Introduction to the `stset` command

Paul C. Lambert
Centre for Biostatistics and Genetic Epidemiology
Department of Health Sciences
University of Leicester
Leicester, UK
paul.lambert@le.ac.uk

December 2006

1 Survival analysis using Stata

1.1 What is the `stset` command?

The `stset` command is used to tell Stata the format of your survival data. You only have to 'tell' Stata once after which all survival analysis commands (the `st` commands) will use this information. For example, after using `stset`, a Cox proportional hazards model with age and sex as covariates can be fitted using

```
. stcox age sex
```

At a minimum Stata needs to know the time at risk (e.g., time from diagnosis to death or censoring) and the failure indicator (e.g., whether or not the patient died). However, the `stset` command is very flexible and powerful for setting up more complicated survival data. I will explain the use of the `stset` command through a number of examples.

1.2 Syntax of the `stset` command

`stset` *timevar* `[if] [weight] , failure(failvar[==numlist]) [`*options*`]`

For example,

```
stset survtime, failure(dead==1)
```

would be appropriate if the time at risk for each individual is in the variable `survtime` and the variable `dead` is an indicator for death.

- The *timevar* variable is compulsory. It is the survival time (or a date) of the event/censoring time.

- The `failure(`*failvar = numlist*`)` option is optional, but it is good practice to always use it. If this option is omitted then it is assumed that all subjects experience the event. It is a number list (`numlist` giving the values indicating a failure. In many cases this will be a single number, but the use of a number list is useful if, for example, you have different codings for different causes of death.

- The `exit` option gives the latest time at which the subject is at risk. The default is `exit(failure)`, i.e. the subject is removed from the risk set after their event. This command is useful if you want to restrict follow-up time. For example if you are using dates to define your survival times, but you want to restrict follow-up time to 31/12/2005, you can use `exit(time mdy(12,31,2005))`. If you have multiple failures then you need to specify `exit(time .)` as the default is to remove the subject from the risk set after their first failure.

- The `origin` option gives the time origin of the time-scale, that is, it is used to define when time is zero. The default is zero. For example, if we have variables representing date of diagnosis and date of exit and wish to analyse time since diagnosis then the time origin should be defined as the date of diagnosis (since the day of diagnosis is time zero for each individual). Similarly, if we wish to use attained age as the timescale then the time origin is the date of birth.

- The `enter` option gives the time at which the subject becomes at risk. You are likely to use this option if using age as the time scale. For example, if there is a date of diagnosis then you will use `enter(datediag)`. It is also useful if patients are only considered to be at risk after a certain date (e.g., in period analysis). For example, if we only want to consider time at risk after 1/1/2001 use `enter(time mdy(1,1,2001))`.

- The `scale(#)` option transforms the survival time. For example to transform the timescale from days to years use `scale(365.25)`.

- The `id(`*varname*`)` option specifies an identification number for each subject. This option is not compulsory, but it is good practice to specify it as the `stsplit` command requires an ID variable. If there are multiple failures the the `id` option must be specified.

The above are the most common options - see the manual or online help for other options.

1.3 Variables created by the `stset` command

The `stset` command creates 4 variables. These variables contain all the necessary information for the survival data. These variables are

`_t0`	-	analysis time when record begins (time at which individual becomes at risk)
`_t`	-	analysis time when record ends (time at which individual stops being at risk)
`_d`	-	failure indicator: 1 if failure, 0 if censored
`_st`	-	1 if the record is included in `st` analyses, 0 if excluded

All the survival analysis (`st`) commands use these variables, as all information regarding survival times is contained within these four variables.

1.4 Examples of using `stset`

I will use an example data set to illustrate how to use the `stset` command. This consists of three subjects where dates of birth, diagnosis, event (death) and treatment change are known. The data is listed below

```
. list, noobs ab(10) linesize(200)
```

id	event	datebirth	datediag	dateexit	datetreat	survdays	survyears
1	0	27mar1969	18jun2000	31dec2006	05jul2002	2387	6.53525
2	1	05sep1975	16apr1999	03jun2004	06sep2000	1875	5.13347
3	1	13feb1974	02nov2001	19jan2005	.	1174	3.214237

One subject did not change treatment and `datetreat` is recorded as missing for this subject.

The variables are as follows;

`id`	-	identification number
`event`	-	event indicator (0 = censored, 1 = dead)
`datebirth`	-	date of birth
`datediag`	-	date of diagnosis
`dateexit`	-	date of death/censoring
`datetreat`	-	date of change in treatment
`survdays`	-	survival time in days (`dateexit - datediag`)
`survyears`	-	survival time in years (`(dateexit - datediag)/365.25`)

The variables `survdays` and `survyears` were calculated using

```
. gen survdays = dateexit - datediag
. gen survyears = survdays/365.25
```

The `datetreat` variable will be used to demonstrate how to incorporate time-dependent covariates in an analysis.

