Survival analysis for Integrative Biologists (Workshop)

SICB, 4 January 2012

Jesse Brunner

Washington State University

We are interested in studying the time to some event, often death. I will use the word "death" throughout, but keep in mind that it is a short hand for many other binary, non-reversible events of interest (e.g., time to metamorphosis, time to mating, the time until a seed is consumed).

A very simple example: rolling dice

Let us consider a simple example where we know the underlying process and can ignore a lot of messy (if interesting) biology. I simply took 100 dice and rolled each one until it turned up a 1, at which point it was "dead." Note that I'm going to call each die an animal so I can avoid talking about dice dying! In any case, I just recorded how many rolls occurred until each animal died.

Let's load in and look at that data, which I've called "full" because we fully observed all of the dice/animals:

```
setwd(file.choose())
full <- read.csv("RollDice_complete.csv")
```

We can get a look at the distribution of rolls until death using the `table` command.

```
table(full)
```

```
      Death
Roll   1
  1   20
  2   11
  3   10
  4   10
  5    9
  6    7
  7    5
  8   10
  9    4
  10   3
  11   3
  12   1
  13   1
  16   1
  17   1
  19   3
  23   1
```

Note: I've used different fonts and colors for `R code` and the `output`. I have excluded the ">" prompt you will see in [R] to make it easier to copy-paste.

You can see that even though each animal had a 1 in 6 chance for dying during any given roll, and so it should take on average 6 rolls (=1/rate = 1/(1/6) to kill each animal, there were some individuals who survived for a long time. The average number of rolls to death is not far from the expected value of 6, though:

```
mean(full$Roll)
```

```
[1] 5.51
```

Because there may be a few individuals that take a really long time to die, the median, which is less affected by outliers, is often a better description of the average

```
median(full$Roll)
```

```
[1] 4
```

*Why isn't the median also close to 1/(1/6) = 6? With a constant rate of death, the distribution of times to death follows an exponential distribution, and while the mean of the exponential is 1/rate, the median is 1/rate*ln(2), which in our case is 4.159. We're pretty darn close to perfect!*

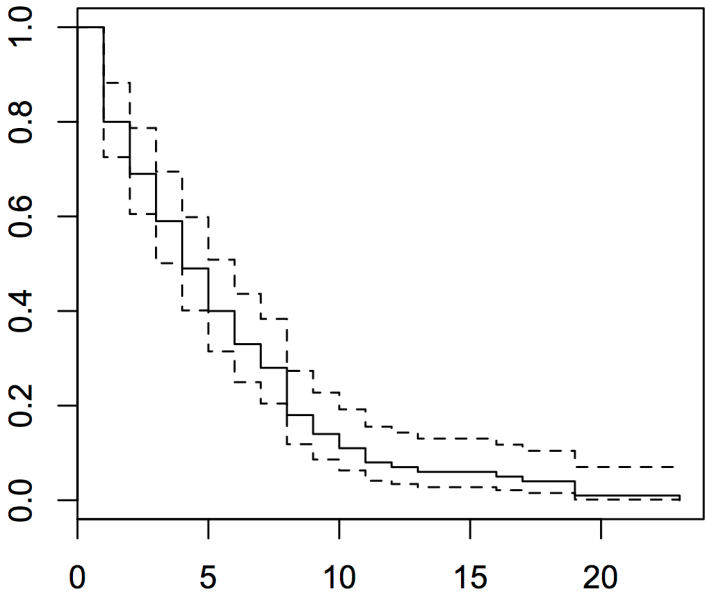
We can jump through the hoops of survival analysis to find the mean or median time to death for this data set, but we get exactly the same answer. For completeness, I've included the steps to do this, but again, it's a bit silly. **You can skip this section and start paying attention to the code in the next example.**

```
library(survival)
s1 <- Surv(full$Roll, full$Death)
survfit( s1 ~ 1)
```

```
Call: survfit(formula = Surv(full$Roll, full$Death) ~ 1)

records   n.max n.start  events  median 0.95LCL 0.95UCL
    100     100     100     100       4       4       6
```

```
plot( survfit( s1 ~ 1)  )
```



Other than creating this nifty graph of the cumulative loss of dice/animals (and the 95% confidence intervals around those estimates in dashed lines), what do we get out of this?

