ctset — Declare data to be count-time data

Description

ct refers to count-time data and is described here and in [ST] **ct**. Do not confuse count-time data with counting-process data, which can be analyzed using the st commands; see [ST] **st**.

When specified with a *timevar* and *nfailvar*, `ctset` declares the data in memory to be ct data. When you `ctset` your data, `ctset` also checks that what you have declared makes sense.

`ctset, noshow` will suppress display of the identities of the key ct variables before the output of other ct commands. By default, this information is shown. If you type `ctset, noshow` and then wish to restore the default behavior, type `ctset, show`.

`ctset, clear` is used mostly by programmers and causes Stata to no longer consider the data to be ct data. The dataset itself remains unchanged. It is not necessary to type `ctset, clear` before doing another `ctset`.

`ctset` typed without arguments—which can be abbreviated `ct`—displays the identities of the key ct variables and reruns the checks on your data. Thus `ct` can remind you of what you have `ctset` (especially if you have `ctset, noshow`) and reverify your data if you make changes to the data.

Quick start

Declare count-time data with number of failures, `fail`, at each time in `tvar`

```
ctset tvar fail
```

Same as above, and specify the number censored, `cens`, at each time

```
ctset tvar fail cens
```

Same as above, and specify the number entering, `enter`, at each time

```
ctset tvar fail cens enter
```

Specify that the number of failures and the number censored are recorded for groups identified by `v1`

```
ctset tvar fail cens, by(v1)
```

Display previous `ct` settings, and verify that any changes to data correspond to settings

```
ctset
```

Do not display information on variables specified in `ctset` when `ct` commands are run

```
ctset, noshow
```

Menu

Statistics > Survival analysis > Setup and utilities > Declare data to be count-time data

Syntax

Declare data in memory to be count-time data and run checks on data

```
ctset timevar nfailvar [ncensvar [nentvar]] [, by(varlist) noshow]
```

Specify whether to display identities of key ct variables

```
ctset, {show | noshow}
```

Clear ct setting

```
ctset, clear
```

Display identity of key ct variables and rerun checks on data

```
{ctset | ct}
```

where *timevar* refers to the time of failure, censoring, or entry. It should contain times ≥ 0.

nfailvar records the number failing at time *timevar*.

ncensvar records the number censored at time *timevar*.

nentvar records the number entering at time *timevar*.

Stata sequences events at the same time as

at *timevar*	*nfailvar* failures occurred,
then at *timevar* + 0	*ncensvar* censorings occurred, and
finally at *timevar* + 0 + 0	*nentvar* subjects entered the data.

Options

`by(`*varlist*`)` indicates that counts are provided by group. For instance, consider data containing records such as

```
t      fail      cens       sex     agecat
5        10         2         0          1
5         6         1         1          1
5        12         0         0          2
```

These data indicate that, in the category `sex` $= 0$ and `agecat` $= 1$, 10 failed and 2 were censored at time 5; for `sex` $= 1$, 1 was censored and 6 failed; and so on.

The above data would be declared

```
. ctset t fail cens, by(sex agecat)
```

The order of the records is not important, nor is it important that there be a record at every time for every group or that there be only one record for a time and group. However, the data must contain the full table of events.

`show` and `noshow` specify whether the identities of the key ct variables are to be displayed at the start of every ct command. Some users find the report reassuring; others find it repetitive. In any case, you can set and unset `show`, and you can always type `ct` to see the summary.

`clear` makes Stata no longer consider the data to be ct data.

Remarks and examples

Remarks are presented under the following headings:

Examples

About all you can do with ct data in Stata is convert it to survival-time (st) data so that you can use the survival analysis commands. To analyze count-time data with Stata,

```
. ctset ...
. cttost
. (now use any of the st commands)
```

▷ Example 1: Simple ct data

We have data on generators that are run until they fail:

```
. use https://www.stata-press.com/data/r19/ctset1
. list, sep(0)
```

	failtime	fail
1.	22	1
2.	30	1
3.	40	2
4.	52	1
5.	54	4
6.	55	2
7.	85	7
8.	97	1
9.	100	3
10.	122	2
11.	140	1

For instance, at time 54, four generators failed. To `ctset` these data, we could type

```
. ctset failtime fail
Count-time data settings
          Time: failtime
      Failures: fail
   Number lost: <none>
Number entered: All enter at time 0
```

It is not important that there be only 1 observation per failure time. For instance, according to our data, at time 85 there were seven failures. We could remove that observation and substitute two in its place—one stating that at time 85 there were five failures and another that at time 85 there were two more failures. `ctset` would interpret that data just as it did the previous data.

