Biostatistics Task 2

Danyu Zhang & Daniel Alonso

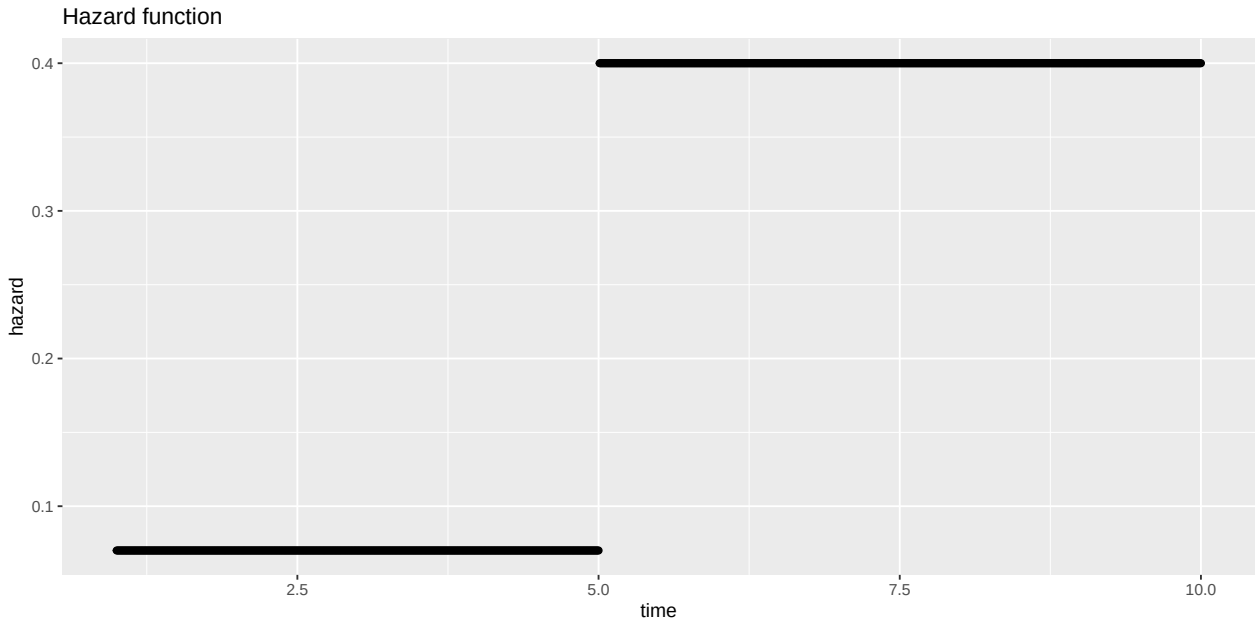
May 28th, 2021

```
library(ggplot2)
library(survival)
library(ggfortify)
library(coin)
library(MASS)
library(survminer)
```

Exercise 1

Hazard function plot

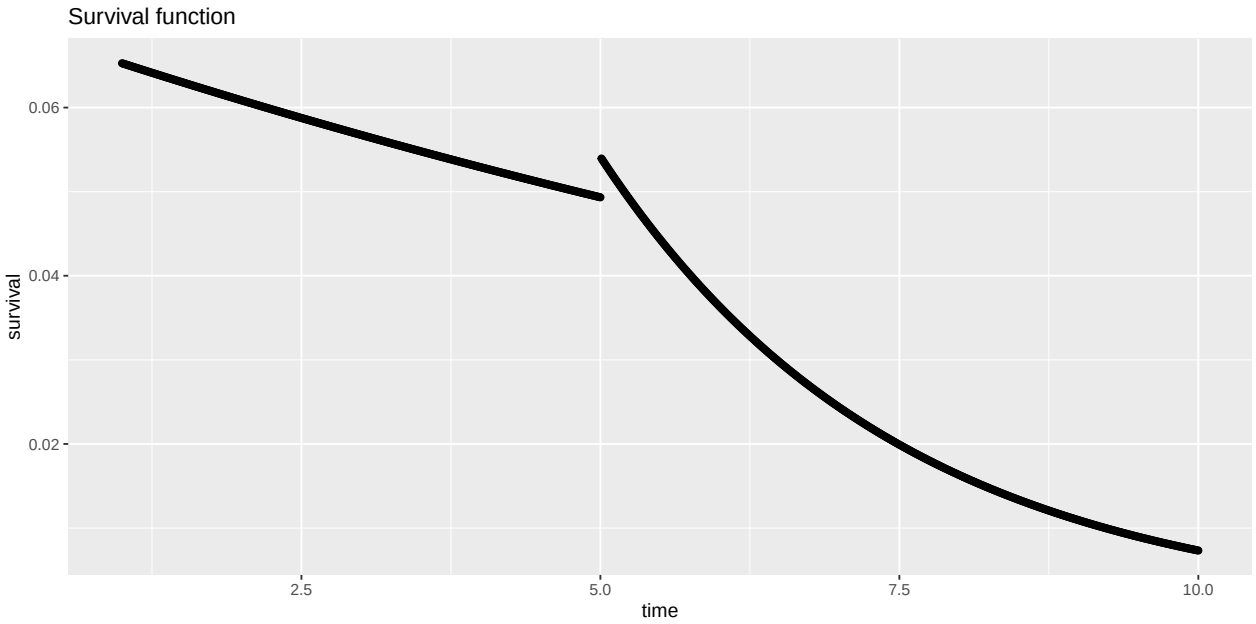
We have a piecewise hazard function as follows:



Survival plot

Given that the survival function must be a smooth function, we obtain a survival function

We obtain a piecewise survival function whose first chunk corresponds

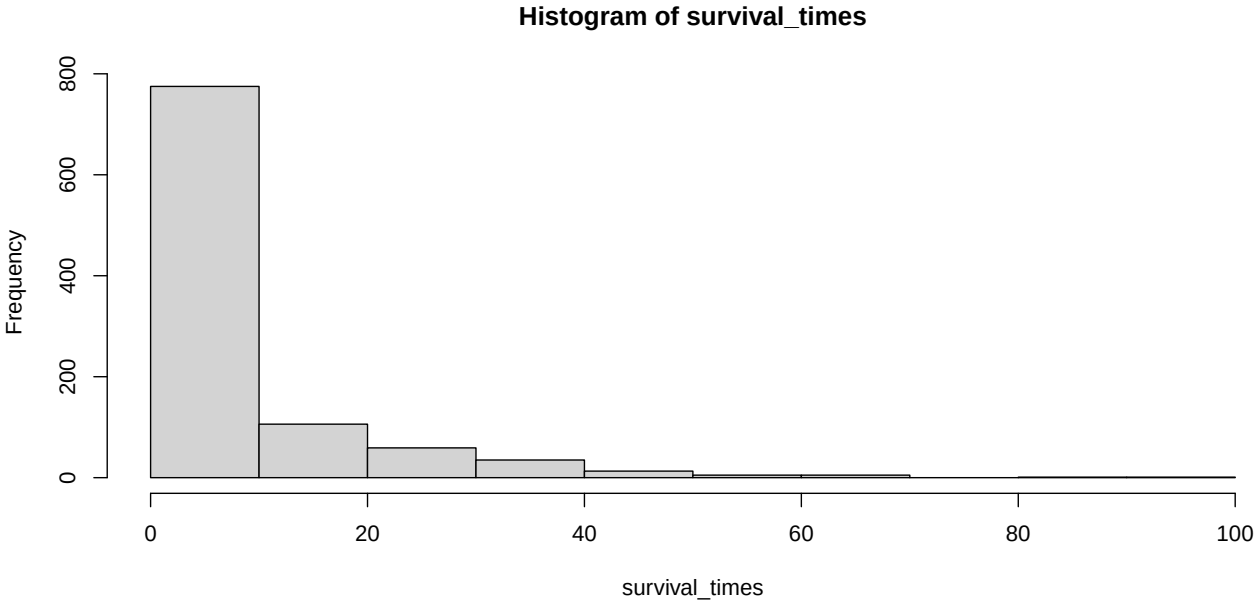


Survival times simulation

```
# number of trials
survival_times <- sapply(theta, function(lambda) {rexp(1,rate=1/lambda)})
```

