Survival analysis project

DC, RD, RL, RR

2023-07-07

1/ENVIRONMENT PREPARATION

First, let's install the libraries that will be required in our analysis

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.5
## v tibble  3.1.8     v dplyr   1.0.10
## v tidyr   1.2.1     v stringr 1.4.1
## v readr   2.1.3     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
library(survival)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

2/DATA PREPARATION

First, we need to specify the path where the dataset is located. You need to amend in with your own path

```
setwd ('C:/Users/romai/Documents/DSTI/21-Survival Analysis/UTMB')
data_utmb17 <- read_csv("utmb_2017.csv", col_names = TRUE)
```

```
## New names:
## Rows: 2535 Columns: 33
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
```

```
## (4): name, team, category, nationality dbl (3): ...1, bib, rank time (26):
## time, timediff, Delevret, St-Gervais, Contamines, La Balme, Bonho...
## i Use ‘spec()‘ to retrieve the full column specification for this data. i
## Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
## * ‘‘ -> ‘...1‘
```

```r
head(data_utmb17)
```

```
## # A tibble: 6 x 33
##    ...1   bib name        team  categ~1  rank natio~2 time     timediff Delevret
##   <dbl> <dbl> <chr>       <chr> <chr>   <dbl> <chr>   <time>   <time>   <time>
## 1     0     4 D'HAENE Fr~ Salo~ SE H        1 FR      19:01:54 00:00:00 01:11:50
## 2     1     2 JORNET BUR~ Salo~ SE H        2 ES      19:16:59 00:15:05 01:10:00
## 3     2    14 TOLLEFSON ~ Hoka  SE H        3 US      19:53:00 00:51:06 01:15:24
## 4     3     7 THEVENARD ~ Asics SE H        4 FR      20:03:39 01:01:45 01:11:51
## 5     4     1 WALMSLEY J~ Hoka  SE H        5 US      20:11:38 01:09:44 01:09:59
## 6     5    17 CAPELL Pau  The ~ SE H        6 ES      20:12:43 01:10:49 01:13:16
## # ... with 23 more variables: ‘St-Gervais‘ <time>, Contamines <time>,
## #   ‘La Balme‘ <time>, Bonhomme <time>, Chapieux <time>, ‘Col Seigne‘ <time>,
## #   ‘Lac Combal‘ <time>, ‘Mt-Favre‘ <time>, Checruit <time>, Courmayeur <time>,
## #   Bertone <time>, Bonatti <time>, Arnouvaz <time>, ‘Col Ferret‘ <time>,
## #   ‘La Fouly‘ <time>, ‘Champex La‘ <time>, ‘La Giète‘ <time>, Trient <time>,
## #   ‘Les Tseppe‘ <time>, Vallorcine <time>, ‘Col Montet‘ <time>,
## #   Flégère <time>, Arrivée <time>, and abbreviated variable names ...
```

Let’s check if we get some problems during the data import

```r
problems(data_utmb17)
```

```
## # A tibble: 0 x 5
## # ... with 5 variables: row <int>, col <int>, expected <chr>, actual <chr>,
## #   file <chr>
```

Let’s have a quick look on the dataset. What are the columns?

```r
colnames(data_utmb17)
```

```
##  [1] "...1"         "bib"          "name"         "team"         "category"
##  [6] "rank"         "nationality"  "time"         "timediff"     "Delevret"
## [11] "St-Gervais"   "Contamines"   "La Balme"     "Bonhomme"     "Chapieux"
## [16] "Col Seigne"   "Lac Combal"   "Mt-Favre"     "Checruit"     "Courmayeur"
## [21] "Bertone"      "Bonatti"      "Arnouvaz"     "Col Ferret"   "La Fouly"
## [26] "Champex La"   "La Giète"     "Trient"       "Les Tseppe"   "Vallorcine"
## [31] "Col Montet"   "Flégère"      "Arrivée"
```

