adjustfor_option — Adjust survivor and related functions for covariates at specific values

Description Quick start Syntax Options
Remarks and examples Also see

Description

The sts, stphplot, and stintphplot commands support the adjustfor(*varlist*) option to adjust the estimates of the survivor and related functions for covariates in *varlist* at specific values. You can adjust the functions to overall means, group-specific means, and zero covariate values. And you can use the at() option, specified within adjustfor() with sts, stphplot, or stintphplot and directly with stcurve, to adjust for specific covariate values and other summary statistics.

Quick start

Adjust for average values of covariates x1, x2, and factor variable x3

```
st_command ..., ... adjustfor(x1 x2 i.x3, atmeans)
```

Adjust for x1 = 0, x2 = 0, and the base level of x3

```
st_command ..., ... adjustfor(x1 x2 i.x3, atzeros)
```

Adjust for x1 = 3, x2 = 4, and x3 = 1

```
st_command ..., ... adjustfor(x1 x2 i.x3, at(x1=3 x2=4 x3=1))
```

Adjust for the means of x1, x2, and the base level of factor variable x3

```
st_command ..., ... adjustfor(x1 x2 i.x3, atmeans atbase)
```

The same as above, but using the at() specification

```
st_command ..., ... adjustfor(x1 x2 i.x3, at((mean) x1 x2 (base) x3))
```

The same as above, but for stcurve

```
stcurve, survival at((mean) x1 x2 (base) x3)
```

Adjust for group-specific means of covariates x1 and x2

```
st_command ..., ... adjustfor(x1 x2, atmeans) by(group)
```

Adjust for overall means of covariates x1 and x2

```
st_command ..., ... adjustfor(x1 x2, atomeans) by(group)
```

Syntax

Adjust for covariates in sts commands, stphplot, and stintphplot

st_command ... [, ... `adjustfor(`*varlist* [, *suboptions*]`)` ...]

where *st_command* is one of `sts graph`, `sts list`, `sts generate`, `stphplot`, or `stintphplot`.

Adjust for specific covariate values in stcurve

`stcurve` ... [, ... `at(`*atspec*`)` ...]

suboptions	Description
`atmeans`	adjust for *varlist* at group-specific means
`atomeans`	adjust for *varlist* at overall means
`atzeros`	adjust for *varlist* at zeros
`atbase`	adjust for factor variables in *varlist* at base levels
`at(`*atspec*`)`	adjust for *varlist* at specified values

`at()` overrides `atmeans`, `atomeans`, `atzeros`, and `atbase`; see *Syntax of at()*.

`atomeans` is the default for `stphplot` and `stintphplot`, and `atzeros` is the default for `sts graph`, `sts list`, and `sts generate`.

For `stcurve`, `at()` is a standalone option and can be repeated.

Options

`atmeans` adjusts the estimates of the survivor and related functions to group-specific means of *varlist*. It may not be combined with `atomeans` or `atzeros`. `atmeans` is equivalent to `atomeans` when option `by()` is not specified with *st_command*.

`atomeans` adjusts the estimates of the survivor and related functions to the overall means of *varlist*. It may not be combined with `atmeans` or `atzeros`.

`atzeros` adjusts the estimates of the survivor and related functions to zero values of continuous variables in *varlist* and base levels of factor variables. It may not be combined with `atomeans` or `atmeans`.

`atbase` adjusts the estimates of the survivor and related functions to the base levels of the factor variables in *varlist*. Without this option, the factor variables are set to their mean values, unless `atzeros` is specified or assumed by the command.

`at(`*atspec*`)` adjusts the estimates of the survivor and related functions to the specified values of covariates in *varlist*. `at()` may be used for continuous or factor covariates. With `stcurve`, multiple `at()` options can be specified, and each will produce a different set of estimates.

`at(x1=20)` uses the specified value of covariate `x1`.