Histogram for survival times

We plot a histogram for the survival times



Median survival time

As we can see, sampling from the distribution results in the following median survival time:

```
#> [1] 3.287133
```

Exercise 2

Given the following density function:

$$f(y) = (\lambda_0 + \lambda_1 y)e^{-\lambda_0 y - \frac{1}{2}\lambda_1 y^2}$$

We obtain the survival function as follows:

$$\begin{aligned} S(t) = P(T > t) &= \int_t^\infty (\lambda_0 + \lambda_1 y)e^{-\lambda_0 y - \frac{1}{2}\lambda_1 y^2} dy \\ &= \lim_{b\to\infty} [-e^{\frac{\lambda_1 b^2}{2} - \lambda_0 b}] + e^{\frac{-\lambda_1 t^2}{2} - \lambda_0 t} \\ &= 0 + e^{\frac{-\lambda_1 t^2}{2} - \lambda_0 t} \\ S(t) &= e^{\frac{-\lambda_1 t^2}{2} - \lambda_0 t},\ \lambda_1 \in \mathbb{R}, \lambda_0 > 0 \end{aligned}$$

We obtain the hazard function as follows:

$$h(t) = \frac{f(t)}{S(t)} = \frac{(\lambda_0 + \lambda_1 t)e^{-\lambda_0 t - \frac{1}{2}\lambda_1 t^2}}{e^{-\lambda_0 t - \frac{1}{2}\lambda_1 t^2}} = \lambda_0 + \lambda_1 t$$

$$h(t) = \lambda_0 + \lambda_1 t$$

And the cumulative hazard function:

$$H(t) = -log(S(t)) = \frac{\lambda_1 t^2}{2} + \lambda_0 t$$

Exercise 3

KM estimator implementation of the survival function

Our implementation is as follows:

Parameters:

- **dataset**: Dataset to obtain the KM estimation from
- **events**: specific column of the dataset corresponding to the events (deemed *status* for the *aml* dataset)

Algorithm:

- Obtain length of the dataset by selecting the first column and utilizing *length* to obtain it
- Create the survival vector with (by definition) survival probability of 1 in the first time instance
- Initilize a counter j to keep track of events occurring in the column of the dataset passed as the **events** parameter
- Iterate over the length of the dataset (*1:length(dataset[,1])*) using i as iteration variable
  - During the loop we check if the i-th element of **events** does corespond to an event (1) or not (0)
  - If so, we add one to the counter and calculate the survival probability as a product of the previous survival probability obtained in the previous positive (1) event

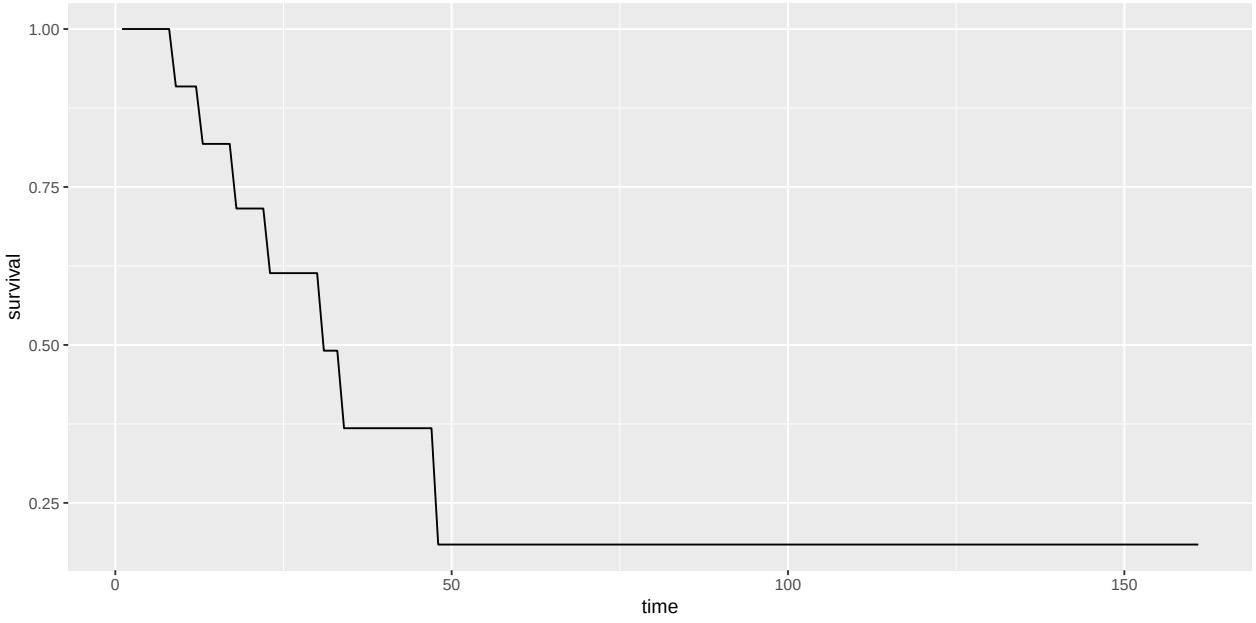
        * We append the survival probability to the survival vector
    - 1 is subtracted from the total length of the dataset as one element in the dataset has been traversed. We use this *n_c* variable as a component of our survival probability calculation.

- Return the survival probability vector, the times where the survival probability changes correspond to the original time of events in the original passed on **dataset**.

```
km_est <- function(dataset, events) {
    # calculate survival probabilities
    n_c <- length(dataset[,1])
    survival <- c(1)
    j = 0
    for (i in 1:length(dataset[,1])) {
        if (events[i] == 1) {
            j = j+1
            prob <- (1-(1/n_c))*survival[j]
            survival <- c(survival, prob)
        }
        n_c <- n_c - 1
    }
    return(survival)
}
```

Utilizing the function to obtain the survival function for the leukemia dataset

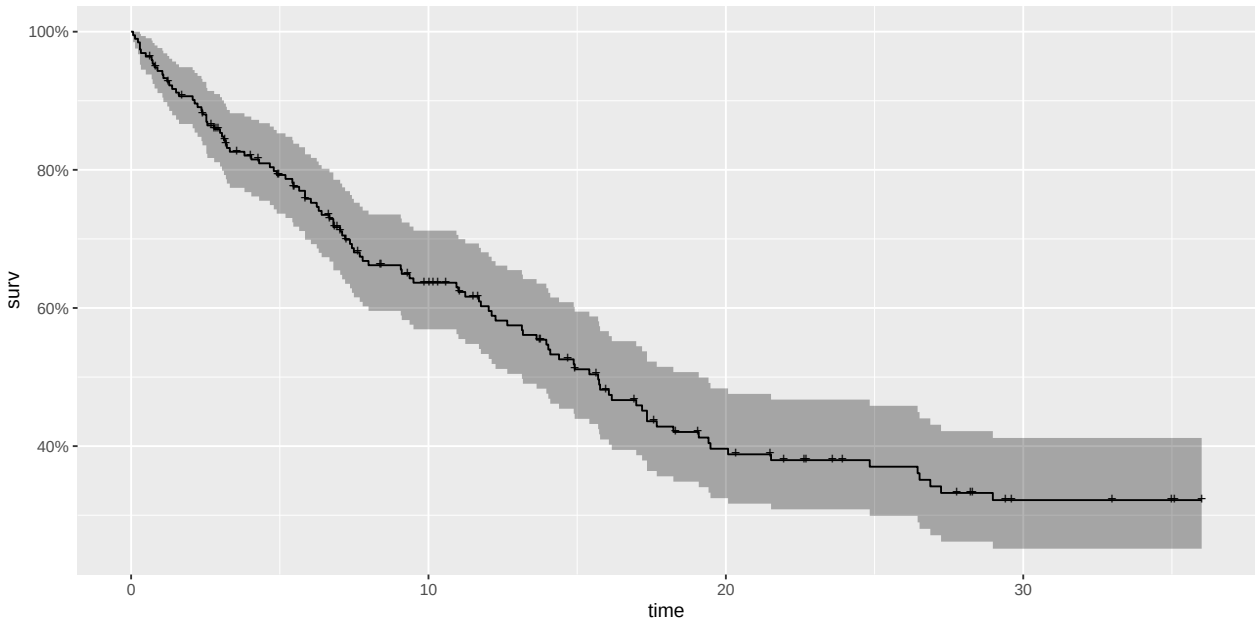
The survival probabilities are as follows, and these change over time at the times displayed on the *time* column of this table:



