TMA4275 Lifetime Analysis
Obligatory project 1

Jim Totland

2/4/2022

In this exercise we consider results from an old investigation to evaluate a histochemical marker that discriminates between primary breast cancer that has metastasized and that which has not. The marker under study is denoted HPA. Each tumor was treated with this marker and hence classified as either positively or negatively stained. The data which will be used is given below. The survival times of each woman is given in months and classified according to whether their tumor was negatively or positively stained. Censored survival times are labeled with an asterisk (*).

Negative	Positive
23	5
47	8
69	10
70*	13
71*	18
100*	24
101*	26
148	31
181	35
198*	50
208*	59
212*	61
224*	76*
	109*
	116*
	118
	143
	154*
	162*
	225*

In the following we denote patients with negatively stained tumors as group 1 and patients with positively stained tumors as group 2.

Problem 1

a)

First, a data frame containing the data presented above is constructed.

```r
# T.tilde corresponds to the right censored lifetimes,
# while D indicates if the lifetime is right-censored.
# This notation is introduced in problem 2.

df <- data.frame(
  T.tilde = c(23,47,69,70,71,100,101,148,181,198,208,212,224,
              5,8,10,13,18,24,26,31,35,50,59,61,76,109,116,118,143,154,162,225),
  stained =  c(0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),
  D = c(0,0,0,1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,1,1,1))
```

Next, the risk set for each of the two groups, $Y_1(t)$ and $Y_2(t)$ respectively, are plotted as functions of the study time, t. The result is given in figure 1.

```r
group1 <- filter(df, stained == 0)
group2 <- filter(df, stained == 1)
n <- 250 # Points on the time-axis
t <- 1:n
y1 <- rep(0,n)
y2 <- rep(0,n)
for(i in 1:length(t)){
  y1[i] <- sum(group1$T.tilde > t[i])
  y2[i] <- sum(group2$T.tilde > t[i])
}
ggplot(data.frame(t = t, y1 = y1, y2 = y2)) + geom_step(aes(t, y1, color = "1")) + geom_step(aes(t, y2,
  ylab("Number of Individuals") + xlab("t (study time)") +
  scale_color_manual(name = "Group", values = c("1" = "#e0474c", "2" = "#7ab8d6"))
```

From figure 1, we observe that group 2 has a much steeper trend than group 1.

b)

we want to compute the Nelson-Aalen estimator of the integrated hazard rate, $A(t)$, over the grid t and a corresponding confidence interval with significance level α is given. The Nelson-Aalen estimator is defined as

$$\hat{A}(t) = \sum_{T_j \leq t} \frac{1}{Y(T_j)},$$

where $Y(t)$ is the number of individuals at risk just before time t, and T_j are the so-called jump times. To find confidence intervals, we use the following estimate of the variance:

$$\hat{\sigma}^2(t) = \sum_{T_j \leq t} \frac{1}{Y(T_j)^2}.$$

We implement two types of confidence intervals; a 'regular' one given by

$$\hat{A}(t) \pm z_{1-\alpha/2} \cdot \hat{\sigma}(t),$$

which we reference to as a type 1 confidence interval. Additionally, we implement a confidence interval based on the log-transformation, which is given as

$$\hat{A}(t) \exp\left(\pm z_{1-\alpha/2} \cdot \hat{\sigma}(t)/\hat{A(t)}\right),$$

and which we refer to as a type 2 confidence interval. The code which implements this is given below.