1.4.1 'Standard' survival data

If the survival time and censoring indicator have already been created then `stset` can be used as follows

```
        . stset survyears, failure(event == 1) id(id)
                id:  id
     failure event:  event == 1
obs. time interval:  (survyears[_n-1], survyears]
 exit on or before:  failure
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 14.88296  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =   6.53525
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	6.5352497	0	1
2	0	5.1334701	1	1
3	0	3.2142367	1	1

The `id` option is not compulsory here as there should only be one row of data per subject. However, it is good practice to include it, as if splitting the data later using `stsplit` then the data must previously have been `stset` using the `id` option.

The output gives some summary information. You should check this output to see if there are any exclusions (e.g. for zero or negative survival times), that the number of events corresponds to what you expect etc.

The `stset` command has created four new variables. For this example `_t0` is 0 for all subjects; this is the default value (we have not used the enter option) and corresponds to all subjects being at risk from time 0, i.e., when they are diagnosed. The variable `_t` gives the survival or censoring time, i.e. when the subject stops being at risk due to death or censoring. The `_d` variable is the event indicator (0 if censored and 1 if an event). The `_st` variable specifies whether the observation should be included in the analysis (1 = include, 0 = exclude). `_st` will be zero if survival times are recorded as zero (or are negative) or if an `if` or `in` option was specified in the `stset` command.

1.4.2 Using the `scale` option

If survival time is measured in days and you would like the analysis time to be in years then use the `scale` option. For example

```
. stset survdays, failure(event == 1) id(id) scale(365.25)
                id:  id
     failure event:  event == 1
obs. time interval:  (survdays[_n-1], survdays]
 exit on or before:  failure
    t for analysis:  time/365.25
--------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
--------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 14.88296  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =   6.53525
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	6.5352498	0	1
2	0	5.1334702	1	1
3	0	3.2142368	1	1

The survival time (in days) is divided by 365.25 to give survival time in years. This is noted in the output from the `stset` command.

The variables created by `stset` (`_t0 _t _d _st`) are exactly the same as the previous example. This is to be expected as the `survyears` variable was calculated in same way as used by `stset`. It is usually safer to let `stset` to do the rescaling for you. There are other advantages, for example when using the `stsplit` command you are able to specify some options that need to remember that you have rescaled the data.

1.4.3 Using date of diagnosis and date of exit

It is common to have data that record various dates. For example, the date of diagnosis of a particular disease, the date of death or end of follow-up, the date of birth or the date patients were given particular treatments. It is of course fairly easy to use any package to calculate various times from these dates, but the `stset` command can do most of this work for you.

It is important to note that Stata records dates as the number of days from 1 January 1960 and you need to ensure that you have either read in or converted your dates to this format. I usually either read the date in as a string (e.g. "27/3/1969") and then use the `date` function, i.e.,

```
. gen datediag = date(sdatediag, "dmy")
```

or I read in the the day, month and year separately and use the `mdy` function, i.e.,

```
. gen datediag = mdy(monthdiag, daydiag, yeardiag)
```

When using dates you need to make use of the `origin` option. If you do not do this then the time origin will be 1/1/1960. The `stset` command is as follows,

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag)
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 exit on or before:  failure
    t for analysis:  (time-origin)
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
     5436  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =      2387
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	2387	0	1
2	0	1875	1	1
3	0	1174	1	1

In the output from `stset` it is reported that `t for analysis: time - origin`, which is what we want. As the dates are stored in units of days, the analysis time is also in units of days. If we want to have our analysis time in units of years then we need to use the `scale` option.

1.4.4 Using date of diagnosis and date of exit with the scale option

By adding the `scale` option we can transform the analysis time to units of years, which is usually easier for interpretation.

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag) scale(365.25)
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 exit on or before:  failure
    t for analysis:  (time-origin)/365.25
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 14.88296  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =   6.53525
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	6.5352498	0	1
2	0	5.1334702	1	1
3	0	3.2142368	1	1

Note that the variables created by `stset` (`_t0 _t _d _st`) are exactly the same as in sections 1.4.1 and 1.4.2.

1.4.5 Restricting the follow-up time

In some instances it may be necessary to define the maximum follow-up time. This may be because follow-up information after a certain date may be unreliable. Alternatively, you may only be interested in follow-up to a certain time after diagnosis. For example, if there are only a few individuals alive after five years, you may want to restrict follow-up to 5 years.

In the following example the censoring date is 31/12/2005 and anyone still alive at this date will be censored at this time. We need to use the `mdy` function with the `exit` option.

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag) scale(365.25) exi
> t(time mdy(12,31,2005))
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 exit on or before:  time mdy(12,31,2005)
    t for analysis:  (time-origin)/365.25
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 13.88364  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =  5.535934
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	5.5359343	0	1
2	0	5.1334702	1	1
3	0	3.2142368	1	1

The option `exit(time mdy(12,31,2005))` truncates the time scale at this date. This affects subject 1 who had a censoring data of 31/12/2006, so their survival time has been reduced by a year. The other two individuals are unaffected as they were not at risk at this date, as they had already experienced an event.

