**4. Estimation of the (integrated) hazard and survivor functions: Kaplan-Meier product-limit and lifetable methods**

1. **Aim**

The aim of this lesson is to illustrate how to use Stata to estimate (integrated) hazard and survival functions using Kaplan-Meier product-limit and lifetable methods. Again, the data are assumed to be the simple single-spell type considered in earlier Lessons (with right censoring but not left censoring or left truncation).

1. **Introduction: sts and ltable**

Estimation of the Kaplan-Meier empirical hazard and survival functions is done very easily in Stata by using either the **sts** collection of commands or by using the **ltable** (‘lifetable’) command.

Many of the differences between **sts** and **ltable** derive from the underlying assumptions about the nature of the survival time data. With **sts**, survival times are treated as observations on a continuous variable. In the **ltable** case, the technique is based on survival data that have been grouped into intervals (or implicitly assumed to be).

Observe that the continuous time hazard rate is defined with reference to an instant of time (and not a probability), whereas the discrete time hazard rate is a probability, the definition of which refers to an interval of time, by construction. One might try and estimate the continuous time hazard from the slope of the integrated hazard function (a step function): a discrete approximation based on the ‘hazard contribution’ – the change in the integrated hazard over the interval of time between the dates between two successive observed failure times. One might then estimate the hazard rate from the ratio of the hazard contribution to the length of the time interval. However this estimator is known to have relatively poor properties and this has led analysts to focus instead on estimates of a smoothed hazard rate.

The **sts** commands consist of **sts graph**, **sts list**, **sts generate**, and **sts test**. (See **help sts**.) The first two of these are likely to be of most use to you: they provide a graph and listing of the survivor function (estimated by the Kaplan-Meier product-limit method), the integrated hazard function (estimated by the Nelson-Aalen method) plus associated statistics such as standard errors. (Note that Stata refers to the integrated hazard as the ‘cumulative hazard’.) The command **sts** used by itself is a synonym for **sts graph** (and understands all the standard options from **graph**). Estimates of smoothed hazard rates can be derived using **sts graph** and its **hazard** option. This uses kernel smoothing, and there are options to choose kernels and bandwidths other than the default ones.

The **sts** commands can only be used with data which has already been **stset** (see Lesson 3), but this has substantial pay-offs. Once the data have been correctly **stset**, then estimates can be derived very straightforwardly – regardless of whether the data are organised by person or by person-month. (**stset** takes care of any potential complications on that score. See Lesson 3.)

The **ltable** command displays and graphs estimates of survivor functions estimated using lifetable methods. It can be applied to data organised in either person or person-month form. Estimates of the empirical hazard can be shown in a table, but cannot be graphed. (This reflects the issues raised earlier about estimation of continuous hazard rates.) By default, **ltable** uses the so-called actuarial adjustment for the number of subjects at risk (this may vary within intervals if survival times are grouped). To get the unadjusted estimates, corresponding to the Kaplan-Meier assumption, which are also the ones produced by **sts list**, you need to use the **noadjust** option to **ltable**.

I will now work through an illustrative example based on the Cancer data. There are exercises at the end that ask you to work through the same material but with other course data sets.

1. **Illustration using the Cancer data set: (i) sts**

First we first **stset** the data (as in Lesson 3):

su

de

stset studytim , failure(died)

ge id = \_n

lab var id "subject identifier"

We also recode the drug variable (for use later)

. recode drug 1=0 2/3=1

(drug: 48 changes made)

. lab var drug "receives drug?"

. lab def drug 0 "placebo" 1 "drug"

. lab val drug drug

Now we simply:

. sts list

Failure \_d: died

Analysis time \_t: studytime

Kaplan–Meier survivor function

At Survivor Std.

Time risk Fail Lost function error [95% conf. int.]