A slightly more realistic example: censored rolling dice

Let us now consider a slightly more realistic and complex example of how we might end up with survival data. We'll keep the same underlying process (roll a one and you're dead), but this time I stopped the experiment after only 10 rolls. We often have to end our experiments after

some reasonable amount of time, even though some events (e.g., deaths) have yet to occur. (Who has time for all of that rolling, or watching a few straggler animals die?)

The dice/animals that were still alive were given a value of Roll = 10 and Death = 0 (i.e., they did not die). Animals that are alive at the end of a study are "**right censored**", or just censored for short. (Left censoring, where you don't know when an individual entered the experiment [e.g., was infected with the agent of interest] is possible, but is beyond this workshop.)

In this case, we know that the animal was alive at day 10, but nothing else. It could have died the next roll, or the roll after that, or who knows when? And just to make things tricky, on the second and seventh rolls (days) I removed three dice/animals from the population. We often do this on purpose when we have to destructively sample a few individuals here or there to get some information from them at a certain point in time (e.g., hormones or infection status), but it can also happen accidentally, when, for instance, the bird you were studying hits a window—it's dead, but that's probably not the cause you were interested in! There are right censored, too.

Let's load in this new experimental data:

```
cens <- read.csv("RollDice_censored.csv")
```

And get a sense of when animals died (the "1" column, the "0" column were the censored animals):

```
table(cens)
```

```
      Death
Roll   0  1
  1    0 16
  2    3 16
  3    0 10
  4    0 10
  5    0  5
  6    0  8
  7    3  7
  8    0  4
  9    0  4
  10 13  1
```

Again, the column for Death = 0 are the censored animals, the Death = 1 column is for those that actually died.

So here is where things get tricky. We want to know the average time to death, but we need to decide whether to include in our calculations those animals that were censored (either in the middle or at the end of the experiment). If we do, we are acting as if they died on the tenth roll, but they did not; they might have survived a long, long time!

```
mean(cens$Roll)
```

```
[1] 4.75
```

```
median(cens$Roll)
```

```
[1] 4
```

If we exclude them, we ignore the fact that

You may have noticed that this median is exactly what we had in the previous example, so maybe censoring is not such a big deal afterwards. Actually, it will depend a great deal on how many animals are censored and when, as well as the distribution of times to death. I wouldn't trust this happy accident to repeat itself in your data!

there were animals that did not die in the 10 days of the experiment. And we are essentially throwing away data!

```
mean(subset(cens, Death == 1)$Roll)
```

```
   [1] 3.925926
```

```
median(subset(cens, Death == 1)$Roll)
```

```
   [1] 3
```

Either way, we bias our estimates, sometimes by quite a lot. Survival analyses can account for censored data, and thus use all of the data and avoid biased estimates. They can also yield some useful information on the rate at which your events occur, which can be terribly useful

(One other reason not to use a normal regression or anova on times to death is that these methods assume a normal distribution. The distribution of times to death is rarely normal (in this example, fir instance, it is exponential) and sometimes not knowable, or even of interest. Survival analysis side-steps these issues.)

Survival analysis: Kaplan-Meier estimates and curves

So let's dig in, then. If you haven't already, you'll want to load the survival package

```
library(survival)
```

The first thing we need to do in [R] is to turn our data into a *survival object*, a different sort of data format that takes into account both when an event occurred and whether the individual was censored or not. Note the capital "S" on the `Surv()` function. It takes the time to event in the first position, and then the censoring/event code. By default, "1" means an event occurred and "0" means the individual was censored. You can specify a different coding system and other stats packages (e.g., JMP) reverse this coding.

```
s2 <- Surv(cens$Roll, cens$Death)
```

```
  [1]  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
 [16]  1   2   2   2   2   2   2   2   2   2   2   2   2   2   2
 [31]  2   2   2+  2+  2+  3   3   3   3   3   3   3   3   3   3
 [46]  4   4   4   4   4   4   4   4   4   4   5   5   5   5   5
 [61]  6   6   6   6   6   6   6   6   7   7   7   7   7   7   7
 [76]  7+  7+  7+  8   8   8   8   9   9   9   9   10  10+ 10+ 10+
 [91] 10+ 10+ 10+ 10+ 10+ 10+ 10+ 10+ 10+ 10+
```

This looks just like the list of times to death, only now the censored individuals have a plus after the time when they were censored to indicate that they survived for at least this many rolls.