Exercise 4

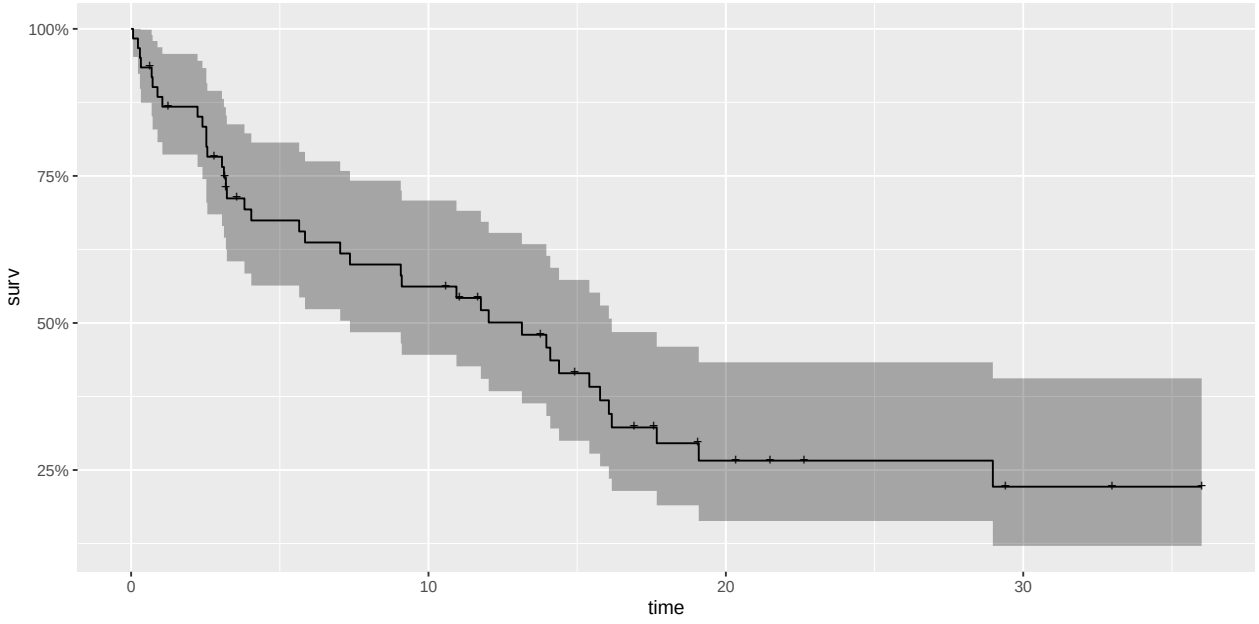
KM estimate of the survival function

We can see the estimate as follows:

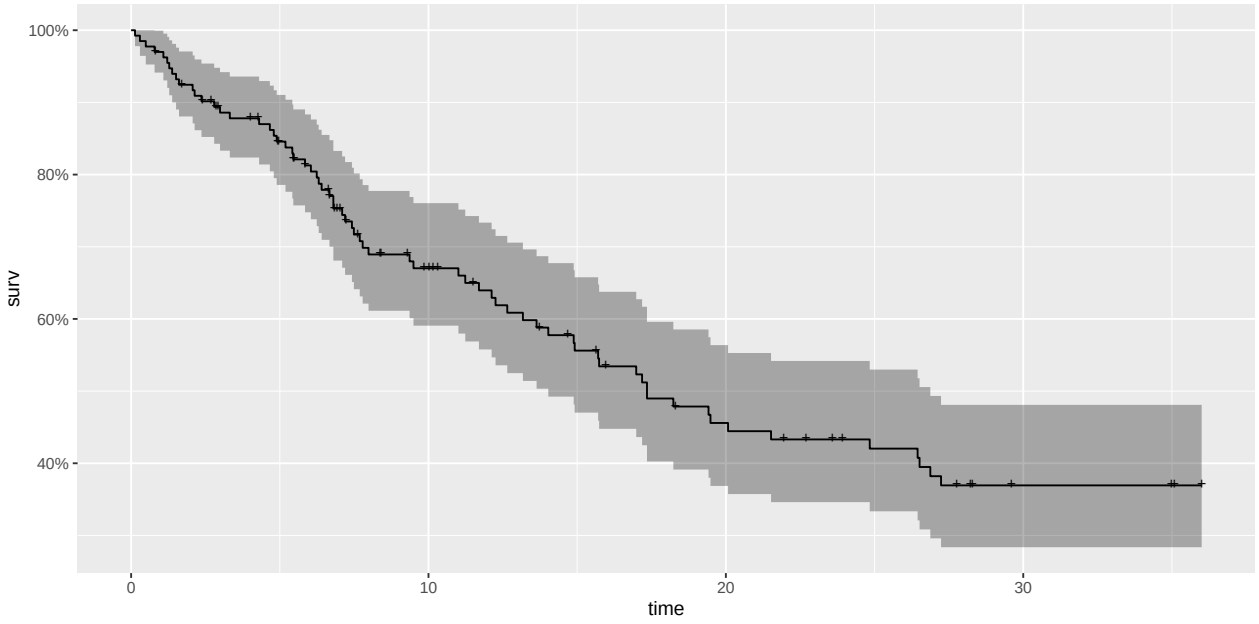


In order to achieve this fit, we have modified the dataset as to convert the columns *personal* and *property* to factors, and then switch the 0s for 1s and vice versa in the *censor* column, given that these are reversed.

