create year, month, week, and day of week variables

Given a variable in date format, you can create year, month, week, and day of week variables:

```
sysuse tsline2, clear
gen year = year(day)
gen month = month(day)
gen week = week(day)
gen dow = dow(day)
```

Keep in mind that the first week is the first seven days of the year and the last week of the year (week 52) can include more than seven days. For the dow variable, 0 is Sunday and 6 is Saturday; I sometimes like to recode Sunday to be 7.

	day	calories	ucalories	lcalories	year	month	week	dow
355	21dec2002	3689.6	3789.6	3589.6	2002	12	51	6
356	22dec2002	3685.6	3785.6	3585.6	2002	12	51	0
357	23dec2002	3787.4	3887.4	3687.4	2002	12	51	1
358	24dec2002	3790.2	3890.2	3690.2	2002	12	52	2
359	25dec2002	4340.2	4440.2	4240.2	2002	12	52	3
360	26dec2002	3785.4	3885.4	3685.4	2002	12	52	4
361	27dec2002	3795.2	3895.2	3695.2	2002	12	52	5
362	28dec2002	3896.8	3996.8	3796.8	2002	12	52	6
363	29dec2002	3895.6	3995.6	3795.6	2002	12	52	0
364	30dec2002	3864.6	3964.6	3764.6	2002	12	52	1
365	31dec2002	3898.2	3998.2	3798.2	2002	12	52	2

125 freqindex

Use *freqindex* to compute the frequency of each word in a string variable:

```
sysuse githubfiles, clear
freqindex packagename
gsort -freq
```

	grams	freq
1	all	246
2	brewscheme	181
3	wbopendata	165
4	ceq	148
5	finance	82
6	ftools	78
7	listutil	75
8	CEQStataPackage	73
9	egenmore	71
10	asmie02	71
11	elabel	70
12	circular	65
13	ol_tools_casen	61
14	sq	57
15	[illegible]	57

126 revrs

To reverse the order of value orders, use *revrs*:

```
ssc install revrs
sysuse sandstone, clear
fre collection
revrs collection, replace
fre collection
```

```
. fre collection

collection ── Type of collection method
```

		Freq.	Percent	Valid	Cum.
Valid	1 Measured	**17**	**0.27**	**0.27**	**0.27**
	2 Estimated	**59**	**0.92**	**0.92**	**1.19**
	3 Interpolated	**6324**	**98.81**	**98.81**	**100.00**
	Total	**6400**	**100.00**	**100.00**	

```
. revrs collection, replace

. fre collection

collection ── Type of collection method
```

		Freq.	Percent	Valid	Cum.
Valid	1 Interpolated	**6324**	**98.81**	**98.81**	**98.81**
	2 Estimated	**59**	**0.92**	**0.92**	**99.73**
	3 Measured	**17**	**0.27**	**0.27**	**100.00**
	Total	**6400**	**100.00**	**100.00**	

127 educationdata

Use *educationdata* to access various education datasets, including IPEDS, College Scoreboard, Common Core of Data, Civil Rights Data Collection, and EdFacts:

```
ssc install educationdata
educationdata using "college ipeds salaries-instructional-staff",
   sub(year=2020) clear csv
```

128 wbopendata

You can access a great deal of World Bank data using *wbopendata*:

```
ssc install wbopendata
wbopendata, indicator(6.0.GDP_current) clear
```

129 freduse

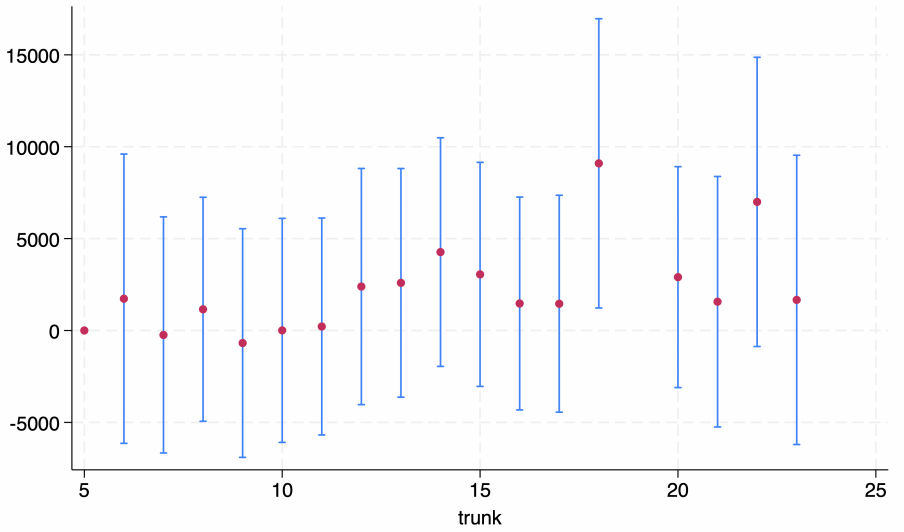
To access data from FRED, use *freduse*:

```
ssc install freduse
freduse CPIAUCSL, clear
```