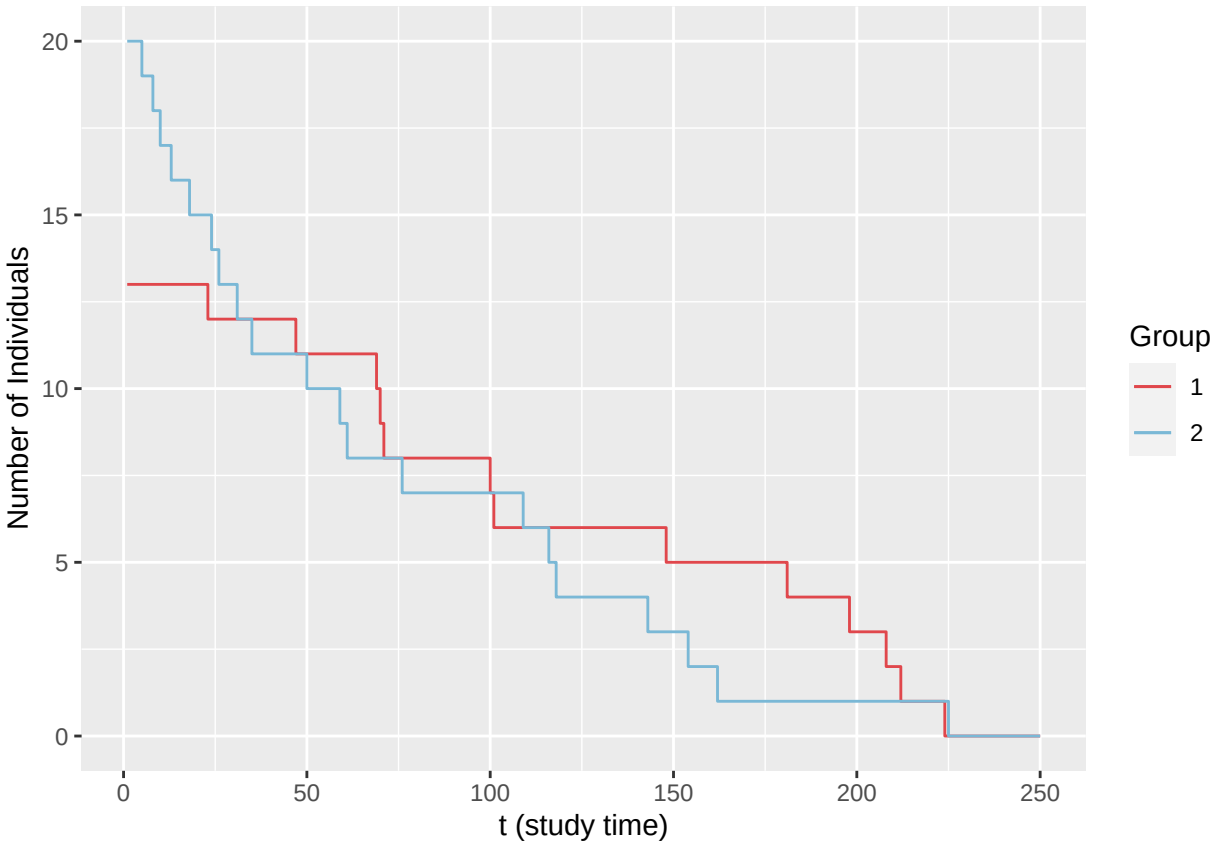

Figure 1: The risk set of group 1 and 2.

```r
#library(survival)


compute.A.hat <- function(y, t){
  idx <- c(0,(diff(y, lag = 1) < 0)) > 0
  jumps <- rep(Inf, length(y))
  jumps[idx] <- y[idx] + 1
  result <- cumsum(1/jumps)
  return(result)
}

compute.sigma.hat <- function(y,t){
  idx <- c(0,(diff(y, lag = 1) < 0)) > 0
  jumps <- rep(Inf, length(y))
  jumps[idx] <- y[idx] + 1
  result <- sqrt(cumsum(1/jumps^2))
  return(result)
}

Nelson.Aalen <- function(df, alpha, conf.int.type){

    # df: data frame with survival times
    # alpha: significance level of conf.int
    # conf.int.type: type of confidence interval, (1) regular or (2) log-type.
    # t: time grid

  times <- df$T.tilde
  n <- length(times) # Number of individuals
  status <- !(df$D) # True if not censored
  m <- sum(status) # Number of uncensored times
  event.times <- sort(times[status]) # Event times
  #print(event.times)
  y <- rep(n, m)
  for(i in 1:m){
    y[i] <- sum(times >= event.times[i])
  }
  # comp <- basehaz(coxph(Surv(T.tilde,D)~1,data=df))
  # print("heyo")
  # print(comp)
  #print(y)
  A.hat <- rep(0, m + 1)
  A.hat[2:(m + 1)] <- cumsum(1/y)
  #print(A.hat)
  sigma.hat <- rep(0, m + 1)
  sigma.hat[2:(m + 1)] = cumsum(1/y^2)
  z = qnorm(1 - alpha/2)
  upper <- lower <- NA
  if(conf.int.type == 1){
    upper <-  A.hat + z*sigma.hat
    lower <- A.hat - z*sigma.hat
  }
  else if(conf.int.type == 2){
    exponent <- z*sigma.hat/A.hat
```

```
    exponent[is.na(exponent)] <- 0
    upper <- A.hat * exp(exponent)
    lower <- A.hat * exp(-exponent)
  }
  else{
    stop("Invalid conf.int.type")
  }

  if(length(event.times) < 1){ # No event times (for simulation)
    event.times <- 0
    A.hat <- lower <- upper <- c(0,0)}
  return(list(A.hat = stepfun(event.times, A.hat),
              conf.int.lower = stepfun(event.times, lower),
              conf.int.upper = stepfun(event.times, upper)))
}
```

In the following, the Nelson-Aalen estimator is computed and plotted for the two groups. Figure 2 uses regular confidence intervals, while in figure 3, the confidence interval based on the log-transform is used.

```
t <- seq(1, 200, by = 0.1)

funcs.1 <- Nelson.Aalen(group1, 0.05, 1)
df.1 <- as_tibble(data.frame(t = t, A.hat = funcs.1$A.hat(t),
                   lower = funcs.1$conf.int.lower(t),
                  upper = funcs.1$conf.int.upper(t)))

funcs.2 <- Nelson.Aalen(group2, 0.05, 1)
df.2 <- as_tibble(data.frame(t = t, A.hat = funcs.2$A.hat(t),
                   lower = funcs.2$conf.int.lower(t),
                   upper = funcs.2$conf.int.upper(t)))

ggplot(df.1) + geom_step(aes(x = t, y = A.hat, color = "1")) +
  geom_stepconfint(aes(x = t, ymin=lower, ymax=upper),fill = "#e0474c", alpha = 0.2) +
  geom_step(data = fortify(df.2), aes(x = t, y = A.hat, color = "2")) +
  geom_stepconfint(data = fortify(df.2), aes(x = t, ymin=lower, ymax=upper), fill = "#7ab8d6", alpha =
  scale_color_manual(name = "Group", values = c("1" = "#e0474c", "2" = "#7ab8d6"))
```

```
t <- seq(1, 200, by = 0.1)

funcs.1 <- Nelson.Aalen(group1, 0.05, 2)
df.1 <- as_tibble(data.frame(t = t, A.hat = funcs.1$A.hat(t),
                   lower = funcs.1$conf.int.lower(t),
                  upper = funcs.1$conf.int.upper(t)))

funcs.2 <- Nelson.Aalen(group2, 0.05, 2)
df.2 <- as_tibble(data.frame(t = t, A.hat = funcs.2$A.hat(t),
                   lower = funcs.2$conf.int.lower(t),
                   upper = funcs.2$conf.int.upper(t)))

ggplot(df.1) + geom_step(aes(x = t, y = A.hat, color = "1")) +
  geom_stepconfint(aes(x = t, ymin=lower, ymax=upper),fill = "#e0474c", alpha = 0.2) +
  geom_step(data = fortify(df.2), aes(x = t, y = A.hat, color = "2")) +
```

