### svy — The survey prefix command

#### Syntax

```
svy [vcetype] [ , svy_options eform_option ] : command
```

#### vcetype Description

<table>
<thead>
<tr>
<th>vcetype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE</td>
<td></td>
</tr>
<tr>
<td><strong>linearized</strong></td>
<td>Taylor-linearized variance estimation</td>
</tr>
<tr>
<td><strong>bootstrap</strong></td>
<td>bootstrap variance estimation; see [SVY] svy bootstrap</td>
</tr>
<tr>
<td><strong>brr</strong></td>
<td>BRR variance estimation; see [SVY] svy brr</td>
</tr>
<tr>
<td><strong>jackknife</strong></td>
<td>Jackknife variance estimation; see [SVY] svy jackknife</td>
</tr>
<tr>
<td><strong>sdr</strong></td>
<td>SDR variance estimation; see [SVY] svy sdr</td>
</tr>
</tbody>
</table>

Specifying a `vcetype` overrides the default from `svyset`.

#### svy_options Description

<table>
<thead>
<tr>
<th>if/in</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>subpop([varname] [if])</code></td>
<td>identify a subpopulation</td>
</tr>
</tbody>
</table>

SE

<table>
<thead>
<tr>
<th>dof(#)</th>
<th>design degrees of freedom</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bootstrap_options</strong></td>
<td>more options allowed with bootstrap variance estimation; see [SVY] bootstrap_options</td>
</tr>
<tr>
<td><strong>brr_options</strong></td>
<td>more options allowed with BRR variance estimation; see [SVY] brr_options</td>
</tr>
<tr>
<td><strong>jackknife_options</strong></td>
<td>more options allowed with jackknife variance estimation; see [SVY] jackknife_options</td>
</tr>
<tr>
<td><strong>sdr_options</strong></td>
<td>more options allowed with SDR variance estimation; see [SVY] sdr_options</td>
</tr>
</tbody>
</table>

Reporting

<table>
<thead>
<tr>
<th>level(#)</th>
<th>set confidence level; default is level(95)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nocnsreport</strong></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><strong>display_options</strong></td>
<td>control column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>noheader</th>
<th>suppress table header</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nolegend</strong></td>
<td>suppress table legend</td>
</tr>
<tr>
<td><strong>noadjust</strong></td>
<td>do not adjust model Wald statistic</td>
</tr>
<tr>
<td><strong>noisily</strong></td>
<td>display any output from command</td>
</tr>
<tr>
<td><strong>trace</strong></td>
<td>trace <code>command</code></td>
</tr>
<tr>
<td><strong>coeflegend</strong></td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>
svy requires that the survey design variables be identified using svyset; see [SVY] svyset.
mi estimate may be used with svy linearized if the estimation command allows mi estimate; it may not be
used with svy bootstrap, svy brr, svy jackknife, or svy sdr.
noheader, nolegend, noadjust, noisily, trace, and coeflegend are not shown in the dialog boxes for estimation
commands.
Warning: Using if or in restrictions will often not produce correct variance estimates for subpopulations. To compute
estimates for subpopulations, use the subpop() option.
See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

Description

svy fits statistical models for complex survey data. Typing

```
.svy: command
```

executes command while accounting for the survey settings identified by svyset.

command defines the estimation command to be executed. Not all estimation commands are
supported by svy. See [SVY] svy estimation for a list of Stata’s estimation commands that are
supported by svy. See [P] program properties for a discussion of what is required for svy to support
an estimation command. The by prefix may not be part of command.

Options

```
subpop(subpop)
```

specifies that estimates be computed for the single subpopulation identified by subpop, which is

```
[ varname ] [ if ]
```

Thus the subpopulation is defined by the observations for which varname ≠ 0 that also meet
the if conditions. Typically, varname = 1 defines the subpopulation, and varname = 0 indicates
observations not belonging to the subpopulation. For observations whose subpopulation status is
uncertain, varname should be set to a missing value; such observations are dropped from the
estimation sample.

See [SVY] subpopulation estimation and [SVY] estat.

```
dof(#)
```

specifies the design degrees of freedom, overriding the default calculation, df = N_{psu} - N_{strata}.
bootstrap_options are other options that are allowed with bootstrap variance estimation specified by svy
bootstrap or specified as svyset using the vce(bootstrap) option; see [SVY] bootstrap_options.

```
brr_options
```

are other options that are allowed with BRR variance estimation specified by svy brr or
specified as svyset using the vce(brr) option; see [SVY] brr_options.

```
jackknife_options
```

are other options that are allowed with jackknife variance estimation specified by svy
jackknife or specified as svyset using the vce(jackknife) option; see [SVY] jackknife_options.

```
sdr_options
```

are other options that are allowed with SDR variance estimation specified by svy sdr or
specified as svyset using the vce(sdr) option; see [SVY] sdr_options.
Reporting

level(\#) specifies the confidence level, as a percentage, for confidence intervals. The default is level(95) or as set by set level; see [U] 20.7 Specifying the width of confidence intervals.

nocnsreport; see [R] estimation options.

display_options: noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofilabel, fvwrap(\#), fvwrapposition(style), cformat(\%fmt), pformat(\%fmt), sformat(\%fmt), and nolstretch; see [R] estimation options.

