UNIVERSITY OF ESSEX
INSTITUTE FOR SOCIAL AND ECONOMIC RESEARCH
Professor Stephen P. Jenkins <stephenj@essex.ac.uk>

Essex Summer School course 'Survival Analysis'
and
EC968. Part II: Introduction to the analysis of spell duration data

Lesson 3. Preparing survival time data for analysis and estimation

Contents

1 Aims

The aim of this lesson is to illustrate how to use Stata to prepare survival time data for analysis.

2 Introduction

The heading refers to 'preparation and organisation of data for analysis'. That is, this lesson is not about inputting original data in Stata (see Lesson 1 about this task). Here I assume that there is a Stata format data set ready to use. The 'preparation and organisation' refers to

manipulation of this data set in ways which facilitate summary and analysis of survival time data.

Stata's **st** (*s*urvival *t*ime) suite of commands provide sophisticated tools for these tasks. In short, with continuous survival time data, once you have 'stset' them – declared the variables summarising the spell length and censoring status – then you can go straight ahead and summarise and analyse the data (including multivariate hazard modelling) without referring to those key variables again. It should be noted, however, that the Stata st suite is designed with an emphasis on analysis of continuous survival time data. Although discrete (grouped duration) data may be usefully summarised using st tools, estimation of discrete time hazard models is typically done outside this framework. It remains relatively straightforward however. More specifically, the data preparation and organisation we do in order to subsequently apply 'easy estimation' methods for discrete time models are closely related to the tasks which one has to do if incorporating time-varying covariates in estimation of a continuous time model. The common task is what is known as 'episode splitting', and we shall examine several ways of doing this.

3 Assumptions about the initial data structure

Throughout this part of the course we focus on data about time-to-absorbing-event data, i.e. single failure data, with a single record per 'subject'. Hence there are no complications arising from left censoring, gaps, left truncation ('delayed entry'), or multiple events, etc. (These complications can also be handled using Stata's **st** suite: see **st stset** in the Reference Manuals.) There are no missing values for simplicity's sake. And the data do not need to be weighted.

We shall also assume that the survival time and censoring variables already exist. Thus we do not consider, for example, what to do if we had to derive survival times from information about spell start and end dates – with one exception. One part of Exercise 3.1 contains a simple example of this task. (For more general cases, see the Stata Reference Manuals.)

Finally, we shall suppose – for the moment anyway – that there are no time-varying covariates. In this case, all the explanatory variables in our regressions have a fixed value for each subject. See the final section for discussion of how to incorporate these.

These various assumptions about the nature of the data imply that our data have a very simple structure. There is one row in the data set for each 'subject' (e.g. person or firm). Columns in the data set (variables) contain at least two types of information for each subject:

(1) the length of time in the state (the survival time = the length of time the subject was exposed to the risk of experiencing a 'failure'); and
(2) censoring status (a variable typically equal to 1 if the person experienced a 'failure', and equal to 0 otherwise).

Other columns in the data set typically include variables used as regressors in estimation of multivariate hazard models. All the survival analysis data sets for this course have this structure.

For example, for the Cancer data set, we have:

```
. use cancer,clear
(Patient Survival in Drug Trial)
. de
Contains data from cancer.dta
  obs:            48                          Patient Survival in Drug Trial
 vars:             4                          16 Nov 1998 11:49
 size:           576 (100.0% of memory free)
-------------------------------------------------------------------------------
   1. studytim  int    %8.0g                  Months to death or end of exp.
   2. died      int    %8.0g                  1 if patient died
   3. drug      int    %8.0g                  Drug type (1=placebo)
   4. age       int    %8.0g                  Patient's age at start of exp.
-------------------------------------------------------------------------------
```

In this case, ‘studytim’ is the survival time and ‘died’ is the censoring indicator; ‘drug’ and ‘age’ are potential explanatory variables.

4 Continuous time models with only fixed covariates: using stset

If you wish to estimate a continuous time model and there are no time-varying covariates, then with this simple data structure, it is very easy to prepare the data for description and analysis: **stset** is the only command required to organise the data. The command syntax is:

```
stset timevar , failure(censvar)
```

where **failure(censvar)** specifies the failure event – there is a failure whenever variable censvar is not equal to zero and not missing. Alternatively use the syntax **failure(censvar==*numlist*)** in which case the failure cases are those with censvar equal to numlist . If **failure(.)** is not specified, every record is assumed to end in a failure. For the full syntax of **stset**, with options, **help stset**.