`at(x1=20 x2=1)` uses the specified values of covariates `x1` and `x2`.

`at(x1=(20 30 40 50))` (`stcurve` only) first uses the value of 20 for `x1`, then the value of 30, and so on. `stcurve` produces separate results for each specified value.

`at(x1=(20(10)50))` (`stcurve` only) does the same as `at(x1=(20 30 40 50))`; that is, you may specify a numlist.

at((mean) x1 (median) x2) uses the specified summary statistics as the values for x1 and x2. at((p25) _all) uses the respective 25th percentile values for all covariates. See *Syntax of at()* for the full list of summary-statistic modifiers.

at((mean) _all (median) x x2=1.2 z=(1 2 3)) is processed from general to specific, with settings for named covariates overriding general settings specified via _all. Thus, all covariates are set to their means except for x (set to its median), x2 (set to 1.2), and z (set to 1, then to 2, and finally to 3).

See *Syntax of at()* for more information.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Using adjustfor() with sts, stphplot, and stintphplot
Syntax of at()

For examples of using the adjustfor() option, see *Covariate-adjusted estimates* in [ST] **sts**, example 1 in [ST] **stcox PH-assumption tests**, and example 3 in [ST] **stintcox PH-assumption plots**. For examples of specifying at(), see *Using at() with stcurve* in [ST] **stcurve**.

Using adjustfor() with sts, stphplot, and stintphplot

When you specify the adjustfor(*varlist*) option with sts, stphplot, or stintphplot, the command fits a Cox model with *varlist* as predictors and computes the survivor (or related) function for the specified values of *varlist*. By default, sts adjusts to zero covariate values (computes the baseline functions), and stphplot and stintphplot adjust to overall means. That is, when adjustfor(*varlist*) is specified, sts assumes adjustfor(*varlist*, atzeros), and stphplot and stintphplot assume adjustfor(*varlist*, atomeans).

For instance, the following two commands are producing the same "log–log" plots.

```
. stphplot, by(group) adjustfor(x1 x2)
. stphplot, by(group) adjustfor(x1 x2, atomeans)
```

To use group-specific means instead of the overall means, we could specify

```
. stphplot, by(group) adjustfor(x1 x2, atmeans)
```

To use zero values, we could specify

```
. stphplot, by(group) adjustfor(x1 x2, atzeros)
```

As we mentioned earlier, sts or, more specifically, sts graph, sts list, and sts generate adjust the function estimates to zero covariate values by default. For instance, the following two commands plot the same baseline survivor function.

```
. sts graph, by(drug) adjustfor(age)
. sts graph, by(drug) adjustfor(age, atzeros)
```

If we wanted the survivor curves adjusted for age and scaled to age 50, we could generate a new variable centered at 50 and adjust the baseline survivor curve to that new variable:

```
. generate double age50 = age-50
. sts graph, by(drug) adjustfor(age50)
```

More conveniently, we could simply use the at() suboption of the adjustfor() option to adjust for age 50:

```
. sts graph, by(drug) adjustfor(age, at(age=50))
```

Syntax of at()

The `at()` option can be specified within the `adjustfor()` option with the `sts`, `stphplot`, and `stintphplot` commands. It can also be used directly with `stcurve`.

In `at(`*atspec*`)`, *atspec* may contain one or more of the following specifications,

varname `=` *#*

varname `= (`*numlist*`)`   (`stcurve` only)

`_frailty = (`*numlist*`)`   (`stcurve` only)

