Survival Analysis

Randy Johnson

3/9/2017

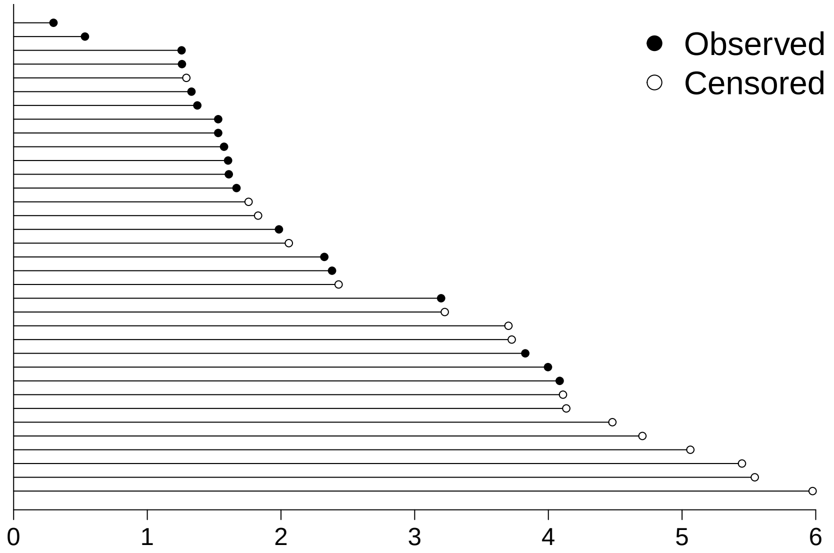
Setup

```r
library(car)
library(survival)
library(tidyverse)
theme_set(theme_classic() +
          theme(axis.line.x = element_line(color = 'black'),
                axis.line.y = element_line(color = 'black'),
                text = element_text(size = 20)))

# colorbline palette
cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73",
                "#F0E442", "#0072B2", "#D55E00", "#CC79A7")

# load the multiplot function
source('../Scripts/multiplot.R')
```

Survival: Definition

Survival time is the amount of time that elapses from a common starting piont until an event occurs. Examples of events include: time to HIV acquisition, time to recurrence of cancer, time to successful pregnancy.

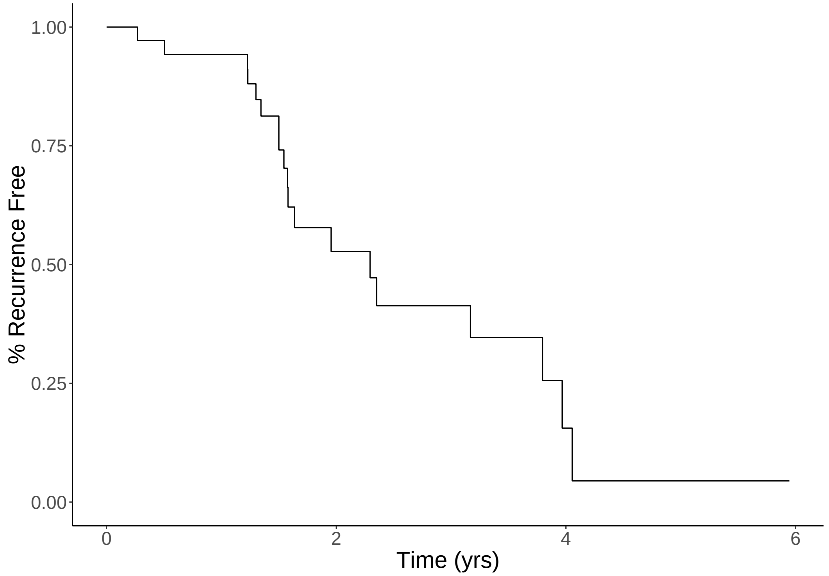
Survival: Graphical example of raw survival data (GBSG)