If we are interested in restricting the follow-up time to 5 years then we can use

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag) scale(365.25) exi
> t(time datediag + 365.25*5)
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 exit on or before:  time datediag + 365.25*5
    t for analysis:  (time-origin)/365.25
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        1  failure in single failure-per-subject data
```

```
 13.21424  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =         5
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	0	5	0	1
2	0	5	0	1
3	0	3.2142368	1	1

Note the use of `exit(time datediag + 365.25*5)`. This is on the original time scale (in days) and so I have multiplied the number of days per year (365.25) by my desired follow-up time.

The analysis time (`_t`) is now 5 years for subject 1. Subject 2 also has an analysis time of 5 years, however their event indicator (`_d`) has changed from 1 to 0 as their event was after 5 years.

1.4.6 Left truncation

We can left truncate the time scale using the `enter` option. This will also be used when we use age as the time scale in section 1.4.7. An example of when left truncation is used is in *period analysis* where only the survival experience of subjects who are at risk in a recent time period are included in the analysis. For example, if we only want to include the survival times after 1/1/2001 we can use `enter(time mdy(1,1,2001))`.

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag) scale(365.25) ent
> er(time mdy(1,1,2001))
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 enter on or after:  time mdy(1,1,2001)
 exit on or before:  failure
    t for analysis:  (time-origin)/365.25
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 12.62971  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =   6.53525
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	.53935661	6.5352498	0	1
2	1.7138946	5.1334702	1	1
3	0	3.2142368	1	1

This is the first time we have observed that `_t0` is not zero. This is because the first two subjects were diagnosed before 1/1/2001 and we have specified that we are only interested in analyzing the survival times after this date. The variable `_t0` is still 0 for subject 3 as they were diagnosed after 1/1/2001.

1.4.7 Age as the timescale

When using age as the timescale we need to make use of the `enter` and `origin` options. As we are interested in age, the time origin must be the date of birth and the entry time in the study is the date of diagnosis.

```
        . stset dateexit, failure(event == 1) id(id) origin(datebirth) enter(datediag)
> scale(365.25)
                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 enter on or after:  time datediag
 exit on or before:  failure
    t for analysis:  (time-origin)/365.25
            origin:  time datebirth
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 14.88296  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =  23.61123
                                  last observed exit t =  37.76318
. list id _t0 _t _d _st, noobs
```

id	_t0	_t	_d	_st
1	31.227926	37.763176	0	1
2	23.611225	28.744695	1	1
3	27.718001	30.932238	1	1

In the above results the variable `_t0` denotes the age at which the subject was diagnosed with the disease. The variable `_t` denotes the age at which the subject died or was stopped being at risk due to censoring.

1.4.8 Time-Varying covariates

When incorporating time-varying covariates in survival analysis we must split the follow-up at the time where the covariate changes value. Note that this time will usually be different between subjects. We can use `stsplit`, but need to invoke a new facility, splitting along *another* timescale.

The origin of another timescale can be specified by the option `after()`. In this case we use `datetreat` as the origin of the new timescale. Then we ask to have the data split at only one point on this timescale, 0, which by definition equals the date of treatment start.

The variable created (`changetx`) will have values corresponding to the left endpoint of the intervals. Stata codes the left endpoint as -1 for intervals prior to `datetreat`.

```
        . stset dateexit, failure(event == 1) id(id) origin(datediag) scale(365.25)

                id:  id
     failure event:  event == 1
obs. time interval:  (dateexit[_n-1], dateexit]
 exit on or before:  failure
    t for analysis:  (time-origin)/365.25
            origin:  time datediag
------------------------------------------------------------------------------
        3  total obs.
        0  exclusions
------------------------------------------------------------------------------
        3  obs. remaining, representing
        3  subjects
        2  failures in single failure-per-subject data
 14.88296  total analysis time at risk, at risk from t =         0
                             earliest observed entry t =         0
                                  last observed exit t =   6.53525

. replace datetreat = dateexit + 1 if datetreat == .
(1 real change made)

. stsplit changetx, after(datetreat) at(0)
(2 observations (episodes) created)

. replace changetx = changetx + 1
(5 real changes made)

. list id _t0 _t _d _st changetx, noobs
```

id	_t0	_t	_d	_st	changetx
1	0	2.0451745	0	1	0
1	2.0451745	6.5352498	0	1	1
2	0	1.3935661	0	1	0
2	1.3935661	5.1334702	1	1	1
3	0	3.2142368	1	1	0

After the `stsplit` command `changetx` will have the value -1 for before the treatment change and 0 for the time of the treatment change and thus the `replace` command changes these to 0 and 1 respectively. Note that the subject who does not change treatment only has one record

If there are more treatment changes at other dates or there are other time-varying covariates then these must be declared in another variable and the process repeated.