The following options are available with svy but are not shown in the dialog boxes:

nolegend prevents the table header from being displayed. This option implies nolegend.

noadjust specifies that the model Wald test be carried out as $W/k \sim F(k, d)$, where $W$ is the Wald test statistic, $k$ is the number of terms in the model excluding the constant term, $d$ is the total number of sampled PSUs minus the total number of strata, and $F(k, d)$ is an $F$ distribution with $k$ numerator degrees of freedom and $d$ denominator degrees of freedom. By default, an adjusted Wald test is conducted: $(d - k + 1)W/(kd) \sim F(k, d - k + 1)$.

See Korn and Graubard (1990) for a discussion of the Wald test and the adjustments thereof. Using the noadjust option is not recommended.

noisily requests that any output from command be displayed.

trace causes a trace of the execution of command to be displayed.

ccoeflegend; see [R] estimation options.

The following option is usually available with svy at the time of estimation or on replay but is not shown in all dialog boxes:

eform_option; see [R] eform_option.

Remarks and examples

The svy prefix is designed for use with complex survey data. Typical survey design characteristics include sampling weights, one or more stages of clustered sampling, and stratification. For a general discussion of various aspects of survey designs, including multistage designs, see [SVY] svyset.

Below we present an example of the effects of weights, clustering, and stratification. This is a typical case, but drawing general rules from any one example is still dangerous. You could find particular analyses from other surveys that are counterexamples for each of the trends for standard errors exhibited here.

Example 1: The effects of weights, clustering, and stratification

We use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981) as our example. This is a national survey, and the dataset has sampling weights, strata, and clustering. In this example, we will consider the estimation of the mean serum zinc level of all adults in the United States.

First, consider a proper design-based analysis, which accounts for weighting, clustering, and stratification. Before we issue our svy estimation command, we set the weight, strata, and PSU identifier variables:
. use http://www.stata-press.com/data/r13/nhanes2f
. svyset psuid [pweight=finalwgt], strata(stratid)
    pweight: finalwgt
    VCE: linearized
    Single unit: missing
    Strata 1: stratid
    SU 1: psuid
    FPC 1: <zero>

We now estimate the mean by using the proper design-based analysis:

. svy: mean zinc
(running mean on estimation sample)
Survey: Mean estimation
Number of strata = 31 Number of obs = 9189
Number of PSUs = 62 Population size = 104176071
        Design df = 31

                      Linearized
                      Mean   Std. Err.  [95% Conf. Interval]
                      zinc   87.18207   .4944827    86.17356    88.19057

If we ignore the survey design and use mean to estimate the mean, we get

. mean zinc
Mean estimation Number of obs = 9189

                      Mean   Std. Err.  [95% Conf. Interval]
                      zinc   86.51518   .1510744    86.21904    86.81132

The point estimate from the unweighted analysis is smaller by more than one standard error than the proper design-based estimate. Also, design-based analysis produced a standard error that is 3.27 times larger than the standard error produced by our incorrect analysis.

Example 2: Halfway is not enough—the importance of stratification and clustering

When some people analyze survey data, they say, “I know I have to use my survey weights, but I will just ignore the stratification and clustering information.” If we follow this strategy, we will obtain the proper design-based point estimates, but our standard errors, confidence intervals, and test statistics will usually be wrong.
To illustrate this effect, suppose that we used the \texttt{svy: mean} procedure with \texttt{pweights} only.

\begin{verbatim}
. svyset [pweight=finalwgt]
pweight: finalwgt
VCE: linearized
Single unit: missing
Strata 1: <one>
  SU 1: <observations>
  FPC 1: <zero>

. svy: mean zinc
    (running mean on estimation sample)
Survey: Mean estimation
Number of strata =  1  Number of obs =  9189
Number of PSUs =  9189  Population size = 104176071
    Design df =  9188

<table>
<thead>
<tr>
<th></th>
<th>Linearized</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>zinc</td>
<td>87.18207</td>
<td>.1828747</td>
</tr>
</tbody>
</table>
<pre><code>                               | 86.82359  87.54054 |
</code></pre>
\end{verbatim}