------------------------------------------------------------------------

1 48 2 0 0.9583 0.0288 0.8435 0.9894

2 46 1 0 0.9375 0.0349 0.8186 0.9794

3 45 1 0 0.9167 0.0399 0.7930 0.9679

4 44 2 0 0.8750 0.0477 0.7427 0.9418

5 42 2 0 0.8333 0.0538 0.6943 0.9129

6 40 2 1 0.7917 0.0586 0.6474 0.8820

7 37 1 0 0.7703 0.0608 0.6236 0.8656

8 36 3 1 0.7061 0.0661 0.5546 0.8143

9 32 0 1 0.7061 0.0661 0.5546 0.8143

10 31 1 1 0.6833 0.0678 0.5302 0.7957

11 29 2 1 0.6362 0.0708 0.4807 0.7564

12 26 2 0 0.5872 0.0733 0.4304 0.7145

13 24 1 0 0.5628 0.0742 0.4060 0.6931

15 23 1 1 0.5383 0.0749 0.3821 0.6712

16 21 1 0 0.5127 0.0756 0.3570 0.6483

17 20 1 1 0.4870 0.0761 0.3326 0.6249

19 18 0 2 0.4870 0.0761 0.3326 0.6249

20 16 0 1 0.4870 0.0761 0.3326 0.6249

22 15 2 0 0.4221 0.0786 0.2680 0.5684

23 13 2 0 0.3572 0.0788 0.2087 0.5083

24 11 1 0 0.3247 0.0780 0.1809 0.4771

25 10 1 1 0.2922 0.0767 0.1543 0.4449

28 8 1 1 0.2557 0.0753 0.1247 0.4093

32 6 0 2 0.2557 0.0753 0.1247 0.4093

33 4 1 0 0.1918 0.0791 0.0676 0.3634

34 3 0 1 0.1918 0.0791 0.0676 0.3634

35 2 0 1 0.1918 0.0791 0.0676 0.3634

39 1 0 1 0.1918 0.0791 0.0676 0.3634

------------------------------------------------------------------------

The second column (beg. total) is the total number of subjects at risk of failure (death) at the time shown in the first column. The third column (Fail) shows the number dying at each time. The fourth column (Net lost) gives the number of subjects censored (and thence no longer entering the risk set). The estimates of the survivor function together with estimates of their statistical significance are shown in the remaining columns.

The table shows that 48% of the sample remained alive after *t* = 17 (with a 95 percent. confidence interval of 0.33, 0.62). You could change the confidence interval shown if you wished using the level option. (This applies to all Stata’s estimation commands: see **help level**.)

Our estimate of the median duration is *t* between 16 and 17. At the end of the (fictional) drug trial which provided the data, only about one fifth of the sample remained alive (four-fifths had died). We can see this directly from a graph of the survivor function derived from **sts graph**:

. sts graph, title("Survivor function, Cancer data (sts)") saving(surv1, replace)

failure \_d: died

analysis time \_t: studytim

**Figure 4.1 Survivor function derived from the Cancer data using sts**



If you wanted instead an estimate of the integrated hazard function, add the **na** option to the command above. (‘na’ stands for Nelson-Aalen, the progenitors of the method used to derive the estimates of the integrated hazard. The estimates could in principle also be derived via the survivor function but the literature has shown that the Nelson-Aalen estimator of the integrated hazard is to be preferred.)

. sts graph, title("Cumulative hazard function, Cancer data (sts)") ///

> na saving(integh1, replace).

**Figure 4.2 Cumulative hazard function derived from the Cancer data using sts**



Using **sts gen**, we can retrieve the estimates shown in the graphs above, together with the estimate of the hazard contribution (defined earlier):

. sts gen s = s

. lab var s "KM survivor function, from -sts gen-"

. sts gen cumh = na

. lab var cumh "NA cumulative hazard function, from -sts gen-"

. sts gen deltach = h

. lab var deltach "Hazard contribution, from -sts gen-"

We can of course also graph these variables against survival time too, and get pictures that look like Figures 4.1 and Figure 4.2. Here follow some examples.