Next we need to estimate the survival function for these data using the Kaplan-Meier (K-M) estimator of the survival function. This is actually quite simple to calculate. For a given time, $t = i$, it is the number at risk just before time i (i.e., those that have survived and have not been censored by a given time—n_i) minus those that die at time i (d_i), divided by the number at risk. In other words, it is the probability that an individual who has survived up to time $t = i$ will survive beyond time $t = i$. The K-M estimators is:

$$S(t) = \prod_{t_i < t} \frac{n_i - d_i}{n_i}$$

To calculate the probability of surviving to time, *t*, we simply multiply the (conditional) probabilities for all of the previous intervals and the interval of interest.

While we can calculate this by hand, the [R] function `survfit()` will do it all for us. This function was really set up to compare survival among groups, so we have to use the formula interface. (that's the what the tilde, "~" means). We'll get there, but for now, we're saying that there are no groups, just an intercept (`~1`).

```
survfit(s2~1)
```

```
  Call: survfit(formula = s2 ~ 1)

  records   n.max n.start  events  median 0.95LCL 0.95UCL
      100     100     100      81       4       3       6
```

Notice that we recovered the "correct" median of ~4 rolls. We can get more information about the survival probability through time, $S(t)$, and confidence intervals on those probabilities using the `summary()` function.

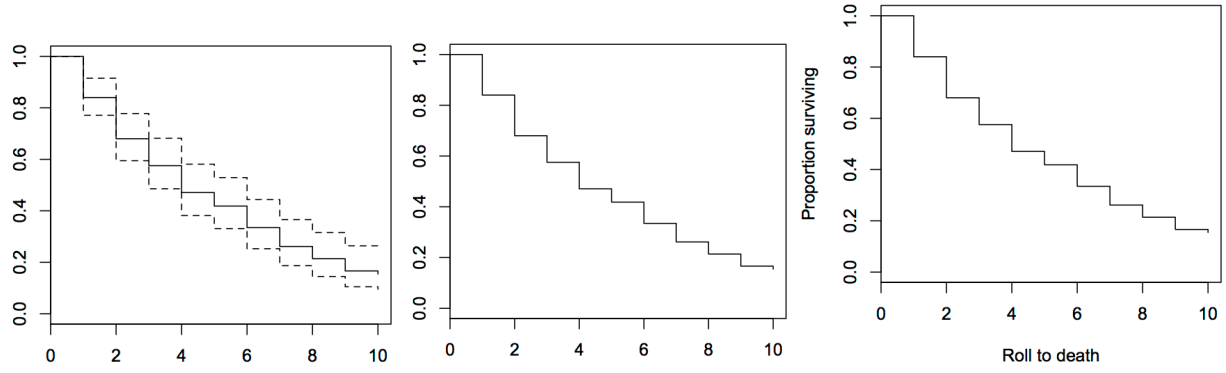
```
summary(survfit(s2~1))
```

```
  Call: survfit(formula = s2 ~ 1)

   time n.risk n.event survival std.err lower 95% CI upper 95% CI
      1    100      16    0.840  0.0367        0.7711        0.915
      2     84      16    0.680  0.0466        0.5945        0.778
      3     65      10    0.575  0.0498        0.4855        0.682
      4     55      10    0.471  0.0506        0.3814        0.581
      5     45       5    0.418  0.0501        0.3310        0.529
      6     40       8    0.335  0.0480        0.2527        0.443
      7     32       7    0.262  0.0448        0.1870        0.366
      8     22       4    0.214  0.0425        0.1450        0.316
      9     18       4    0.166  0.0391        0.1050        0.264
     10     14       1    0.155  0.0381        0.0953        0.251
```

And we can make a plot of this K-M survival curve. While the goal here is not making pretty pictures (this is very do-able in [R], but can take a bit of futzing), I've shown you a few iterations making more useful versions.

```
plot(survfit(s2~1))
plot(survfit(s2~1), conf.int = F)
plot(survfit(s2~1), conf.int = F, xlab = "Roll to death", ylab = "Proportion
   surviving")
```



Adding to our simple example: coin flips

We now want to work with multiple groups or types of individuals. To keep our example simple I have just added the survival of a different group (species?) to our data: coins! I flipped 100 coins ten times, as above, but if they got a head then they were dead. Again, I also removed three coins from the population at times 2 and intended to at time 7, but there none left! So these data are very, very similar, only we expect each coin to have a 50:50 shot of dying after each coin flip, three-times greater than that for the dice rolls.

