

**SC706: Longitudinal Data Analysis**  
**Instructor: Natasha Sarkisian**

**Event history analysis: Multiple Event Types and Repeated Events**

**Multiple Event Types**

In event history analysis, sometimes we are interested in more than just one 0/1 type event, like birth or death – we might be interested in a number of outcomes that can take place – e.g. after a spell of unemployment, the unemployed person might either get a full-time job or a part-time job; a married person might get separated, divorced, or widowed, etc. To deal with these multiple kinds of events, we can extend the same methodology, but to do that, we need to consider how the risk of each kind of event is related to the risk of other events.

*1. One process determines whether some kind of event happens, then another process determined which kind of event happens.*

E.g. one process may determine whether one goes to college, and another – what type of school. The appropriate analysis strategy here is to model the two processes separately – first, use event history to model the occurrence of an event (any event, regardless of the type), and second, use binary or multinomial logistic regression (depending on whether there are only two kinds of events or more than two) predict which kind of event happened. Obviously, this second analysis should be done on a subsample of those who experienced an event, and your dataset should only contain one record per person for this part of the analysis. You may, for example, only keep the records for the time period when the event took place.

*2. Different kinds of events happen independently from one another – the occurrence of one kind does not affect either the risk of another event nor the observation of other kinds of events.*

E.g. getting a pet and getting into a car accident. We can never say that events are absolutely unrelated, but there are many cases where for all practical purposes we can say that there is no relationship. To deal with such cases, we will just estimate separate event history models that make no use of the information about whether other kind of event took place or not.

*3. When one kind of event happens, it increases or lowers (but not to zero!) the risk of another kind of event.*

E.g. getting married increases the risk of having a child. In this situation, we'll model the events separately, but when modeling the second event (e.g. giving birth), we will take the first one (marital status) into account by using a time-varying independent variable in the model.

*4. When one kind of event happens, it removes the individual from observation of other event types.*

E.g. if we track residential mobility both in terms of mobility inside the country and in terms of moving abroad, we may lose those who move abroad and cannot track whether they

subsequently also move within the country. This is similar to censoring – we analyze the two events separately, and ignore the fact that the reason the observation is censored is because the other kind of event happened. But this type of analysis is only fully appropriate if the two types of events are actually independent from one another– i.e. the fact that the first type happened does not raise or lower the risk for the second (that is, events are like in #2 rather in #3 above). This is because event history methods require that censoring times have to be independent of event times. If that is not the case, the analyses are somewhat problematic.

Note that you don't need to change your dataset to deal with multiple models like that – stset command allows you to state what defines an observation is censored by specifying it in the “exit” option.

*5. If one type of event happens, it removes the individual from risk of the other type of event.*

E.g. if you are unemployed, you can either get a part-time or a full-time job. This is usually described as “multi-state” process or as “competing risks.” The classic example is death from different causes. Such models are common in the literature.

Here, for each event, the other types of event again serve as censoring mechanisms. So we develop separate event history models for each type of event (i.e. we model type-specific hazard rate), and then can either estimate them separately or simultaneously.

An example of separate estimation:

Metraux, Stephen and Dennia P. Culhane. 1999. Recurring Homelessness Among Women. *Journal of Family Issues*, 20(3): 371-396.

An example of simultaneous estimation:

De Graaf, Paul M. and Matthijs Kalmijn. 2003. Alternative Routes in the Remarriage Market: Competing-Risk Analyses of Union Formation after Divorce. *Social Forces*, 81(4):1459-1498.

Simultaneous estimation is especially easy if we use discrete-time approach – we can use multinomial logit instead of binary logit. That's what the article does. Also note that they are constraining coefficients of certain variables to be equal – it is best to first test whether they are, but once that is established, use “constraint” command in Stata to specify your constraints.

But this is also possible with semi-parametric or parametric models. If you are interested only in one event and want only to control for the competing event, you can use `stcrreg` (see `help stcrreg`) but it is very limiting. It is typically better to adapt standard techniques to work for us.

We need to set up the data so that all the observations are replicated so many times as many types of events we have (i.e., we'll have one case per event type, but that case can contain multiple lines if the data are multi-record). These lines will be identical except for two things: (1) an indicator whether the event of that type happened or the case was censored, and (2) an indicator of what type of event that observation represents. Let's examine an example using another subset of the dataset that you are using for your homework.

```
. use "C:\Documents and Settings\SARKISIN\My Documents\part_fulltime.dta"
```

```

. reshape long  mar educ interv ptime ftime newage emp enrol, i(id) j(year)
(note: j = 79 81 82 83 84 85 86 87 88 89 90 91 92 93 94)
(note: mar79 not found)
(note: educ79 not found)
(note: ptime79 not found)
(note: ftime79 not found)
(note: newage79 not found)
(note: emp79 not found)
(note: enrol79 not found)
(note: mar81 not found)
(note: educ81 not found)
(note: ptime81 not found)
(note: ftime81 not found)
(note: newage81 not found)
(note: emp81 not found)
(note: enrol81 not found)