`(`*stat*`)` *varlist*

where

1. *varname* and *varlist* must contain covariates from *varlist* specified in `adjustfor()` or, for `stcurve`, *varlist* specified with the survival regression model.
2. variables (whether in *varname* or *varlist*) may be continuous variables, factor variables, or specific-level variables, such as `age`, `group`, or `3.group`.
3. *varlist* may also be one of three standard lists:
   a. `_all` (all covariates);
   b. `_factor` (all factor-variable covariates); or
   c. `_continuous` (all continuous covariates).
4. *stat* can be any of the following:

stat	Description	Variables allowed
`mean`	means	all
`median`	medians	continuous
`p1`	1st percentile	continuous
`p2`	2nd percentile	continuous
`...`	3rd–49th percentiles	continuous
`p50`	50th percentile (same as `median`)	continuous
`...`	51st–97th percentiles	continuous
`p98`	98th percentile	continuous
`p99`	99th percentile	continuous
`min`	minimums	continuous
`max`	maximums	continuous
`zero`	zeros	all
`base`	base level	factors

   Any *stat* except `zero` and `base` may be prefixed with an `o` to get the overall statistic such as `omean`, `omedian`, and `op25`. Overall statistics differ from their correspondingly named statistics only when the `by()` option is specified. For factor variables, `zero` is a synonym for `base`. If *stat* is not followed by a varlist, *stat* is ignored.

5. `_frailty` is allowed only in the `at()` option with `stcurve` after fitting a shared-frailty model with `stcox`. When `_frailty` is specified, frailties are set to the values in the corresponding numlist. When `_frailty` is not specified, frailties are set to 1.

atspec can involve settings for multiple covariates and, with `stcurve`, multiple settings for one covariate. The following rules are applied when more than one covariate or value is included:

1. When more than one covariate is referenced in *atspec* but each covariate is set to only one value, all settings are applied in combination. For example, `at(x1=5 x2=0)` results in one scenario, with `x1` set to 5, `x2` set to 0, and all other covariates set to their defaults.
2. When multiple values are specified for a covariate, the covariate will be set to each of the values in turn. For example, `at(x1=5 x1=10)` or, equivalently, `at(x1=(5 10))` specifies that `x1` be set first to 5 and then to 10. This is allowed only with the `stcurve` command.
3. When multiple values are specified for more than one covariate, all possible combinations of settings are applied in turn. For example, `at(x1=(5 10) x2=(0 1))` results in four scenarios: `x1` = 5 and `x2` = 0; `x1` = 5 and `x2` = 1; `x1` = 10 and `x2` = 0; and `x1` = 10 and `x2` = 1. This is allowed only with the `stcurve` command.
4. Settings may be specified for groups of covariates using three general varlists—`_all`, `_factor`, and `_continuous`. When *atspec* includes both specifications with general varlists and specifications with named covariates, the specifications for named covariates take precedence over general ones. For example, `at(x1=10 (means) _all)` sets `x1` to 10 while setting all other covariates to their means.
5. Only one (*stat*) *varlist* specification can be applied to a covariate. If more than one is specified, the rightmost specification is respected. For example, `at((means) x1 x2 (medians) x1 x2)` sets both `x1` and `x2` to their medians.
6. When both a (*stat*) specification and another specification are included for a named covariate, the other specification takes precedence. For example, `at(x1=5 (means) x1)` sets `x1` to 5.

In addition, with the `stcurve` command, `at()` can be repeated. When multiple `at()` options are specified, *atspec*s are processed sequentially. For instance, `at(x1=5) at(x2=0)` results in `stcurve` producing two curves. The first sets `x1` to 5 and all other covariates, including `x2`, to their means. The second sets `x2` to 0 and all other covariates to their means. Note that this is different from the single `at(x1=5 x2=0)` specification, which sets `x1` and `x2` to the specified values simultaneously.

Also see

[ST] **stcox PH-assumption tests** — Tests of proportional-hazards assumption after stcox

[ST] **stcurve** — Plot the survivor or related function after streg, stcox, and more

[ST] **sts** — Generate, graph, list, and test the survivor and related functions

[ST] **sts generate** — Create variables containing survivor and related functions

[ST] **sts graph** — Graph the survivor or related function

[ST] **sts list** — List the survivor or related function

For suggested citations, see the FAQ on citing Stata documentation.