Analysis of recurrent events

Terry Therneau

2020-08-14

In response to the article on recurrent events in the July LIDA newsletter, this is a look at producing a portion of the results using the base survival package. Many people do not realize that it has capabilities in this regard. The purpose is to show how the standard package can compliment the more specialized routines of the article.

There are 3 subjects who die on the day of their final readmission. My own inclination would be to move said deaths to time+.1 to break the tie, i.e., treat these as valid readmission events. Second would be to remove only those three events, arguing that each was *only a death* who happened to pass through the hospital while doing so. But we will follow the article's lead and remove all observations for those subjects.

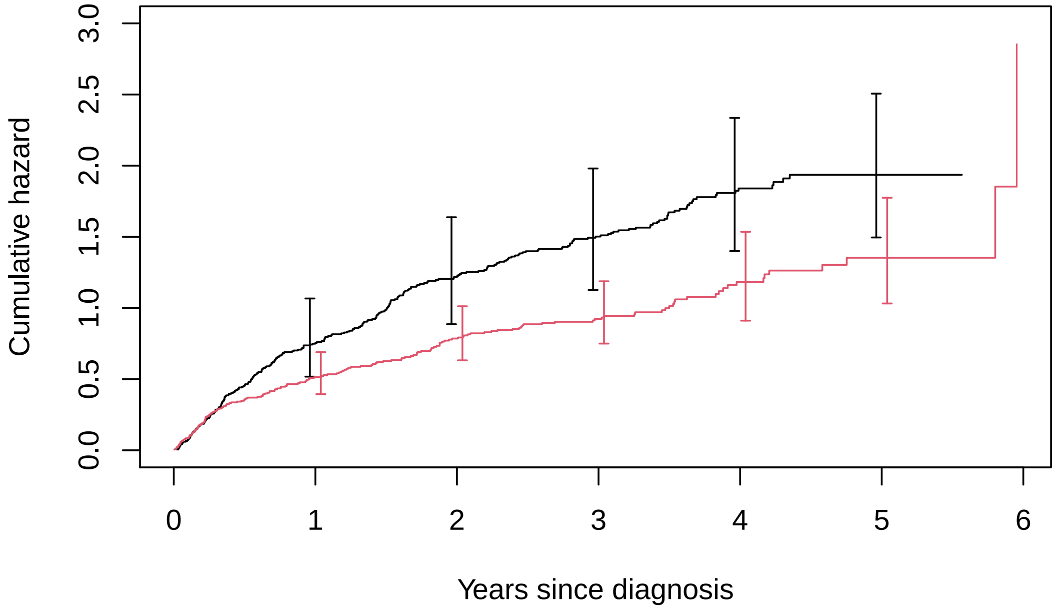
```
library(survival)
data(readmission, package='frailtypack')
both <- subset(readmission, event==1 & death==1)  # those with both
d0 <- subset(readmission, !(id %in% both$id))  # not both
d0$state <- with(d0, factor(event + 2*death, 0:2,
                 c("censor", "admit", "death")))

check <- survcheck(Surv(t.start, t.stop, state) ~1, d0, id=id)
check
```

```
## Call:
## survcheck(formula = Surv(t.start, t.stop, state) ~ 1, data = d0,
##     id = id)
##
## Unique identifiers       Observations        Transitions
##                 400                  852                  852
##
## Transitions table:
##        to
## from    admit death (censored)
##   (s0)    201    36        163
##   admit   251    70        131
##   death     0     0          0
##
## Number of subjects with 0, 1, ... transitions to each state:
##        count
## state     0   1  2  3  4 5 6 8 9 10 11 16 17 22 23
##   admit 199 103 45 21 14 8 4 1 1  1  1  1  0  1  0
##   death 294 106  0  0  0 0 0 0 0  0  0  0  0  0  0
##   (any) 163 108 60 27 19 9 8 0 1  2  1  0  1  0  1
```

Most subjects have 0 or one hospital admissions, but one subject has 22. No one goes from death to another state (which is good). Compute the Nelson-Aalen estimate of the cumulative hazard.

```
hfit <- survfit(Surv(t.start, t.stop, event) ~ chemo, data=d0, id=id)
## plot(ch1, cumhaz=TRUE, col=1:2, conf.times=(1:5)*365.25,
plot(hfit, cumhaz=TRUE, col=1:2, conf.times=(1:5)*365.25,
     xscale=365.25, ylim=c(0,3),
     xlab="Years since diagnosis", ylab="Cumulative hazard")
```

