**Description**

`statsby` collects statistics from `command` across a by list. Typing

```
.statsby exp_list, by(varname): command
```

executes `command` for each group identified by `varname`, building a dataset of the associated values from the expressions in `exp_list`. The resulting dataset replaces the current dataset, unless the `saving()` option is supplied. `varname` can refer to a numeric or a string variable.

`command` defines the statistical command to be executed. Most Stata commands and user-written programs can be used with `statsby`, as long as they follow standard Stata syntax and allow the `if` qualifier; see [U] 11 Language syntax. The `by` prefix cannot be part of `command`.

`exp_list` specifies the statistics to be collected from the execution of `command`. If no expressions are given, `exp_list` assumes a default depending upon whether `command` changes results in `e()` and `r()`. If `command` changes results in `e()`, the default is `_b`. If `command` changes results in `r()` (but not `e()`), the default is all the scalars posted to `r()`. It is an error not to specify an expression in `exp_list` otherwise.

**Quick start**

Replace data in memory with estimates of the coefficient of `x` and constant for each value of `catvar`

```
statsby, by(catvar): regress y x
```

As above, but name new variables `b` and `cons`

```
statsb y b=_b[x] cons=_b[_cons], by(catvar): regress y x
```

Add standard errors of the estimates and use default variable names

```
statsby _b _se, by(catvar): regress y x
```

As above, but retain data in memory and save estimates to `myest.dta`

```
statsby _b _se, by(catvar) saving(myest): regress y x
```

As above, and include estimate for entire dataset

```
statsby _b _se, by(catvar) saving(myest) total: regress y x
```

Note: Any command that accepts the `statsby` prefix may be substituted for `regress` above.

**Menu**

Statistics > Other > Collect statistics for a command across a by list
statsby — Collect statistics for a command across a by list

Syntax

```stata
statsby [exp_list] [ , options ] : command
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td></td>
</tr>
<tr>
<td>*by(varlist [ , missing ])</td>
<td>equivalent to interactive use of by varlist:</td>
</tr>
<tr>
<td>Options</td>
<td>replace data in memory with results</td>
</tr>
<tr>
<td>clear</td>
<td>save results to filename; save statistics in double precision; save results to filename every # replications</td>
</tr>
<tr>
<td>saving(filename, ...)</td>
<td>include results for the entire dataset</td>
</tr>
<tr>
<td>total</td>
<td>include all combinations of subsets of groups</td>
</tr>
<tr>
<td>subsets</td>
<td></td>
</tr>
<tr>
<td>Reporting</td>
<td>suppress replication dots</td>
</tr>
<tr>
<td>nodots</td>
<td>display dots every # replications</td>
</tr>
<tr>
<td>dots(#)</td>
<td>display any output from command</td>
</tr>
<tr>
<td>noisily</td>
<td>suppress trace command</td>
</tr>
<tr>
<td>trace</td>
<td>suppress table legend</td>
</tr>
<tr>
<td>nolegend</td>
<td>display the full table legend</td>
</tr>
<tr>
<td>Advanced</td>
<td>restrict initializing sample to exp; seldom used</td>
</tr>
<tr>
<td>basepop(exp)</td>
<td>do not check for svy commands; seldom used</td>
</tr>
<tr>
<td>force</td>
<td>retain only observations in by-groups when calling command; seldom used</td>
</tr>
<tr>
<td>forcedrop</td>
<td></td>
</tr>
</tbody>
</table>

* by() is required on the dialog box because statsby is useful to the interactive user only when using by().

All weight types supported by command are allowed except pweights; see [U] 11.1.6 weight.

exp_list contains (name: elist)

list
exp

elist contains newvarname = (exp)

(exp)

eexp is specname

[eqno]specname

specname is _b

_b[ []

_se []

_eqno[ []

eqno is ##

name

exp is a standard Stata expression; see [U] 13 Functions and expressions.

Distinguish between [], which are to be typed, and [], which indicate optional arguments.
Statsby — Collect statistics for a command across a by list

Options

Main

by(varlist [, missing]) specifies a list of existing variables that would normally appear in the by varlist section of the command if you were to issue the command interactively. By default, statsby ignores groups in which one or more of the by() variables is missing. Alternatively, missing causes missing values to be treated like any other values in the by-groups, and results from the entire dataset are included with use of the subsets option. If by() is not specified, command will be run on the entire dataset. varlist can contain both numeric and string variables.

total specifies that command be run on the entire dataset, in addition to the groups specified in the by() option.

subsets specifies that command be run for each group defined by any combination of the variables in the by() option.

Advanced

basepop(exp) specifies a base population that statsby uses to evaluate the command and to set up for collecting statistics. The default base population is the entire dataset, or the dataset specified by any if or in conditions specified on the command.

total specifies that command be run on the entire dataset, in addition to the groups specified in the by() option.

subsets specifies that command be run for each group defined by any combination of the variables in the by() option.

Options

clear specifies that it is okay to replace the data in memory, even though the current data have not been saved to disk.

saving(filename[, suboptions]) creates a Stata data file (.dta file) consisting of (for each statistic in exp_list) a variable containing the replicates.

double specifies that the results for each replication be stored as doubles, meaning 8-byte reals. By default, they are stored as floats, meaning 4-byte reals.

every(#) specifies that results be written to disk every #th replication. every() should be specified in conjunction with saving() only when command takes a long time for each replication. This will allow recovery of partial results should your computer crash. See [P] postfile.

total specifies that command be run on the entire dataset, in addition to the groups specified in the by() option.

subsets specifies that command be run for each group defined by any combination of the variables in the by() option.

Nodots and dots(#) specify whether to display replication dots. By default, one dot character is displayed for each by-group. A red ‘x’ is displayed if command returns an error or if any value in exp_list is missing. You can also control whether dots are printed using set dots; see [R] set.
	nodots suppresses display of the replication dots.

dots(#) displays dots every # replications. dots(0) is a synonym for nodots.

noisily causes the output of command to be displayed for each by-group. This option implies the nodots option.

trace causes a trace of the execution of command to be displayed. This option implies the noisily option.

nolegend suppresses the display of the table legend, which identifies the rows of the table with the expressions they represent.

verbose requests that the full table legend be displayed. By default, coefficients and standard errors are not displayed.

Advanced

basepop(exp) specifies a base population that statsby uses to evaluate the command and to set up for collecting statistics. The default base population is the entire dataset, or the dataset specified by any if or in conditions specified on the command.
One situation where basepop() is useful is collecting statistics over the panels of a panel dataset by using an estimator that works for time series, but not panel data, for example,