Survival function: with crimes against persons

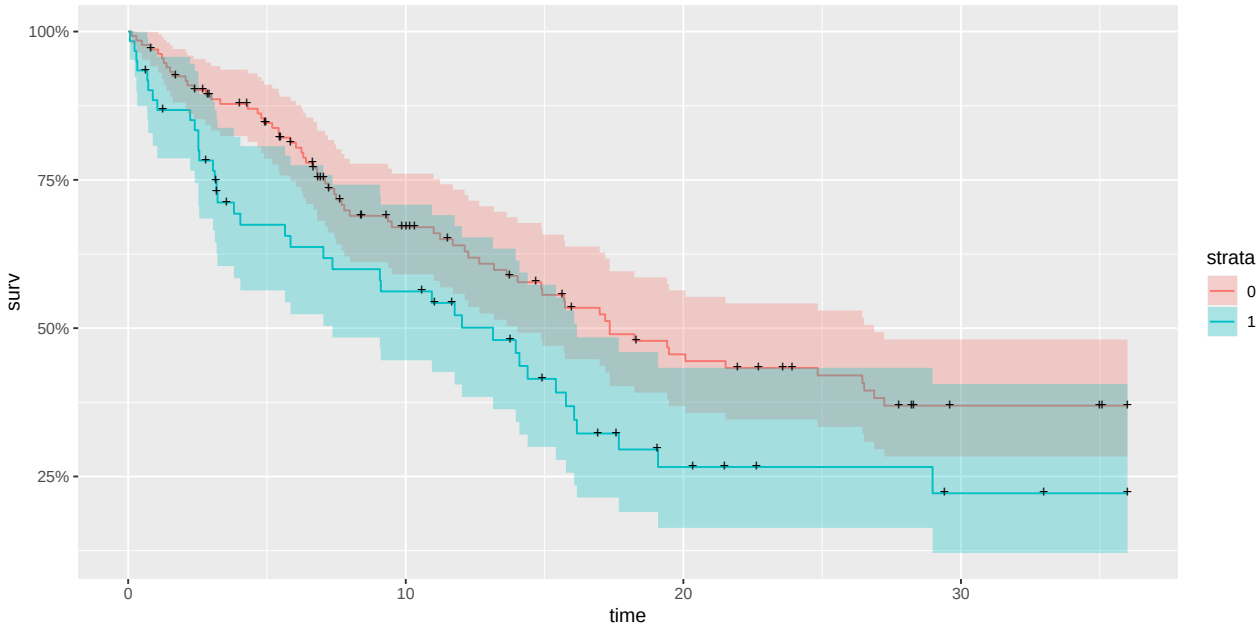


Survival function: without crimes against persons



Comparing both curves

We can see that the curve for nonpersonal crimes decays faster overall, as opposed to the personal crimes.



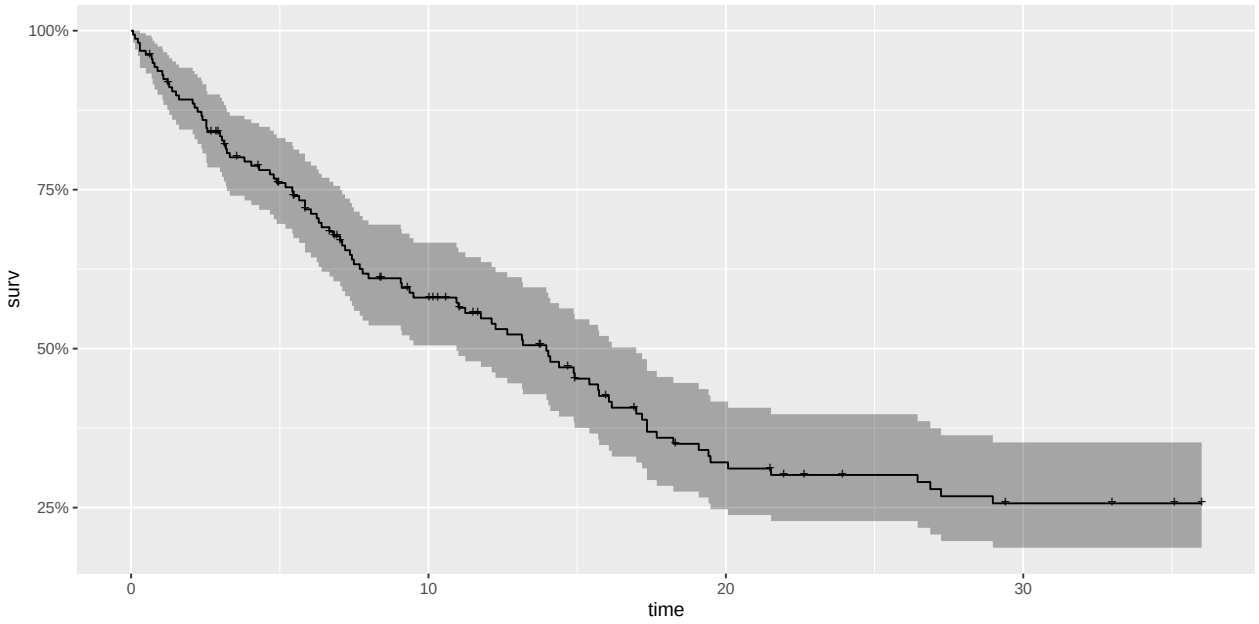
Low-rank test

According to the low-rank test. Where our null hypothesis is that both groups are the same. We obtain a p-val of ~0.02 which is <0.05 (setting our confidence level at 95%), therefore we reject the null hypothesis. Therefore there are differences among groups.

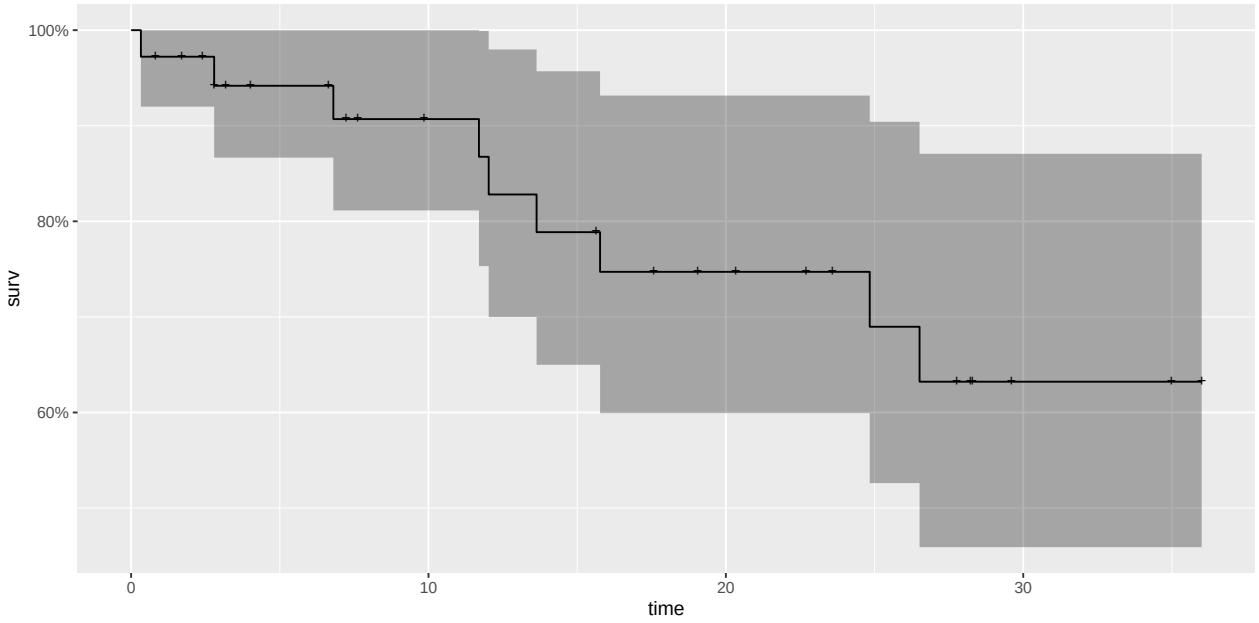
```
#> Call:
#> survdiff(formula = Surv(months, censor) ~ personal, data = henning)
```

```
#>
#>                N Observed Expected (O-E)^2/E (O-E)^2/V
#> personal=0 133       67     77.8      1.50       5.7
#> personal=1  61       39     28.2      4.14       5.7
#>
#>  Chisq= 5.7  on 1 degrees of freedom, p= 0.02
```