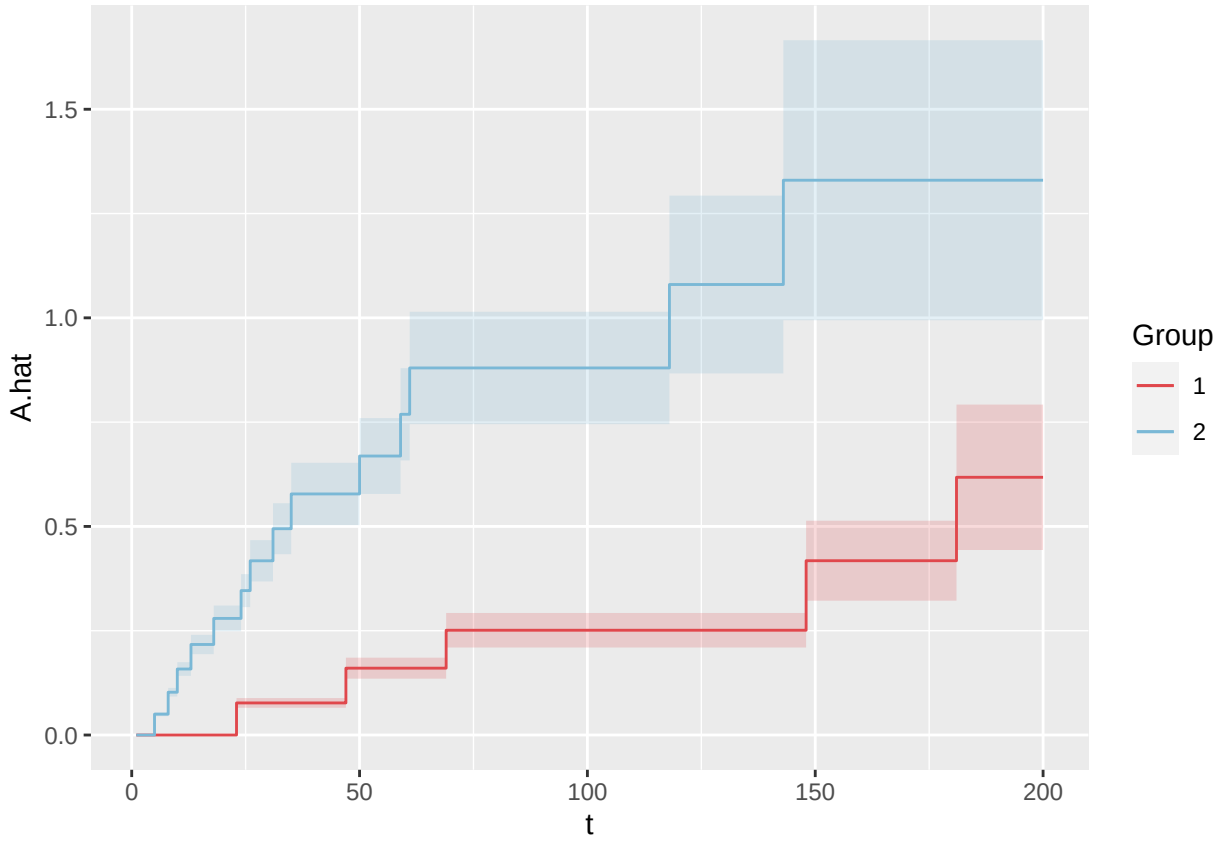

Figure 2: The Nelson-Aalen estimator of the integrated hazard rate with 95%-confidence intervals of type 1

```
geom_stepconfint(data = fortify(df.2), aes(x = t, ymin=lower, ymax=upper), fill = "#7ab8d6", alpha = 
scale_color_manual(name = "Group", values = c("1" = "#e0474c", "2" = "#7ab8d6"))
```