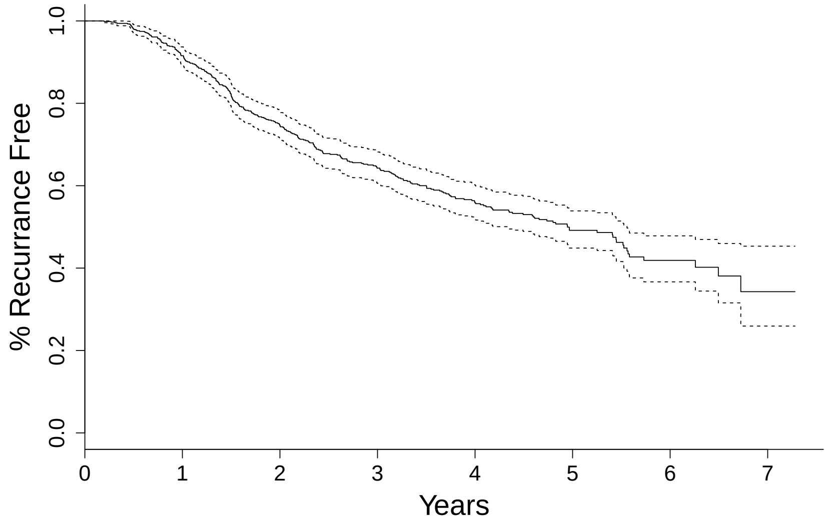
Kaplan-Meier Curves: Creation

```r
# derivation of a survival curve, the hard way
curve <- data.frame(t = 0, # time
                    s = 1) # survival proportion

# go through random sample of 65 individuals
for(i in 1:dim(gbsg.sub)[1])
{
  # proportion who survived until now
  curve <- bind_rows(curve,
          data_frame(t = gbsg.sub$t[i],
                     s = min(curve$s)))

  # if there is an event, decrease survival
  if(gbsg.sub$d[i] == 1)
  {
    t.curr <- gbsg.sub$t[i]
    prop_lost_to_event = 1 / dim(filter(gbsg.sub, t >= t.curr))[1]
    curve <- bind_rows(curve,
            data_frame(t = t.curr,
                       s = min(curve$s) - prop_lost_to_event))
  }
}
```