This approach gives us the same point estimate as our design-based analysis, but the reported standard error is less than one-half the design-based standard error. If we accounted only for clustering and weights and ignored stratification in NHANES II, we would obtain the following analysis:

\begin{verbatim}
. svyset psuid [pweight=finalwgt]
pweight: finalwgt
VCE: linearized
Single unit: missing
Strata 1: <one>
  SU 1: psuid
  FPC 1: <zero>

. svy: mean zinc
    (running mean on estimation sample)
Survey: Mean estimation
Number of strata =  1  Number of obs =  9189
Number of PSUs =  2  Population size = 104176071
    Design df =  1

<table>
<thead>
<tr>
<th></th>
<th>Linearized</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>zinc</td>
<td>87.18207</td>
<td>.7426221</td>
</tr>
</tbody>
</table>
<pre><code>                               | 77.74616  96.61798 |
</code></pre>
\end{verbatim}

Here our standard error is about 50% larger than what we obtained in our proper design-based analysis.

\section*{Example 3}

Let’s look at a regression. We model zinc on the basis of age, weight, sex, race, and rural or urban residence. We compare a proper design-based analysis with an ordinary regression (which assumes independent and identically distributed error).
Here is our design-based analysis:

```
.svyset psuid [pweight=finalwgt], strata(stratid)
pweight: finalwgt
VCE: linearized
Single unit: missing
Strata 1: stratid
SU 1: psuid
FPC 1: <zero>
```

```
.svy: regress zinc age c.age#c.age weight female black orace rural
(running regress on estimation sample)
```

Survey: Linear regression

| Number of strata | 31 |
| Number of PSUs | 62 |
| Number of obs | 9189 |
| Population size | 104176071 |
| Design df | 31 |
| F( 7, 25) | 62.50 |
| Prob > F | 0.0000 |
| R-squared | 0.0698 |

| Coef. | Std. Err. | t | P>|t| | [95% Conf. Interval] |
|-------|-----------|---|-----|-----------------|
| age | -0.1701161 | 0.0844192 | -2.02 | 0.053 | -0.3422901 | 0.002058 |
| c.age#c.age | 0.0008744 | 0.0008655 | 1.01 | 0.320 | -0.0008907 | 0.0026396 |
| weight | 0.0535225 | 0.0139115 | 3.85 | 0.001 | 0.0251499 | 0.0818951 |
| female | -6.134161 | 0.4403625 | -13.93 | 0.000 | -7.032286 | -5.236035 |
| black | -2.881813 | 1.0759858 | -2.68 | 0.012 | -5.076244 | -0.687381 |
| orace | -4.118051 | 1.621121 | -2.54 | 0.016 | -7.424349 | -0.811758 |
| rural | -0.5386327 | 0.6171836 | -0.87 | 0.390 | -1.797387 | 0.720126 |
| _cons | 92.47495 | 2.228263 | 41.50 | 0.000 | 87.93038 | 97.01952 |

If we had improperly ignored our survey weights, stratification, and clustering (that is, if we had used the usual Stata `regress` command), we would have obtained the following results:

```
.regress zinc age c.age#c.age weight female black orace rural
```

<table>
<thead>
<tr>
<th>Source</th>
<th>SS</th>
<th>df</th>
<th>MS</th>
<th>Number of obs</th>
<th>9189</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>110417.827</td>
<td>7</td>
<td>15773.9753</td>
<td>Prob &gt; F</td>
<td>0.0000</td>
</tr>
<tr>
<td>Residual</td>
<td>1816535.3</td>
<td>9181</td>
<td>197.858111</td>
<td>R-squared</td>
<td>0.0573</td>
</tr>
<tr>
<td>Total</td>
<td>1926953.13</td>
<td>9188</td>
<td>209.724982</td>
<td>Root MSE</td>
<td>14.066</td>
</tr>
</tbody>
</table>

| Coef. | Std. Err. | t | P>|t| | [95% Conf. Interval] |
|-------|-----------|---|-----|-----------------|
| age | -0.090298 | 0.0638452 | -1.41 | 0.157 | -0.2154488 | 0.0348528 |
| c.age#c.age | -0.000324 | 0.0006788 | -0.05 | 0.962 | -0.0013631 | 0.0012983 |
| weight | -0.0324226 | 0.0399565 | -0.08 | 0.939 | -0.1012891 | 0.0364441 |
| female | -4.134161 | 0.4403625 | -9.38 | 0.000 | -5.032286 | -3.236035 |
| black | -2.881813 | 1.0759858 | -2.68 | 0.012 | -5.076244 | -0.687381 |
| orace | -4.118051 | 1.621121 | -2.54 | 0.016 | -7.424349 | -0.811758 |
| rural | -0.5386327 | 0.6171836 | -0.87 | 0.390 | -1.797387 | 0.720126 |
| _cons | 89.49465 | 1.477528 | 60.57 | 0.000 | 86.59836 | 92.39093 |
The point estimates differ by 3%–100%, and the standard errors for the proper designed-based analysis are 30%–110% larger. The differences are not as dramatic as we saw with the estimation of the mean, but they are still substantial.