```

```

Data                                wide  ->  long
-----
Number of obs.                      1824  ->  27360
Number of variables                  112   ->    15
j variable (15 values)              ->   year
xij variables:
      mar79 mar81 ... mar94  ->   mar
      educ79 educ81 ... educ94 ->   educ
      interv79 interv81 ... interv94 ->   interv
      ptime79 ptime81 ... ptime94 ->   ptime
      ftime79 ftime81 ... ftime94 ->   ftime
      newage79 newage81 ... newage94 ->   newage
      emp79 emp81 ... emp94 ->   emp
      enrol79 enrol81 ... enrol94 ->   enrol
-----

```

```

. drop if missing(interv)
(62 observations deleted)

```

```

. snapspan id  interv  ftime ptime emp educ newage emp enrol, gen(interv0) replace

```

```

. format interv %d

```

```

. format interv0 %d

```

```

. expand 2
(27298 observations created)

```

```

. sort id year

```

```

. by id year: gen type=_n

```

```

. tab type

```

type	Freq.	Percent	Cum.
1	27,298	50.00	50.00
2	27,298	50.00	100.00
Total	54,596	100.00	

```

. tab ptime ftime

```

ptime	ftime	Total
0	1	

	0	1	Total
16,266	22,562	38,828	
8,596	0	8,596	
24,862	22,562	47,424	

```

. gen event=.
(54596 missing values generated)

. replace event=ptime if type==1
(23712 real changes made)

. replace event=ftime if type==2
(23712 real changes made)

. di 16*365
5840

. stset interv, time0(interv0) failure(event==1) origin(time newage==5840)
exit(failure)
      failure event:  event == 1
obs. time interval:  (interv0, interv]
exit on or before:  failure
      t for analysis:  (time-origin)
              origin:  time newage==5840
-----
54596 total obs.
3648  entry time missing (interv0>=.)          PROBABLE ERROR
-----
50948 obs. remaining, representing
15579 failures in single record/single failure data
2.05e+07 total analysis time at risk, at risk from t =          0
              earliest observed entry t =          6943
              last observed exit t =          12775

```

Important: although multiple observations are linked by id, we do not specify that id in stset command – otherwise, we would get an error because time intervals within id values overlap. Once the data are set up, four different models can be estimated, depending on the assumptions about the underlying processes. For all of them, make sure to enter ID as the clustering variable!

A. The baseline shape and the effects of the covariates are the same, the risk is just uniformly higher or lower for one type of event than for another.

```

. gen typed=type-1

. stcox typed black hispanic mar educ, cluster(id)
      failure _d:  event == 1
      analysis time _t:  (interv-origin)
              origin:  time newage==5840
Iteration 0:  log pseudolikelihood = -115798.86
Iteration 1:  log pseudolikelihood = -113892.7
Iteration 2:  log pseudolikelihood = -113883.8
Iteration 3:  log pseudolikelihood = -113883.8
Refining estimates:
Iteration 0:  log pseudolikelihood = -113883.8

Cox regression -- Breslow method for ties

```

```

No. of subjects      =          43636          Number of obs   =          43636
No. of failures     =          14284
Time at risk        =          16468558
Log pseudolikelihood = -113883.8          Wald chi2(5)     =          1282.60
                                                Prob > chi2      =          0.0000

```

(Std. Err. adjusted for 1824 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
typed	2.758947	.0967388	28.94	0.000	2.575711	2.955219
black	.9381391	.0268197	-2.23	0.026	.887019	.9922053
hispanic	1.067778	.0317175	2.21	0.027	1.007388	1.131788
mar	.7682082	.014617	-13.86	0.000	.740087	.797398
educ	1.067955	.0050345	13.95	0.000	1.058133	1.077868

B. The baseline hazard shape is essentially the same, but the risk is higher or lower for one type of event than for another and the effects of covariates differ for different events.

```

. xi: stcox typed i.typed*black i.typed*hispanic i.typed*mar i.typed*educ,
cluster(id)
i.typed      _Ityped_0-1          (naturally coded; _Ityped_0 omitted)
i.typed*black  _ItypXblack_#      (coded as above)
i.typed*hispa~c  _ItypXhispa_#    (coded as above)
i.typed*mar     _ItypXmar_#       (coded as above)
i.typed*educ    _ItypXeduc_#      (coded as above)
failure _d: event == 1
analysis time _t: (interv-origin)
origin: time newage==5840
note: _Ityped_1 dropped due to collinearity
note: _Ityped_1 dropped due to collinearity
note: _Ityped_1 dropped due to collinearity
note: _Ityped_1 dropped due to collinearity
Iteration 0: log pseudolikelihood = -115798.86
Iteration 1: log pseudolikelihood = -113893.34
Iteration 2: log pseudolikelihood = -113806.86
Iteration 3: log pseudolikelihood = -113806.81
Refining estimates:
Iteration 0: log pseudolikelihood = -113806.81