130 Put error bands on graph

To put error bands on a graph, use *rcap*:

```
sysuse auto2, clear
reg price i.trunk
parmest, norestore
drop if parm=="_cons"
gen trunk = substr(parm, 1, 1) if _n==1 | strlen(parm)==7
replace trunk = substr(parm, 1, 2) if missing(trunk)
destring trunk, replace
twoway (rcap min95 max95 trunk) ///
     (scatter estimate trunk), legend(off)
```



131 rowmiss

To count the number of missing values each row has, use *rowmiss*:

```
sysuse gitget, clear
drop homepage dependency
egen rowmiss = rowmiss(*)
tab rowmiss
```

```
. egen rowmiss = rowmiss(*)

. tab rowmiss
```

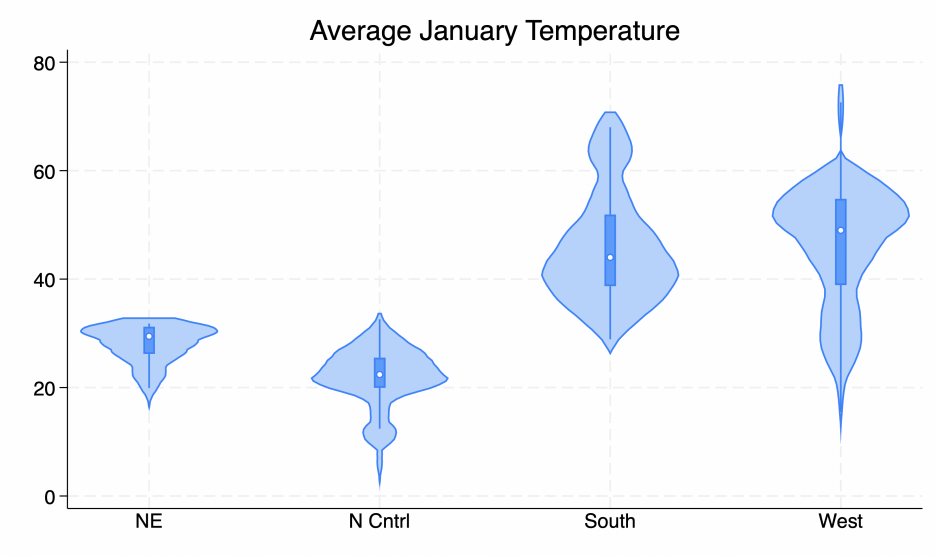
rowmiss	Freq.	Percent	Cum.
0	250	47.62	47.62
1	218	41.52	89.14
2	21	4.00	93.14
5	11	2.10	95.24
6	24	4.57	99.81
7	1	0.19	100.00
Total	525	100.00	

rownonmiss counts the number of nonmissing values.

132 vioplot

To create a violin plot, use *vioplot*:

```
ssc install vioplot
sysuse citytemp4, clear
vioplot tempjan, over(region) title(Average January Temperature)
```



133 Winsorize a variable

To winsorize a variable, use *winsor2*, which replaces values beyond the indicated percentiles with the values at those percentiles:

```
ssc install winsor2
sysuse nlsw88, clear
keep if _n<=1000
sort hours
winsor2 hours, cuts(1 99)
order hours_w, after(hours)
```

hours	hours_w
2	4
2	4
2	4
2	4
2	4
3	4
3	4
3	4
3	4
4	4
4	4
4	4
5	5
5	5
5	5
5	5
5	5
7	7
7	7

If you use this command, it would be good to understand exactly how it obtains the values for the specified percentiles.

Use the *trim* option to trim as opposed to winsorize.