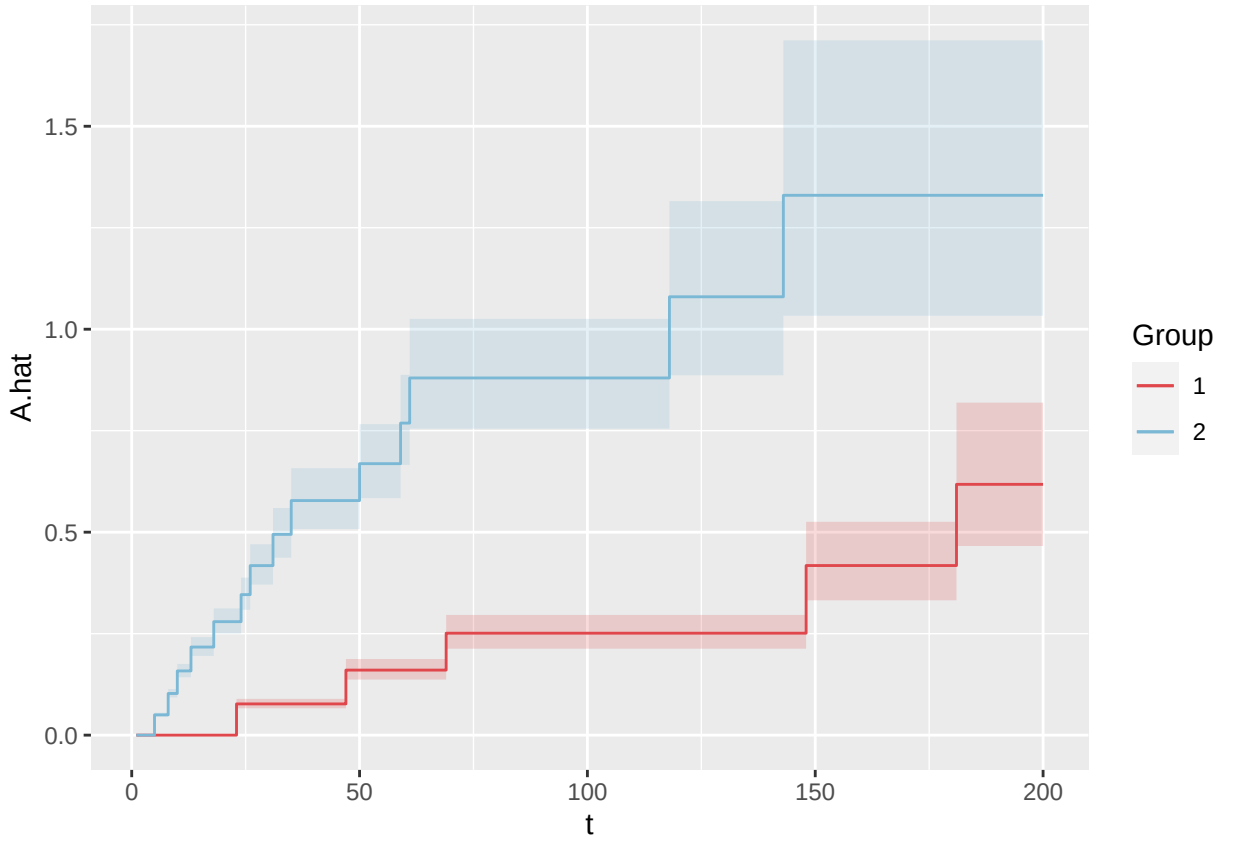

Figure 3: The Nelson-Aalen estimator of the integrated hazard rate with 95%-confidence intervals of type 2

From figure 2 and 3, we observe that $\hat{A}(t)$ grows much faster for group 2 than for group 1.

Problem 2

a)

We now want to evaluate the quality of the confidence intervals found in problem 1. We assume to have a group of n individuals with independent and identically distributed lifetimes. Let the hazard rate of the individuals be given by

$$\lambda(t) = 0.027 \cdot t^{-0.4}.$$

To find the CDF and PDF of the lifetime(s), T, of the individuals, we utilize that

$$\begin{aligned} P(T > t) := S_T(t) &= \exp\left(-\int_0^t \lambda(s)ds\right) \\ &= \exp\left(-\frac{0.027}{0.6}s^{0.6}\Big|_0^t\right) = \exp\left(-\frac{9}{200}t^{0.6}\right). \end{aligned}$$

Then the CDF is given as,

$$F_T(t) = 1 - S_T(t) = 1 - \exp\left(-\frac{9}{200}t^{0.6}\right),$$

and the PDF is given as

$$f_T(t) = \frac{d}{dt}F_T(t) = 0.027t^{-0.4}\exp\left(-\frac{9}{200}t^{0.6}\right) = \lambda(t)\exp\left(-\frac{9}{200}t^{0.6}\right).$$

The censoring times, C, are exponentially distributed with PDF

$$f_C(c;\lambda) = \lambda e^{-\lambda c}.$$

Consequently, their CDF is given as

$$F_C(c;\lambda) = \int_0^c \lambda e^{-\lambda x}dx = 1 - e^{-\lambda c},$$

and

$$S_C(c;\lambda) = 1 - F_C(c;\lambda) = e^{-\lambda c}.$$

We denote the hazard rate of the censoring times by $\alpha(c)$ and find it through the following relation.

$$\alpha(c) = -\frac{S_C'(c)}{S_C(c)} = \frac{-\lambda e^{-\lambda c}}{e^{-\lambda c}} = \lambda.$$

A plot of $\lambda(t)$ and $\alpha(c)$ is given in figure 4.

```
T.hazard <- function(t){
  return(0.027*t^(-0.4))
}

lambda <- 0.02
C.hazard <- function(c){
  return(lambda)
}

ggplot(data.frame(t = seq(0, 10, by = 0.01)), aes(t)) +
  geom_function(fun = T.hazard, aes(color = "T")) +
  geom_function(fun = C.hazard, aes(color = "C")) +
  scale_color_manual(name = "", values = c("T" = "#e0474c", "C" = "#7ab8d6")) + ylab("Hazard rate")
```

The densities, $f_C(c)$ and $f_T(t)$ are plotted in figure 5.

```
f.T <- function(t){
  return(T.hazard(t)*exp(-9/200 * t^(0.6)))
}

f.C <- function(c){
  return(lambda*exp(-lambda*c))
}

ggplot(data.frame(t = seq(0, 500, by = 0.01)), aes(t)) +
  geom_function(fun = f.T, aes(color = "T")) +
  geom_function(fun = f.C, aes(color = "C")) +
  scale_color_manual(name = "", values = c("T" = "#e0474c", "C" = "#7ab8d6")) + ylab("Density")
```