Let’s get a bit more details on columns (type, etc)

```r
str(data_utmb17)
```

```
## spec_tbl_df [2,535 x 33] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ...1       : num [1:2535] 0 1 2 3 4 5 6 7 8 9 ...
```

```
##  $ bib        : num [1:2535] 4 2 14 7 1 17 9 13 8 32 ...
##  $ name       : chr [1:2535] "D'HAENE François" "JORNET BURGADA Kilian" "TOLLEFSON Tim" "THEVENARD Xa
##  $ team       : chr [1:2535] "Salomon" "Salomon" "Hoka" "Asics" ...
##  $ category   : chr [1:2535] "SE H" "SE H" "SE H" "SE H" ...
##  $ rank       : num [1:2535] 1 2 3 4 5 6 7 8 9 10 ...
##  $ nationality: chr [1:2535] "FR" "ES" "US" "FR" ...
##  $ time       : 'hms' num [1:2535] 19:01:54 19:16:59 19:53:00 20:03:39 ...
##   ..- attr(*, "units")= chr "secs"
##  $ timediff   : 'hms' num [1:2535] 00:00:00 00:15:05 00:51:06 01:01:45 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Delevret   : 'hms' num [1:2535] 01:11:50 01:10:00 01:15:24 01:11:51 ...
##   ..- attr(*, "units")= chr "secs"
##  $ St-Gervais : 'hms' num [1:2535] 01:45:05 01:44:21 01:48:38 01:45:08 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Contamines : 'hms' num [1:2535] 02:41:09 02:41:01 02:45:17 02:41:11 ...
##   ..- attr(*, "units")= chr "secs"
##  $ La Balme   : 'hms' num [1:2535] 03:33:40 03:33:45 03:41:50 03:33:45 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Bonhomme   : 'hms' num [1:2535] 04:28:07 04:29:18 04:41:04 04:38:06 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Chapieux   : 'hms' num [1:2535] 04:53:31 04:54:39 05:10:05 05:07:23 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Col Seigne : 'hms' num [1:2535] 06:18:02 06:18:04 06:40:51 06:41:10 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Lac Combal : 'hms' num [1:2535] 06:37:51 06:37:54 07:02:40 07:04:45 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Mt-Favre   : 'hms' num [1:2535] 07:15:35 07:15:37 07:42:45 07:45:38 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Checruit   : 'hms' num [1:2535] 07:39:09 07:39:16 08:08:05 08:11:11 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Courmayeur : 'hms' num [1:2535] 08:02:18 08:02:49 08:33:53 08:37:54 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Bertone    : 'hms' num [1:2535] 08:54:29 08:57:30 09:29:48 09:38:22 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Bonatti    : 'hms' num [1:2535] 09:44:00 09:48:28 10:21:27 10:31:58 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Arnouvaz   : 'hms' num [1:2535] 10:17:44 10:23:53 10:55:21 11:09:38 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Col Ferret : 'hms' num [1:2535] 11:11:12 11:18:54 NA 12:09:17 ...
##   ..- attr(*, "units")= chr "secs"
##  $ La Fouly   : 'hms' num [1:2535] 12:04:26 12:12:40 12:46:12 13:00:59 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Champex La : 'hms' num [1:2535] 13:24:20 13:33:52 14:08:23 14:22:44 ...
##   ..- attr(*, "units")= chr "secs"
##  $ La Giète   : 'hms' num [1:2535] 14:55:05 15:13:06 15:45:55 15:58:54 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Trient     : 'hms' num [1:2535] 15:24:59 15:41:22 16:12:00 16:28:53 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Les Tseppe : 'hms' num [1:2535] 16:06:17 16:23:16 16:56:16 17:12:35 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Vallorcine : 'hms' num [1:2535] 16:51:13 17:05:14 17:39:45 17:55:20 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Col Montet : 'hms' num [1:2535] 17:20:02 17:34:21 18:09:03 18:23:24 ...
##   ..- attr(*, "units")= chr "secs"
```

```
##  $ Flégère    : 'hms' num [1:2535] 18:23:09 18:39:27 19:17:41 19:28:04 ...
##   ..- attr(*, "units")= chr "secs"
##  $ Arrivée    : 'hms' num [1:2535] 19:01:54 19:16:59 19:53:00 20:03:39 ...
##   ..- attr(*, "units")= chr "secs"
##  - attr(*, "spec")=
##   .. cols(
##   ..   ...1 = col_double(),
##   ..   bib = col_double(),
##   ..   name = col_character(),
##   ..   team = col_character(),
##   ..   category = col_character(),
##   ..   rank = col_double(),
##   ..   nationality = col_character(),
##   ..   time = col_time(format = ""),
##   ..   timediff = col_time(format = ""),
##   ..   Delevret = col_time(format = ""),
##   ..   `St-Gervais` = col_time(format = ""),
##   ..   Contamines = col_time(format = ""),
##   ..   `La Balme` = col_time(format = ""),
##   ..   Bonhomme = col_time(format = ""),
##   ..   Chapieux = col_time(format = ""),
##   ..   `Col Seigne` = col_time(format = ""),
##   ..   `Lac Combal` = col_time(format = ""),
##   ..   `Mt-Favre` = col_time(format = ""),
##   ..   Checruit = col_time(format = ""),
##   ..   Courmayeur = col_time(format = ""),
##   ..   Bertone = col_time(format = ""),
##   ..   Bonatti = col_time(format = ""),
##   ..   Arnouvaz = col_time(format = ""),
##   ..   `Col Ferret` = col_time(format = ""),
##   ..   `La Fouly` = col_time(format = ""),
##   ..   `Champex La` = col_time(format = ""),
##   ..   `La Giète` = col_time(format = ""),
##   ..   Trient = col_time(format = ""),
##   ..   `Les Tseppe` = col_time(format = ""),
##   ..   Vallorcine = col_time(format = ""),
##   ..   `Col Montet` = col_time(format = ""),
##   ..   Flégère = col_time(format = ""),
##   ..   Arrivée = col_time(format = "")
##   .. )
##  - attr(*, "problems")=<externalptr>
```