```stata
.statsby, by(mypanels) basepop(mypanels==2): arima ...
```

force suppresses the restriction that command not be a svy command. statsby does not perform subpopulation estimation for survey data, so it should not be used with svy. statsby reports an error when it encounters svy in command if the force option is not specified. This option is seldom used, so use it only if you know what you are doing.

forcedrop forces statsby to drop all observations except those in each by-group before calling command for the group. This allows statsby to work with user-written programs that completely ignore if and in but do not return an error when either is specified. forcedrop is seldom used.

Remarks and examples

Remarks are presented under the following headings:

- Collecting coefficients and standard errors
- Collecting stored results
- All subsets

Collecting coefficients and standard errors

Example 1

We begin with an example using auto2.dta. In this example, we want to collect the coefficients from a regression in which we model the price of a car on its weight, length, and mpg. We want to run this model for both domestic and foreign cars. We can do this easily by using statsby with the extended expression _b.

```stata
.use https://www.stata-press.com/data/r16/auto2
(1978 Automobile Data)
.statsby _b, by(foreign) verbose nodots: regress price weight length mpg
```

This allows statsby to work with user-written programs that completely ignore if and in but do not return an error when either is specified. forcedrop is seldom used.

If we were interested only in the coefficient of a particular variable, such as mpg, we would specify that particular coefficient; see [U] 13.5 Accessing coefficients and standard errors.
. use https://www.stata-press.com/data/r16/auto2, clear
(1978 Automobile Data)
.statsby mpg=_b[mpg], by(foreign) nodots: regress price weight length mpg
\hspace{1cm} \text{command: } \texttt{regress price weight length mpg}
\hspace{1cm} \text{mpg: } _b[mpg]
\hspace{1cm} \text{by: } \texttt{foreign}
.
\verb|. list|
\begin{verbatim}
 foreign  mpg
1. Domestic 142.7663
2. Foreign -18.4072
\end{verbatim}

The extended expression $\_se$ indicates that we want standard errors.

. use https://www.stata-press.com/data/r16/auto2, clear
(1978 Automobile Data)
.statsby $\_se$, by(foreign) verbose nodots: regress price weight length mpg
\hspace{1cm} \text{command: } \texttt{regress price weight length mpg}
\hspace{1cm} $\_se\_weight$: $\_se[\texttt{weight}]$
\hspace{1cm} $\_se\_length$: $\_se[\texttt{length}]$
\hspace{1cm} $\_se\_mpg$: $\_se[\texttt{mpg}]$
\hspace{1cm} $\_se\_cons$: $\_se[\texttt{cons}]$
\hspace{1cm} \text{by: } \texttt{foreign}
.
\verb|. list|
\begin{verbatim}
 foreign  $\_se\_we$~t  $\_se\_le$h  $\_se\_mpg  $\_se\_cons$
1. Domestic 1.226326 39.48193 134.7221 7770.131
2. Foreign 1.670006 50.70229 59.37442 6337.952
\end{verbatim}

\textbf{Example 2}

For multiple-equation estimations, we can use $[\texttt{eqno}]\_b ([\texttt{eqno}]\_se)$ to get the coefficients (standard errors) of a specific equation or use $\_b (\_se)$ to get the coefficients (standard errors) of all the equations. To demonstrate, we use \texttt{heckman} and a slightly different dataset.

. use https://www.stata-press.com/data/r16/statsby, clear
.statsby $\_b$, by(group) verbose nodots: \texttt{heckman price mpg, sel(trunk)}
\hspace{1cm} \text{command: } \texttt{heckman price mpg, sel(trunk)}
\hspace{1cm} \texttt{price\_b\_mpg: [price]_b[mpg]}
\hspace{1cm} \texttt{price\_b\_cons: [price]_b[\_cons]}
\hspace{1cm} \texttt{select\_b\_tr=rk: [select]_b[trunk]}
\hspace{1cm} \texttt{select\_b\_cons: [select]_b[\_cons]}
\hspace{1cm} \texttt{\_eq3\_b\_athrho: [/]_b[athrho]}
\hspace{1cm} \texttt{\_eq3\_b\_lnsi-a: [/]_b[lnsigma]}
\hspace{1cm} \text{by: } \texttt{group}
To collect the coefficients of the first equation only, we would specify `[price]_b` instead of `._b`.

