Laboratorio con R - 5

Metodi e Modelli per l'Inferenza Statistica - Ing. Matematica - a.a. 2018-19

06/06/2019

---

0. Librerie

```
library( survival )
```

Reference:

Therneau, T. M., & Grambsch, P. M. (2013). Modeling survival data: extending the Cox model. Springer Science & Business Media.

---

1. Estimating the survival and hazard functions

The hazard function $\lambda(t)$ is the nonparametric part of the Cox model. In this section we are going to show the most common estimator (in case of no covariates) of the cumulative hazard function: the Nelson-Aalen estimate. The cumulative hazard funtion is defined as:

$$\Lambda(t) = \int_0^t \lambda(s)\,\mathrm{d}s$$

The Nelson-Aalen estimate is:

$$\hat{\Lambda}(t) = \sum_{i:t_i \leq t} \frac{\Delta \bar{N}(t_i)}{\bar{Y}(t_i)}$$

in which $\Delta \bar{N}(t_i) = \bar{N}(t_i) - \bar{N}(t_i^-)$ represents the total number of events happened at time t_i, while $\bar{Y}(t_i)$ represents the total number of subjects at risk at time t_i.

We show the computation of N-A, both manually and automatically, with the dataset used in the paper where this estimate has been proposed for the first time. The data regard the failure time of diesel generator fans. The goal of the study was to decide whether to replace the working fans with higher quality fans to prevent future failures. The engineering problem was to determine whether the failure rate was decreasing over time.

We load the data.

```
load( 'fans.RData' )

dim( fans )
## [1] 70  2

head( fans )
```

```
##   time fail
## 1  4.5    1
## 2  4.6    0
## 3 11.5    1
## 4 11.5    1
## 5 15.6    0
## 6 16.0    1
```

