**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

```
statsby 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
Syntax

\texttt{statsby [exp\_list] [ , options]: command}

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

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

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

\texttt{exp\_list} contains

\begin{itemize}
\item \texttt{(name: elist)}
\item \texttt{elist}
\item \texttt{eexp}
\end{itemize}

\texttt{elist} contains

\begin{itemize}
\item \texttt{newvarname = (exp)}
\item \texttt{(exp)}
\end{itemize}

\texttt{eexp} is

\begin{itemize}
\item \texttt{specname}
\item \texttt{[eqno]specname}
\end{itemize}

\texttt{specname} is

\begin{itemize}
\item \texttt{\_b}
\item \texttt{\_b[]}
\item \texttt{\_se}
\item \texttt{\_se[]}
\end{itemize}

\texttt{eqno} is

\begin{itemize}
\item \texttt{##}
\item \texttt{name}
\end{itemize}

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

Distinguish between [], which are to be typed, and [], which indicate optional arguments.
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.

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.

Reporting

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,

```
. statsby, by(mypannels) basepop(mypannels==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`.

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

```
command:  regress price weight length mpg
_b_weight:  _b[weight]
_b_length:  _b[length]
_b_mpg:  _b[mpg]
_b_cons:  _b[_cons]
by:  foreign
```

```
list

<table>
<thead>
<tr>
<th>foreign</th>
<th>_b_weight</th>
<th>_b_length</th>
<th>_b_mpg</th>
<th>_b_cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>6.767233</td>
<td>-109.9518</td>
<td>142.7663</td>
<td>2359.475</td>
</tr>
<tr>
<td>Foreign</td>
<td>4.784841</td>
<td>13.39052</td>
<td>-18.4072</td>
<td>-6497.49</td>
</tr>
</tbody>
</table>
```

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
    command:  regress price weight length mpg
               mpg:  _b[mpg]
               by:  foreign

. list

<table>
<thead>
<tr>
<th>foreign</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>142.7663</td>
</tr>
<tr>
<td>Foreign</td>
<td>-18.4072</td>
</tr>
</tbody>
</table>

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
    command:  regress price weight length mpg
               _se_weight:  _se[weight]
               _se_length:  _se[length]
               _se_mpg:  _se[mpg]
               _se_cons:  _se[_cons]
               by:  foreign

. list

<table>
<thead>
<tr>
<th>foreign</th>
<th>_se_weight</th>
<th>_se_length</th>
<th>_se_mpg</th>
<th>_se_cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domestic</td>
<td>1.226326</td>
<td>39.48193</td>
<td>134.7221</td>
<td>7770.131</td>
</tr>
<tr>
<td>Foreign</td>
<td>1.670006</td>
<td>50.70229</td>
<td>59.37442</td>
<td>6337.952</td>
</tr>
</tbody>
</table>

Example 2

For multiple-equation estimations, we can use [eqno]_b ([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 heckman and a slightly different dataset.

. use https://www.stata-press.com/data/r16/statsby, clear
. statsby _b, by(group) verbose nodots: heckman price mpg, sel(trunk)
    command:  heckman price mpg, sel(trunk)
               price_b_mpg:  [price]_b[mpg]
               price_b_cons:  [price]_b[_cons]
               select_b_tr-k:  [select]_b[trunk]
               select_b_cons:  [select]_b[_cons]
               _eq3_b_athrho:  [/]_b[athrho]
               _eq3_b_lnsi-a:  [/]_b[lnsigma]
               by:  group
. list, compress noobs

<table>
<thead>
<tr>
<th>group</th>
<th>price_b~g</th>
<th>price_s~s</th>
<th>select_k</th>
<th>select~s</th>
<th>eq3_b~o</th>
<th>eq3_b~a</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-253.9293</td>
<td>11836.33</td>
<td>-.0122223</td>
<td>1.248342</td>
<td>-.31078</td>
<td>7.895351</td>
</tr>
<tr>
<td>2</td>
<td>-242.5759</td>
<td>11906.46</td>
<td>-.0488969</td>
<td>1.943078</td>
<td>-1.399222</td>
<td>8.000272</td>
</tr>
<tr>
<td>3</td>
<td>-172.6499</td>
<td>9813.357</td>
<td>-.0190373</td>
<td>1.452783</td>
<td>-.3282423</td>
<td>7.876059</td>
</tr>
<tr>
<td>4</td>
<td>-250.7318</td>
<td>10677.31</td>
<td>.0525965</td>
<td>.3502012</td>
<td>.6133645</td>
<td>7.96349</td>
</tr>
</tbody>
</table>

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)

   command:  heckman price mpg, sel(trunk)
   price_b_mpg:  [price]_b[mpg]
   price_b_cons:  [price]_b[_cons]
   by:  group

. list

<table>
<thead>
<tr>
<th>group</th>
<th>price_b~g</th>
<th>price_s~s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-253.9293</td>
<td>11836.33</td>
</tr>
<tr>
<td>2</td>
<td>-242.5759</td>
<td>11906.46</td>
</tr>
<tr>
<td>3</td>
<td>-172.6499</td>
<td>9813.357</td>
</tr>
<tr>
<td>4</td>
<td>-250.7318</td>
<td>10677.31</td>
</tr>
</tbody>
</table>

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
```

<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>
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

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>74</td>
<td>6165.257</td>
<td>2949.496</td>
<td>3291</td>
<td>15906</td>
</tr>
</tbody>
</table>
```

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

Statsby groups
- running (summarize price) on group 1
. summarize price

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>52</td>
<td>6072.423</td>
<td>3097.104</td>
<td>3291</td>
<td>15906</td>
</tr>
</tbody>
</table>

running (summarize price) on group 2

. summarize price

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>price</td>
<td>22</td>
<td>6384.682</td>
<td>2621.915</td>
<td>3748</td>
<td>12990</td>
</tr>
</tbody>
</table>

. list

<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