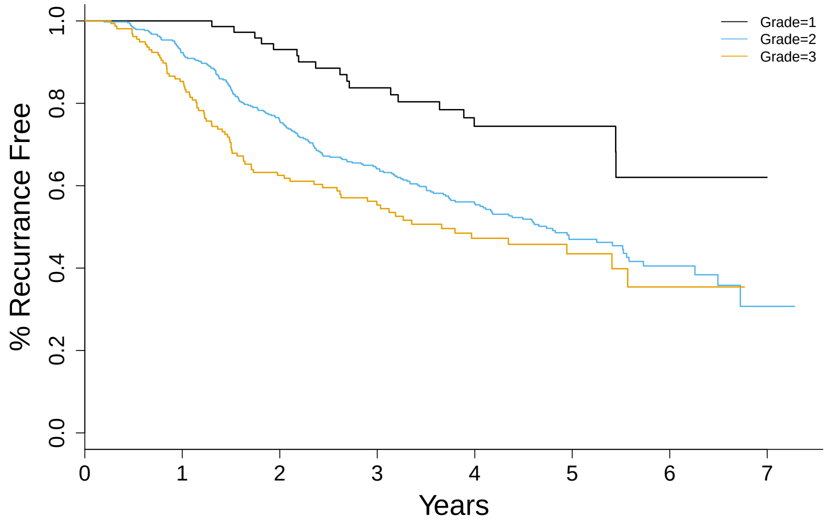
Kaplan-Meier Curve: Creation



Kaplan-Meier Curves: Full GBSG data set



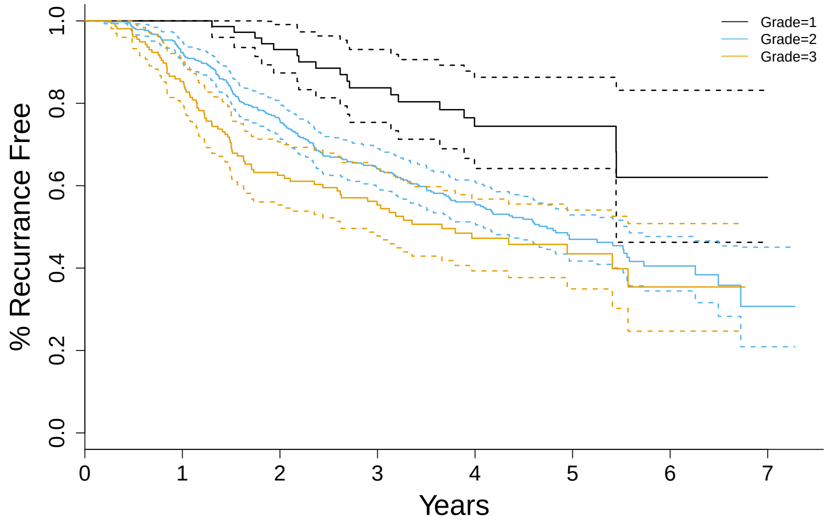
Kaplan-Meier Curves



Kaplan-Meier Curves: Code

```r
# create survival variable
gbsg$surv <- Surv(gbsg$t, gbsg$d)

# plot single curve
survfit(surv ~ 1, data = gbsg) %>%
  plot(bty = 'l', cex.axis = 1.5, ylab = '% Recurrance Free',
       cex.lab = 2, xlab = 'Years')

# plot curves by tumor grade
survfit(surv ~ grade, data = gbsg) %>%
  plot(bty='l', col=cbbPalette[c(1,3,2)], lwd=1.5, cex.lab=2,
       cex.axis=1.5, ylab='% Recurrance Free', xlab='Years')

legend('topright', c('Grade=1', 'Grade=2', 'Grade=3'),
       col = cbbPalette[c(1,3,2)], lty = 1, bty = 'n')

# noisy plot by tumor grade
survfit(surv ~ grade, data = gbsg) %>%
  plot(bty='l', col=cbbPalette[c(1,3,2)], conf.int=TRUE, lwd=1.5
       cex.axis=1.5, cex.lab=2, ylab='% Recurrance Free', xlab='
```

Kaplan-Meier Curves



Survival Analysis

The analysis of survival until a particular event of interest needs to take 4 key measures into account (with definitions for the GBSG data set):

- Target event (recurrence of breast cancer)
- Time origin (tumor resection)
- Time Metric (years - continuous)
- Censoring Causes (mostly at end of study? - would need to look more into this)

Survival Analysis: Time Metric

The time metric can fall two different categories, continuous or discrete. There are four main ways in which we will get discrete time values:

- Truly descrete: Events that may only occur at discrete time points (e.g. graduation time).
- Partially discrete: These events can occur continuously, but tend to clump into discrete groups (e.g. teachers leaving jobs at the end of the year).
- Discrete due to measurement & data collection constraints: These event times are continuous, but we only discrete times are available to us (e.g. CD4 measurements, or recall of event times on a questionare).
- Aggregation: Summary of incident events over short periods of time (e.g. clinical events since last visit).

Survival Analysis: Censoring Causes

We need to be aware of the cause of censoring whenever possible. One importaint thing to watch out for is emmigrative selection bias, when the risk among individuals being censored is somehow different than among those not censored (e.g. a side effect of the treatment is myocardial infarction, that causes patients to drop out of the study).

- Administrative censoring: Censoring at the end of the study (no worries about emmigrative selection bias).
- Drop-out or Loss to follow-up: Emmigrative bias must be addressed, but need not be a problem.
- Competing risks: This can cause emmigrative bias, and the assumption that un/censored groups are similar is indefensible.

Survival Analysis: Hazard Function

The discrete hazard function can be approximated by the following formula when no censoring or late entries exist.

$$\begin{aligned} h(t) &= \mathrm{P}(T = t | T \geq t) \\ &\approx \frac{\#\ \textit{events during time period } t}{\#\ \textit{individuals surviving up to time period } t}. \end{aligned}$$

Survival Analysis: Survival Function

The discrete survival function is the probability of surviving until time t without the event occuring.

$$S(t) = \prod_{i=0}^{t} 1 - h(i)$$

Cox Proportional Hazards Model

- Response: Continuous, time to event
- Interpretation:
  - $\beta_{1\ldots n}$ are the log hazard ratios comparing the likelihood of an event in those exposed to X to the reference group.
- Caveats: Hazards are assumed to be proportional. This can be checked using the `cox.zph()` function or by looking at the log cumulative hazard function. This can be done in R by using the fun = 'cloglog' option in `plot.survfit` (see ?plot.survfit for more details). The baseline hazard, though, is arbitrary and isn't even estimated. One important thing to keep in mind when analyzing survival data is that late entries should have thier immortal person time censored from the analysis. For time varying markers, we can split an individual's time into pieces and include later time periods as late entries.