First column seems useless (it looks like a row numbering)

```
data_utmb17 <- data_utmb17[,-1]
```

We can see that column (category) contains 2 interesting information: age category and gender. Therefore, we can create 2 new columns for gender & age In addition, we add a column "status" (1 = finisher; 0 = DNF / did not finish) based on the presence or not of a time in the column "Arrivée"

```
data_utmb17 <-data_utmb17 |>
  mutate(gender = case_when(
    endsWith(category, " H") ~ "Male",
```

```r
    endsWith(category, " F") ~ "Female"),
    age = substring(data_utmb17$category, first=1, last=2),
    status = case_when(time != 'NA' ~ 1, TRUE ~ 0),
    .after ="category")
```

We can observe that there is no column capturing the latest/highest time for all individuals. Column "Arrivée" (Arrival <=> finish line) capture only finisher (status =1). Non-finisher individuals (status = 0) have only the last time corresponding to the time where they stop the race. Therefore, we create a new column a new column "HighestTime" to capture the information about the time-to-event regardless the status.

```r
data_utmb17$highesttime <- apply(data_utmb17[11:35], 1, function(x) max(x, na.rm = TRUE))
```

```
## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA

## Warning in max(x, na.rm = TRUE): no non-missing arguments, returning NA
```

```r
data_utmb17<-data_utmb17|>
  mutate(highesttime = replace_na(highesttime, '00:00:00'))
```

Format of the newly-created column "highesttime" is character preventing to apply survival analysis.

```r
str(data_utmb17$highesttime)
```

```
##  chr [1:2535] "19:01:54" "19:16:59" "19:53:00" "20:03:39" "20:11:38" ...
```

Therefore, we convert it in time format (expressed in seconds) creating a the final time column "timetoevent'

```r
data_utmb17$timetoevent<- lubridate::hms(data_utmb17$highesttime)
data_utmb17$timetoevent<- period_to_seconds(data_utmb17$timetoevent)
```

Then, we remove all intermediate checkpoints time that are not useful anymore for our analysis

```r
data_utmb17<- data_utmb17[,-11:-34]
colnames(data_utmb17)
```

```
##  [1] "bib"         "name"        "team"        "category"    "gender"
##  [6] "age"         "status"      "rank"        "nationality" "time"
## [11] "Arrivée"     "highesttime" "timetoevent"
```

We keep removing others useless columns * name * team : only few individuals show that the information * category : we split it in 2 new columns (gender and age) * nationality: removed because we don't have the information for all censored individuals * Arrivée (arrival): we capture it in the timetoevent column * highestime: not the appropriate format -> convert in time format (seconds) above

We keep only useful columns: bib (or ID), gender, age, status and timetoevent

```
data_utmb17<-  data_utmb17[,-c(2,3,4,8,9,10)]
```

Then, we convert the age category (SE, V1, V2, V3, V4) in age range (in years) using the international age ranking for running trail

```
age_range <- tibble('age' =
                      c("V1", "V2","V3", "V4", "SE"),
                    'age_range' = c("40-49", "50-59", "60-69", "70+", "23-39")
                    )
data_utmb17 <- data_utmb17 |> inner_join(age_range,by = "age")

#Move column "age_range" just after column "age"
data_utmb17 <- data_utmb17  %>% relocate("age_range", .after = "age")
```

```
table(data_utmb17$age_range)
```

```
##
## 23-39 40-49 50-59 60-69   70+
##   853  1144   472    58     5
```