```

Cox regression -- Breslow method for ties

```

No. of subjects      =          43636          Number of obs   =          43636
No. of failures     =          14284
Time at risk        =          16468558
Log pseudolikelihood = -113806.81          Wald chi2(9)     =          1518.87
                                                Prob > chi2      =          0.0000

```

(Std. Err. adjusted for 1824 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
typed	1.154667	.2264584	0.73	0.463	.7861681	1.695892
black	.7186947	.0495474	-4.79	0.000	.6278587	.8226724
_ItypXblac~1	1.431	.1236375	4.15	0.000	1.208082	1.695051
hispanic	.8494534	.0649131	-2.14	0.033	.7312954	.9867025
_ItypXhispa~1	1.365134	.1327803	3.20	0.001	1.128191	1.651839
mar	.8703612	.0426091	-2.84	0.005	.7907303	.9580114
_ItypXmar_1	.8434304	.053987	-2.66	0.008	.7439861	.9561668

```

educ | 1.019531 .011436 1.72 0.085 .9973613 1.042193
_ItypXeduc_1 | 1.065276 .0149359 4.51 0.000 1.036401 1.094956
-----

```

C. The baseline hazard shape is different, but the effects of the covariates are the same.

```

. stcox black hispanic mar educ, cluster(id) strata(typed)
  failure _d: event == 1
  analysis time _t: (interv-origin)
  origin: time newage==5840
Iteration 0: log pseudolikelihood = -105897.94
Iteration 1: log pseudolikelihood = -105609.57
Iteration 2: log pseudolikelihood = -105609.47
Iteration 3: log pseudolikelihood = -105609.47
Refining estimates:
Iteration 0: log pseudolikelihood = -105609.47

Stratified Cox regr. -- Breslow method for ties
No. of subjects = 43636 Number of obs = 43636
No. of failures = 14284
Time at risk = 16468558
Wald chi2(4) = 461.18
Log pseudolikelihood = -105609.47 Prob > chi2 = 0.0000
(Std. Err. adjusted for 1824 clusters in id)
-----

```

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]
black	.9381391	.0268197	-2.23	0.026	.887019 .9922053
hispanic	1.067778	.0317175	2.21	0.027	1.007388 1.131788
mar	.7682082	.014617	-13.86	0.000	.740087 .797398
educ	1.067955	.0050345	13.95	0.000	1.058133 1.077868

Stratified by typed

D. The baseline hazard shape is different, and the effects of covariates differ for different events.

```

. xi: stcox i.typed*black i.typed*hispanic i.typed*mar i.typed*educ, cluster(id)
strata(typed)
i.typed      _Ityped_0-1      (naturally coded; _Ityped_0 omitted)
i.typed*black  _ItypXblack_#    (coded as above)
i.typed*hispa~c  _ItypXhispa_#    (coded as above)
i.typed*mar      _ItypXmar_#    (coded as above)
i.typed*educ     _ItypXeduc_#    (coded as above)

  failure _d: event == 1
  analysis time _t: (interv-origin)
  origin: time newage==5840
note: _Ityped_1 dropped due to collinearity
note: _Ityped_1 dropped due to collinearity
note: _Ityped_1 dropped due to collinearity
Iteration 0: log pseudolikelihood = -105897.94
Iteration 1: log pseudolikelihood = -105521.55
Iteration 2: log pseudolikelihood = -105521.31
Iteration 3: log pseudolikelihood = -105521.31
Refining estimates:
Iteration 0: log pseudolikelihood = -105521.31

Stratified Cox regr. -- Breslow method for ties

No. of subjects = 43636 Number of obs = 43636
No. of failures = 14284
Time at risk = 16468558

```



```

Iteration 1:  log pseudolikelihood = -105521.55
Iteration 2:  log pseudolikelihood = -105521.31
Iteration 3:  log pseudolikelihood = -105521.31
Refining estimates:
Iteration 0:  log pseudolikelihood = -105521.31

```

Stratified Cox regr. -- Breslow method for ties

```

No. of subjects      =          43636          Number of obs      =          43636
No. of failures     =           14284
Time at risk        =       16468558
Log pseudolikelihood = -105521.31          Wald chi2(8)       =          489.70
                                          Prob > chi2        =          0.0000