Go ahead and read in the data and take a look at it, as before:

```
dicecoin <- read.csv("DiceCoin.csv")
table(dicecoin)
```

Create a K-M estimates of survival (all in one-fell swoop this time):

```
dc <- survfit(Surv(Time, Death) ~ Type, data = dicecoin)
dc
```

```
  Call: survfit(formula = Surv(Time, Death) ~ Type, data = dicecoin)

            records n.max n.start events median 0.95LCL 0.95UCL
  Type=Coin     100   100     100     97      2       1       2
  Type=Die      100   100     100     81      4       3       6
```

Now, let's plot the K-M curves.

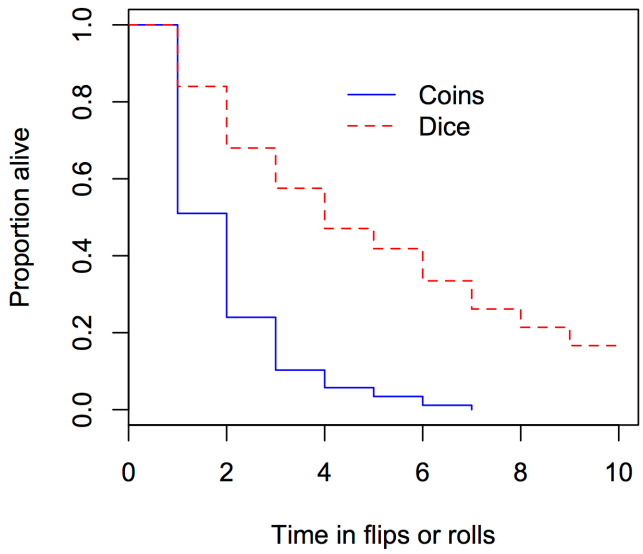
```
plot(dc)
```

OK. There's a bit of a problem here. We cannot differentiate the lines for the two types of animals. We'll add color (`col = c("blue", "red")`) and different line types (`lty = 1:2`) to make this more transparent, along with axis labels.

```
plot(dc, lty = 1:2, col = c("blue",
  "red"), xlab = "Time in flips or
  rolls", ylab = "Proportion alive")
```

Let's add a legend for clarity. Note: `bty = "n"` means do not put a box around the legend.

```
legend(4, 0.9, c("Coins", "Dice"), lty
  = 1:2, col = c("blue", "red"), bty =
  "n")
```



Survival analysis: The log-rank or Mantel-Haenszel test

Now *know* that these two survival curves are different since we know the actual survival probabilities and can see that the curves are no where near each other. But it would be good to have a more formalized way to do this. Our first step is the log-rank test, which is essentially a contingency table approach. We just calculate the number of individuals at risk in each group at each time and the number of events that occur. If there is no difference between the groups then the number of events in, say, group A should be predicted by the overall proportion of events

(regardless of groups) times the number of individuals that are in group A. It is almost identical in logic to finding the expectations for a chi-square test. The contingency table is:

	Group A	Group B	Total
Event	d_{Ai}	d_{Bi}	d_i
No Event	$n_{Ai} - d_{Ai}$	$n_{Bi} - d_{Bi}$	$n_i - d_i$
At Risk	n_{Ai}	n_{Bi}	n_i

where i refers to the time t_i and subscripts A and B are the two groups

The *expected* number of deaths at time i in group A, if both groups are identical in terms of their survival functions, is simply, $\hat{e}_{Ai} = (d_i \times n_{Ai})/n_i$. We can then see how the expected number of deaths compares to the *actual* number of deaths in group A at time i, and do repeat this over each of the $\boldsymbol{m}$ times. (Since we only have two groups, we don't need to do this for both groups... deviation in one group means the deviation in the other.) The actual test statistic is:

$$Q = \frac{\left(\sum_{i=1}^{m} d_{Ai} - \sum_{i=1}^{m} \hat{e}_{Ai}\right)^2}{\sum_{i=1}^{m} \hat{V}(\hat{e}_{Ai})},$$

where V is the variance of the expected number. This is chi-square distributed with 1 d.f.