$$\log h(t) = \log h_0(t) + \beta_1 X_1 + \cdots + \beta_n X_n$$

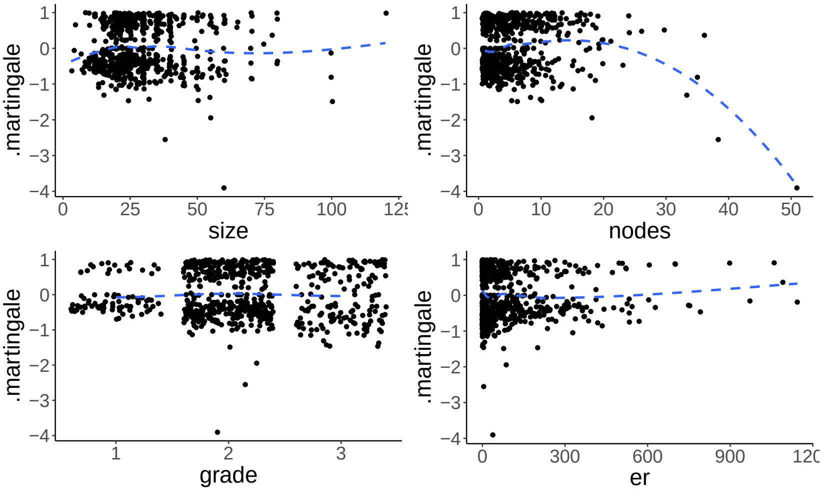
Cox Proportional Hazards Model: Example

```r
(model <- coxph(surv ~ size + grade + nodes + er, data = gbsg)) %>%
  summary()
```

```
## Call:
## coxph(formula = surv ~ size + grade + nodes + er, data = gbsg)
##
##   n= 686, number of events= 299
##
##              coef  exp(coef)   se(coef)      z Pr(>|z|)
## size   0.0061306  1.0061495  0.0038223  1.604    0.109
## grade  0.4011126  1.4934854  0.1025339  3.912 9.15e-05 ***
## nodes  0.0525033  1.0539060  0.0074769  7.022 2.19e-12 ***
## er    -0.0006043  0.9993958  0.0004321 -1.399    0.162
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##       exp(coef) exp(-coef) lower .95 upper .95
## size     1.0061     0.9939    0.9986     1.014
## grade    1.4935     0.6696    1.2216     1.826
## nodes    1.0539     0.9489    1.0386     1.069
## er       0.9994     1.0006    0.9985     1.000
##
## Concordance= 0.66  (se = 0.018 )
## Rsquare= 0.099   (max possible= 0.995 )
## Likelihood ratio test= 71.61  on 4 df,   p=1.044e-14
```

Cox Proportional Hazards Model: Linearity Assumption

The Martingale plot should have a flat trend line. Most of these look OK, but the nodes should probably be log transformed.



Cox Porportional Hazards Model: Linearity Assumption

R code:

```r
gbsg$.martingale <- residuals(model)

multiplot(
  ggplot(gbsg, aes(size, .martingale)) +
    geom_jitter() +
    geom_smooth(linetype = 2, se = FALSE),

  ggplot(gbsg, aes(grade, .martingale)) + geom_jitter() + g

  ggplot(gbsg, aes(nodes, .martingale)) + geom_jitter() + g

  ggplot(gbsg, aes(er, .martingale)) + geom_jitter() + geom

  cols = 2)
```

Cox Proportional Hazards Model: Proportionality

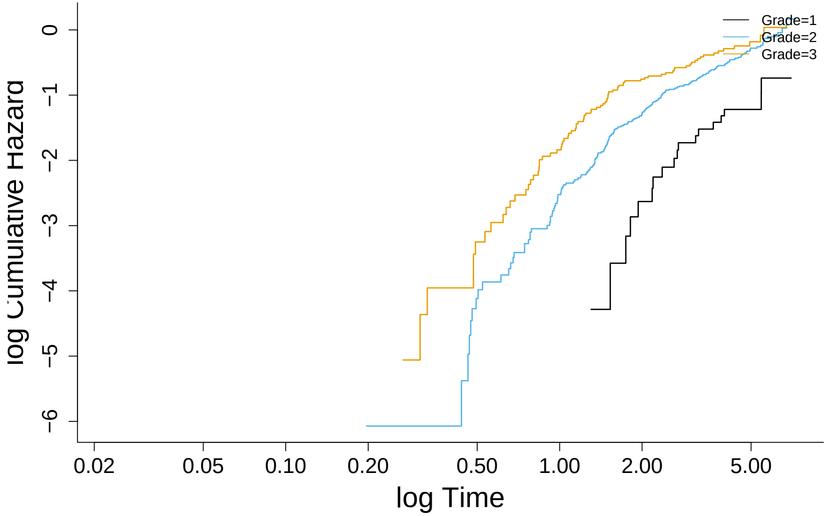
Proportionality assumption seems to be violated for grade and er measure.

```r
cox.zph(model)
```

```
##             rho  chisq       p
## size    -0.0279  0.222 0.63718
## grade   -0.1734  7.516 0.00611
## nodes    0.0678  0.786 0.37536
## er       0.1005  3.810 0.05096
## GLOBAL       NA 13.161 0.01051
```

Cox Proportional Hazards Model: Proportionality

This isn't really necessary, but is interesting to take a look at. We will use the same code as above, but use the `fun='cloglog'` argument. These lines should be roughly parallel.



Cox Proportional Hazards Model: Collinearity

We are familiar with this function. There are some potential problems with it, but as long as the VIFs are greater than 1, we can interpret them as before.

```
vif(model)
```

```
## Warning in vif.default(model): No intercept: vifs may not be
```

```
##     size    grade    nodes       er
## 1.173467 1.008437 1.168106 1.012601
```