Survival function: with crimes against property

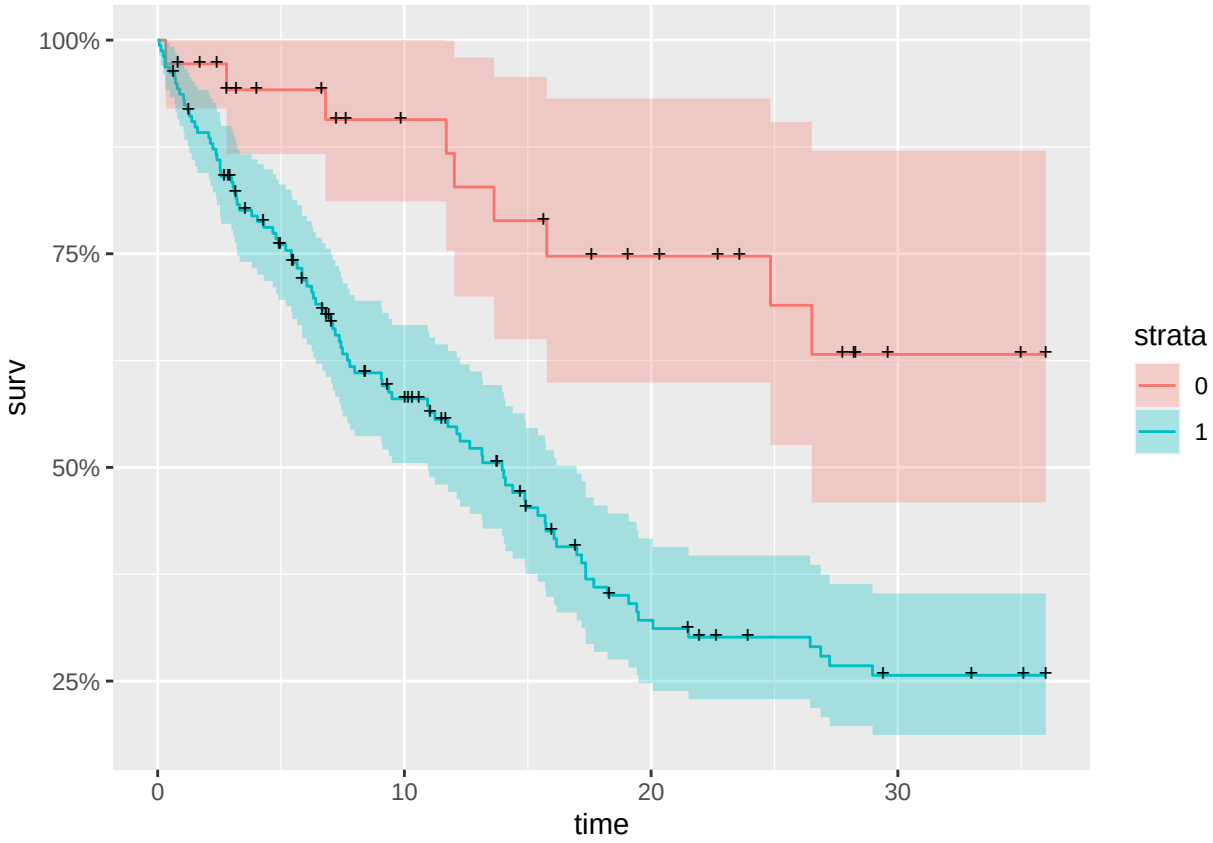


Survival function: with crimes against property



Comparing both curves

We can see that the curve for non-property related crimes overall decays significantly faster than the opposite one.



Low-rank test

```
#> Call:
#> survdiff(formula = Surv(months, censor) ~ property, data = henning)
#>
#>              N Observed Expected (O-E)^2/E (O-E)^2/V
#> property=0  36        9     24.7      9.97      13.1
#> property=1 158       97     81.3      3.02      13.1
#>
#>  Chisq= 13.1  on 1 degrees of freedom, p= 3e-04
```

As before, we must reject the null hypothesis. Therefore we can say there is a significant statistical difference among both groups.

Fitting a Cox regression

Converting personal and property to leveled factors with labels yes/no.

```
henning$personal=factor(henning$personal,levels=c("0","1"),labels=c("no","yes"))
henning$property=factor(henning$property,levels=c("0","1"),labels=c("no","yes"))
head(henning)
#>   id     months censor personal property       cage
#> 1  1 0.06570842      1      yes      yes  -1.675198
#> 2  2 0.13141684      1       no      yes -10.482864
#> 3  3 0.22997947      1      yes      yes  -4.426738
#> 4  4 0.29568789      1       no      yes -11.328860
```

```
#> 5  5 0.29568789      1      yes      yes  -7.164589
#> 6  6 0.32854209      1      yes       no  -2.868901
```

Running the cox regression fit.

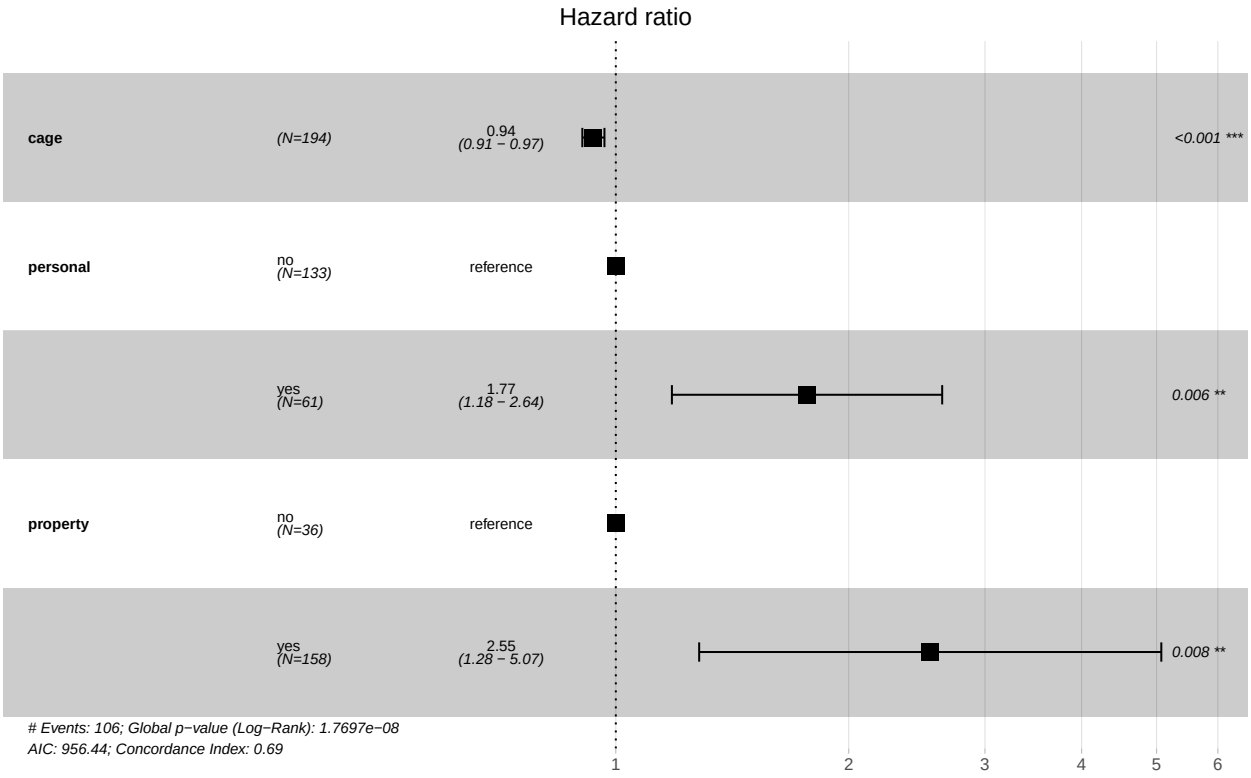
```
fit.all = coxph(Surv(months,censor) ~ cage + personal + property , henning)
summary(fit.all)
#> Call:
#> coxph(formula = Surv(months, censor) ~ cage + personal + property,
#>     data = henning)
#>
#>   n= 194, number of events= 106
#>
#>                 coef exp(coef) se(coef)      z Pr(>|z|)
#> cage        -0.06671   0.93546  0.01678 -3.976 7.01e-05 ***
#> personalyes  0.56914   1.76674  0.20521  2.773  0.00555 **
#> propertyyes  0.93579   2.54922  0.35088  2.667  0.00765 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>             exp(coef) exp(-coef) lower .95 upper .95
#> cage           0.9355     1.0690    0.9052    0.9667
#> personalyes    1.7667     0.5660    1.1817    2.6415
#> propertyyes    2.5492     0.3923    1.2816    5.0708
#>
#> Concordance= 0.694  (se = 0.027 )
#> Likelihood ratio test= 38.96  on 3 df,   p=2e-08
#> Wald test            = 29.02  on 3 df,   p=2e-06
#> Score (logrank) test = 30.3  on 3 df,   p=1e-06
```

According to the p-value on our Wald test, we can see that all three varaibles are significant. With cage holding the largest significance by a lot.