In more realistic examples, the generators might differ from one another. For instance, the following data show the number failing with old-style and new-style bearings:

```
. use https://www.stata-press.com/data/r19/ctset2
. list, sepby(bearings)
```

	bearings	failtime	fail
1.	Old-style	22	1
2.	Old-style	40	2
3.	Old-style	54	1
4.	Old-style	84	2
5.	Old-style	97	2
6.	Old-style	100	1
7.	New-style	30	1
8.	New-style	52	1
9.	New-style	55	1
10.	New-style	100	3
11.	New-style	122	2
12.	New-style	140	1

That the data are sorted on bearings is not important. The `ctset` command for these data is

```
. ctset failtime fail, by(bearings)
Count-time data settings
          Time: failtime
      Failures: fail
   Number lost: <none>
Number entered: All enter at time 0
Group variable: bearings
```

◁

▷ Example 2: ct data with censoring

In real data, not all units fail in the time allotted. Say that the generator experiment was stopped after 150 days. The data might be

```
. use https://www.stata-press.com/data/r19/ctset3
. list
```

	bearings	failtime	fail	censored
1.	Old-style	22	1	0
2.	Old-style	40	2	0
3.	Old-style	54	1	0
4.	Old-style	84	2	0
5.	New-style	97	2	0
6.	Old-style	100	1	0
7.	Old-style	150	0	2
8.	New-style	30	1	0
9.	New-style	52	1	0
10.	New-style	55	1	0
11.	New-style	122	2	0
12.	New-style	140	1	0
13.	New-style	150	0	3

The ctset command for these data is

```
. ctset failtime fail censored, by(bearings)
Count-time data settings
          Time: failtime
      Failures: fail
   Number lost: censored
Number entered: All enter at time 0
Group variable: bearings
```

In some other data, observations might also be censored along the way; that is, the value of censored would not be 0 before time 150. For instance, a record might read

```
bearings   failtime       fail   censored
       0         84          2          1
```

This would mean that at time 84, two failed and one was lost because of censoring. The failure and censoring occurred at the same time, and when we analyze these data, Stata will assume that the censored observation could have failed, that is, that the censoring occurred after the two failures.

◁

▷ Example 3: ct data with delayed entry

Data on survival time of patients with a particular kind of cancer are collected. Time is measured as time since diagnosis. After data collection started, the sample was enriched with some patients from hospital records who had been previously diagnosed. Some of the data are

time	die	cens	ent	*other variables*
0	0	0	50	
1	0	0	5	...
⋮				
30	0	0	3	...
31	0	1	2	...
32	1	0	1	...
⋮				
100	1	1	0	...
⋮				

Fifty patients entered at time 0 (time of diagnosis); five patients entered 1 day after diagnosis; and three, two, and one patients entered 30, 31, and 32 days after diagnosis, respectively. On the 32nd day, one of the previously entered patients died.

If the other variables are named sex and agecat, the ctset command for these data is

```
. ctset time die cens ent, by(sex agecat)
Count-time data settings
           Time: time
       Failures: die
    Number lost: cens
 Number entered: ent
Group variables: sex agecat
```

◁

The count-time format is an inferior way to record data like these—data in which every subject does not enter at time 0—because some information is already lost. When did the patient who died on the 32nd day enter? There is no way of telling.

For traditional survival analysis calculations, it does not matter. More modern methods of estimating standard errors, however, seek to identify each patient, and these data do not support using such methods.

This issue concerns the robust estimates of variance and the `vce(robust)` options on some of the st analysis commands. After converting the data, you must not use the `vce(robust)` option, even if an st command allows it, because the identities of the subjects—tying together when a subject starts and ceases to be at risk—are assigned randomly by `cttost` when you convert your ct to st data. When did the patient who died on the 32nd day enter? For conventional calculations, it does not matter, and `cttost` chooses a time randomly from the available entry times.

Data errors flagged by ctset

`ctset` requires only two things of your data: that the counts all be positive or zero and, if you specify an entry variable, that the entering and exiting subjects (failure + censored) balance.

If all subjects enter at time 0, we recommend that you do not specify a number-that-enter variable. `ctset` can determine for itself the number who enter at time 0 by summing the failures and censorings.

Also see

[ST] **ct** — Count-time data

[ST] **cttost** — Convert count-time data to survival-time data