The graphs could have been made more ‘pretty’ using **graph twoway** options for labelling and scaling axes, and so on. Note the use of the **connect(J)** option: this ensures that points are connected in steps, rather than a straight line between points.

**Figure 4.3 Survivor function derived from the Cancer data using -sts gen-**



Eyeballing the (slope of the) cumulative hazard function shown above suggests that the hazard rate might be relatively constant, decline and then increase (remember that the continuous time hazard rate is the slope of the continuous time integrated hazard rate). But this is indeed just ‘eyeballing’, and made harder to tell given the staircase nature of the function arising from real data. We shall have another look using the estimate of the smoothed hazard below.

Estimates of the (continuous time) hazard rate might be derived by dividing each value of the hazard contribution by the length of the time interval between the current failure time and the previous failure time – this is an attempt to look at the slope of the cumulative hazard function. But this usually generates poor estimates. (Things would be easier if failures times could be guaranteed to occur at equal-spaced intervals of short length!) Here’s what the hazard contributions look like (remember, from earlier, that these are the changes in the cumulative hazard over the intervals of time between successive failure times observed in the data):

**Figure 4.4 Hazard contribution function derived from the Cancer data using sts**



One can get an estimate of the smoothed hazard with the command that follows below. The estimate of the hazard uses a kernel-based smoothing of the hazard contributions defined earlier (the change in the cumulative hazard between successive failures). With kernel smoothing, the smoothed value at a given time is based on a weighted average of the values in the neighbourhood of that point. The choice of kernel function determines how much weight is given to observations that are close to the one being considered relative to ones that are far away (most kernel functions give greater weights to closer points), and the ‘bandwidth’ determines how wide the window of observation is when considering which observations are used for the averaging. The larger the bandwidth, the greater the smooth. (Choice of the bandwidth is typically much more important than choice of the kernel function.) I start with the default bandwidth and function.

\* smoothed hazard, default bandwidth (and default kernel)  
sts graph, hazard title("Smoothed hazard function, Cancer data (sts)") saving(smoothhaz1, replace)

**Figure 4.5 Smoothed hazard derived from the Cancer data using sts**



This illustrates how eyeballing the cumulative hazard function can be potentially misleading about the shape of the (smoothed) hazard! Of course, perceptions also depend on the amount of smoothing. Try the following commands and see how the picture differs, noting that the width option specifies the bandwidths:

sts graph, hazard width(2)

sts graph, hazard width(10)

Note that one can look directly at the estimates of the various functions by **list**ing the data. What we want is one estimate presented for each value of *t*, but we know that there are estimates for each subject. So, if we simply listed all the data, we would get more than we really want to see: there would be repeated observations on the variables of interest for each subject. The trick is to generate a variable that will select one observation on *t* for each group of observations with the same *t* value. We use the **egen tag()** function to create such a variable (called *tagt*), and then observations with values of *tagt* = 1. We sort the data by \_t before listing them.

. egen tagt = tag(\_t)