Stored results

`svy` stores the following in `e()`:

Scalars

- `e(N)` number of observations
- `e(N_sub)` subpopulation observations
- `e(N_strata)` number of strata
- `e(N_strata_omit)` number of strata omitted
- `e(singleton)` 1 if singleton strata, 0 otherwise
- `e(census)` 1 if census data, 0 otherwise
- `e(F)` model $F$ statistic
- `e(df_m)` model degrees of freedom
- `e(df_r)` variance degrees of freedom
- `e(N_pop)` estimate of population size
- `e(N_subpop)` estimate of subpopulation size
- `e(N_psu)` number of sampled PSUs
- `e(stages)` number of sampling stages
- `e(k_eq)` number of equations in $e(b)$
- `e(k_aux)` number of ancillary parameters
- `e(p)` $p$-value
- `e(rank)` rank of $e(V)$

Macros

- `e(prefix)` `svy`
- `e(cmdname)` command name from `command`
- `e(cmd)` same as `e(cmdname)` or `e(vce)`
- `e(command)` `command`
- `e(cmdline)` command as typed
- `e(wtype)` weight type
- `e(wexp)` weight expression
- `e(wvar)` weight variable name
- `e(singleunit)` `singleunit()` setting
- `e(strata)` `strata()` variable
- `e(strata#)` variable identifying strata for stage #
- `e(psu)` `psu()` variable
- `e(su#)` variable identifying sampling units for stage #
- `e(fpc)` `fpc()` variable
- `e(fpc#)` FPC for stage #
- `e(title)` title in estimation output
- `e(poststrata)` `poststrata()` variable
- `e(postweight)` `postweight()` variable
- `e(vce)` `vcetype` specified in `vce()`
- `e(vcetype)` title used to label Std. Err.
- `e(mse)` `mse`, if specified
- `e(subpop)` `subpop` from `subpop()`
- `e(adjust)` `noadjust`, if specified
- `e(properties)` `b V`
- `e(estat_cmd)` program used to implement `estat`
- `e(predict)` program used to implement `predict`
- `e(marginsnotok)` predictions disallowed by `margins`

Matrices

- `e(b)` estimates
- `e(V)` design-based variance
- `e(V_srs)` simple-random-sampling-without-replacement variance, $\hat{V}_{srswor}$
- `e(V_srs_sub)` subpopulation simple-random-sampling-without-replacement variance, $\hat{V}_{srswor}$ (created only when `subpop()` is specified)
svy — The survey prefix command

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>e(V)srswr)</code></td>
<td>simple-random-sampling-with-replacement variance, (\hat{V}_{srswr}) (created only when <code>fpc()</code> option is <code>svyset</code>)</td>
</tr>
<tr>
<td><code>e(V)srssubwr)</code></td>
<td>subpopulation simple-random-sampling-with-replacement variance, (\hat{V}_{srswr}) (created only when <code>subpop()</code> is specified)</td>
</tr>
<tr>
<td><code>e(V)modelbased)</code></td>
<td>model-based variance</td>
</tr>
<tr>
<td><code>e(V)msp)</code></td>
<td>variance from misspecified model fit, (\hat{V}_{msp})</td>
</tr>
<tr>
<td><code>e(N_strata_single)</code></td>
<td>number of strata with one sampling unit</td>
</tr>
<tr>
<td><code>e(N_strata_certain)</code></td>
<td>number of certainty strata</td>
</tr>
<tr>
<td><code>e(N_strata)</code></td>
<td>number of strata</td>
</tr>
</tbody>
</table>

Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>e(sample)</code></td>
<td>marks estimation sample</td>
</tr>
</tbody>
</table>

svy also carries forward most of the results already in `e()` from `command`.

Methods and formulas

See [SVY] variance estimation for all the details behind the point estimate and variance calculations made by svy.

References


Also see

- [SVY] svy estimation — Estimation commands for survey data
- [SVY] svy postestimation — Postestimation tools for svy
- [SVY] svy bootstrap — Bootstrap for survey data
- [SVY] svy brr — Balanced repeated replication for survey data
- [SVY] svy jackknife — Jackknife estimation for survey data
- [SVY] svy sdr — Successive difference replication for survey data
- [SVY] svyset — Declare survey design for dataset
- [P] _robust — Robust variance estimates
- [U] 20 Estimation and postestimation commands
- [SVY] poststratification — Poststratification for survey data
- [SVY] subpopulation estimation — Subpopulation estimation for survey data
- [SVY] variance estimation — Variance estimation for survey data
- [P] program properties — Properties of user-defined programs