Workshop on Survival Analysis
Frederick J. Boehmke
Exercise #2: Continuous Duration Models

Our second exercise will focus on analyzing continuous time duration data. We will begin with the basics of Stata's built-in suite of `st` commands for survival time data. Following that we will compare the results of different ways of specifying duration dependence via a handful of parametric models then with the semi-parametric Cox model.

We will use data from King, Alt, Burns and Laver's (1990) study of cabinet dissolution in parliamentary governments.[1] I supplied a Stata copy of the data set here. In particular, we replicate the list of variables in model 2.4.

## Part I

Let's start by learning some of the basic commands for declaring, summarizing, analyzing, and interpreting continuous time duration models. We do this using Stata's built-in suite of `st` commands.

1. Open up the data set `exercise02continuous.dta` and explore its structure.

2. Use the `st` set of commands to declare the data to be survival time and summarize the duration outcome.
   ```
   .stset y
   .stsum
   .stdescribe
   ```

3. Then use the `st` set of commands to graph the duration variable.
   ```
   .sts graph, survival
   .sts graph, hazard
   ```

4. Now use `streg` to estimate the model.
   ```
   .streg invest fract polar numst2 format eltime2 caretk2
   ```
   This is a good time to play around with the different reporting options via `nohr` and `time`.

5. Use the distribution option to select an alternate parametric model.
   ```
   .streg invest fract polar numst2 format eltime2 caretk2, distribution(weibull)
   ```
   Again, play around with the different reporting options via `nohr` and `time` since include duration dependence affects the differences.

6. Use the `stcurve` command to plot the estimates survival and hazard functions.
   ```
   .stcurve, survival
   ```

[1] For those interested, the data for this study are available via Gary King's Dataverse.

`.stcurve, hazard`
We can also use this command to compare the hazard at different values of `polar`, say 3 and 35 using the `at()` option. If you specify more than one `at()` you can compare the hazard for different values in the same figure. Note that the hazard always increases or decreases by a proportional amount.

7. We can estimate a variety of variety of parametric models – try a few. In particular make sure to run the generalized (two parameter) gamma via the `ggamma` distribution.

8. Finally, run a Cox semi-parametric model. Rather than use `streg` we use the `stcox` command. Note that the Cox model does not allow the `time` interpretation.

**Part II**

Now let's account for right censoring. Because of limits on the constitutionally mandated maximum inter-election period (CIEP), some governments are forced to end at a specific point in time. King et al. argue that this effectively right-censors government since they might otherwise last past the CIEP. This also has important consequences for our understanding of duration dependence, as they also discuss and show.

Given our knowledge of these two issues, let's compare the consequences of estimating models with different assumptions. Specifically, let's account for censoring and compare the results of new models to the ones we just estimated.

1. To account for censoring we do so via the `stset` command using the `fail` option.
   ```
   .stset y, fail(cieptw=1)
   .stsum
   .stdescribe
   ```

2. Then use the `sts graph` command to graph the duration variable and compare to before.

3. Let's re-run our parametric and Cox models and compare. The commands do not change since right censoring is already accounted for.

**Part III [Illustration]**

Given the importance of the differing forms of duration dependence in this section we will walk through creating a graph to compare the baseline hazard function across models and with and without accounting for right censoring. This involves using the `saving()` option for the `stcurve` command to output data sets containing the predicted values. We then combine these data sets to overlay them in one plot. For the sake of time I will just demonstrate this and leave you with the code for later exploration.