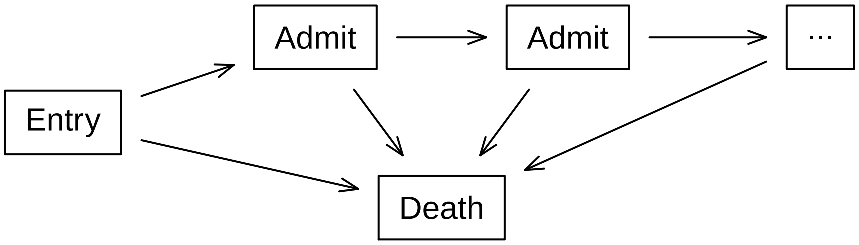


The survfit function computes both the Nelson-Aalen and the Kaplan-Meier. The default is to plot the latter, hence the cumhaz argument. Standard errors of the curves are based on a grouped (infinitesimal) jackknife estimate with each subject as a group. I chose to use confidence bars since it makes the plot a little less busy. The routine sets the default y range based on a confidence band over the entire curve; I use ylim to override this. (And no, I don't think the KM makes sense for this data.) Since each subject is in either one curve or the other, the curves are independent and the variance of the difference is the sum of the variances. Plotting the difference is a nuisance, however. The two separate curves don't share exactly the same time points so we have to force an interpolation.

The survival routine does not include Cook's test for cumulative hazards, nor am I familiar with it. We can fit a joint hazards model using the coxph function.

```
states <- c("Entry", "Admit", "Admit", "Death", "...")
cmat <- matrix(0, 5, 5, dimnames=list(states, states))
cmat[1,2] <- cmat[2,3] <- cmat[3,5] <-1
```

```
cmat[-4,4] <- 1
statefig(cbind((c(1,2,3, 2.5,4)-.5)/4, c(1.5,2,2,1,2)/3), cmat)
```



```
cfit <- coxph(Surv(t.start, t.stop, state) ~ chemo, data=d0,
              id =id)
cfit
```

```
## Call:
## coxph(formula = Surv(t.start, t.stop, state) ~ chemo, data = d0,
##     id = id)
##
##
## 1:2              coef exp(coef) se(coef) robust se      z      p
##   chemoTreated -0.2954    0.7442   0.1421    0.1416 -2.087 0.0369
##
##
## 2:2              coef exp(coef) se(coef) robust se      z     p
##   chemoTreated -0.3154    0.7295   0.1350    0.2452 -1.286 0.198
##
##
## 1:3             coef exp(coef) se(coef) robust se     z       p
##   chemoTreated 1.0203    2.7741   0.3855    0.3772 2.705 0.00682
##
##
## 2:3             coef exp(coef) se(coef) robust se     z     p
##   chemoTreated 0.1985    1.2195   0.2412    0.2552 0.778 0.437
##
##  States:  1= (s0), 2= admit, 3= death
##
## Likelihood ratio test=18.69  on 4 df, p=0.0009059
## n= 852, number of events= 558
```

This shows patients treated with chemotherapy have a lower rate of both initial (.74) and recurrent hospitalization (.73). Their risk of death is much higher (2.3 fold) before hospitalization occurs, but much closer to non-chemo patients after that. The coxph function is using an approach similar to a GEE model with the working indepence assumption, i.e., fit the model ignoring correlation, and then create a valid variance post-fit using the infinitesimal jackknife. Overall, the numeric result has much in common with the more sophisticated approach in the article, while using familiar tools.

We can force a common coefficient estimate for the periods pre and post the the hospitalization state by adding constraints.

```
cfit2 <- coxph(list(Surv(t.start, t.stop, state) ~ 1,
                    1:2 + 2:2 ~ chemo / common,
                    1:3 + 2:3 ~ chemo / common),
               data=d0, id=id)
cfit2
```

```
## Call:
## coxph(formula = list(Surv(t.start, t.stop, state) ~ 1, 1:2 + 
##     2:2 ~ chemo/common, 1:3 + 2:3 ~ chemo/common), data = d0, 
##     id = id)
## 
## 
## 1:2, 2:2          coef exp(coef) se(coef) robust se      z      p
##   chemoTreated -0.3059    0.7365   0.0978    0.1434 -2.134 0.0328
## 
## 
## 1:3, 2:3         coef exp(coef) se(coef) robust se     z      p
##   chemoTreated 0.4491    1.5668   0.1970    0.1985 2.262 0.0237
## 
##  States:  1= (s0), 2= admit, 3= death 
## 
## Likelihood ratio test=15.2  on 2 df, p=0.0005014
## n= 852, number of events= 558
```