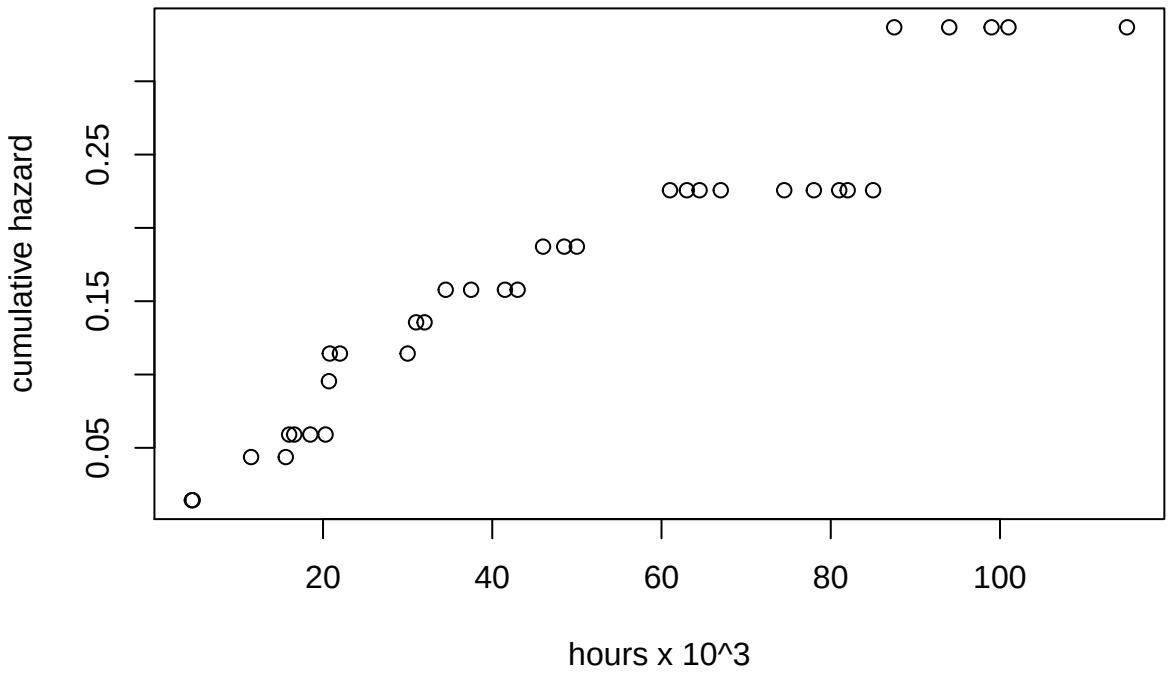
Nelson-Aalen

```
time = sort( unique ( fans[ , 1 ] ) )

length( time )
## [1] 35

table( fans[ ,2 ] ) #12 eventi, ergo 12 salti
##
##  0  1
## 58 12

fans[ which( fans[ ,2 ] == "1" ), ]
##    time fail
## 1   4.5    1
## 3  11.5    1
## 4  11.5    1
## 6  16.0    1
## 16 20.7    1
## 17 20.7    1
## 18 20.8    1
## 24 31.0    1
## 26 34.5    1
## 37 46.0    1
## 46 61.0    1
## 63 87.5    1

n = dim( fans )[ 1 ]

Nelson_Aalen = rep( 0, length( time ) )
var = rep( 0, length( time ) )
N = rep( 0, length( time ) )
Y = rep( 0, length( time ) )

for( i in 1 : length( time ) )
{
  N[ i ] = length( which ( fans[ ,1 ] == time[ i ] & fans[ ,2 ] == "1" ) )
  Y[ i ] = length( which ( fans[ ,1 ] >= time[ i ] ) )

  if( i ==1 )
  {
    Nelson_Aalen[ i ] = N[ i ]/Y[ i ]
    var[ i ] = N[ i ]/( Y[ i ] )^2
  }
  else
  {
```

```
    Nelson_Aalen[ i ] = Nelson_Aalen[ i-1 ] + N[ i ]/Y[ i ]
    var[ i ] = var[ i-1 ] + N[ i ]/( Y[ i ] )^2
  }
  print( c( i, Nelson_Aalen[ i ], var[ i ], time[ i ], N[ i ], Y[ i ] ) )
}
## [1] 1.000000e+00 1.428571e-02 2.040816e-04 4.500000e+00 1.000000e+00
## [6] 7.000000e+01
## [1] 2.000000e+00 1.428571e-02 2.040816e-04 4.600000e+00 0.000000e+00
## [6] 6.900000e+01
## [1] 3.000000e+00 4.369748e-02 6.366076e-04 1.150000e+01 2.000000e+00
## [6] 6.800000e+01
## [1] 4.000000e+00 4.369748e-02 6.366076e-04 1.560000e+01 0.000000e+00
## [6] 6.600000e+01
## [1]  5.000000000  0.059082094  0.000873294 16.000000000  1.000000000
## [6] 65.000000000
## [1]  6.000000000  0.059082094  0.000873294 16.600000000  0.000000000
## [6] 64.000000000
## [1]  7.000000000  0.059082094  0.000873294 18.500000000  0.000000000
## [6] 63.000000000
## [1]  8.000000000  0.059082094  0.000873294 20.300000000  0.000000000
## [6] 58.000000000
## [1]  9.000000000  0.095445731  0.001534451 20.700000000  2.000000000
## [6] 55.000000000
## [1] 10.00000000  0.11431366  0.00189045 20.80000000  1.00000000 53.00000000
## [1] 11.00000000  0.11431366  0.00189045 22.00000000  0.00000000 52.00000000
## [1] 12.00000000  0.11431366  0.00189045 30.00000000  0.00000000 51.00000000
## [1] 13.000000000  0.135590251  0.002343143 31.000000000  1.000000000
## [6] 47.000000000
## [1] 14.000000000  0.135590251  0.002343143 32.000000000  0.000000000
## [6] 46.000000000
## [1] 15.00000000  0.15781247  0.00283697 34.50000000  1.00000000 45.00000000
## [1] 16.00000000  0.15781247  0.00283697 37.50000000  0.00000000 44.00000000
## [1] 17.00000000  0.15781247  0.00283697 41.50000000  0.00000000 42.00000000
## [1] 18.00000000  0.15781247  0.00283697 43.00000000  0.00000000 38.00000000
## [1] 19.000000000  0.187224238  0.003702022 46.000000000  1.000000000
## [6] 34.000000000
## [1] 20.000000000  0.187224238  0.003702022 48.500000000  0.000000000
## [6] 33.000000000
## [1] 21.000000000  0.187224238  0.003702022 50.000000000  0.000000000
## [6] 29.000000000
## [1] 22.000000000  0.225685776  0.005181312 61.000000000  1.000000000
## [6] 26.000000000
## [1] 23.000000000  0.225685776  0.005181312 63.000000000  0.000000000
## [6] 22.000000000
## [1] 24.000000000  0.225685776  0.005181312 64.500000000  0.000000000
## [6] 21.000000000
## [1] 25.000000000  0.225685776  0.005181312 67.000000000  0.000000000
## [6] 19.000000000
## [1] 26.000000000  0.225685776  0.005181312 74.500000000  0.000000000
## [6] 18.000000000
## [1] 27.000000000  0.225685776  0.005181312 78.000000000  0.000000000
## [6] 17.000000000
## [1] 28.000000000  0.225685776  0.005181312 81.000000000  0.000000000
```