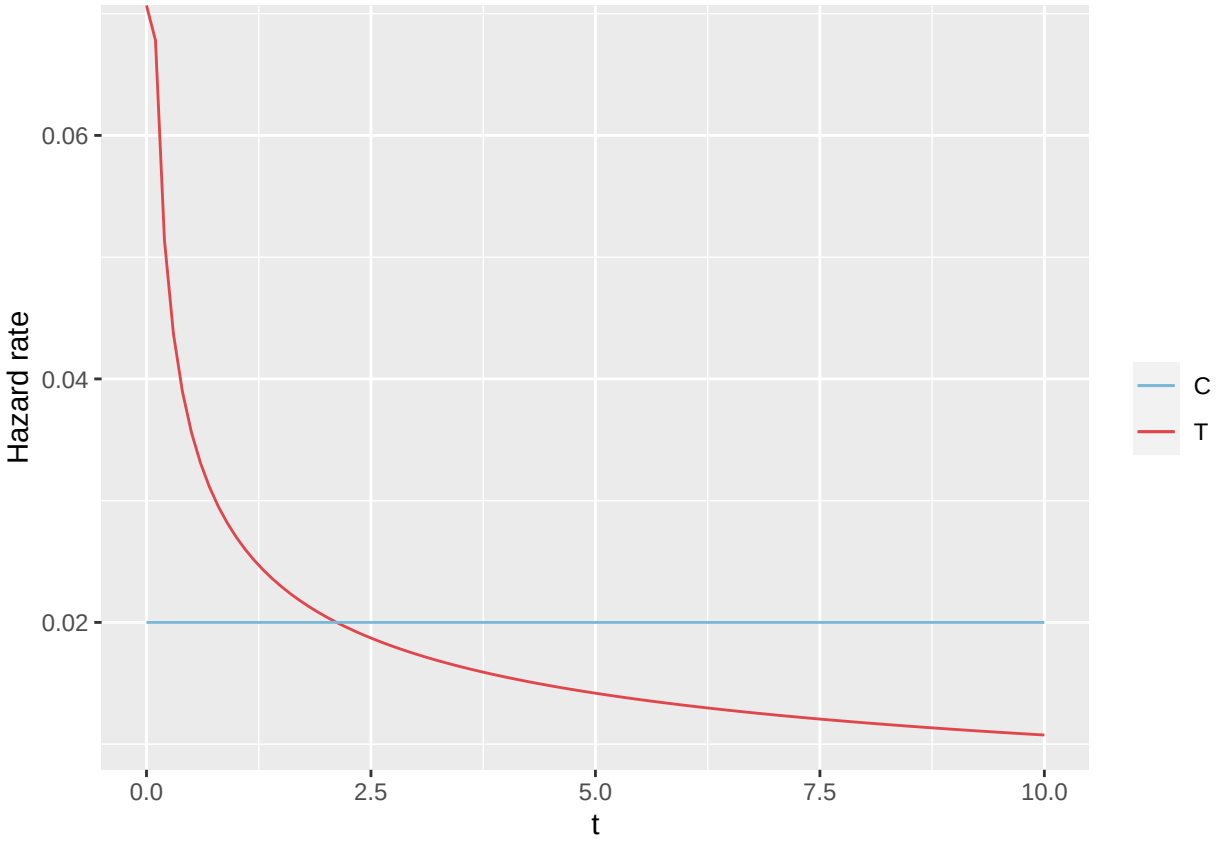

Figure 4: The hazard rates of the lifetimes and censoring times.

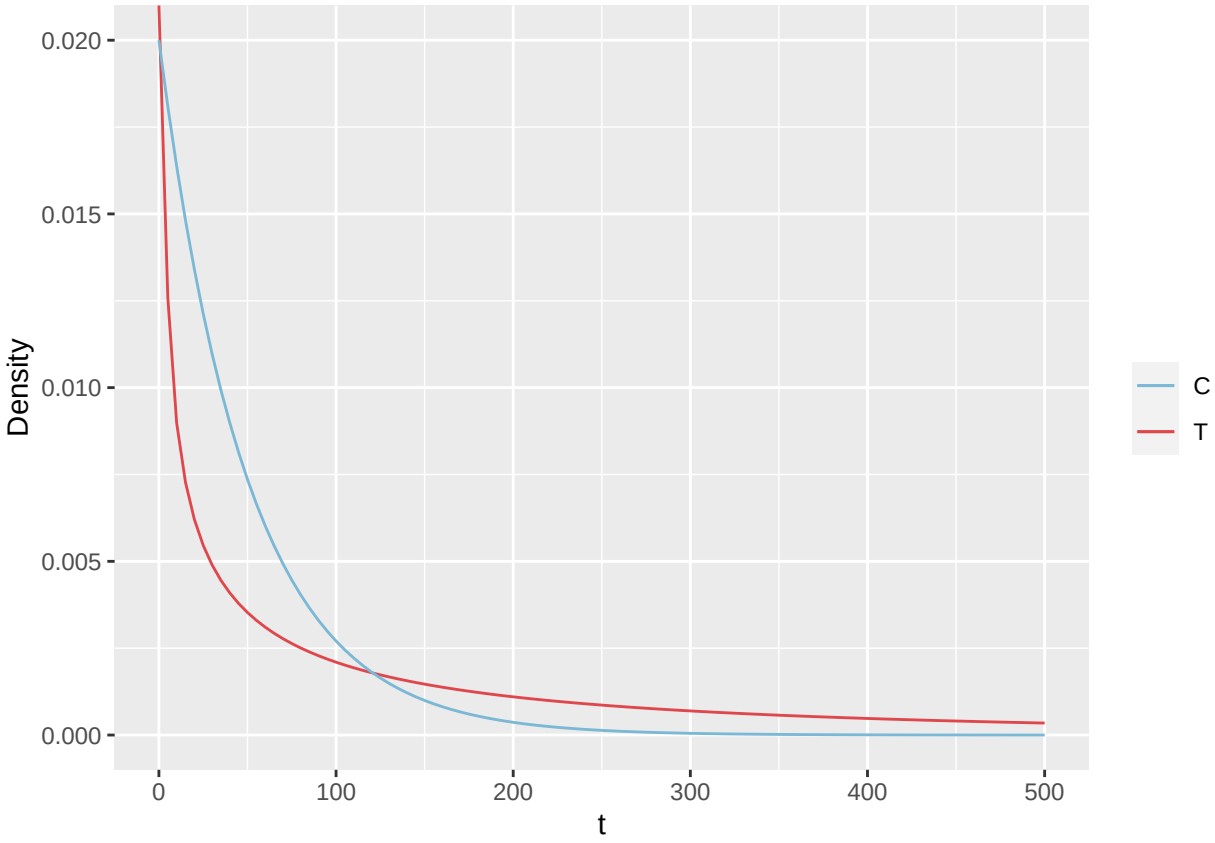

Figure 5: The PDFs of the lifetimes and censoring times.