```

(Std. Err. adjusted for 1824 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
black0	.7363472	.0508897	-4.43	0.000	.6430656	.8431601
hispanic0	.8610583	.0657061	-1.96	0.050	.7414445	.9999689
educ0	1.027529	.0118273	2.36	0.018	1.004607	1.050973
mar0	.9613765	.0475129	-0.80	0.425	.8726209	1.05916
black1	1.018721	.0379412	0.50	0.618	.9470069	1.095866
hispanic1	1.15273	.0461918	3.55	0.000	1.06566	1.246914
educ1	1.082722	.0066088	13.02	0.000	1.069847	1.095753
mar1	.7089811	.0193178	-12.62	0.000	.6721121	.7478726

Stratified by typed

## Repeated Events

Most event history data contains repeated events for each individual – multiple job changes, marriages, births, etc.

### Unordered repeated events

In rare cases, the events are unordered, that is, we do not know which one might happen first. Typically, this happens when we actually have multiple levels within the units we study, e.g., in medical data, if we study the occurrence of blindness, we can do so separately for the two eyes. Thus, the eyes are the subunits of a larger level unit – an individual. In social science data, such situations can happen if we study related groups of individuals, e.g., if we study sibling pairs and examine marriages, it can happen that the older sibling will get married first or it can happen that the younger one will. Here, we assume that the timing of marriage for family members can be correlated because of some similar social factors or even similar genetic predispositions.

The data setup for the analysis of unordered repeated events is relatively simple. Each sampling unit (e.g., an eye in the first example or a sibling in the second) appears as a separate observation in the dataset (of course, if we have time-varying covariates for each, each observation itself can have multiple records/lines in the dataset). Important: although multiple observations (people) are linked by id, we do not specify that id in stset command – otherwise, we would get an error because time intervals within id values overlap.

Single-record data setup:

```
. list famid sibid siborder time time0 event gender pareduc
```

	famid	sibid	siborder	time	time0	event	gender	pareduc
1.	1	1	2	22	17	1	0	16
2.	1	2	1	25	17	1	1	16
3.	2	1	1	24	17	1	1	12
4.	2	2	2	27	17	0	1	12

```
. stset time, failure(event) time0(time0)
```

```
failure event: event != 0 & event < .
obs. time interval: (time0, time]
exit on or before: failure
```

```
-----
4 total obs.
0 exclusions
```

```
-----
4 obs. remaining, representing
3 failures in single record/single failure data
30 total analysis time at risk, at risk from t = 0
earliest observed entry t = 17
last observed exit t = 27
```

```
. stcox gender, cluster(famid)
```

[Output omitted]

Multi-record data with two intervals per person (to accommodate the time-varying employment variable):

```
. list famid sibid siborder time time0 event gender pareduc empl
```

	famid	sibid	siborder	time	time0	event	gender	pareduc	empl
1.	1	1	2	20	17	0	0	16	0
2.	1	1	2	22	20	1	0	16	1
3.	1	2	1	21	17	0	1	16	0
4.	1	2	1	25	21	1	1	16	1
5.	2	1	1	19	17	0	1	12	1
6.	2	1	1	24	19	1	1	12	0
7.	2	2	2	21	17	0	1	12	0
8.	2	2	2	27	21	0	1	12	1

```
. stset time, failure(event) time0(time0)
```

```
failure event: event != 0 & event < .
obs. time interval: (time0, time]
exit on or before: failure
```

```
-----
8 total obs.
0 exclusions
```

```
-----
8 obs. remaining, representing
3 failures in single record/single failure data
30 total analysis time at risk, at risk from t = 0
earliest observed entry t = 17
last observed exit t = 27
```

```
. stcox empl, cluster(famid)
[Output omitted]
```

Note that the processes leading to different events within larger-level units (e.g. family) do not have to be the same, as is assumed by the model above or model (0) below. We can have some characteristics of the units within clusters shape the risk accumulation – e.g., we can have different processes for younger/older siblings. We can (1) have that characteristic (sibling order) as a regular variable in the model, or (2) we can include interactions with that characteristic, or (3) we can stratify by that characteristic (e.g. allow different baseline hazard), or (4) both stratify and introduce interactions. For example:

```
(0) stcox empl gender, cluster(famid)
(1) stcox siborder empl gender, cluster(famid)
(2) xi: stcox siborder i.siborder*empl i.siborder*gender, cluster(famid)
(3) stcox empl gender, strata(siborder) cluster(famid)
(4) xi: stcox i.siborder*empl i.siborder*gender, strata(siborder) cluster(famid)
```

### Ordered repeated events

Ordered repeated events are much more common than the unordered repeated events.