For example, for the Cancer data set, and treating survival time recorded in studytim as a continuous variable:

```
. stset studytim , failure(died)

     failure event:  died != 0 & died < .
obs. time interval:  (0, studytim]
 exit on or before:  failure

-------------------------------------------------------------------------------
       48  total obs.
        0  exclusions
-------------------------------------------------------------------------------
       48  obs. remaining, representing
       31  failures in single record/single failure data
      744  total analysis time at risk, at risk from t =         0
                            earliest observed entry t =         0
                                 last observed exit t =        39
```

Note that **stset** creates a set of new variables in the data, all prefaced with “_”. These always have the same names: that’s how Stata’s survival time estimation commands is able to work, because it knows that, if the data have been **stset**, then the key variables (duration, censoring indicator, etc.) are available. Here are the variables:

```
. de _*

              storage  display     value
variable name   type   format      label      variable label
------------------------------------------------------------------------------
_st             byte   %8.0g
_d              byte   %8.0g
_t              byte   %10.0g
_t0             byte   %10.0g

. su _*

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         _st |        48           1           0          1          1
          _d |        48    .6458333    .4833211          0          1
          _t |        48        15.5    10.25629          1         39
         _t0 |        48           0           0          0          0
```

The **_st** variable is a 0/1 variable, equal to 1 for observations whose data has been **stset** (it would be zero if one had excluded some cases with an **if** qualifier, for instance). The **_d** variable is the censoring indicator, another 0/1 variable, and corresponds to the variable died in this case. The variable **_t** is the duration variable, corresponding to studytim. Finally, **_t0** is a variable recording the date of entry to the study for each case, i.e. when they were first observed at risk of the event. For this dataset, all cases were observed from the time of entry to the state, i.e. $t = 0$. If, however, there had been delayed entry (left truncation), then the entry times would have been recorded in this variable if the researcher had declared it using the **enter()** option to **stset** (which we won't use in this course). We will refer to these 'underscore' variables later on.

Typing **st** by itself shows how the data are currently set:

```
. st
-> stset studytim, failure(died)

     failure event:  died ~= 0 & died ~= .
obs. time interval:  (0, studytim]
 exit on or before:  failure
```

That there are 31 failures (deaths in this case), as the output from the **stset** command says, can easily verified using **ta died**. The '744' refers to the total number of time periods for which this sample was observed at risk of dying since time $t = 0$: the sum of studytim across all persons. You can get almost all this information more directly using **stsum**:

```
. stsum

         failure _d:  died
   analysis time _t:  studytim

         |              incidence       no. of    |------ Survival time -----|
         | time at risk     rate        subjects        25%       50%       75%
---------+--------------------------------------------------------------------
   total |          744   .0416667            48          8        17        33
```

The incidence rate, 0.04167 = 31/744. See also **stdes** (which comes into its own for descriptuion of more complicated survival data structures). The median survival time since the start of the study is 17 months (more about deriving estimates of the survival time distribution in the next Lesson). Note that the median survival time reported by **stdes** is derived from the raw data and ignores censoring, and so differs from the median shown by **stsum**.

5 The discrete time hazard model case

Data organisation for estimation of discrete time hazard models is only slightly more complicated. Recall from the Lectures that our 'easy estimation' methods for these models are based on application of standard binary dependent variable models to re-organised data. (In principle one does not have to estimate the models this way, of course, but it is typically the most practical way of doing so.)

The data set must be re-organised so that, for each person, there are as many data rows as there are time intervals at risk of the event occurring for each person. We need to go from the simple data set discussed earlier, with one row of data per person, to another data set in which each person contributes T_i rows, where T_i is the number of time periods (e.g. months) i was at risk of the event. In effect an unbalanced panel data set-up is required.