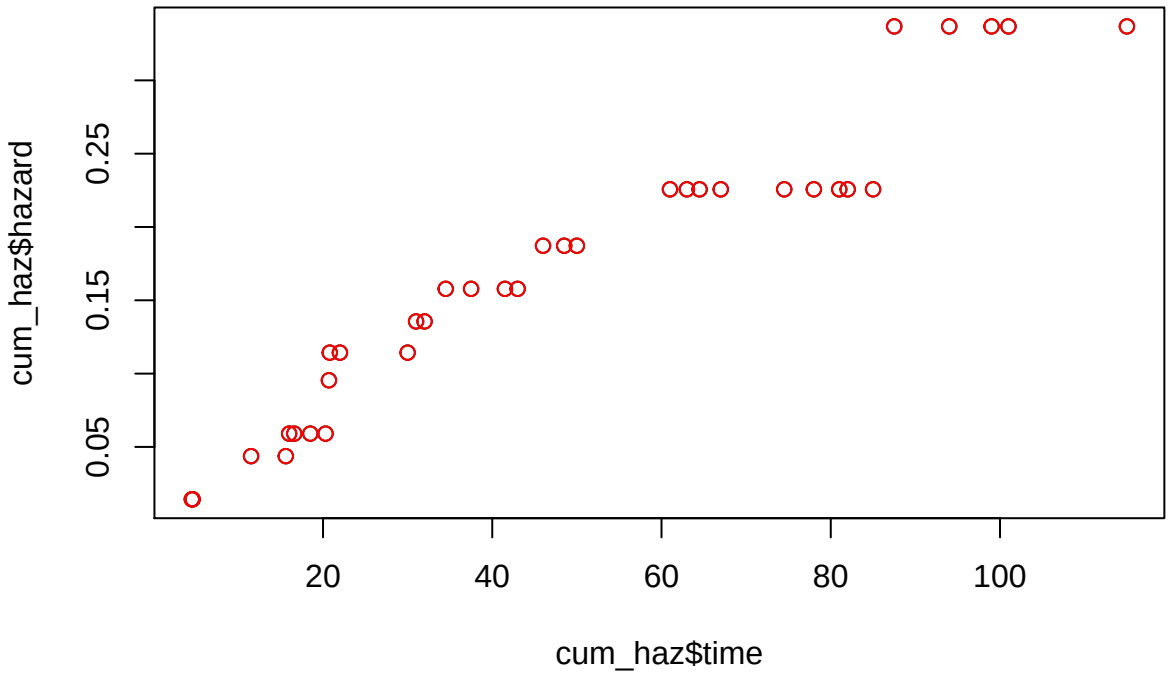
```
## [6] 15.000000000
## [1] 29.000000000  0.225685776  0.005181312 82.000000000  0.000000000
## [6] 13.000000000
## [1] 30.000000000  0.225685776  0.005181312 85.000000000  0.000000000
## [6] 12.000000000
## [1] 31.00000000  0.33679689  0.01752699 87.50000000  1.00000000  9.00000000
## [1] 32.00000000  0.33679689  0.01752699 94.00000000  0.00000000  6.00000000
## [1] 33.00000000  0.33679689  0.01752699 99.00000000  0.00000000  5.00000000
## [1]  34.00000000   0.33679689   0.01752699 101.00000000   0.00000000
## [6]   4.00000000
## [1]  35.00000000   0.33679689   0.01752699 115.00000000   0.00000000
## [6]   1.00000000
```

```
plot( time, Nelson_Aalen, xlab = "hours x 10^3", ylab = "cumulative hazard" )
```




```
check1 = 0
check2 = dim( fans[ which( fans[ ,2 ] == "1" ), ] )[ 1 ]

for ( i in 1 : n )
{
  index = which( time %in% fans[ i, 1] )
  check1 = check1 + Nelson_Aalen[ index ]
}

check1
```

```
## [1] 12
check2
## [1] 12
sum(N)
## [1] 12
lambda_hat = length( which( fans[ ,2] == 1 ) )/sum( fans[ ,1] )

cum_haz = basehaz( coxph( Surv( time, fail ) ~ 1, method = 'breslow', data = fans ), center = T )


plot( cum_haz$time, cum_haz$hazard )
points( time, Nelson_Aalen, col = 'red' )
```



Another important quantity in Survival Analysis is the Survival function:

$$S(t) = \exp\{-\Lambda(t)\}$$

Breslow suggested the following nonparametric estimator of the survival function:

$$\hat{S}_B(t) = \exp\{-\hat{\Lambda}(t)\}$$

The Breslow estimator can be written as:

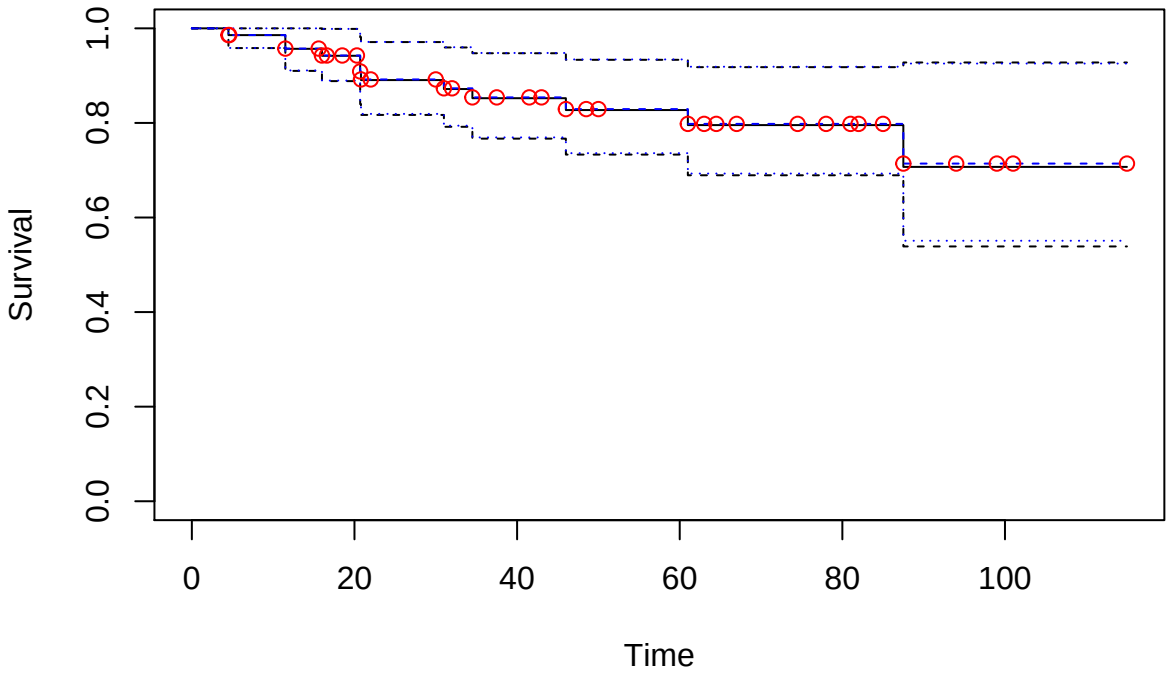
$$\hat{S}_B(t) = \prod_{i:t_i<t} \exp\{-d\hat{\Lambda}(t_i)\}$$