```
. use https://www.stata-press.com/data/r16/statsby, clear
. statsby [price]_b, by(group) verbose nodots: heckman price mpg, sel(trunk)
```

Technical note

If `command` fails on one or more groups, `statsby` will capture the error messages and ignore those groups.

Collecting stored results

Many Stata commands store results of calculations; see [U] 13.6 Accessing results from Stata commands. `statsby` can collect the stored results and expressions involving these stored results, too. Expressions must be bound in parentheses.
Example 3

Suppose that we want to collect the mean and the median of price, as well as their ratios, and we want to collect them for both domestic and foreign cars. We might type

```
. use https://www.stata-press.com/data/r16/auto2, clear
   (1978 Automobile Data)
. statsby mean=r(mean) median=r(p50) ratio=(r(mean)/r(p50)), by(foreign) nodots:
   > summarize price, detail
```

<table>
<thead>
<tr>
<th>foreign</th>
<th>mean</th>
<th>median</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>6072.423</td>
<td>4782.5</td>
<td>1.269717</td>
</tr>
<tr>
<td>Foreign</td>
<td>6384.682</td>
<td>5759</td>
<td>1.108644</td>
</tr>
</tbody>
</table>

Technical note

In `exp_list`, `newvarname` is not required. If no new variable name is specified, `statsby` names the new variables `_stat_1`, `_stat_2`, and so forth.

All subsets

Example 4

When there are two or more variables in `by(varlist)`, we can execute `command` for any combination, or subset, of the variables in the `by()` option by specifying the `subsets` option.

```
. use https://www.stata-press.com/data/r16/auto2, clear
   (1978 Automobile Data)
. statsby mean=r(mean) median=r(p50) n=r(N), by(foreign rep78) subsets nodots:
   > summarize price, detail
```
In the above dataset, observation 6 is for domestic cars, regardless of the repair record; observation 10 is for foreign cars, regardless of the repair record; observation 11 is for both foreign cars and domestic cars given that the repair record is 1; and the last observation is for the entire dataset.

Technical note

To see the output from *command* for each group identified in the by() option, we can use the noisily option.

```
. use https://www.stata-press.com/data/r16/auto2, clear
   (1978 Automobile Data)
. statsby mean=r(mean) se=(r(sd)/sqrt(r(N))), by(foreign) noisily nodots:
   > summarize price
   statsby: First call to summarize with data as is:
   . summarize price

Variable | Obs  | Mean   | Std. Dev. | Min  | Max  
---------|------|--------|-----------|------|------
price    | 74   | 6165.26| 2949.496  | 3291 | 15906
```

*statsby* legend:

- command: summarize price
- mean: r(mean)
- se: r(sd)/sqrt(r(N))
- by: foreign

Statsby groups

running (summarize price) on group 1
The Stata command `.summarize price` is used to calculate summary statistics for the variable `price` across different groups. Here is the output for two different groups:

```
Variable   |    Obs    |    Mean   |   Std. Dev. |   Min  |   Max
------------|-----------|-----------|-------------|--------|--------
price       |    52     | 6072.423  | 3097.104    |  3291  | 15906  
```

Running `.summarize price` on group 2 produces the following output:

```
Variable   |    Obs    |    Mean   |   Std. Dev. |   Min  |   Max
------------|-----------|-----------|-------------|--------|--------
price       |    22     | 6384.682  | 2621.915    |  3748  | 12990  
```

The results are also displayed in a table format:

<table>
<thead>
<tr>
<th>foreign</th>
<th>mean</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>6072.423</td>
<td>429.4911</td>
</tr>
<tr>
<td>Foreign</td>
<td>6384.682</td>
<td>558.9942</td>
</tr>
</tbody>
</table>

Acknowledgment

Speed improvements in `statsby` were based on code written by Michael Blasnik of Nest Labs.

References


Also see

[D] `by` — Repeat Stata command on subsets of the data
[D] `collapse` — Make dataset of summary statistics
[P] `postfile` — Post results in Stata dataset
[R] `bootstrap` — Bootstrap sampling and estimation
[R] `jackknife` — Jackknife estimation
[R] `permute` — Monte Carlo permutation tests