We also require a unique identifier variable for each subject (if it doesn't already exist), plus a spell month identifier variable for each subject. The binary dependent variable also needs to be created. If subject i's survival time is censored, the binary dependent variable is equal to 0 for all of i's spell months; if subject i's survival time is not censored, the binary dependent variable is equal to 0 for all but the last of i's spell months (month 1,..., T_i–1) and equal to 1 for the last month (month T_i).

Consider how to use Stata to re-organise the data set and create the new variables. We use the Cancer data set for illustration, but now suppose that studytim refers to discrete intervals of time (months).

To understand what is going on, look at how the data for the first four people is currently organised.

```
. ge id = _n  /* create unique person identifier */
. list id studytim died drug age in 1/4

     +----------------------------------+
     | id   studytim   died   drug   age |
     |----------------------------------|
  1. |  1          1      1      1    61 |
  2. |  2          1      1      1    65 |
  3. |  3          2      1      1    59 |
  4. |  4          3      1      1    52 |
     +----------------------------------+
```

5.1 Episode splitting using expand

Now **expand** the data set so that there's one data row per person per month at risk of dying.

```
. expand studytim   /* see -help expand- */
(696 observations created)
```

Now create the spell month identifier for each subject and the binary dependent variable

```
. bysort id: ge seqvar = _n   // -bysort- is like -sort- followed by -by-
. lab var seqvar "spell month identifier, by subject"
        // NB generation of variable using a logical condition
. bysort id: ge dead = died == 1 & _n==_N
. lab var dead "binary depvar for discrete hazard model"
```

The **bysort id:** … command is a convenient way of combining two operations **sort id**, followed by **by id:**. (If you are going to ask Stata to undertake calculations within groups of observations, here groups defined by values of *id*, the data have to be sorted by the group variable. The rest of the command line **ge seqvar = _n** generates a new variable equal to the current observation number. Because we are doing a 'by group' operation, **_n** in this case is the current observation number *within the relevant group of cases* (as identified by *id*). The next **bysort** command generates a 0/1 variable with the value of *dead* depending on whether the two logical conditions on the right hand side of the '=' sign are satisfied: it is equal to one if both statements are true, and equal to zero otherwise. The first condition requires the subject to have died and second requires the current observation number to be equal to the maximum within all rows of data for each subject. Thus *dead* = 1 only if the subject died and it's the last row of data for that subject. (The combination of **by** and **_n**, **_N** is extensively used in the analysis of panel data. See also the **egen** (*e*xtended *gen*erate) commands for related bygroup operations.)

Note the importance of creating the subject identifier variable (id) *before* expanding the data. If we had not done so, we would not know which spell months in the new data referred to which person.

Compare the new data format with the earlier one, taking the same four persons:

```
. list id studytim seqvar died dead age if id <= 4

    +-----------------------------------------+
    | id   studytim   seqvar   died   dead   age |
    |-----------------------------------------|
 1. |  1          1        1      1      1    61 |
 2. |  2          1        1      1      1    65 |
 3. |  3          2        1      1      0    59 |
 4. |  3          2        2      1      1    59 |
 5. |  4          3        1      1      0    52 |
    |-----------------------------------------|
 6. |  4          3        2      1      0    52 |
 7. |  4          3        3      1      1    52 |
    +-----------------------------------------+
```

Check that there are 744 observations in the new data set.

```
. de  /* There should be 744 obs in the data set now */

Contains data from cancer.dta
  obs:           744                          Patient Survival in Drug Trial
 vars:            11                          16 Nov 1998 11:49
 size:        20,832 (99.9% of memory free)
-------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
-------------------------------------------------------------------------------
studytim        int    %8.0g                  Months to death or end of exp.
died            int    %8.0g                  1 if patient died
drug            int    %8.0g                  Drug type (1=placebo)
age             int    %8.0g                  Patient's age at start of exp.
_st             byte   %8.0g
_d              byte   %8.0g
_t              byte   %10.0g
_t0             byte   %10.0g
id              float  %9.0g                  subject identifier
seqvar          float  %9.0g                  spell month identifier, by
                                                subject
dead            float  %9.0g                  binary depvar for discrete
                                                hazard model
-------------------------------------------------------------------------------
Sorted by:  id
     Note:  dataset has changed since last saved
```

We could **stset** the data now if we wished but, because we are supposing that we are interested in estimating discrete time models, there is no need to do so. Estimation of discrete time models does not require the data to be **stset**; this command is essentially a utility for *continuous* survival time data.