The Kaplan-Meier estimator is:

$$\hat{S}_{KM}(t) = \prod_{i:t_i<t} [1 - d\hat{\Lambda}(t_i)]$$

Kaplan-Meier

```
afit = survfit( Surv( time, fail ) ~ 1, data = fans, type = 'fleming-harrington')
kfit = survfit( Surv( time, fail ) ~ 1, data = fans, type = 'kaplan-meier')
summary( afit )
## Call: survfit(formula = Surv(time, fail) ~ 1, data = fans, type = "fleming-harrington")
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##   4.5     70       1    0.986  0.0141        0.959        1.000
##  11.5     68       2    0.957  0.0242        0.911        1.000
##  16.0     65       1    0.943  0.0279        0.890        0.999
##  20.7     55       2    0.909  0.0356        0.842        0.982
##  20.8     53       1    0.892  0.0388        0.819        0.971
##  31.0     47       1    0.873  0.0423        0.794        0.960
##  34.5     45       1    0.854  0.0455        0.769        0.948
##  46.0     34       1    0.829  0.0505        0.736        0.934
##  61.0     26       1    0.798  0.0574        0.693        0.919
##  87.5      9       1    0.714  0.0945        0.551        0.926
summary( kfit )
## Call: survfit(formula = Surv(time, fail) ~ 1, data = fans, type = "kaplan-meier")
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##   4.5     70       1    0.986  0.0142        0.958        1.000
##  11.5     68       2    0.957  0.0244        0.910        1.000
##  16.0     65       1    0.942  0.0282        0.888        0.999
##  20.7     55       2    0.908  0.0361        0.840        0.981
##  20.8     53       1    0.891  0.0392        0.817        0.971
##  31.0     47       1    0.872  0.0427        0.792        0.960
##  34.5     45       1    0.852  0.0460        0.767        0.947
##  46.0     34       1    0.827  0.0510        0.733        0.933
##  61.0     26       1    0.795  0.0581        0.689        0.918
##  87.5      9       1    0.707  0.0980        0.539        0.928


plot( kfit, mark.time = F ,  xlab = "Time", ylab="Survival" )
lines( afit, lty=2, mark.time=F, col = 'blue')
points( cum_haz$time, exp( -cum_haz$hazard ), col = 'red')
```



---

2. Cox model

In this section we are going to fit a Cox model (also known as proportional hazard model).

The Cox model is a semiparametric model because there is a nonparametric part: the **baseline hazard** function ($\lambda_0(t)$) and a parametric part: ($\exp\{\mathbf{x}_i\} \cdot \boldsymbol{\beta}$), in which $\mathbf{x}_i$ is the vector of covariates that characterize a statistical unit and $\boldsymbol{\beta}$ is the vectro of predictors. The Cox model is a nonlinear regression that allows us to fit the hazard rate ($\lambda(t; \mathbf{x}_i)$).

$$\lambda(t; \mathbf{x}_i) = \lambda_0(t) \cdot \exp\{x_i \cdot \boldsymbol{\beta}\} \qquad i = 1 : n$$

It is also known as **proportional hazard** model, because the hazard ratio for two subjects with fixed covariate vectors $\mathbf{x}_i$ and $\mathbf{x}_j$:

$$\frac{\lambda(t; \mathbf{x}_i)}{\lambda(t; \mathbf{x}_j)} = \frac{\lambda_0(t) \cdot \exp\{\mathbf{x}_i \cdot \boldsymbol{\beta}\}}{\lambda_0(t) \cdot \exp\{\mathbf{x}_j \cdot \boldsymbol{\beta}\}} = \exp\left(\mathbf{x}_i - \mathbf{x}_j\right)\boldsymbol{\beta}$$

is fixed over time.

In order to show the model and its inetrpretation, we study the PBC data set. The data come from a Mayo Clinic trial in primary biliary cirrhosis of the liver conducted between 1974 and 1984. PBC is a progressive

disease thought to be of an autoimmune origin; the subsequent inflammatory process eventually leads to cirrhosis and destruction of the liver's bile ducts and death of the patient.