. sort \_t

. list \_t deltach cumh s if tagt == 1

+----------------------------------------+

| \_t deltach cumh s |

|----------------------------------------|

2. | 1 .04166667 .04166667 .95833333 |

3. | 2 .02173913 .0634058 .9375 |

4. | 3 .02222222 .08562802 .91666667 |

6. | 4 .04545455 .13108256 .875 |

7. | 5 .04761905 .17870161 .83333333 |

|----------------------------------------|

9. | 6 .05 .22870161 .79166667 |

12. | 7 .02702703 .25572864 .77027027 |

14. | 8 .08333333 .33906197 .70608108 |

17. | 9 . .33906197 .70608108 |

19. | 10 .03225806 .37132004 .68330427 |

|----------------------------------------|

21. | 11 .06896552 .44028555 .63617984 |

24. | 12 .07692308 .51720863 .58724293 |

25. | 13 .04166667 .5588753 .56277447 |

26. | 15 .04347826 .60235356 .53830602 |

28. | 16 .04761905 .64997261 .5126724 |

|----------------------------------------|

30. | 17 .05 .69997261 .48703878 |

32. | 19 . .69997261 .48703878 |

33. | 20 . .69997261 .48703878 |

34. | 22 .13333333 .83330594 .42210027 |

36. | 23 .15384615 .98715209 .35716177 |

|----------------------------------------|

38. | 24 .09090909 1.0780612 .32469252 |

40. | 25 .1 1.1780612 .29222327 |

42. | 28 .125 1.3030612 .25569536 |

44. | 32 . 1.3030612 .25569536 |

45. | 33 .25 1.5530612 .19177152 |

|----------------------------------------|

46. | 34 . 1.5530612 .19177152 |

47. | 35 . 1.5530612 .19177152 |

48. | 39 . 1.5530612 .19177152 |

+----------------------------------------+

*Stratification*

One might suppose that survival times will vary according to whether the subjects received a drug (drug = 2 or 3) or a placebo (drug = 1). One can use **sts** with the **by(byvar)** option to derive separate estimates for sample subgroups stratified by a variable such as type of drug received:

. sts graph, title("Survivor functions, by drug, Cancer data (sts)") ///

> by(drug) lost saving(surv5, replace)

**Figure 4.6 Survivor function, stratified by drug (Cancer data), derived using sts**



Subjects receiving the placebo appear to have shorter survival times. One can look at the estimates themselves, stratified by subgroup, using the command **sts list, by(drug)**. The resulting table has the same format as the earlier **sts list** table except that there is a separate panel of estimates for each subgroup. An alternative and more succinct listing can be produced with

*Graphical checks regarding model specification*

Our final exercise with data assumed to be continuous is illustration of the graphical checks that one may use to check whether the data are consistent with a proportional hazards model, with a Weibull model, or with a log-logistic model. See the Lecture Notes for details.

The check concerning the validity of the PH assumption is based on graphs of the log of non- parametric estimates of the cumulative hazard against time for different subgroups: recall that we are hoping to see the graphs move in parallel. Throughout, we shall simply compare values according to one explanatory variable *drug*, which has two values corresponding to whether the subject received the drug or the placebo.

The first step in implementation is creation of the subgroup variables referring to the log of the estimated cumulative hazard. We use **sts generate** to create a variable *cumhg* which contains values of the cumulative hazard calculated separately for each group. The ‘**= na**’ part of the command tells Stata that we want the cumulative hazard; the **by(.)** option tells Stata that we want separate calculations for the groups defined by the byvar (two groups in this case corresponding to drug = 0 (placebo recipient) and drug = 1 (drug recipient)). We then compute the log of this variable.

sts gen cumhg = na, by(drug)

ge lch = log(cumhg)

separate lch, by(drug) // creates lch0 lch1

In order to graph the log cumulative variables with separate lines for each group, we need two new variables. The first is equal to lch if drug = 0 and missing otherwise, and the second is equal to lch if drug = 1 and missing otherwise. The **separate** command creates these two new variables, naming lch0 and lch1 by default (the suffix to the name corresponds to the values of drug). You could name the variables using your own variable name stub if you wished, using the **generate()** option.

**Figure 4.7 Informal graphical check of the PH assumption**



Do you think the graphs have the same slope at each survival time (are ‘parallel’)? To my eyes, they do not! Note the difficulty of making a firm decision on the basis of a visual inspection – this issue reminds us that these sorts of checks are relatively informal.

The graphical check of the Weibull specification is implemented similarly (recall that it is a PH model): the variable on the vertical axis is the same, but we graph it against the log of survival time on the horizontal axis. We create the log(time) variable first and then draw the graph.

**Figure 4.8 Informal graphical check of the Weibull assumption**



If the Weibull model is appropriate, the two lines should be parallel straight lines – are they? To my eyes, they are not. (And I hadn’t expected them to be because I was already sceptical of the PH assumption on the basis of the previous graph.)