In practice, the next stage in estimating a discrete time model would be creation of time-varying covariates. At the very least, this will include variables used to characterise the pattern of duration dependence. These will be functions of each subject's survival time, which is here recorded in the integer-valued variable **seqvar** (it corresponds to '*j*' used in the Lecture Notes). Illustrations of how to generate these variables are provided later in this Lesson (see also Lesson 6). Other time-varying covariates might also be created at this stage (if, for example, we'd had additional information in the dataset).

5.2 Allocate sufficient memory to hold the expanded data set

Note that creation of a larger re-organised data set may result in an error message:

```
no room to add more observations
r(901);
```

In this case, you need to increase the memory available to Stata above the initial level. First either **drop _all** or **clear**, and then **set memory X**, where X is typically a number slightly larger than your data set size but smaller than the RAM memory installed in your PC. You may need first to **clear** or **drop _all** (see **help memory**). Examples of X include '10000k' and '30m'. You should also make sure that you have first **compress**ed your data.

5.3 Episode splitting using stsplit

One can use **stsplit** to derive the same expanded data structure. Although it is a command developed for episode splitting with continuous time survival data, it can also be used in the discrete time case because the data structure we require in this case is a special sort of episode splitting. Consider the following code:

```
. use cancer, clear
(Patient Survival in Drug Trial)

. ge id = _n

. stset studytim, f(died) id(id)

                id:  id
     failure event:  died ~= 0 & died ~= .
obs. time interval:  (studytim[_n-1], studytim]
 exit on or before:  failure

--------------------------------------------------------------------------------
       48  total obs.
        0  exclusions
--------------------------------------------------------------------------------
       48  obs. remaining, representing
       48  subjects
       31  failures in single failure-per-subject data
      744  total analysis time at risk, at risk from t =         0
                            earliest observed entry t =         0
                                 last observed exit t =        39

. ge T = studytim

. stsplit time, at(1(1)39)
(696 observations created)
```

The number list in the **at()** option tells Stata that we want to split the data at each month in the data from the smallest month number to the largest. This requires knowing what these are. It's much easier to get the same effect by using the **every()** option instead.

```
. * stsplit time, every(1)  /* alternative specification */
. stset
-> stset studytim, id(id) failure(died)

                id:  id
     failure event:  died ~= 0 & died ~= .
obs. time interval:  (studytim[_n-1], studytim]
 exit on or before:  failure

--------------------------------------------------------------------------------
      744  total obs.
        0  exclusions
--------------------------------------------------------------------------------
      744  obs. remaining, representing
       48  subjects
       31  failures in single failure-per-subject data
      744  total analysis time at risk, at risk from t =         0
                            earliest observed entry t =         0
                                 last observed exit t =        39
```

Now let us check that the data structure is as we expect.

```
. sort id time

. ge seqvar = time + 1

. by id: list id died T studytim time seqvar , noobs

-------------------------------------------------------------------------------------------------
-> id = 1

  +---------------------------------------+
  | id   died   T   studytim   time   seqvar |
  |---------------------------------------|
  |  1      1   1          1      0        1 |
  +---------------------------------------+

-------------------------------------------------------------------------------------------------
-> id = 2

  +---------------------------------------+
  | id   died   T   studytim   time   seqvar |
  |---------------------------------------|
  |  2      1   1          1      0        1 |
  +---------------------------------------+

-------------------------------------------------------------------------------------------------
-> id = 3

  +---------------------------------------+
  | id   died   T   studytim   time   seqvar |
  |---------------------------------------|
  |  3      .   2          1      0        1 |
  |  3      1   2          2      1        2 |
  +---------------------------------------+

-------------------------------------------------------------------------------------------------
-> id = 4

  +---------------------------------------+
  | id   died   T   studytim   time   seqvar |
  |---------------------------------------|
  |  4      .   3          1      0        1 |
  |  4      .   3          2      1        2 |
  |  4      1   3          3      2        3 |
  +---------------------------------------+

-------------------------------------------------------------------------------------------------
-> id = 5

  +---------------------------------------+
  | id   died   T   studytim   time   seqvar |
  |---------------------------------------|
  |  5      .   4          1      0        1 |
  |  5      .   4          2      1        2 |
  |  5      .   4          3      2        3 |
  |  5      1   4          4      3        4 |
  +---------------------------------------+
< output omitted >

. stsum

         failure _d:  died
   analysis time _t:  studytim
                 id:  id

         |              incidence       no. of    |------ Survival time -----|
         | time at risk     rate        subjects        25%       50%       75%
---------+-------------------------------------------------------------------
   total |          744   .0416667            48          8        17        33
```

