“How can I access information stored after I run a command in Stata (returned results)?”

Authored by

stats writer

July 1, 2024

The process of accessing information stored after running a command in Stata refers to the ability to retrieve and view the results of a previously executed command. This can be done by using the appropriate commands within the Stata software, which allows users to access and manipulate the returned results for further analysis and interpretation. By accessing this information, users can effectively utilize the data generated by the command to inform their research and decision-making processes. This feature of Stata enhances the efficiency and functionality of the software, making it a valuable tool for data analysis and management.

How can I access information stored after I run a command in Stata (returned results)? | Stata FAQ

In addition to the output in the shown in the results window, many of Stata's commands store information about the command and it's results in memory. This allows the user, as well as other Stata commands, to easily make use of this information. Stata calls these returned results. Returned results can be very useful when you want to use information produced by a Stata command to do something else in Stata. For example, if you want to mean center a variable, you can use summarize to calculate the mean, then use the value of the mean calculated by summarize to center the variable. Using returned results will eliminate

the need to retype or cut and paste the value of the mean.

Another example of

how returned results can be useful is if you want to generate predicted values of the outcome

variable when the predictor variables are at a specific set of values, again

here, you could retype the coefficients or use cut and paste, but returned results

make the task much easier.

The best way to get a sense of how returned results work is to jump right in

and start looking at and using them. The code below opens an example dataset and

uses summarize (abbreviated sum) to generate descriptive statistics for the variable read. This produces the

expected output, but more importantly for our purposes, Stata now has results from the

summarize command stored in memory. But how do you know what information has

been stored? A listing of the information saved by each command is included in the help file and/or printed

manual, so I could look
there, but I can also just type return list, which
will list all the returned results in memory.

```
use https://stats.idre.ucla.edu/stat/stata/notes/hsb2, clear
sum read

Variable | Obs Mean Std. Dev. Min Max
-------------+--------------------------------------------------------
read | 200 52.23 10.25294 28 76

return list
scalars:
r(N) = 200
r(sum_w) = 200
r(mean) = 52.23
r(Var) = 105.1227135678392
r(sd) = 10.25293682648241
r(min) = 28
r(max) = 76
r(sum) = 10446
```

Above is a list of the returned results, as you can see each result is of the

form r(...) where the ellipses ("...") is a short label. We could
see the help file for the summarize command to find out what each item on
the list is, but it is often easy to figure out what value is assigned to what result, for example, r(mean), not surprisingly contains the mean of
read (you can check
this against the output), but others are not as obvious, for example
r(sum_w), for these, you may need to consult the manual if you think you
might want to use them. Most of the time the process will be relatively easy
because you'll know what
result you want to access, you will be looking at the list to find out what name it is stored under,
rather than looking at the list and trying to figure out what each item is.

As you might imagine, different commands, and even the same command with different options,
store different results. Below we summarize the variable read again, but add the detail option.

Then we use return list to get the list of returned results. Just as the
detail option adds additional information to the output, it also results in
additional information stored in the returned results. The new list includes all of the information
returned by the sum command
above, plus skewness; kurtosis; and a number of percentiles, including the 1st (
r(p25))and 3rd
(r(p75)) quartiles and the median (r(p50)).

sum read, detail
reading score

Percentiles Smallest
1% 32.5 28
5% 36 31
10% 39 34 Obs 200
25% 44 34 Sum of Wgt. 200

50% 50 Mean 52.23
Largest Std. Dev. 10.25294
75% 60 73

90% 67 73 Variance 105.1227

95% 68 76 Skewness .1948373

99% 74.5 76 Kurtosis 2.363052

return list

scalars:

r(N) = 200

r(sum_w) = 200

r(mean) = 52.23

r(Var) = 105.1227135678392

r(sd) = 10.25293682648241

r(skewness) = .1948372909440272

r(kurtosis) = 2.363051990033788

r(sum) = 10446

r(min) = 28

r(max) = 76

r(p1) = 32.5

r(p5) = 36

r(p10) = 39

r(p25) = 44

r(p50) = 50

r(p75) = 60

r(p90) = 67

r(p95) = 68
r(p99) = 74.5

Now that we have some sense of what results are returned by the summarize command, we can make use of the returned results. Following through with one of the examples mentioned above, we will mean center the variable read. Assuming that the last command we ran was the summarize command above, the code below uses generates a new variable, c_read that contains the mean centered values of read. Notice that instead of using the actual value of the mean of read in this command, we used the name of the returned result (i.e. r(mean)), Stata knows when it sees r(mean) that we actually mean the value stored in that system variable. On the next line we summarize the new variable c_read, while the mean is not exactly equal to zero, it is within rounding error of

zero, so we know that we have properly mean centered the variable read.

```
gen c_read = read - r(mean)
sum c_read

Variable | Obs Mean Std. Dev. Min Max
-------------+--------------------------------------------------------
c_read | 200 2.18e-07 10.25294 -24.23 23.77
```