There are two main issues that one has to decide when selecting a model for ordered repeated events:

1. Do we assume that (a) there is one continuous process where the risk continues to accumulate regardless of the fact that one event took place, or (b) the risk starts accumulating all over again each time after one event happens?
2. Are the processes shaping the risk (a) the same for the first and each subsequent event or (b) different depending on the number of event?

The answers to these questions determine the data setup.

### Single record data:

First, let us see how the setup works if we have single record data. E.g., the original data may look like this:

```
. list id time mar1 mar2 mar3 mar4 pareduc
+-----+
|      id  time  mar1  mar2  mar3  mar4  pareduc |
+-----+-----+
| 1.      1     5     0     0     0     0     16     |
| 2.      2    10     5     0     0     0     16     |
| 3.      3    11     2     8     0     0     13     |
| 4.      4    12     3     5    11     0     12     |
| 5.      5    12     0     0     0     0     18     |
+-----+-----+
```

### A. Continuous risk accumulation data setup:

If the original data are single record (i.e., no time-varying variables), but we want to look at repeated events, we need to create a separate record for each possible subsequent event. In case of continuous risk accumulation, two setups are possible, depending on whether we think the process depends on the sequential number of event.

(a) If we think the event number does not matter, we set it up like this:

	id	time	mar1	mar2	mar3	pareduc	dur	event
1.	1	5	0	0	0	16	5	0
2.	2	10	5	0	0	16	5	1
3.	2	10	5	0	0	16	10	0
4.	3	11	2	8	0	13	2	1
5.	3	11	2	8	0	13	8	1
6.	3	11	2	8	0	13	11	0
7.	4	12	3	5	11	12	3	1
8.	4	12	3	5	11	12	5	1
9.	4	12	3	5	11	12	11	1
10.	4	12	3	5	11	12	12	0
11.	5	12	0	0	0	18	12	0

We could then svset it and run models:

```
. svset dur, failure(event) id(id)

. stcox pareduc, robust
```

(b) If we think the event number does matter, we have a separate line for each event number regardless how many events that person had:

	id	time	mar1	mar2	mar3	pareduc	dur	event	number
1.	1	5	0	0	0	16	5	0	1
2.	1	5	0	0	0	16	5	0	2
3.	1	5	0	0	0	16	5	0	3
4.	1	5	0	0	0	16	5	0	4
5.	2	10	5	0	0	16	5	1	1
6.	2	10	5	0	0	16	10	0	2
7.	2	10	5	0	0	16	10	0	3
8.	2	10	5	0	0	16	10	0	4
9.	3	11	2	8	0	13	2	1	1
10.	3	11	2	8	0	13	8	1	2
11.	3	11	2	8	0	13	11	0	3
12.	3	11	2	8	0	13	11	0	4
13.	4	12	3	5	11	12	3	1	1
14.	4	12	3	5	11	12	5	1	2
15.	4	12	3	5	11	12	11	1	3
16.	4	12	3	5	11	12	12	0	4
17.	5	12	0	0	0	18	12	0	1
18.	5	12	0	0	0	18	12	0	2
19.	5	12	0	0	0	18	12	0	3
20.	5	12	0	0	0	18	12	0	4

The svset command is the same:

```
. svset dur, failure(event) id(id)
```

But when we run the models, we want to take the number of event into account. We can take it into account in one of the following four ways discussed above:

- (1) `xi: stcox i.number pareduc, robust`
- (2) `xi: stcox i.number*pareduc, robust`
- (3) `stcox pareduc, strata(number) robust`
- (4) `xi: stcox i.number*pareduc, strata(number) robust`

***B. Separate risk accumulation data setup:***

In case of separate risk accumulation, again, two setups are possible, depending on whether we think the process depends on the sequential number of event. Note that here, we only know the timing of marriages, we don't know when the previous marriage ended, so we assume that the person is at risk for the second marriage as soon as they entered their first one. A more reasonable assumption would be to restart risk accumulation after the previous marriage actually ends, whether in divorce or death of spouse.

(a) If we think the event number does not matter, we set the data up like this:

	id	time	mar1	mar2	mar3	pareduc	dur	event
1.	1	5	0	0	0	16	5	0
2.	2	10	5	0	0	16	5	1
3.	2	10	5	0	0	16	5	0
4.	3	11	2	8	0	13	2	1
5.	3	11	2	8	0	13	6	1
6.	3	11	2	8	0	13	3	0
7.	4	12	3	5	11	12	3	1
8.	4	12	3	5	11	12	2	1
9.	4	12	3	5	11	12	11	1
10.	4	12	3	5	11	12	6	0
11.	5	12	0	0	0	18	1	0

Again, the svset command is the same:

```
. svset dur, failure(event) id(id)
. stcox pareduc, robust
```