We can notice that for individuals which committed personal and property crimes, there is a positive increase in hazard, with property crimes yielding the largest increase in hazard.

As for cage, there is a small decrease in hazard given a higher value of *cage*. Hazard is associated with reincidence in crime.

This can be visualized in the *ggforest* plot from the *survminer* library.



We can see that while personal crimes increase the probability of reincidence significantly, property crimes increase it to even a larger extent.

Exercise 5

Given a hazard function $h(t) = c$, where $c > 0$:

We obtain the cumulative hazard function $H(t)$:

$$\begin{aligned} H(t) &= \int_0^t h(u)du \\ &= c\int_0^t du \\ &= ct \end{aligned}$$

With this, we derive the survival function $S(t)$:

$$\begin{aligned} H(t) &= ct \\ H(t) &= -log(S(t)) \\ ct &= -log(S(t)) \\ S(t) &= e^{-ct} \end{aligned}$$

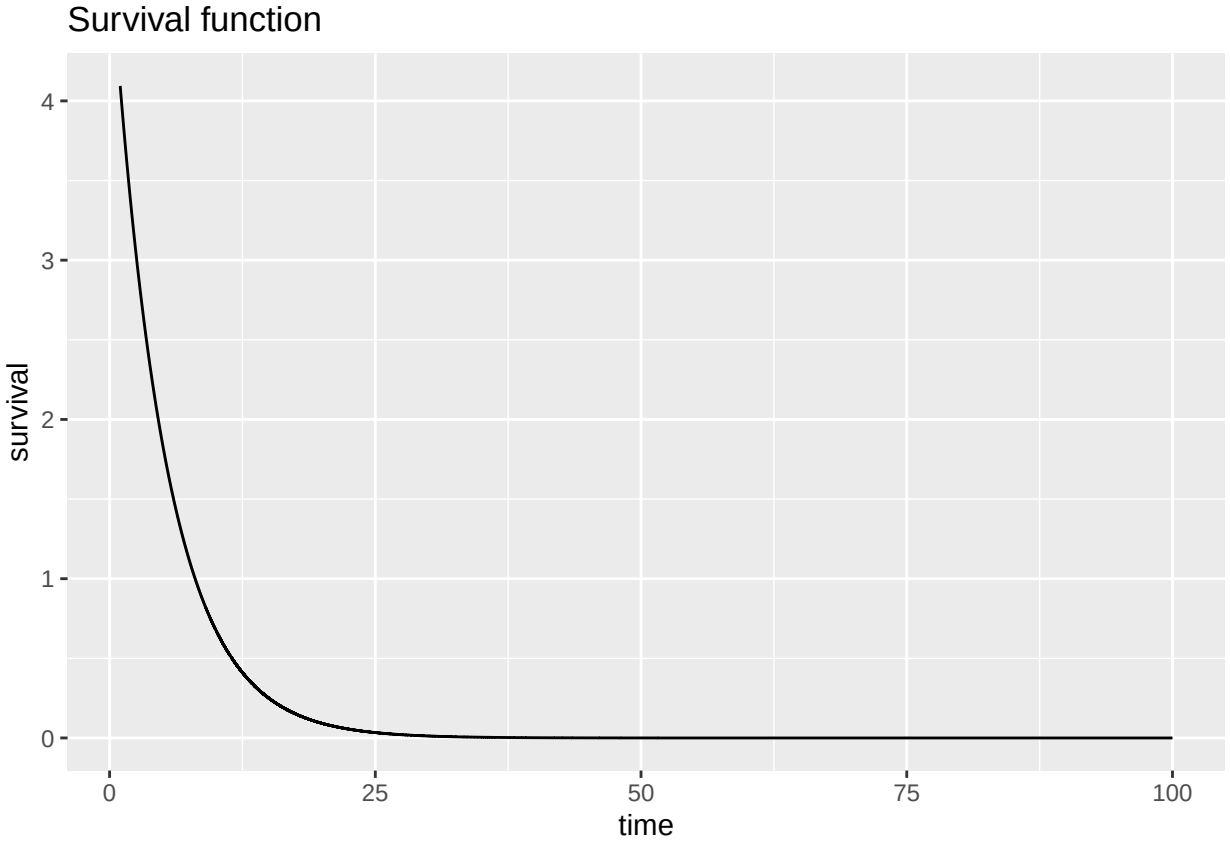
And then we obtain the density function $f(t)$:

$$h(t) = \frac{f(t)}{S(t)}$$
$$c = \frac{f(t)}{e^{-ct}}$$
$$f(t) = ce^{-ct}$$

Calculating median failure time with $c = 5$

We note the functions with $c = 5$ are:

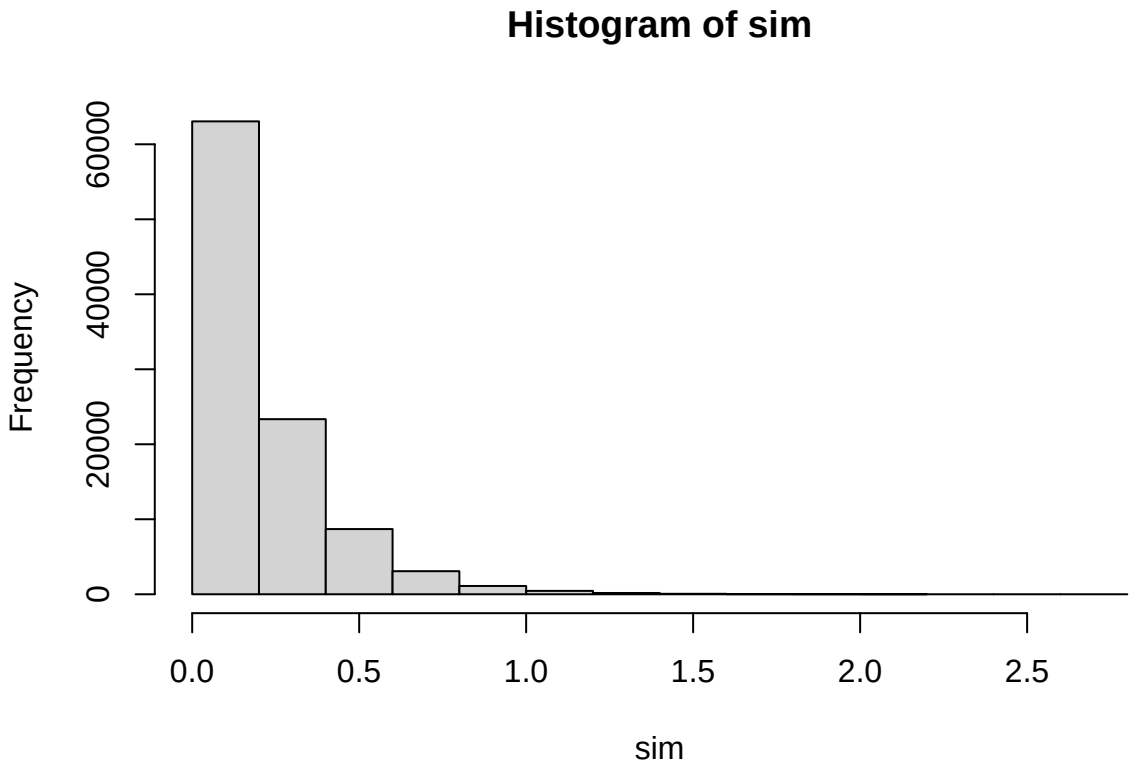
$$h(t) = 5$$
$$H(t) = 5t$$
$$S(t) = e^{-5t}$$
$$f(t) = 5e^{-5t}$$



The median failure time, according to 10,000 simulations, with a max length of time of 100, is as follows:

```
#> [1] 0.1390805
```

Which we can visualize in a histogram:



Therefore failure times are most commonly below 0.2 (~63% of the data) and almost always below 0.6 (~95% of the data).