b)

For individual number i, we define the right censored survival time

$$\widetilde{T}_i = \min\{T_i, C_i\},$$

which is what we observe. We also define the censoring indicator

$$D_i = \begin{cases} 1, & \text{if } T \leq C_i \\ 0, & \text{otherwise.} \end{cases}$$

An R-function which simulates $\widetilde{T}$ and D_i for n individuals is given below. To simulate from f_T and f_C, we use the probability integral transform method.

```
sim.C <- function(n, lambda){
  u <- runif(n)
  c <- -log(1-u)/lambda
  return(c)
}

sim.T <- function(n){
  u <- runif(n)
  t <- (-200/9 * log(1 - u))^(5/3)
  return(t)
}

sim.TD <- function(n, lambda, floor = TRUE){
  t <- sim.T(n)
  c <- sim.C(n, lambda)
  t.tilde = pmin(t,c)
  if(floor){
    t.tilde = floor((t.tilde))
  }
  d <- (c < t)

  df <- data.frame(T.tilde = t.tilde, D = d)
  if(max(t.tilde) < 150){
    sim.TD(n, lambda)
  } else{
    return(df)
  }
}
```

c)

Below is code which simulated and plots the Nelson-Aalen estimator. Figure 6 and figure 7 shows two such simulations plotted with the two different types of confident intervals.

```
n1 <- 13
n2 <- 20

sim1 <- sim.TD(n1, lambda, floor = FALSE)
```

```r
sim2 <- sim.TD(n2, lambda, floor = FALSE)

tmax <- max(sim1,sim2)
t <- seq(1, tmax + 10, by = 0.1)

funcs.1 <- Nelson.Aalen(sim1, 0.05, 1)
df.1 <- as_tibble(data.frame(t = t, A.hat = funcs.1$A.hat(t),
                    lower = funcs.1$conf.int.lower(t),
                   upper = funcs.1$conf.int.upper(t)))

funcs.2 <- Nelson.Aalen(sim2, 0.05, 1)
df.2 <- as_tibble(data.frame(t = t, A.hat = funcs.2$A.hat(t),
                    lower = funcs.2$conf.int.lower(t),
                    upper = funcs.2$conf.int.upper(t)))

ggplot(df.1) + geom_step(aes(x = t, y = A.hat, color = "1")) +
  geom_stepconfint(aes(x = t, ymin=lower, ymax=upper),fill = "#e0474c", alpha = 0.2) +
  geom_step(data = fortify(df.2), aes(x = t, y = A.hat, color = "2")) +
  geom_stepconfint(data = fortify(df.2), aes(x = t, ymin=lower, ymax=upper),
                   fill = "#7ab8d6", alpha = 0.2) +
  scale_color_manual(name = "Group", values = c("1" = "#e0474c", "2" = "#7ab8d6"))
```