As the code above suggests, we can use returned results pretty much the same way we would use an actual number. This is because Stata uses the r(...) as a placeholder for a real value. For another example of this, say that we want to calculate the variance of read from its standard deviation (ignoring the fact that summarize returns the variance in r(Var)). We can do this on the fly using the display command as a calculator. The second line of code below does this. We can even check the result by cutting and pasting the value of the

standard deviation from
the output, which is done in the third command below. The results are basically
the same, the very slight difference is rounding error because the stored
estimate r(sd) contains more digits of accuracy than the value of the
standard deviation displayed in the output.

display r(sd)^2
105.12271

display 10.25294^2
105.12278

Types of returned results, r-class and e-class

Now that you know a little about returned results and how they work you are
ready for a little more information about them. Returned results come in two
main types, r-class, and e-class (there are also s-class and c-class results/variables, but we will not discuss them here). Commands that perform
estimation, for example regressions of all types, factor

**analysis, and anova are
e-class commands. Other commands, for example summarize, correlate and post-estimation
commands, are r-class commands. The distinction between r-class and e-class commands is important because
Stata stores results from e-class and r-class commands in
different "places." This has two ramifications for you as a user.
First, you need to know whether results are stored in r() or e() (as well as the
name of the result) in order to make use of them. If you're not sure which class a
command you've run is in, you can either look it up in the help file, or "look"
in one place (using the appropriate command to list results), if the results are not
stored there they are probably in the other. A potentially more important
ramification of the difference in how results from r-class and e-class commands
are returned is that returned results are held in memory only until another**

command of the same class is run. That is, returned results from previous commands are
replaced by subsequent commands of the same class. In contrast, running a command of
another class will not affect the returned results. For example, if I run a
regression, and then a second regression, the results of the first regression
(stored in e()) are replaced by those for the second regression (also
stored in e()) . However, if instead of a second regression, I ran a post-estimation command, the results from the regression would remain in
e()
while the results from the post estimation command would be placed in r().

While there is a distinction between the two, the actual use of results from r-class
and e-class commands is very similar. For starters, the commands are parallel, to list
the r-class results stored in memory the command is return list, to do the
same for e-class results the command ereturn list.

Further, except for
the difference in naming conventions (r() vs. e()), the results are accessed in the same way.
The example below demonstrates this, first we regresswrite on female and read, and then use ereturn list to look at
the returned results.

regress write female read

Source | SS df MS Number of obs = 200
----------+---------- F(2, 197) = 77.21
Model | 7856.32118 2 3928.16059 Prob > F = 0.0000
Residual | 10022.5538 197 50.8759077 R-squared = 0.4394
----------+---------- Adj R-squared = 0.4337
Total | 17878.875 199 89.843593 Root MSE = 7.1327

--
write | Coef. Std. Err. t P>|t|
----------+---
female | 5.486894 1.014261 5.41 0.000 3.48669 7.487098
read | .5658869 .0493849 11.46 0.000 .468496 .6632778
_cons | 20.22837 2.713756 7.45 0.000 14.87663 25.58011

---

ereturn list

scalars:
e(N) = 200
e(df_m) = 2
e(df_r) = 197
e(F) = 77.21062421518363
e(r2) = .439419213038750 6
e(rmse) = 7.132734938503835
e(mss) = 7856.321182518186
e(rss) = 10022.5538174818
e(r2_a) = .433728037536605 9
e(ll) = -675.2152914029985
e(ll_0) = -733.0934827146213

macros:
e(cmdline) : "regress write female read"
e(title) : "Linear regression"
e(vce) : "ols"
e(depvar) : "write"
e(cmd) : "regress"
e(properties) : "b V"
e(predict) : "regres_p"

e(model) : "ols"
e(estat_cmd) : "regress_estat"

matrices:
e(b) : 1 x 3
e(V) : 3 x 3

functions:
e(sample)