Exercise 6

First we read the data and remove missing values:

```
# reading the data
lungcancer = read.table(file="http://www.mayo.edu/research/documents/lungdat/DOC-10027697", sep="",header=F, col.names=c("inst",
lungcancer[lungcancer=='.'] <- NA
lungcancer <- lungcancer[complete.cases(lungcancer), ]
```

Then we convert the corresponding variables into factors/numeric:

```
factors <- c('status','sex','inst')
nums <- c('weight.loss','calories','Karnofsky.physician','Karnofsky.patient','ECOG')
for (name in factors) {lungcancer[,name] <- as.factor(lungcancer[,name])}
for (name in nums) {lungcancer[,name] <- as.numeric(lungcancer[,name])}
```

Fitting a Cox PH model with all covariates

We fit the model and check its summary:

```
#> Call:
#> coxph(formula = Surv(time) ~ ., data = lungcancer)
#>
#>   n= 167, number of events= 167
#>
#>                coef  exp(coef)   se(coef)      z Pr(>|z|)
#> inst10     0.1537466  1.1661954  0.5497850  0.280  0.77975
#> inst11    -0.1752310  0.8392631  0.3470949 -0.505  0.61366
#> inst12    -0.1487126  0.8618168  0.3354185 -0.443  0.65750
#> inst13    -0.3561554  0.7003638  0.3494027 -1.019  0.30805
#> inst15    -0.3158451  0.7291724  0.4789985 -0.659  0.50965
#> inst16    -0.7492780  0.4727077  0.3907460 -1.918  0.05517 .
#> inst2      0.3669028  1.4432576  0.5531854  0.663  0.50717
#> inst21    -0.0969015  0.9076454  0.4305881 -0.225  0.82194
#> inst22    -1.0226541  0.3596391  0.3623982 -2.822  0.00477 **
#> inst26     0.0756597  1.0785955  0.5570521  0.136  0.89196
#> inst3     -0.3325248  0.7171109  0.3554011 -0.936  0.34946
#> inst32     0.7573949  2.1327131  0.4741282  1.597  0.11017
#> inst4     -1.1380700  0.3204369  0.5545112 -2.052  0.04013 *
#> inst5     -0.2397618  0.7868153  0.4539193 -0.528  0.59736
#> inst6     -0.1466783  0.8635718  0.3503709 -0.419  0.67548
```

```
#> inst7               -0.2043859  0.8151477  0.4463130 -0.458  0.64699
#> status2              0.2726741  1.3134722  0.2202012  1.238  0.21561
#> age                 -0.0017526  0.9982490  0.0106912 -0.164  0.86979
#> sex2                -0.3486300  0.7056542  0.1792656 -1.945  0.05180 .
#> ECOG                 0.6496109  1.9147956  0.2228797  2.915  0.00356 **
#> Karnofsky.physician  0.0253203  1.0256436  0.0111669  2.267  0.02336 *
#> Karnofsky.patient   -0.0108162  0.9892420  0.0074541 -1.451  0.14676
#> calories            -0.0002739  0.9997261  0.0002453 -1.117  0.26419
#> weight.loss         -0.0098622  0.9901862  0.0069817 -1.413  0.15778
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>                     exp(coef) exp(-coef) lower .95 upper .95
#> inst10                 1.1662     0.8575    0.3970    3.4257
#> inst11                 0.8393     1.1915    0.4251    1.6571
#> inst12                 0.8618     1.1603    0.4466    1.6631
#> inst13                 0.7004     1.4278    0.3531    1.3891
#> inst15                 0.7292     1.3714    0.2852    1.8645
#> inst16                 0.4727     2.1155    0.2198    1.0167
#> inst2                  1.4433     0.6929    0.4881    4.2679
#> inst21                 0.9076     1.1018    0.3903    2.1107
#> inst22                 0.3596     2.7806    0.1768    0.7317
#> inst26                 1.0786     0.9271    0.3620    3.2138
#> inst3                  0.7171     1.3945    0.3573    1.4391
#> inst32                 2.1327     0.4689    0.8421    5.4015
#> inst4                  0.3204     3.1207    0.1081    0.9500
#> inst5                  0.7868     1.2709    0.3232    1.9154
#> inst6                  0.8636     1.1580    0.4346    1.7161
#> inst7                  0.8151     1.2268    0.3399    1.9550
#> status2                1.3135     0.7613    0.8531    2.0224
#> age                    0.9982     1.0018    0.9775    1.0194
#> sex2                   0.7057     1.4171    0.4966    1.0027
#> ECOG                   1.9148     0.5222    1.2371    2.9637
#> Karnofsky.physician    1.0256     0.9750    1.0034    1.0483
#> Karnofsky.patient      0.9892     1.0109    0.9749    1.0038
#> calories               0.9997     1.0003    0.9992    1.0002
#> weight.loss            0.9902     1.0099    0.9767    1.0038
#>
#> Concordance= 0.65  (se = 0.024 )
#> Likelihood ratio test= 38.05  on 24 df,   p=0.03
#> Wald test            = 39.61  on 24 df,   p=0.02
#> Score (logrank) test = 40.33  on 24 df,   p=0.02
```