(b) If we think the event number does matter, we have the same setup but with a number of event variable a separate line for each event number regardless how many events that person had:

	id	time	mar1	mar2	mar3	pareduc	dur	event	number
1.	1	5	0	0	0	16	5	0	1
2.	2	10	5	0	0	16	5	1	1
3.	2	10	5	0	0	16	5	0	2
4.	3	11	2	8	0	13	2	1	1
5.	3	11	2	8	0	13	6	1	2
6.	3	11	2	8	0	13	3	0	3
7.	4	12	3	5	11	12	3	1	1
8.	4	12	3	5	11	12	2	1	2
9.	4	12	3	5	11	12	11	1	3
10.	4	12	3	5	11	12	6	0	4
11.	5	12	0	0	0	18	1	0	1

The svset command is the same:

```
. svset dur, failure(event) id(id)
```

But again, when we run the models, we want to take the number of event into account.

```
(1) xi: stcox i.number pareduc, robust
(2) xi: stcox i.number*pareduc, robust
(3) stcox pareduc, strata(number) robust
(4) xi: stcox i.number*pareduc, strata(number) robust
```

Note that another solution would be to just run separate models for each successive event -- e.g. a separate model for the first marriage, a separate model for the second marriage (here, the analysis time could start with the time of first marriage or with the time of the first divorce/death of spouse), and so on. By doing this, we assume that the processes that govern the risk of each subsequent event are completely different. When we only focused on the first marriage in our earlier examples, we assumed that it was unique and it didn't make sense to mix it with the second marriage. If that's the assumption, then it might be easier to just estimate the models separately.

### Multirecord data:

If the data setup is already multirecord, in the case of separate risk accumulation (which is more common in social sciences), we usually do not need to create extra observations, but we do need to appropriately indicate the durations and event numbers.

#### A. Separate risk accumulation data setup:

In this method, the data setup is similar to the one we used before, but we need to specify that we want to keep using data even after failure occurs as well as specify that we only want the actual transition to marriage to count as an event, not the fact of marriage itself:

```
. gen mart=(mar==1)

. by id: replace mart=0 if mar==1 & mar[_n-1]==1
(6405 real changes made)

. gen dob=interv-newage
(4304 missing values generated)

. by id: egen dob1=mean(dob)

. replace newage=interv-dob1 if newage==.
(4304 real changes made)

. di 365*17
6205

. gen enttime=dob1+6205
```

To be able to distinguish the process for the first marriage, second marriage, etc. if we want to, we need to create a variable identifying the sequential number of transition:

```
. gen period=1

. by id: replace period=period[_n-1]+mart[_n-1] if mart[_n-1]~=.
(8050 real changes made)
```

```
. tab period
```

period	Freq.	Percent	Cum.
1	9,984	55.36	55.36

2	6,934	38.45	93.81
3	1,059	5.87	99.68
4	57	0.32	100.00
-----			
Total	18,034	100.00	

Now to restart time of origin each time an event happens, we modify enttime variable:

```
. sort id period

. by id period: gen seq=_n

. replace enttime=interv0 if period>1 & seq==1
(1223 real changes made)

. by id period: replace enttime=enttime[_n-1] if period>1 & seq>1
(6827 real changes made)

. stset interv, time0(interv0) failure(mart==1) origin(time enttime) exit(time .)

      failure event:  mart == 1
obs. time interval:  (interv0, interv]
exit on or before:  time .
  t for analysis:    (time-origin)
      origin:        time enttime

-----
18034 total obs.
1204 entry time missing (interv0>=.)          PROBABLE ERROR
111 obs. end on or before enter()
-----
16719 obs. remaining, representing
1254 failures in single record/single failure data
6551460 total analysis time at risk, at risk from t =          0
                                         earliest observed entry t =          0
                                         last observed exit t =          7564
```

Note that we did not specify the id variable in this syntax – we’ll specify it using cluster command. That is done to prevent Stata from enforcing uniform time of origin for all observations within a given id value.

We can then run one of the five possible models specified in the single-record example:

```
(0) stcox emp pared, cluster(id)
(1) xi: stcox i.period emp pared, cluster(id)
(2) xi: stcox i.period*emp i.period*pared, cluster(id)
(3) stcox emp pared, strata(period) cluster(id)
(4) xi: stcox i.period*emp i.period*pared, strata(period) cluster(id)

. stcox emp pared, cluster(id)

      failure _d:  mart == 1
analysis time _t:  (interv-origin)
      origin:    time enttime
exit on or before:  time .

Iteration 0:  log pseudolikelihood = -8378.9358
Iteration 1:  log pseudolikelihood = -8367.2904
Iteration 2:  log pseudolikelihood = -8367.2664
Refining estimates:
Iteration 0:  log pseudolikelihood = -8367.2664
```