The list of returned results for regress includes several types of returned results listed under the headings scalars, macros, matrices and functions. We will discuss the types of returned results below, but for now we will show how you can use the scalar returned results the same way that we used the returned results from summarize. For example, one way to calculate the variance of the errors after a regression is to divide the residual sum of squares by the total degrees of freedom (i.e. n-1). The residual sum of squares is stored in e(rss) and that the n for the analysis is

stored in e(N). Below we use the display command as a calculator, along with the
returned results to calculate the variance of the errors.

```
display e(rss)/(e(N)-1)
50.364592
```

How results are returned: Scalars, strings, matrices and functions

As mentioned above, for both r-class and e-class commands, there are multiple types of returned
results, including scalars, strings, matrices, and functions. In the lists of returned results, each type is listed under its own heading. The results listed under the heading "scalars" are just that, a single
numeric value. Their usage is discussed above, so we won't say anymore about
them in this section.

Returned results listed under "macros" are generally strings
that give information about the command that was run. For example, in the
returned results of for the regression shown above, e(cmd_line)

contains the command the user issued (without any abbreviations). These are generally used in programming Stata.

Results listed under "matrices" are, as you would expect, matrices. While the list of results returned by return list and erturn list show you the values taken on by most of the returned results, this is not practical with matrices, instead the dimensions of the matrices are listed. To see the contents of matrices you must display them using matrix commands. We do this below with the matrix of coefficients (e(b)) using the command matrix list e(b). (Note that there is another way to access coefficients and their standard errors after you fit a model, this is discussed below.) If we would like to perform matrix operations on returned matrices, or wish to access individual elements of the matrix, we can move the matrix stored as a returned

result to a normal Stata matrix.
This is done in the final line of syntax below.

matrix list e(b)

e(b)
female read _cons
y1 5.486894 .56588693 20.228368

matrix b = e(b)

Finally, the results returned under the heading "functions" contain functions
that can be used in a manner similar to other Stata functions. The most common function
returned by Stata estimation commands is probably e(sample). This function marks the
sample used in estimation of the last analysis, this is useful as datasets often
contain
missing values resulting in not all cases in the dataset being used in a given
analysis. Assuming that the last
estimation command run was the regression of write on female and

read shown
above, the first line of code below uses e(sample) to find the mean of read among those cases used in the model. The second line of code uses e(sample) to
create a new variable called flag which is equal to 1 for cases that were
used in the analysis, and zero otherwise. (Note since the example dataset contains no
missing data, all of the cases are included in the analysis, and flag is
a constant equal to one.)

```
sum read if e(sample)==1

Variable | Obs Mean Std. Dev. Min Max
-------------+--------------------------------------------------------
read | 200 52.23 10.25294 28 76

gen flag = e(sample)
```

Coefficients and their standard errors

As discussed above, after one fits a model, coefficients and their standard errors are stored
in e() in matrix form. These matrices allow the user

access to the coefficients, but Stata
gives you an even easier way to access this information by storing it in the system variables
_b and _se. To access the value of a regression coefficient after a regression, all
one needs to do is type _b where varname is the name of the predictor variable whose coefficient you
want to examine. To access the standard error, you can simply type _se.
To access the coefficient and standard error of the constant we use _b
and _se respectively. Below we run the same regression model we
ran above (omitting the output), using female and read to predict write.
Once we have estimated the model, we use the display command to show
that the values in _b are equal to our regression coefficients. Finally,
we calculate the predicted value of write
when a female (female=1) student has a read score of 52.

regress write female read

```
Source | SS df MS Number of obs = 200
-------------+------------------------------ F( 2, 197) = 77.21
Model | 7856.32118 2 3928.16059 Prob > F = 0.0000
Residual | 10022.5538 197 50.8759077 R-squared = 0.4394
-------------+------------------------------ Adj R-squared = 0.4337
Total | 17878.875 199 89.843593 Root MSE = 7.1327

------------------------------------------------------------------------------
write | Coef. Std. Err. t P>|t|
-------------+----------------------------------------------------------------
female | 5.486894 1.014261 5.41 0.000 3.48669 7.487098
read | .5658869 .0493849 11.46 0.000 .468496 .6632778
_cons | 20.22837 2.713756 7.45 0.000 14.87663 25.58011
------------------------------------------------------------------------------

display _b
20.228368

display _b
5.486894

display _b
.56588693
```

```
display _b + _b*1 + _b*52
55.141383
```