Again, we could do this by hand without much trouble, but [R] can do all of the math for us and probably make fewer mistakes:

```
survdiff(Surv(Time, Death) ~ Type, data = dicecoin)
```

```
  Call:
  survdiff(formula = Surv(Time, Death) ~ Type, data = dicecoin)

              N Observed Expected (O-E)^2/E (O-E)^2/V
  Type=Coin 100       97     57.7      26.7      65.3
  Type=Die  100       81    120.3      12.8      65.3

   Chisq= 65.3  on 1 degrees of freedom, p= 6.66e-16
```

The test shows that there is a significant different in these survival curves, which is reassuring. As you might guess, the log-rank test can only compare ≥2 distinct curves or groups; it cannot handle continuous variables or individual covariates. Moreover, it doesn't provide a sense of the magnitude of the difference in survival. So it is a useful tool, but pretty limited. Let us move on.

Survival analysis: The Cox proportional hazard model

The Cox proportional hazard model is a much more general and useful way to test for differences in survival among groups, or even among individuals with continuous covariates. It works very much like a regression. In fact it *is* a regression model, with a key difference. A regression model has a baseline value based on the mean of one group or where the covariate(s) equal zero (i.e., the intercept) and then tests whether being in other groups or having non-zero values of the covariate changes the response variable. The Cox PH has a baseline *hazard* function, again based on one group. It then tests whether individuals in that other group have a higher or lower hazard than the baseline. So what is hazard?

The hazard function or hazard rate, $h(t)$, is the instantaneous rate at which we expect events (i.e., death) to occur. Note that this is not a conditional probability as was $S(t)$—it can be greater than

1 for instance—but rather it is interpreted as the expected number of events per individual per unit time. It is worth noting some relationships between the hazard function, survival function, and the probability density or distribution function (pdf)—the statistical distributions we're used to seeing (e.g., the normal curve, the lognormal curve, the exponential curve).

Hazard function, $h(t)$—instantaneous rate of events at time t.

Survival function, $S(t)$—probability of surviving beyond time t.

Probability density or distribution function (PDF), $f(t)$—essentially the expected distribution of times to death

These are all related to each other in fairly simple, often very useful ways…

$$h(t) = \frac{f(t)}{S(t)} = -\frac{\partial \ln S(t)}{\partial t}$$

$$f(t) = S(t)h(t)$$

$$S(t) = \exp\left[-\int_0^t h(t)\right] = \exp[-H(t)]$$

So this hazard function describes the rate at which we expect events (deaths) to occur through time. Often this hazard function is complex and hard to specify. The cool think about the Cox is that it just estimates a baseline hazard function, $h_0(t)$, from the data (in a nonparametric way) and then focuses on the differences between the baseline group and other groups (or individuals with other covariates). The hazard for an individual with predictors x_1, x_2, … is $h_0(t)e^{\beta_1 x_1 + \beta_2 x_2 + \cdots}$.

Again, if $h_0(t)$ is the baseline hazard (e.g., being a coin), then we want to see whether being in a different group (e.g., a die) changes the hazard. The exponent part of this reduces to 1 if none of those predictors matter, otherwise they change the hazard by some multiple (e.g, twice or half of the baseline). What we are really testing is the proportional change in hazard with these different predictors.

Embedded in this construction is the assumption that the proportional difference in hazard between groups is constant through time. This is a key assumption that bears testing and can sometimes limit the utility of this model, although there are some tricky ways to get around this.