- **case number**;
- number of days between registration and the earlier of death, transplantion, or study analysis time;
- **status**: 0=alive, 1=transplanted, 2=dead;
- **drug**: 1= D-penicillamine, 0=placebo;
- **age** in days, at registration;
- **sex**: 0=male, 1=female;
- **day**: number of days between enrollment and this visit date, remaining values on the line of data refer to this visit;
- presence of ascites: 0=no 1=yes;
- presence of hepatomegaly 0=no 1=yes;
- presence of spiders 0=no 1=yes;
- presence of edema 0=no edema and no diuretic therapy for edema; .5 = edema present without diuretics, or edema resolved by diuretics; 1 = edema despite diuretic therapy;
- serum bilirubin in mg/dl;
- serum cholesterol in mg/dl;
- albumin in gm/dl;
- alkaline phosphatase in U/liter;
- SGOT in U/ml (serum glutamic-oxaloacetic transaminase, the enzyme name has subsequently changed to "ALT" in the medical literature);
- platelets per cubic ml / 1000;
- prothrombin time in seconds;
- histologic stage of disease.

We load the data.

```
data( pbc )

head( pbc )
##   id time status trt      age sex ascites hepato spiders edema bili chol
## 1  1  400      2   1 58.76523   f       1      1       1   1.0 14.5  261
## 2  2 4500      0   1 56.44627   f       0      1       1   0.0  1.1  302
## 3  3 1012      2   1 70.07255   m       0      0       0   0.5  1.4  176
## 4  4 1925      2   1 54.74059   f       0      1       1   0.5  1.8  244
## 5  5 1504      1   2 38.10541   f       0      1       1   0.0  3.4  279
## 6  6 2503      2   2 66.25873   f       0      1       0   0.0  0.8  248
##   albumin copper alk.phos    ast trig platelet protime stage
## 1    2.60    156   1718.0 137.95  172      190    12.2     4
## 2    4.14     54   7394.8 113.52   88      221    10.6     3
## 3    3.48    210    516.0  96.10   55      151    12.0     4
## 4    2.54     64   6121.8  60.63   92      183    10.3     4
## 5    3.53    143    671.0 113.15   72      136    10.9     3
## 6    3.98     50    944.0  93.00   63       NA    11.0     3
```

We fit a Cox proportional hazard model.

```
fit1 = coxph( Surv( time, status == 2 ) ~ age + edema + log( bili ) +
                log( protime ) + log( albumin ), data = pbc )

summary( fit1 )
## Call:
## coxph(formula = Surv(time, status == 2) ~ age + edema + log(bili) +
##     log(protime) + log(albumin), data = pbc)
##
##   n= 416, number of events= 160
##    (2 observations deleted due to missingness)
##
##                    coef exp(coef)  se(coef)      z Pr(>|z|)
## age            0.039609  1.040404  0.007672  5.163 2.43e-07 ***
## edema          0.896311  2.450547  0.271410  3.302 0.000959 ***
## log(bili)      0.863551  2.371566  0.082941 10.412  < 2e-16 ***
## log(protime)   2.386839 10.879054  0.768509  3.106 0.001898 **
## log(albumin) -2.506923  0.081519  0.652916 -3.840 0.000123 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##              exp(coef) exp(-coef) lower .95 upper .95
## age            1.04040    0.96117   1.02488    1.0562
## edema          2.45055    0.40807   1.43959    4.1715
## log(bili)      2.37157    0.42166   2.01575    2.7902
## log(protime)  10.87905    0.09192   2.41232   49.0622
## log(albumin)   0.08152   12.26713   0.02267    0.2931
##
## Concordance= 0.835  (se = 0.017 )
## Likelihood ratio test= 231  on 5 df,   p=<2e-16
## Wald test            = 234.2  on 5 df,   p=<2e-16
## Score (logrank) test = 301.8  on 5 df,   p=<2e-16
```

In order to give a proper interpretation of the model, we have to focus on the `exp(coef)`, which represents the multiplicative change in risk due to each covariate.

In this case, we can conclude that:

- bilirubin is the most important variable and an increment of 1 point in log(bili) leads to a 2.37 increases in the risk of dying;
- getting older leads to a higher risk of dying: getting 10 years elder leads to 1.5 higher risk of death;

```
exp( fit1$coefficients[ 1 ] * 10  )
##      age
## 1.486005
```

- an increasing of albumin level in blood leads to a lower risk of death.

Testing the global null hypothesis

$$H_0 : \boldsymbol{\beta} = \boldsymbol{\beta}_0 \quad vs \quad H_1 : \boldsymbol{\beta} \neq \boldsymbol{\beta}_0$$

In this case $\boldsymbol{\beta}_0$ is the starting value of the regression parameters.

There are three methods to compute this test:

- **Likelihood Ratio Test**: $2(l(\hat{\boldsymbol{\beta}}) - l(\boldsymbol{\beta}_0))$, twice the difference in the log partial likelihood at the initial and final estimates of β.

- **Wald test**: $(\hat{\beta} - \beta_0) I(\hat{\beta})(\hat{\beta} - \beta_0)$, where $I(\hat{\beta})$ is the estimated information matrix at the solution. For a single variable, this reduces to the usual z-statistic $\hat{\beta}/se(\hat{\beta})$.
- **Efficient Score statistics**: $U^T(\beta_0) I(\beta_0)^{-1} U(\beta_0)$. $U(\beta)$ is the score vector, the partial derivative with respect to β of the log-likelihood. When p = 1 and the single covariate is categorical, the score test is identical to the **log-rank test**.