You can see that **stsplit** and **stset** are a pretty smart combination. If the data had been **stset** you do not have **stset** them again. (Remember that you don't have to **stset** the data if you are doing discrete time modelling.) Stata has appropriately modified the studytim variable. (Remember to create a copy of the variable if you want to retain the original information.) The censoring variable is set equal to missing in the new records too, but that is not a problem. They get counted as censored cases, because failures for **stset** are defined as cases with non-zero non-missing values.

6 Time-varying covariates – episode splitting again

How we organise our data in order to incorporate these variables in the analysis depends on the type of model that we are estimating, whether continuous or discrete time. Let us consider these in turn.

6.1 Discrete time models

In the last section, we organised the data so that there was an observation corresponding to each time interval that a subject was at risk of failure. So, assuming each time-varying covariate has a constant value within each time interval, then one can simply generate, or merge in, the values of the covariates that are appropriate for the relevant period.

The simplest example of a time-varying covariate in discrete time models is perhaps the variable summarising the baseline hazard function ($c(t)$ in Lesson 2). Observe that we have already created t. It is simply the variable 'seqvar' in the last section (or the modified studytim variable in Section 5.2). If we wanted to estimate a model with, say, $c(t) = r.\ln(t)$ as the baseline hazard, we would create a variable as follows:

```
. ge logd = ln(seqvar)
. lab var logd "ln(t)"
```

The variable would be used as a regressor and the estimated coefficient on logd would be our estimate of $r = q-1$ (see Lesson 6).

It is straightforward to add other time-varying variables to a data set, using the variables corresponding to 'seqvar' and 'id' (and possibly some calendar time variables too) in order to ensure that the new variables are correctly matched with the appropriate spell month.

6.2 Continuous time parametric models

For estimation of continuous time models with time-varying covariates, episode splitting need not be at each survival time in the data set – it depends on how often the time-dependent variables are assumed to change (with time and for each subject). Indeed it is often much more economical in terms of data set size not to split at each survival time in the data. And Stata is still smart enough (assuming a correctly specified **stset**) to ensure that the likelihood contribution for each subject is correct.

The piece-wise exponential hazard regression model provides an illustration of how **stsplit** may be employed for episode-splitting in the continuous time case. As the Lectures (and Lesson 5) point out, this model may be estimated by defining new variables that identify each survival time interval over which the hazard rate is assumed to be constant and including them as time-varying covariates.