Cox regression -- Breslow method for ties

```

No. of subjects      =          15184          Number of obs   =          15184
No. of failures     =           1226
Time at risk        =          5722907
Log pseudolikelihood = -8367.2664          Wald chi2(2)    =           28.07
                                                Prob > chi2     =           0.0000

```

(Std. Err. adjusted for 1168 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
emp	.7549793	.0432915	-4.90	0.000	.6747237	.8447808
pared	1.017109	.0067251	2.57	0.010	1.004013	1.030376

```

. xi: stcox i.period emp pared, cluster(id)
i.period      _Iperiod_1-4      (naturally coded; _Iperiod_1 omitted)

      failure _d:  mart == 1
      analysis time _t:  (interv-origin)
                  origin:  time enttime
      exit on or before:  time .

```

```

Iteration 0:  log pseudolikelihood = -8378.9358
...
Iteration 45:  log pseudolikelihood = -8227.0374
Refining estimates:
Iteration 0:  log pseudolikelihood = -8227.0374

```

Cox regression -- Breslow method for ties

```

No. of subjects      =          15184          Number of obs   =          15184
No. of failures     =           1226
Time at risk        =          5722907
Log pseudolikelihood = -8227.0374          Wald chi2(4)    =           .
                                                Prob > chi2     =           .

```

(Std. Err. adjusted for 1168 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
_Iperiod_2	.3334448	.0252443	-14.51	0.000	.2874628	.386782
_Iperiod_3	.2040823	.0490276	-6.62	0.000	.1274434	.3268084
_Iperiod_4	1.23e-20	.	.	.	.	.
emp	.6511501	.0403891	-6.92	0.000	.5766116	.7353243
pared	1.015256	.0081087	1.90	0.058	.9994866	1.031274

We would probably omit the 4<sup>th</sup> spell from these analyses because there are so few of them, but let's keep it for now.

```

. xi: stcox i.period*emp i.period*pared, cluster(id)
i.period      _Iperiod_1-4      (naturally coded; _Iperiod_1 omitted)
i.period*emp  _IperXemp_#      (coded as above)
i.period*pared _IperXpared_#    (coded as above)

      failure _d:  mart == 1
      analysis time _t:  (interv-origin)
                  origin:  time enttime

```

exit on or before: time .

note: \_Iperiod\_2 dropped due to collinearity  
note: \_Iperiod\_3 dropped due to collinearity  
note: \_Iperiod\_4 dropped due to collinearity  
Iteration 0: log pseudolikelihood = -8378.9358  
...  
Iteration 42: log pseudolikelihood = -8221.2998  
Refining estimates:  
Iteration 0: log pseudolikelihood = -8221.2998  
Iteration 1: log pseudolikelihood = -8221.2998

Cox regression -- Breslow method for ties

No. of subjects = 15184 Number of obs = 15184  
No. of failures = 1226  
Time at risk = 5722907  
Log pseudolikelihood = -8221.2998 Wald chi2(9) = .  
Prob > chi2 = .

(Std. Err. adjusted for 1168 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
_Iperiod_3	.0614445	.0543371	-3.15	0.002	.0108578	.3477157
_Iperiod_4	7.02e-19	.	.	.	.	.
emp	.6842908	.0495912	-5.23	0.000	.5936811	.7887296
_IperXemp_2	.7811912	.1196392	-1.61	0.107	.5786235	1.054675
_IperXemp_3	2.329769	1.101368	1.79	0.074	.9223983	5.884467
_IperXemp_4	1.806155	.	.	.	.	.
_Iperiod_2	.2346702	.0598389	-5.68	0.000	.1423669	.3868183
pared	1.005174	.0090548	0.57	0.567	.9875825	1.023078
_IperXpare~2	1.047579	.0226687	2.15	0.032	1.004078	1.092965
_IperXpare~3	1.083824	.0722829	1.21	0.227	.9510213	1.235173
_IperXpare~4	.6650958	.0282291	-9.61	0.000	.6120066	.7227903

. stcox emp pared, strata(period) cluster(id)

failure\_d: mart == 1  
analysis time \_t: (interv-origin)  
origin: time enttime  
exit on or before: time .