134 extremes

To view the lowest and highest values of a variable, use *extremes*:

```
ssc install extremes
sysuse citytemp4, clear
extremes tempjuly
```

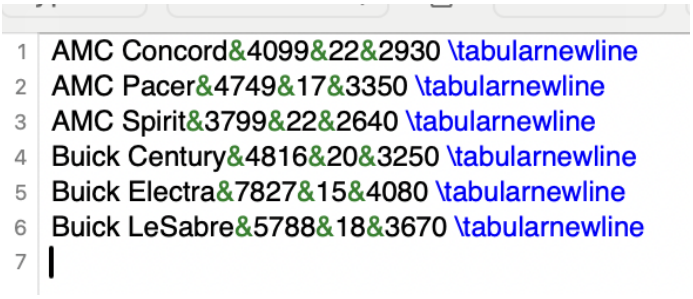
```
. extremes tempjuly

     +-------------------+
     | obs:   tempjuly |
     |-------------------|
     | 701.       58.1 |
     | 848.       59.5 |
     | 849.       59.5 |
     | 850.       59.5 |
     | 851.       61.5 |
     +-------------------+

     +-------------------+
     | 781.       90.3 |
     | 909.       91.7 |
     | 865.       92.3 |
     | 866.       92.3 |
     | 930.       93.6 |
     +-------------------+

note: 2 values of 90.3
```

The default is to display the five lowest and highest values, but you can change this with the *n()* option.

135 Save data to .tex

To save your data to a .tex file, use *texsave*:

```
ssc install texsave
sysuse auto2, clear
keep make price mpg weight
keep if _n<7
texsave using "auto.tex", replace
```



136 Sizes

Here is the list of sizes available for things like the text size:

zero
minuscule
quarter_tiny
third_tiny
half_tiny
tiny
vsmall
small
medsmall
medium
medlarge
large
vlarge
huge
vhuge

You can also choose sizes relative to the size of the graph using:

tenth
quarter
third
half
full

Finally, you can have further control over relative size using numbers such as *4*, *5pt*, *.5in*, *4cm*, *3, and *-.6; see *help relativesize* for details.

137 For loops

If you need to loop over consecutive integers, use *forvalues*:

```
forvalues i = 5/8 {
  di "`i'"
}
```

Use *foreach...in* when you want to write out each element:

```
foreach i in 5 6 7 8 {
  di "`i'"
}
```

You can also use *foreach...in* with locals and globals:

```
local loc 5 6 7 8
foreach i in `loc' {
  di "`i'"
}

global loc2 5 6 7 8
foreach i in $loc2 {
  di "`i'"
}
```

I prefer these less, but the following—*foreach...of local* and *foreach...of global* are equivalent:

```
foreach i of local loc {
  di "`i'"
}

foreach i of global loc2 {
  di "`i'"
}
```

Finally (and there are multiple ways to do this), but *foreach...of varlist _all* allows you to loop through all variables:

```
sysuse auto2, clear
foreach i of varlist _all {
  sum `i'
}
```

138 Create new variable based on value label of another variable

To create a new variable based on the value label of another variable, use *decode*.

First, note that if we just use *gen*, the new variable contains the values, not the value labels:

```
sysuse auto2, clear
keep rep78

gen rep = rep78
```

But if we use *decode*, the new variable contains the value labels:

```
decode rep78, gen(rep2)
```

	rep78	rep	rep2
1	Average	3	Average
2	Average	3	Average
3	.	.	
4	Average	3	Average
5	Good	4	Good
6	Average	3	Average

139 Sorting of string and numeric variables with missing values

When you sort a string variable, the missing values are at the top:

```
sysuse auto2, clear
gen rep_numeric = rep78
decode rep78, gen(rep_string)
keep rep_numeric rep_string
keep if _n<=7
sort rep_string
```

	rep_num…	rep_string
1	.	
2	.	
3	3	Average
4	3	Average
5	3	Average
6	3	Average
7	4	Good

But when you sort a numeric variable, the missing values are at the bottom:

```
sort rep_numeric
```

	rep_num…	rep_string
1	3	Average
2	3	Average
3	3	Average
4	3	Average
5	4	Good
6	.	
7	.	

140 Be careful when creating a new variable using conditional logic

Stata treats missing numeric values as having infinite value. Thus, you need to be careful when defining a new variable with conditional logic based on a variable with missing values. For example, if we want to create a variable, *over3_try1*, that is 0 if *rep* is under 3 and 1 if it is over 3, observations that are missing *rep* would get coded as a *1* (as opposed to being coded as missing, which is probably what we actually want):

```
sysuse auto2, clear
gen rep = rep78
keep if _n<=7
keep rep
sort rep

gen over3_try1 = rep>3
```

If we add *& !missing(rep)*, we will get another coding that is likely also not what we want:

```
gen over3_try2 = rep>3 & !missing(rep)
```

But if we instead restrict the statement to consider only the observations that are not missing the variable (using *if !missing(rep)*), then we get what we (likely) want:

```
gen over3_try3 = rep>3 if !missing(rep)
```

	rep	over3_try1	over3_try2	over3_try3
1	3	0	0	0
2	3	0	0	0
3	3	0	0	0
4	3	0	0	0
5	4	1	1	1
6	.	1	0	.
7	.	1	0	.