The null hypothesis distribution of each of these tests is a $\chi^2(p)$.

```
fit_test = coxph( Surv( time, status == 2 ) ~ edema + log( bili ) +
                    log( protime ) + log( albumin ), data = pbc )

2*( summary(fit1)$logtest - summary(fit_test)$logtest )
##          test            df        pvalue
##  5.208316e+01  2.000000e+00 -6.532394e-43
summary(fit_test)$waldtest
##         test           df       pvalue
## 2.214600e+02 4.000000e+00 9.077814e-47
summary(fit_test)$sctest
##         test           df       pvalue
## 2.799640e+02 4.000000e+00 2.268545e-59
```

Stratified Cox model

A possible extension to Cox model consists in considering k strata and a specific $\lambda_0(t)$ for each stratum:

$$\lambda_k(t; \mathbf{x}_i) = \lambda_{0k}(t) \cdot \exp\{\mathbf{x}_i \cdot \boldsymbol{\beta}\} \qquad i = 1 : n$$

It is often of interest to fit different baseline in study where several healthcare providers are involved.

In this case we show the output that we obtain by stratifing on edema covariate.

```
fit2 = coxph( Surv( time, status == 2 ) ~ log(bili) * strata( edema ), data = pbc  )

summary(fit2)
## Call:
## coxph(formula = Surv(time, status == 2) ~ log(bili) * strata(edema),
##     data = pbc)
##
##   n= 418, number of events= 161
##
##                                     coef exp(coef) se(coef)      z
## log(bili)                           1.02064   2.77498  0.09548 10.690
## log(bili):strata(edema)edema=0.5 -0.37323   0.68851  0.19814 -1.884
## log(bili):strata(edema)edema=1   -0.74906   0.47281  0.25848 -2.898
##                                    Pr(>|z|)
## log(bili)                           < 2e-16 ***
## log(bili):strata(edema)edema=0.5  0.05961 .
## log(bili):strata(edema)edema=1    0.00376 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                                  exp(coef) exp(-coef) lower .95 upper .95
## log(bili)                           2.7750     0.3604    2.3014    3.3460
## log(bili):strata(edema)edema=0.5    0.6885     1.4524    0.4669    1.0152
## log(bili):strata(edema)edema=1      0.4728     2.1150    0.2849    0.7847
##
```

```
## Concordance= 0.78  (se = 0.021 )
## Likelihood ratio test= 122.4  on 3 df,   p=<2e-16
## Wald test            = 129.4  on 3 df,   p=<2e-16
## Score (logrank) test = 148.4  on 3 df,   p=<2e-16

fit3 = coxph( Surv( time, status == 2 ) ~ log(bili),
              data = pbc[ which( pbc$edema == 0 ), ]  )

summary(fit3)
## Call:
## coxph(formula = Surv(time, status == 2) ~ log(bili), data = pbc[which(pbc$edema ==
##     0), ])
##
##   n= 354, number of events= 116
##
##              coef exp(coef) se(coef)     z Pr(>|z|)
## log(bili) 1.02064   2.77498  0.09548 10.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##           exp(coef) exp(-coef) lower .95 upper .95
## log(bili)     2.775     0.3604     2.301     3.346
##
## Concordance= 0.783  (se = 0.021 )
## Likelihood ratio test= 107.6  on 1 df,   p=<2e-16
## Wald test            = 114.3  on 1 df,   p=<2e-16
## Score (logrank) test = 131.6  on 1 df,   p=<2e-16

fit2$coefficients[ 1 ]
## log(bili)
##  1.020643

fit4 <- coxph( Surv( time, status == 2 ) ~ log(bili),
               data = pbc[ which( pbc$edema == 0.5 ), ]  )

summary(fit4)
## Call:
## coxph(formula = Surv(time, status == 2) ~ log(bili), data = pbc[which(pbc$edema ==
##     0.5), ])
##
##   n= 44, number of events= 26
##
##             coef exp(coef) se(coef)     z Pr(>|z|)
## log(bili) 0.6474    1.9106   0.1736 3.729 0.000192 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##           exp(coef) exp(-coef) lower .95 upper .95
## log(bili)     1.911     0.5234      1.36     2.685
##
## Concordance= 0.738  (se = 0.057 )
## Likelihood ratio test= 13.37  on 1 df,   p=3e-04
## Wald test            = 13.91  on 1 df,   p=2e-04
```

```
## Score (logrank) test = 15.53  on 1 df,   p=8e-05

sum( fit2$coefficients[ c( 1, 2 ) ] )
## [1] 0.6474116

fit5 <- coxph( Surv( time, status == 2 ) ~ log(bili),
               data = pbc[ which( pbc$edema == 1 ), ]  )

summary(fit5)
## Call:
## coxph(formula = Surv(time, status == 2) ~ log(bili), data = pbc[which(pbc$edema ==
##     1), ])
##
##   n= 20, number of events= 19
##
##              coef exp(coef) se(coef)     z Pr(>|z|)
## log(bili) 0.2716    1.3120   0.2402 1.131    0.258
##
##           exp(coef) exp(-coef) lower .95 upper .95
## log(bili)     1.312     0.7622    0.8194     2.101
##
## Concordance= 0.59  (se = 0.081 )
## Likelihood ratio test= 1.36  on 1 df,   p=0.2
## Wald test            = 1.28  on 1 df,   p=0.3
## Score (logrank) test = 1.3  on 1 df,   p=0.3

sum( fit2$coefficients[ c( 1, 3 ) ] )
## [1] 0.2715844
```

We can observe that, whether all the covariates vary with the stratum level, it is identical to fit k different models (where k is the number of strata).