In any case, the code to conduct a Cox regression is very similar to any other regression in [R].

```
dc.cox <- coxph(Surv(Time, Death) ~ Type, data = dicecoin)
summary(dc.cox)
```

```
  n= 200, number of events= 178
             coef exp(coef) se(coef)      z Pr(>|z|)
  TypeDie -1.3479    0.2598   0.1725 -7.814 5.55e-15 ***
  ---
  Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

          exp(coef) exp(-coef) lower .95 upper .95
  TypeDie    0.2598      3.849    0.1853    0.3643

  Concordance= 0.677  (se = 0.03 )
  Rsquare= 0.27   (max possible= 1 )
  Likelihood ratio test= 63.07  on 1 df,   p=1.998e-15
  Wald test            = 61.06  on 1 df,   p=5.551e-15
  Score (logrank) test = 67.91  on 1 df,   p=2.22e-16
```

Once again, we find a significant difference in the *hazard* of death between these groups. The value of exp(coef) = 0.2598 is the hazard ratio for the dice. It says that dice have about one-quarter the hazard of coins. Since the confidence interval on this hazard ratio does not overlap with 1 (which would mean that there is no difference in hazard), we can be confident that this estimated difference is not due to random chance.

It is sometimes useful to extract the baseline hazard (here, the hazard through time of coins).

```
basehaz( dc.cox )
```

```
        hazard time
1  0.3401289    1
2  0.7302868    2
3  1.0999080    3
4  1.4558872    4
5  1.6903325    5
6  2.1360030    6
7  2.6457717    7
8  3.0297384    8
9  3.5075668    9
10 3.6477072   10
```

This is somewhat revealing: since we are flipping coins the hazard should remain constant (and ~50%) through time, but the Cox model is estimating an increasing hazard. To "properly" estimate the constant hazard, we will need to move on to different methods. Still, the overall conclusions about differences between coins and dice are robust.

Survival analysis: Parametric accelerated failure time models

Sometimes we actually know the hazard or PDF of the process (e.g., you have good data on the distribution of time-to-death or have a strong reason to think that it, like our coins and dice example, takes on a particular form). Or perhaps we can fit different parametric distributions and see which is best. This brings us to parametric survival models, specifically what are called accelerated failure time models. The key here is that we assume a particular statistical distribution and then we make the *scale* of this distribution (how stretched it is, left-to-right) a function of the parameters of interest. By stretching or scrunching, we are essentially slowing or accelerating time, hence the name.

The advantages of accelerated failure models include being able to extrapolate the hazard beyond the time observed. Also, since we are modeling actual hazards, not certain ratios between groups, we can more easier account for (and interpret) the ways in which hazards are affected by parameters.

Common distributions used in survival analysis include the exponential (with constant hazard), the lognormal, and the Weibull, a very flexible distribution and a good starting point if you don't have good reason to use other distributions. In our case, we will use the exponential, since we know that we (should) have a constant hazard.

```
dc.aft <- survreg(Surv(Time, Death) ~ Type, data = dicecoin,
                  dist = "exponential")
summary(dc.aft)
```

```
Call:
survreg(formula = Surv(Time, Death) ~ Type, data = dicecoin,
    dist = "exponential")
```

```
             Value Std. Error    z        p
(Intercept) 0.672      0.102 6.62 3.56e-11
TypeDie     1.097      0.151 7.29 3.21e-13

Scale fixed at 1

Exponential distribution
Loglik(model)= -386.5   Loglik(intercept only)= -412.6
    Chisq= 52.22 on 1 degrees of freedom, p= 5e-13
Number of Newton-Raphson Iterations: 4
n= 200
```

We can use the parameters to back-calculate the actual hazards for each group. Since our hazards are constant for the exponential model, this is pretty simple. We just exponentiate the negative of the parameter estimate(s). (If your model is complex, the `predict()` functions available for most of these regression-like models will be useful.)

For the coin, the hazard is:

```
exp(-(0.672+0*1.097))
```

```
[1] 0.5106862
```

For the dice it is:

```
exp(-(0.672+1*1.097))
```

```
[1] 0.1705034
```

Both of these estimates are pretty close to what we would expect based on first principles. That is always satisfying!

Summary

So I hope that this has been a useful, informative introduction to survival analyses. This is a big field, with all sorts of special cases for different types of data or designs. It is also an area of active research. The basics that I have presented above should be pretty stable. The next step for you, then, is to settle on a statistics package (I would recommend [R]), figure out which sorts of analyses might be best, and start playing with other worked examples as well as your own data. Once you understand how hazard, survival, and pdfs all relate to each other, the most useful analyses will be the ones you create yourself! Good luck!