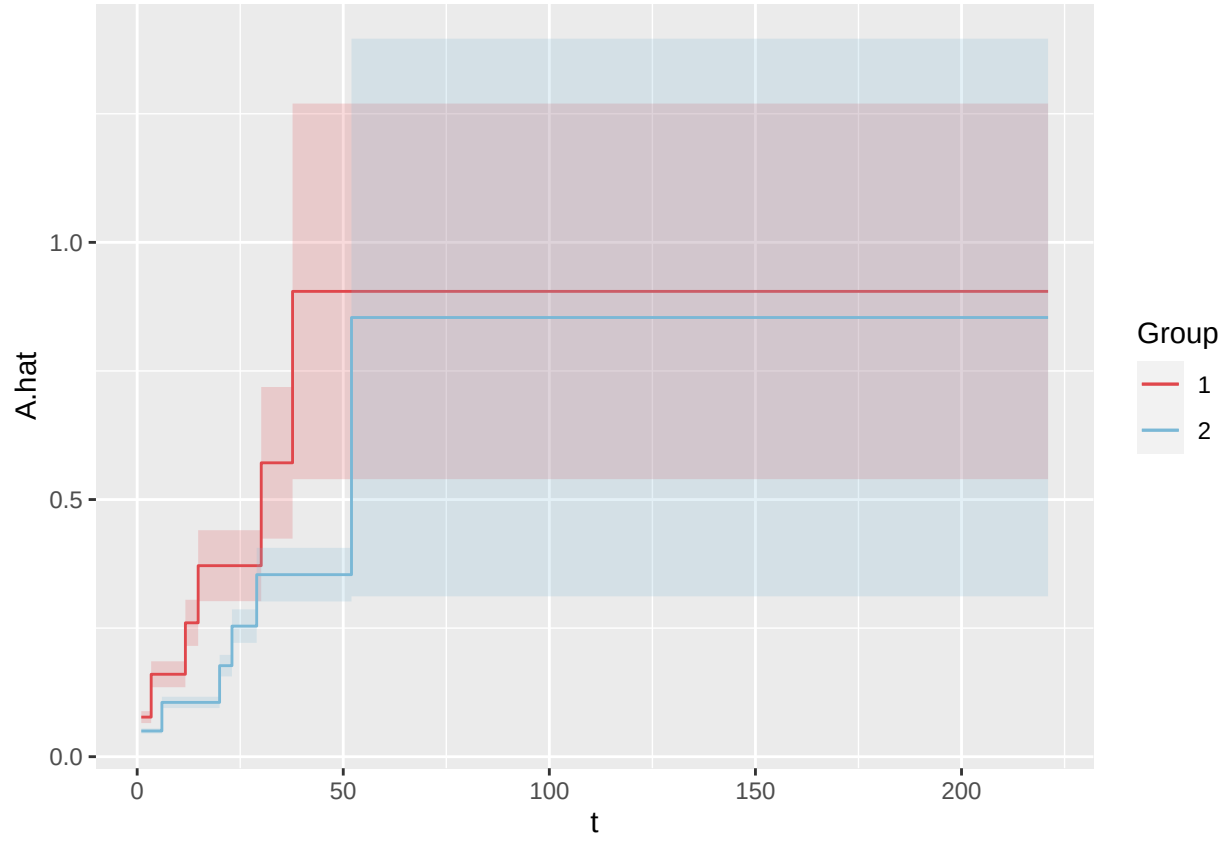

Figure 6: Nelson-Aalen estimator for simulated data. We have used 95% cconfidence intervals of type 1

```r
sim3 <- sim.TD(n1, lambda, floor = FALSE)
sim4 <- sim.TD(n2, lambda, floor = FALSE)

tmax <- max(sim3,sim4)
t <- seq(1, tmax + 10, by = 0.1)

funcs.3 <- Nelson.Aalen(sim3, 0.05, 2)
df.3 <- as_tibble(data.frame(t = t, A.hat = funcs.1$A.hat(t),
                   lower = funcs.1$conf.int.lower(t),
                  upper = funcs.1$conf.int.upper(t)))

funcs.4 <- Nelson.Aalen(sim4, 0.05, 2)
df.4 <- as_tibble(data.frame(t = t, A.hat = funcs.2$A.hat(t),
                   lower = funcs.2$conf.int.lower(t),
                   upper = funcs.2$conf.int.upper(t)))

ggplot(df.1) + geom_step(aes(x = t, y = A.hat, color = "1")) +
  geom_stepconfint(aes(x = t, ymin=lower, ymax=upper),fill = "#e0474c", alpha = 0.2) +
  geom_step(data = fortify(df.2), aes(x = t, y = A.hat, color = "2")) +
  geom_stepconfint(data = fortify(df.2), aes(x = t, ymin=lower, ymax=upper),
                   fill = "#7ab8d6", alpha = 0.2) +
  scale_color_manual(name = "Group", values = c("1" = "#e0474c", "2" = "#7ab8d6"))
```

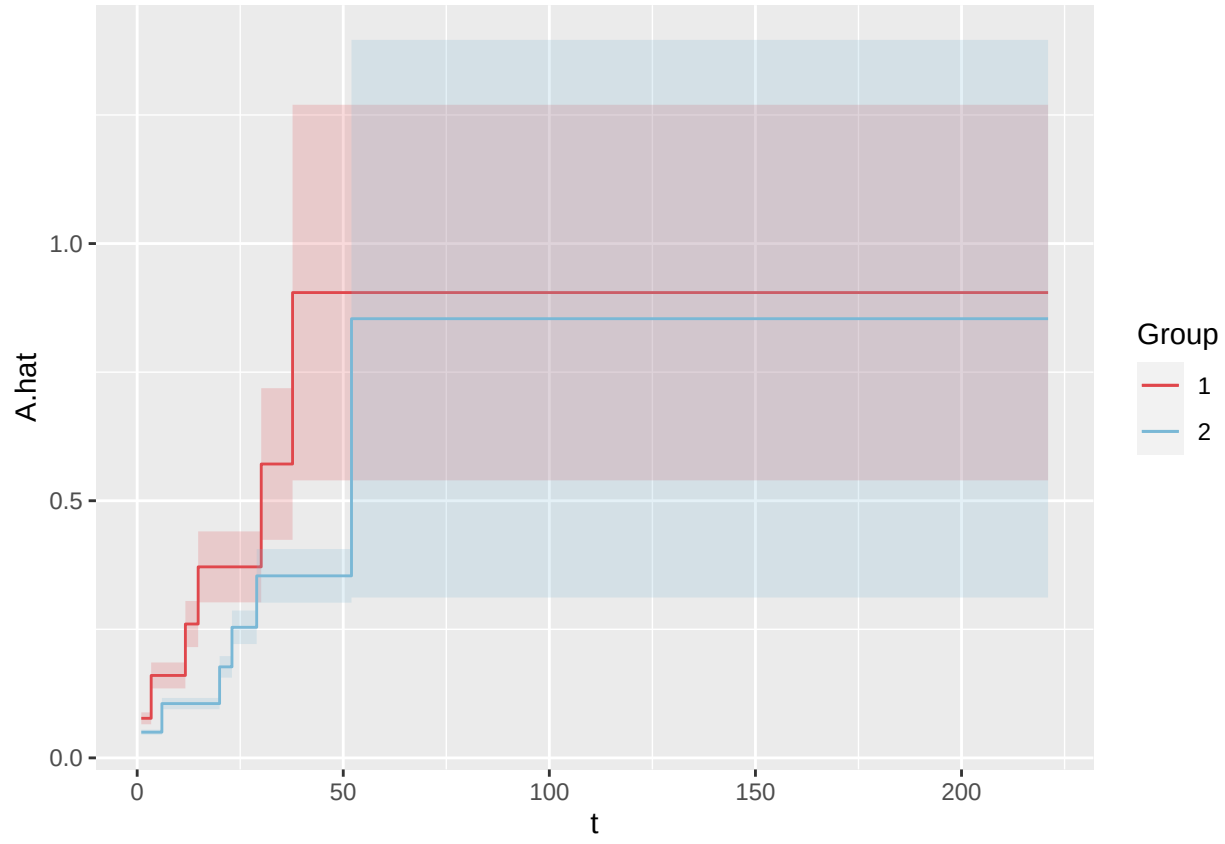

Figure 7: Nelson-Aalen estimator for simulated data. We have used 95% cconfidence intervals of type 2