Iteration 0: log pseudolikelihood = -7646.2082  
Iteration 1: log pseudolikelihood = -7622.4205  
Iteration 2: log pseudolikelihood = -7622.2489  
Iteration 3: log pseudolikelihood = -7622.2488  
Refining estimates:  
Iteration 0: log pseudolikelihood = -7622.2488

Stratified Cox regr. -- Breslow method for ties

No. of subjects = 15184 Number of obs = 15184  
No. of failures = 1226  
Time at risk = 5722907  
Log pseudolikelihood = -7622.2488 Wald chi2(2) = 49.36  
Prob > chi2 = 0.0000

(Std. Err. adjusted for 1168 clusters in id)

	Robust
--	--------

_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
emp	.6496415	.0402527	-6.96	0.000	.5753501	.7335258
pared	1.014003	.0081899	1.72	0.085	.9980778	1.030183

Stratified by period

```
. xi: stcox i.period*emp i.period*pared, strata(period) cluster(id)
i.period      _Iperiod_1-4      (naturally coded; _Iperiod_1 omitted)
i.period*emp  _IperXemp_#      (coded as above)
i.period*pared _IperXpared_#    (coded as above)
```

```
failure _d: mart == 1
analysis time _t: (interv-origin)
origin: time enttime
exit on or before: time .
```

```
note: _Iperiod_2 dropped due to collinearity
note: _Iperiod_3 dropped due to collinearity
note: _Iperiod_4 dropped due to collinearity
Iteration 0: log pseudolikelihood = -7646.2082
Iteration 1: log pseudolikelihood = -7617.4357
Iteration 2: log pseudolikelihood = -7617.2559
Iteration 3: log pseudolikelihood = -7617.2559
Refining estimates:
Iteration 0: log pseudolikelihood = -7617.2559
```

Stratified Cox regr. -- Breslow method for ties

```
No. of subjects      =      15184      Number of obs      =      15184
No. of failures     =      1226
Time at risk        =      5722907
Log pseudolikelihood = -7617.2559
Wald chi2(6)        =      .
Prob > chi2         =      .
```

(Std. Err. adjusted for 1168 clusters in id)

_t	Haz. Ratio	Robust Std. Err.	z	P> z	[95% Conf. Interval]	
_Iperiod_3	1	.	.	.	.	.
_Iperiod_4	1	.	.	.	.	.
emp	.685358	.0501229	-5.17	0.000	.593835	.7909868
_IperXemp_2	.7751409	.1171163	-1.69	0.092	.5764641	1.042291
_IperXemp_3	1.948304	.9300695	1.40	0.162	.7643907	4.965902
_IperXemp_4	1	.	.	.	.	.
_Iperiod_2	1	.	.	.	.	.
pared	1.00422	.009175	0.46	0.645	.9863976	1.022365
_IperXpare~2	1.045884	.0220148	2.13	0.033	1.003614	1.089935
_IperXpare~3	1.079825	.0704902	1.18	0.239	.9501396	1.227211
_IperXpare~4	1	.	.	.	.	.

Stratified by period

Note that here, the risk for the next marriage starts accumulating right after the first marriage. If we want to only restart the risk once the previous marriage ends, we need to change enttime accordingly and only include those observations when the person is single:

```
. gen marend=0
. by id: replace marend=1 if mar[_n-1]==1 & mar==0
(445 real changes made)
. gen single=.
```

```

(18034 missing values generated)
. replace single=1 if period==1
(9984 real changes made)
. by id period: replace single=1 if period>1 & marend[_n-1]==1
(418 real changes made)

. by id period: replace single=1 if period>1 & single[_n-1]==1
(782 real changes made)

. gen enttime2=enttime

. replace enttime2=interv if marend==1
(445 real changes made)

. by id period: replace enttime2=enttime2[_n-1] if period>1 & single==1
(1200 real changes made)

. stset interv, time0(interv0) failure(mart==1) origin(time enttime2) enter(single==1)
exit(time .)

      failure event:  mart == 1
obs. time interval:  (interv0, interv]
enter on or after:  single==1
exit on or before:  time .
      t for analysis:  (time-origin)
                   origin:  time enttime2

```

```

-----
18034 total obs.
1204  entry time missing (interv0>=.)          PROBABLE ERROR
556   obs. end on or before origin()
-----
16274 obs. remaining, representing
1254  failures in single record/single failure data
6389558 total analysis time at risk, at risk from t =          0
                                         earliest observed entry t =          0
                                         last observed exit t =          7564

```

Again, we should not forget to cluster by id when running the models with this setup.

### B. Continuous risk accumulation data setup:

If we assume that risk starts accumulating in the beginning of time and does not restart with each subsequent event, we need a very different data setup. Here, we would need to duplicate the records, creating one for each event, when the data are still in the wide form. We should make sure to create an event number variable so that it is clear which set of records belongs to which event number. After that, we could reshape the dataset into the long form to complete the data management, including the creation of an indicator whether the corresponding event actually took place.

Then, in each of these sets, we would only examine whether the corresponding event happened and ignore other events. That is, once a failure happened for that specific event number, the whole set of observations for that event number exits the sample (that is, we would specify `exit(failure)` option). Again, we would have to cluster by id when running models rather than as a part of the `stset` command.