There is a clear analogy between this output and the output related to a linear regression with a categorical covariate.

---

3. Testing the proportional hazard assumption

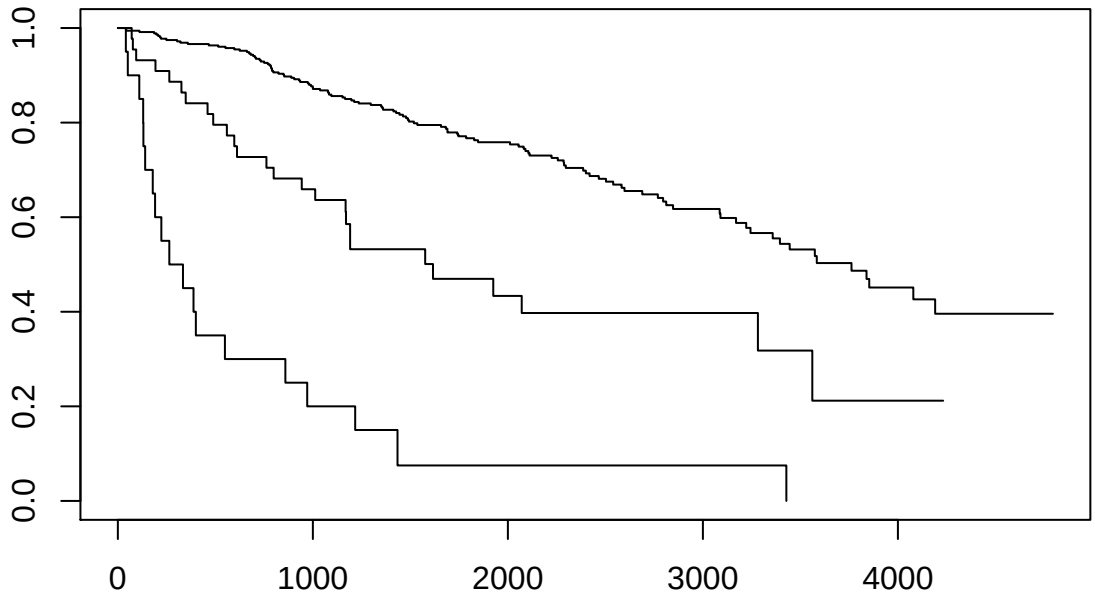
If proportional hazards hold and we consider **discrete** covariates, then the log survival curves should be almost parallel lines. Indeed, the survival function under the model satisfies:

$$S_i(t) = \exp(-\Lambda_0(t)\boldsymbol{\beta}\mathbf{x}_i)$$

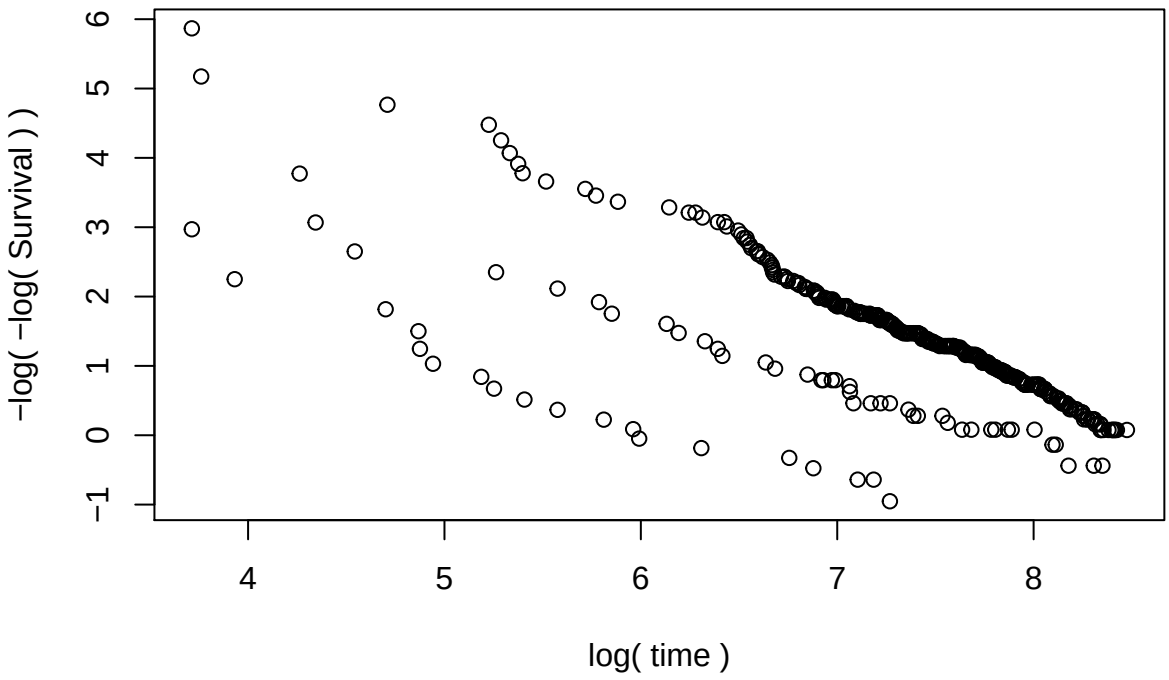
$$\log[-\log(S_i(t))] = \log[\Lambda_0(t)] - \mathbf{x}_i\boldsymbol{\beta}$$