After running the above code a dozen times, it is seems that the confidence intervals will usually overlap for

the two groups, at least for $t > 50$. Since the confidence 95%-confidence intervals of the 'real' groups never overlap, this suggests that the hazard rates do in fact differ.

d)

```
point.conf.int <- function(sim.df, t.point, alpha, conf.int.type){
  funcs <- Nelson.Aalen(sim.df, alpha, conf.int.type)
  return(c(funcs$conf.int.lower(t.point),funcs$conf.int.upper(t.point)))
}
```

e)

Now we want to assess the coverage probability of the different types of confidence intervals we have considered. Firstly, we note that the analytic expression for the integrated hazard rate is

$$A(t) = \int_0^t \lambda(s)ds = \frac{9}{200}t^{0.6}.$$

For all combinations $n \in \{13, 20\}$, $t \in \{50, 100, 150\}$ and $\alpha \in \{0.01, 0.05, 0.10\}$ we want to estimate the coverage probability of the approximate confidence intervals. An R-function which does this for one such combination is given below.

```
A.exact <- function(t){return(9/200 * t^0.6)}

est.coverage <- function(n, t.point, alpha, iter, conf.int.type){
  result.vec = rep(NA, iter)
  A <- A.exact(t.point)
  print(A)
  for(i in 1:iter){
    sim.df <- sim.TD(n, lambda)
    conf.int <- point.conf.int(sim.df, t.point, alpha, conf.int.type)
    #print(conf.int)
    result.vec[i] = (A > conf.int[1] & A < conf.int[2])
  }
  return(sum(result.vec)/length(result.vec))
}
```

```
n <- 13
t.point <- 50
alpha <- 0.10
iter <- 1000
conf.int.type <- 2
est.coverage(n, t.point, alpha, iter, conf.int.type)
```

```
## [1] 0.4705378
```

```
## [1] 0.28
```

We first do the simulation for confidence intervals of type 1:

```r
n <- c(13, 20)
t.points <- c(50, 100, 150)
alpha <- c(0.01, 0.05, 0.10)

params <- expand.grid(n, t.points, alpha)
result.1 <- cbind(params, rep(NA, 18))
colnames(result.1) <- c("n", "t.point", "alpha", "cov.prob")
iter <- 100
conf.int.type <- 1

for(i in 1:nrow(params)){
  #print(result[i, ])
  n <- result.1$n[i]
  alpha <- result.1$alpha[i]
  t.point <- result.1$t.point[i]
  alpha <- result.1$alpha[i]
  result.1$cov.prob[i] <- est.coverage(n, t.point, alpha, iter, conf.int.type)
}
```

```
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
```

```r
result.1
```

```
##     n t.point alpha cov.prob
## 1  13      50  0.01     0.52
## 2  20      50  0.01     0.41
## 3  13     100  0.01     0.49
## 4  20     100  0.01     0.49
## 5  13     150  0.01     0.38
## 6  20     150  0.01     0.34
## 7  13      50  0.05     0.45
## 8  20      50  0.05     0.31
## 9  13     100  0.05     0.39
## 10 20     100  0.05     0.40
## 11 13     150  0.05     0.31
## 12 20     150  0.05     0.38
```

```
## 13 13      50  0.10     0.24
## 14 20      50  0.10     0.31
## 15 13     100  0.10     0.41
## 16 20     100  0.10     0.31
## 17 13     150  0.10     0.41
## 18 20     150  0.10     0.30
```

Then we consider confidence intervals of type 2:

```
result.2 <- cbind(params, rep(NA, 18))
colnames(result.2) <- c("n", "t.point", "alpha", "cov.prob")
iter <- 100
conf.int.type <- 2

for(i in 1:nrow(params)){
  #print(result[i, ])
  n <- result.2$n[i]
  alpha <- result.2$alpha[i]
  t.point <- result.2$t.point[i]
  alpha <- result.2$alpha[i]
  result.2$cov.prob[i] <- est.coverage(n, t.point, alpha, iter, conf.int.type)
}
```

```
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
## [1] 0.4705378
## [1] 0.4705378
## [1] 0.7132019
## [1] 0.7132019
## [1] 0.9096352
## [1] 0.9096352
```

```
result.2
```

```
##     n t.point alpha cov.prob
## 1  13      50  0.01     0.44
## 2  20      50  0.01     0.40
## 3  13     100  0.01     0.54
## 4  20     100  0.01     0.48
## 5  13     150  0.01     0.39
## 6  20     150  0.01     0.48
## 7  13      50  0.05     0.33
## 8  20      50  0.05     0.28
```

```
## 9  13     100  0.05     0.42
## 10 20     100  0.05     0.42
## 11 13     150  0.05     0.30
## 12 20     150  0.05     0.46
## 13 13      50  0.10     0.27
## 14 20      50  0.10     0.29
## 15 13     100  0.10     0.39
## 16 20     100  0.10     0.37
## 17 13     150  0.10     0.29
## 18 20     150  0.10     0.37
```