

SC706: Longitudinal Data Analysis
Instructor: Natasha Sarkisian

Multi-record data management for event history analysis

So far in dealing with continuous-time models, we worked with the data that only have one record per person. However, frequently we'll have to work with multi-record data. We need multirecord data when we have time-varying independent variables; we also need it if we deal with repeatable events as our outcomes. To illustrate the techniques of working with multirecord data, we'll work with another dataset which is a subset of the dataset you'll deal with for your assignment: marriage.dta.

To make sure we'll have enough memory for our data transformations, we want to allow 100Mb for the data:

```
. set memory 100m
. use http://www.sarkisian.net/sc706/marriage.dta
. gen id=_n
```

The first thing we see is that we have separate variables for each year rather than separate records. The data are in wide format, and we need to convert them into long format.

```
. reshape long mar fexp educ interv 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: fexp79 not found)
(note: educ79 not found)
(note: newage79 not found)
(note: emp79 not found)
(note: enrol79 not found)
(note: mar81 not found)
(note: fexp81 not found)
(note: educ81 not found)
(note: newage81 not found)
(note: emp81 not found)
(note: enrol81 not found)
```

Data	wide	->	long
Number of obs.	1204	->	18060
Number of variables	106	->	21
j variable (15 values)		->	year
xij variables:			
mar79 mar81 ... mar94		->	mar
fexp79 fexp81 ... fexp94		->	fexp
educ79 educ81 ... educ94		->	educ
interv79 interv81 ... interv94		->	interv
newage79 newage81 ... newage94		->	newage
emp79 emp81 ... emp94		->	emp
enrol79 enrol81 ... enrol94		->	enrol

Now, for each year, we have the interview date – we need to convert that information into intervals – that is, use each two consecutive lines for an individual and generate an interval (e.g. from the interview date in 1985 to the date in 1986).

The question is, what should be the value of each variable over such interval? For the time-invariant independent variables, it should be the value from the first observation for each person (and right now such values are filled in into every line). For time-variant variables, we would typically want to get values from the second line from each pair – in terms of our dependent variable (event), we are interested in events that happened by the end of the period, so typically we would also want the values of independent variables by the end of the period (but remember our discussion of different ways to code time-variant independent variables). We also need to make sure that we include the data from the first observation for each person. We'll use `snapspan` command to deal with all these issues. Here's its syntax:

```
snapspan id_var tim_var timechanging_vars, gen(new_begin_date)
```

```
. snapspan id interv fexp educ newage emp enrol, gen(interv0) replace
26 observations have interv==.
either fix them or drop them
by typing drop if interv==.
r(459);
```

If our date variable has missing values, Stata cannot generate intervals – we can either impute those in some way or drop them. For now, we'll drop.

```
. drop if missing(interv)
(26 observations deleted)
. snapspan id interv fexp educ newage emp enrol, gen(interv0) replace
and now it worked.
```

The date variable is unreadable – it is usually coded in days since January 1, 1960 (and if it's not, you might want to recode it that way). (See Cleves et al. textbook for more info on dates.) We want to inform Stata that this is how that's coded:

```
. format interv %d
. format interv0 %d
```

Next, I need to decide on the starting point. The starting point for analysis time is really important – it determines when the subjects start accumulating the risk. It's especially important for parametric models, but it also can affect which cases are used in the analysis! Here, I decided to start when respondent is age 17. Need to generate a variable that contains the (approximate) date when the respondent turns 17.

```
. 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
```

Now we can `stset` the data – although now it's much more complex.

```
. stset interv, id(id) time0(interv0) failure(mar==1) origin(time enttime)
exit(failure)
```

```

            id: id
failure event: mar == 1
obs. time interval: (interv0, interv]
exit on or before: failure
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()
8050 obs. begin on or after (first) failure
-----
8669 obs. remaining, representing
1204 subjects
998 failures in single failure-per-subject data
3514127 total analysis time at risk, at risk from t = 0
                                         earliest observed entry t = 0
                                         last observed exit t = 7564

```

We know that missing entry times are those first observations that we generated. 8050 observations that begin on or after first failure is what we call left-censored observations – they experienced their event before we started observing them.

Now we can use these data in our models, e.g.:

```

. stcox parpres pared black hispanic educ emp fexp, robust
failure _d: mar == 1
analysis time _t: (interv-origin)
origin: time enttime
id: id
Iteration 0: log pseudolikelihood = -6070.6225
Iteration 1: log pseudolikelihood = -5984.7016
Iteration 2: log pseudolikelihood = -5983.2094
Iteration 3: log pseudolikelihood = -5983.2075
Refining estimates:
Iteration 0: log pseudolikelihood = -5983.2075

```

Cox regression -- Breslow method for ties

```

No. of subjects      =          1168          Number of obs      =          7346
No. of failures      =           974
Time at risk         =        2765787
Log pseudolikelihood =    -5983.2075
                                         Wald chi2(7)       =        163.22
                                         Prob > chi2        =         0.0000
                                         (Std. Err. adjusted for 1168 clusters in id)

```

	Robust					
_t	Haz. Ratio	Std. Err.	z	P> z	[95% Conf. Interval]	
parpres	.9966084	.0031329	-1.08	0.280	.9904869	1.002768
pared	1.015492	.0128565	1.21	0.225	.9906037	1.041005
black	.3802374	.0350346	-10.49	0.000	.317414	.4554949
hispanic	.7701662	.0706743	-2.85	0.004	.6433895	.9219237
educ	.9409391	.0156606	-3.66	0.000	.9107401	.9721395
emp	.6765832	.0551938	-4.79	0.000	.5766106	.7938892
fexp	1.002138	.0143569	0.15	0.882	.9743901	1.030676

Note that when working with multirecord data, we should take into account that multirecord nature by adjusting for the fact that the records for each individual are not independent – i.e., adjust for clustering on id by obtaining robust estimates of standard errors (using robust option or cluster option).