Let us use the cancer data again, then, and suppose that we wish to estimate this type of model. And, for the sake of argument, hypothesise that the hazard rate differs for survival times less than 8, between 8 and 17, and over 17, but is constant within those intervals. So we need dummy variables that will identify each of the three intervals, and have to split episodes appropriately for those subjects for whom it is relevant. Consider the following code:

```
. use cancer, clear
(Patient Survival in Drug Trial)

. ge id = _n

. stset studytim, f(died) id(id)

                id:  id
     failure event:  died ~= 0 & died ~= .
obs. time interval:  (studytim[_n-1], studytim]
 exit on or before:  failure

--------------------------------------------------------------------------------
       48  total obs.
        0  exclusions
--------------------------------------------------------------------------------
       48  obs. remaining, representing
       48  subjects
       31  failures in single failure-per-subject data
      744  total analysis time at risk, at risk from t =         0
                            earliest observed entry t =         0
                                 last observed exit t =        39
```

```
.
. * Vble to pick out 3 subjects with illustrative times
. ge pick = id == 3 | id == 17 | id == 48

. sort id

. list id studytim _t died _d _t0 if pick

     +-------------------------------------+
     | id   studytim   _t   died   _d   _t0 |
     |-------------------------------------|
  3. |  3          2    2      1    1     0 |
 17. | 17         15   15      1    1     0 |
 48. | 48         39   39      0    0     0 |
     +-------------------------------------+
.
. * split at survival times 8 and 17
. * ... key intervals are (0,8], (8,17], (17,infinity]
.
. stsplit ehaz, at(8 17)
(50 observations created)

. ta ehaz, ge(e)

       ehaz |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |         48       48.98       48.98
          8 |         32       32.65       81.63
         17 |         18       18.37      100.00
------------+-----------------------------------
      Total |         98      100.00

. sort id ehaz

. list id studytim _t died _d _t0 e* if pick

    +-------------------------------------------------------------+
     | id   studytim   _t   died   _d   _t0   ehaz   e1   e2   e3 |
     |-------------------------------------------------------------|
  3. |  3          2    2      1    1     0      0    1    0    0 |
 21. | 17          8    8      .    0     0      0    1    0    0 |
 22. | 17         15   15      1    1     8      8    0    1    0 |
 96. | 48          8    8      .    0     0      0    1    0    0 |
 97. | 48         17   17      .    0     8      8    0    1    0 |
     |-------------------------------------------------------------|
 98. | 48         39   39      0    0    17     17    0    0    1 |
     +-------------------------------------------------------------+
```

Observe how **stsplit** created 50 new records: 32 for subjects with studytim values in the interval (8, 17] and 18 for subjects with studytim values in the interval (17, 39]. Put another way, subjects with studytim values in the interval (8, 17] had one extra record created and those with studytim values in the interval (17, 39] had two extra records created. The **ta ehaz, ge(e)** command also creates dummy variables enabling us to identify the different records for each person. As Lesson 5 will show, these variables are used to estimate the regression model.

Observe too how **stset** has automatically updated the underlying variables that it uses to ensure that the correct likelihood contributions from each subject will be created: look at _t, _d, and _t0 on the records for the two illustrative cases with multiple records. (Refer back to earlier in the Lesson for a discussion of these variables.) For example, subject 17 with studytim = 15 and died = 1 (failure) now has one record indicating survival from time 0 through to 8 without failure (_t0 = 0, _t = 8, _d = .) and one record indicating survival from time 8 through to time 15 when failure occurred (_t0 = 8, _t = 15, _d = 1). In other words, the spell (episode) for this person has been split into two sub-spells: the first is a right-censored spell; the second is a non-censored spell, but with delayed entry (left truncation) at $t = 8$ (note the value of the _t0 variable in this case).

What about more general sorts of time-varying covariates? The general principles are the same as just explained: split episodes as and where appropriate and create the relevant variables for each episode. In practice of course things can be somewhat more tricky than this bald statement might suggest! The complications depend on the nature of the model being considered. Read the relevant sections in the Reference Manual.

6.3 Continuous time Cox models

Time-varying covariates can also be included in the partial likelihood semi-parametric Cox model. As mentioned in the Lectures and elsewhere, it is a property of the Cox model that estimates are derived using information about the risk pool (subjects at risk) at each failure time only. Thus covariate values at the observed failure times are what is relevant to the model (regardless of their values at any times in between). Time-varying covariates in this context mean covariates changing at different failure times for the members of the relevant risk pools. What one has to do is expand the data for each subject so that there is a record for every survival time when that subject was the member of the relevant risk pool.

There are two situations to distinguish. The first is where the time-varying covariates are assumed to change discretely between time intervals. For example, suppose the cancer data set also had information about the identity of the physician who treated each subject and the this person changed over the time of the study. In this situation, one proceeds along the following lines.

```
. use cancer, clear
(Patient Survival in Drug Trial)

. ge id = _n

. stset studytim, f(died) id(id)

                id:  id
     failure event:  died ~= 0 & died ~= .
obs. time interval:  (studytim[_n-1], studytim]
 exit on or before:  failure

--------------------------------------------------------------------------------
       48  total obs.
        0  exclusions
--------------------------------------------------------------------------------
       48  obs. remaining, representing
       48  subjects
       31  failures in single failure-per-subject data
      744  total analysis time at risk, at risk from t =         0
                            earliest observed entry t =         0
                                 last observed exit t =        39

. stsplit , at(failures)
(21 failure times)
(534 observations (episodes) created)
```