We check the proportionality assumption in case of `pbc` data and `edema` covariate (the plot should be done in log-log scale).

```
km_pbc = survfit( Surv( time, status == 2) ~ edema, data = pbc )
names( km_pbc )
##  [1] "n"          "time"       "n.risk"     "n.event"    "n.censor"
##  [6] "surv"       "std.err"    "cumhaz"     "std.chaz"   "start.time"
## [11] "strata"     "type"       "logse"      "conf.int"   "conf.type"
## [16] "lower"      "upper"      "call"

plot( km_pbc )
```



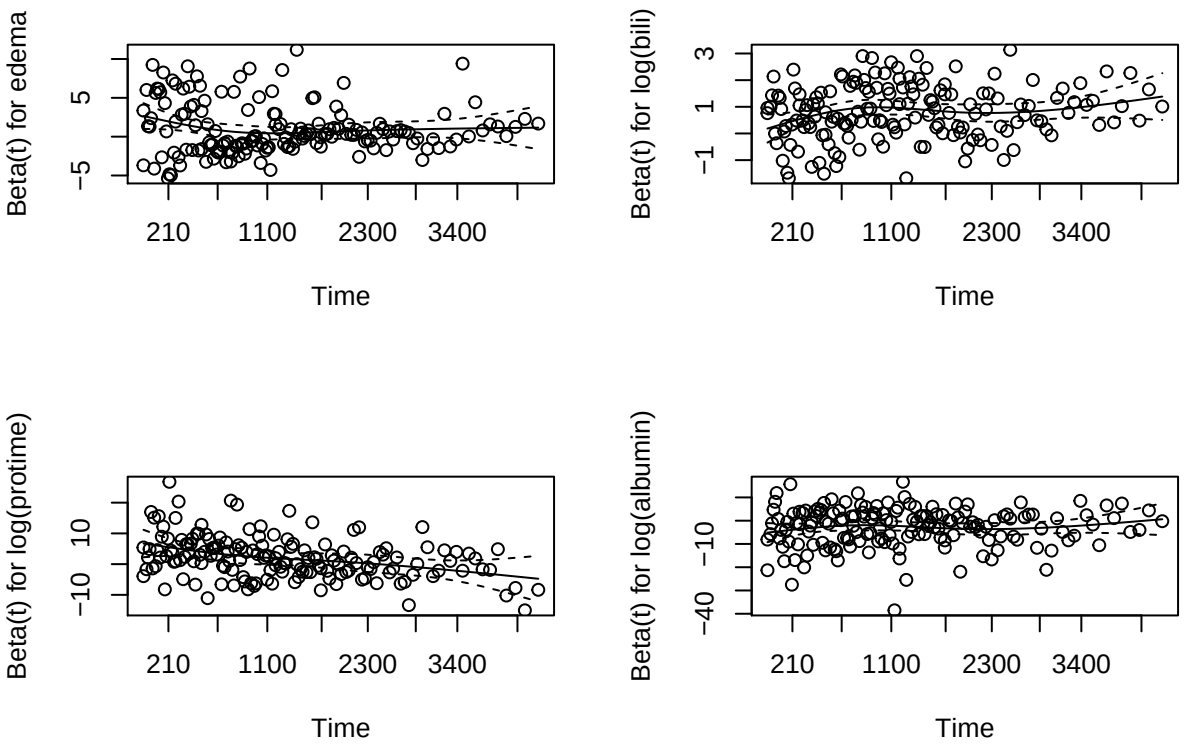

```
plot( log( km_pbc$time ), -log(-log( km_pbc$surv ) ),
      ylab = '-log( -log( Survival ) )', xlab = 'log( time )' )
```



We can state that the observed lines are almost parallel, so edema seems to respect the proportionality assumption.

In case of continuous covariate, it is not possible to have a graphical intuition of the proportionality of the hazard. We apply a `cox.zph` function, whose output is a matrix with one row for each variable, and optionally a last row for the global test. Columns of the matrix contain the correlation coefficient between transformed survival time and the scaled **Schoenfeld residuals**, a chi-square, and the two-sided p-value. For the global test there is no appropriate correlation, so an NA is entered into the matrix as a placeholder.

```
prop_haz_pbc = cox.zph( fit_test,  transform = 'km' )

prop_haz_pbc
##                  rho  chisq       p
## edema        -0.0783  0.968 0.32508
## log(bili)     0.1362  2.815 0.09336
## log(protime) -0.3636 12.312 0.00045
## log(albumin)  0.0468  0.371 0.54227
## GLOBAL            NA 17.353 0.00165

par(mfrow = c(2,2))
plot( prop_haz_pbc )
```



There is strong evidence for nonproportionality as shown by the large global test statistic. `log(protime)` does not respect the proportionality assumption and its impact changes over time, while `edema` has a constant effect with respect to time (see the former graphical interpretation).