Wald test and LRT test for Karnofsky.patient and Karnofsky.physician

According the LRT and Wald test for this model, excluding vs including the variables shows that it is not statistically correct to simply remove these two variables. We cannot, with sufficient authority (and a confidence level of 95%), reject the null hypothesis. Therefore these two variables should remain in the model.

```
#> Call:
#> coxph(formula = Surv(time) ~ . - Karnofsky.physician - Karnofsky.patient,
#>     data = lungcancer)
#>
#>   n= 167, number of events= 167
#>
#>                   coef  exp(coef)   se(coef)      z Pr(>|z|)
#> inst10       0.0364943  1.0371684  0.5477600  0.067  0.94688
#> inst11      -0.1483818  0.8621019  0.3444167 -0.431  0.66660
#> inst12      -0.2406130  0.7861458  0.3394471 -0.709  0.47843
#> inst13      -0.3083409  0.7346649  0.3488753 -0.884  0.37680
#> inst15      -0.4231001  0.6550130  0.4728542 -0.895  0.37091
#> inst16      -0.7419266  0.4761956  0.3901992 -1.901  0.05725 .
#> inst2        0.3155020  1.3709473  0.5522225  0.571  0.56778
#> inst21       0.1670720  1.1818393  0.4159491  0.402  0.68793
#> inst22      -0.9013016  0.4060408  0.3540286 -2.546  0.01090 *
#> inst26       0.0342884  1.0348830  0.5558434  0.062  0.95081
#> inst3       -0.3526006  0.7028578  0.3488422 -1.011  0.31212
#> inst32       0.8352244  2.3053314  0.4720427  1.769  0.07683 .
#> inst4       -0.9861941  0.3729936  0.5497556 -1.794  0.07283 .
#> inst5       -0.0200092  0.9801897  0.4329050 -0.046  0.96313
#> inst6       -0.0773719  0.9255455  0.3477553 -0.222  0.82393
#> inst7       -0.2558607  0.7742498  0.4366983 -0.586  0.55794
#> status2      0.3723943  1.4512051  0.2194513  1.697  0.08971 .
#> age         -0.0074211  0.9926064  0.0104692 -0.709  0.47842
#> sex2        -0.3209401  0.7254667  0.1786380 -1.797  0.07240 .
#> ECOG         0.3904803  1.4776904  0.1417182  2.755  0.00586 **
#> calories    -0.0003072  0.9996929  0.0002417 -1.271  0.20379
#> weight.loss -0.0070079  0.9930166  0.0068565 -1.022  0.30675
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>             exp(coef) exp(-coef) lower .95 upper .95
#> inst10         1.0372     0.9642    0.3545    3.0346
#> inst11         0.8621     1.1600    0.4389    1.6933
#> inst12         0.7861     1.2720    0.4042    1.5291
#> inst13         0.7347     1.3612    0.3708    1.4556
#> inst15         0.6550     1.5267    0.2593    1.6548
#> inst16         0.4762     2.1000    0.2216    1.0231
#> inst2          1.3709     0.7294    0.4645    4.0464
#> inst21         1.1818     0.8461    0.5230    2.6706
#> inst22         0.4060     2.4628    0.2029    0.8127
#> inst26         1.0349     0.9663    0.3481    3.0763
#> inst3          0.7029     1.4228    0.3548    1.3925
#> inst32         2.3053     0.4338    0.9140    5.8148
#> inst4          0.3730     2.6810    0.1270    1.0956
#> inst5          0.9802     1.0202    0.4196    2.2898
#> inst6          0.9255     1.0804    0.4682    1.8298
#> inst7          0.7742     1.2916    0.3290    1.8222
#> status2        1.4512     0.6891    0.9439    2.2311
#> age            0.9926     1.0074    0.9724    1.0132
#> sex2           0.7255     1.3784    0.5112    1.0296
#> ECOG           1.4777     0.6767    1.1193    1.9508
```

```
#> calories       0.9997     1.0003    0.9992    1.0002
#> weight.loss    0.9930     1.0070    0.9798    1.0065
#>
#> Concordance= 0.646  (se = 0.023 )
#> Likelihood ratio test= 31.29  on 22 df,   p=0.09
#> Wald test            = 31.2  on 22 df,   p=0.09
#> Score (logrank) test = 32.06  on 22 df,   p=0.08
```

Finding the best Cox model

I will find the best model using stepBIC, so using the Bayesian Information Criterion in order to sequetially tune the model.

```
#> Start:  AIC=1467.15
#> Surv(time) ~ inst + status + age + sex + ECOG + Karnofsky.physician +
#>     Karnofsky.patient + calories + weight.loss
#>
#>                       Df    AIC
#> - inst                16 1406.4
#> - age                  1 1462.1
#> - calories             1 1463.3
#> - status               1 1463.6
#> - weight.loss          1 1464.1
#> - Karnofsky.patient    1 1464.1
#> - sex                  1 1465.9
#> <none>                   1467.2
#> - Karnofsky.physician  1 1467.3
#> - ECOG                 1 1470.6
#>
#> Step:  AIC=1406.43
#> Surv(time) ~ status + age + sex + ECOG + Karnofsky.physician +
#>     Karnofsky.patient + calories + weight.loss
#>
#>                       Df    AIC
#> - age                  1 1401.3
#> - calories             1 1401.5
#> - status               1 1402.6
#> - sex                  1 1403.0
#> - Karnofsky.patient    1 1403.3
#> - weight.loss          1 1403.4
#> - Karnofsky.physician  1 1405.2
#> <none>                   1406.4
#> - ECOG                 1 1407.4
#>
#> Step:  AIC=1401.35
#> Surv(time) ~ status + sex + ECOG + Karnofsky.physician + Karnofsky.patient +
#>     calories + weight.loss
#>
#>                       Df    AIC
#> - calories             1 1396.4
#> - status               1 1397.5
#> - sex                  1 1397.8
#> - Karnofsky.patient    1 1398.2
#> - weight.loss          1 1398.3
#> - Karnofsky.physician  1 1400.6
#> <none>                   1401.3
#> - ECOG                 1 1402.4
#>
#> Step:  AIC=1396.44
#> Surv(time) ~ status + sex + ECOG + Karnofsky.physician + Karnofsky.patient +
#>     weight.loss
#>
#>                       Df    AIC
#> - status               1 1392.4
#> - sex                  1 1392.9
#> - weight.loss          1 1393.4
#> - Karnofsky.patient    1 1393.6
#> - Karnofsky.physician  1 1395.7
#> <none>                   1396.4
#> - ECOG                 1 1397.7
#>
#> Step:  AIC=1392.44
#> Surv(time) ~ sex + ECOG + Karnofsky.physician + Karnofsky.patient +
#>     weight.loss
#>
#>                       Df    AIC
#> - sex                  1 1389.6
#> - Karnofsky.patient    1 1389.7
#> - weight.loss          1 1389.9
#> <none>                   1392.4
#> - Karnofsky.physician  1 1392.7
#> - ECOG                 1 1395.5
#>
#> Step:  AIC=1389.63
#> Surv(time) ~ ECOG + Karnofsky.physician + Karnofsky.patient +
#>     weight.loss
#>
#>                       Df    AIC
#> - weight.loss          1 1386.7
#> - Karnofsky.patient    1 1387.1
#> - Karnofsky.physician  1 1389.5
#> <none>                   1389.6
#> - ECOG                 1 1391.8
#>
#> Step:  AIC=1386.67
#> Surv(time) ~ ECOG + Karnofsky.physician + Karnofsky.patient
#>
#>                       Df    AIC
#> - Karnofsky.patient    1 1383.3
#> - Karnofsky.physician  1 1386.1
#> <none>                   1386.7
#> - ECOG                 1 1387.6
#>
```

```
#> Step:  AIC=1383.32
#> Surv(time) ~ ECOG + Karnofsky.physician
#>
#>                        Df    AIC
#> - Karnofsky.physician  1 1382.0
#> <none>                   1383.3
#> - ECOG                 1 1387.2
#>
#> Step:  AIC=1382.01
#> Surv(time) ~ ECOG
#>
#>        Df    AIC
#> <none>    1382.0
#> - ECOG  1 1382.4
#> Call:
#> coxph(formula = Surv(time) ~ ECOG, data = lungcancer)
#>
#>        coef exp(coef) se(coef)     z      p
#> ECOG 0.2682    1.3076   0.1142 2.348 0.0189
#>
#> Likelihood ratio test=5.48  on 1 df, p=0.01927
#> n= 167, number of events= 167
```

The best model has a BIC of 1382.01.

The final model has the following formula: *Surv(time) ~ ECOG*

Interpretation in terms of hazard ratios

```
#> Call:
#> coxph(formula = Surv(time) ~ ECOG, data = lungcancer)
#>
#>   n= 167, number of events= 167
#>
#>        coef exp(coef) se(coef)     z Pr(>|z|)
#> ECOG 0.2682    1.3076   0.1142 2.348   0.0189 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#>      exp(coef) exp(-coef) lower .95 upper .95
#> ECOG     1.308     0.7647     1.045     1.636
#>
#> Concordance= 0.575  (se = 0.024 )
#> Likelihood ratio test= 5.48  on 1 df,   p=0.02
#> Wald test            = 5.51  on 1 df,   p=0.02
#> Score (logrank) test = 5.54  on 1 df,   p=0.02
```

As we see in the summary, given that we're only left with the ECOG variable. We can see that the coefficient is 1.3076, therefore, the higher the ECOG score the higher the hazard, to a singificant extent, where basically 1 unit increase in ECOG score represents a ~31% increase in hazard.

And given the confidence intervals, the hazard ratio can be as low as 1.045 and as high as 1.636