Now check that the data structure is as we expect:

```
sort id studytim
```

```
. list id studytim _t died _d _t0  if (id==3|id==17|id==48), noobs
```

id	studytim	_t	died	_d	_t0
3	1	1	.	0	0
3	2	2	1	1	1
17	1	1	.	0	0
17	2	2	.	0	1
17	3	3	.	0	2
17	4	4	.	0	3
17	5	5	.	0	4
17	6	6	.	0	5
17	7	7	.	0	6
17	8	8	.	0	7
17	10	10	.	0	8
17	11	11	.	0	10
17	12	12	.	0	11
17	13	13	.	0	12
17	15	15	1	1	13
48	1	1	.	0	0
48	2	2	.	0	1
48	3	3	.	0	2
48	4	4	.	0	3
48	5	5	.	0	4
48	6	6	.	0	5
48	7	7	.	0	6
48	8	8	.	0	7
48	10	10	.	0	8
48	11	11	.	0	10
48	12	12	.	0	11
48	13	13	.	0	12
48	15	15	.	0	13
48	16	16	.	0	15
48	17	17	.	0	16
48	22	22	.	0	17
48	23	23	.	0	22
48	24	24	.	0	23
48	25	25	.	0	24
48	28	28	.	0	25
48	33	33	.	0	28
48	39	39	0	0	33

After the episode splitting, one creates the relevant time-varying covariates manually (e.g. physician identity – if this was available in our cancer data set!).

The second situation to consider with the Cox model is when the relevant variables vary continuously with time:

$$h(t) = h_0(t)\exp[\beta X + g(t).(\gamma Z)].$$

For example, suppose that $g(t) = \ln(t)$. The standard Cox model is of course the case when $g(t) = 0$.

In cases like this, one can proceed as we did for the discrete time-varying covariate case: use **stsplit** to split the data at failure times and create the variables manually (the $g(t)$ can be characterised using the variable _t that is created by **stset**).

Alternatively, note that there are options to the **stcox** command that allow one to directly specify the variables that comprise Z and the function $g(t)$: **tvc(varlist)** and **texp(expression)**.

This is particularly useful because episode splitting need not be done. If there are a large number of observed survival times then **stsplit**ting the data creates many new records and the resulting data set may not fit into memory. Even if it did fit, estimation of the model would take much longer because of the increased size of the data set.

7 Exercise 3.1

(1) Take the case of no time-varying covariates first, and organise the strike data (kennan.dta) and the marriage duration data (duration.dta) into a form suitable for analysis, assuming that survival times are continuous. What are the median survival times in each case (according to **stsum**)? How would your procedures change if you were to assume that the survival times in these data sets referred to a discrete variable rather than a continuous one?

(2) Now consider how one would organise data in order to allow for a time-varying covariate in modelling. Use the unemployment data (unemp.dta). Recall that the survival time variable is called conmths (measured in months) and the censoring variable is called exit. Variable rn1 gives the net replacement rate during months 1–6 of the spell, and variables give the values for the net replacement rate during months 7–12 and 13–24, respectively. Produce a reorganised data set with a single time-varying replacement rate variable. Suppose that the survival time variable refers to discrete intervals of time, and then (for practice) repeat the exercise supposing that conmths refers to exact times.

(3) Derive a survival time variable from information about spell beginning and ending dates –an introduction to Stata's date functions. Use the HIV data set (hmohiv.dta). There is already a survival time variable in it, time, measuring survival time in months. But I want you to derive a survival time variable measuring in days, using the variables entdate and enddate. Note: these are currently string variables of a form like 03jan91 for the 3rd of January 1991. First you need to generate new variables that convert these dates into an elapsed date format (numbers of days since 01jan1960): see **help dates_and_times**, and format the new variables as date variables (**format newvar %td**). Then generate the new survival time variable as the difference between the two new variables, **stset** the data, and run **stsum**. Compare the results with what happens if you use the monthly survival time variable (**stset time, failure(censor)**, followed by **stsum**).