The 3 oldest categories contains few individuals compared to the 2 others. We could merge the 3 oldest range together.

```
data_utmb17 ["age_range"] [data_utmb17 ["age_range"] == "60-69"]<- "50+"
data_utmb17 ["age_range"] [data_utmb17 ["age_range"] == "70+"]<- "50+"
data_utmb17 ["age_range"] [data_utmb17 ["age_range"] == "50-59"] <- "50+"
```

```
table(data_utmb17$age_range)
```

```
##
## 23-39 40-49   50+
##   853  1144   535
```

```
table(data_utmb17$age_range, data_utmb17$gender)
```

```
##
##         Female Male
##   23-39     95  758
##   40-49    108 1036
##   50+       39  496
```

3/SURVIVAL ANALYSIS

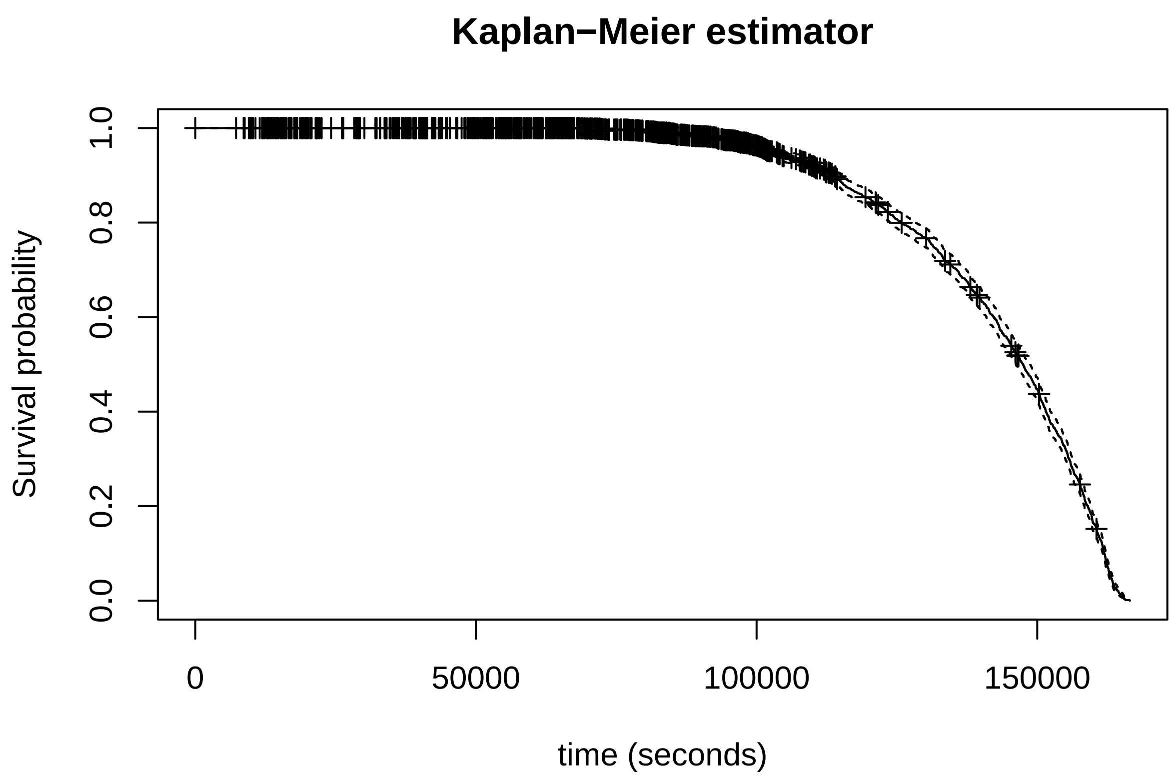
a/Global analysis

Kaplan-Meier

```r
fit.KM <- survfit(Surv(timetoevent, status) ~ 1, data = data_utmb17)
fit.KM
```

```
## Call: survfit(formula = Surv(timetoevent, status) ~ 1, data = data_utmb17)
##
##          n events median 0.95LCL 0.95UCL
## [1,] 2532   1684 147471  146205  148641
```

```r
plot(fit.KM, mark.time = TRUE,
     main = "Kaplan-Meier estimator",
     ylab = "Survival probability",
     xlab = "time (seconds)")
```



Semi-parametric Cox regression

```r
cox.full<- coxph(Surv(timetoevent, status) ~ 1, data = data_utmb17)
summary(cox.full)
```

```
## Call:  coxph(formula = Surv(timetoevent, status) ~ 1, data = data_utmb17)
##
## Null model
##   log likelihood= -10854.15
##   n= 2532
```

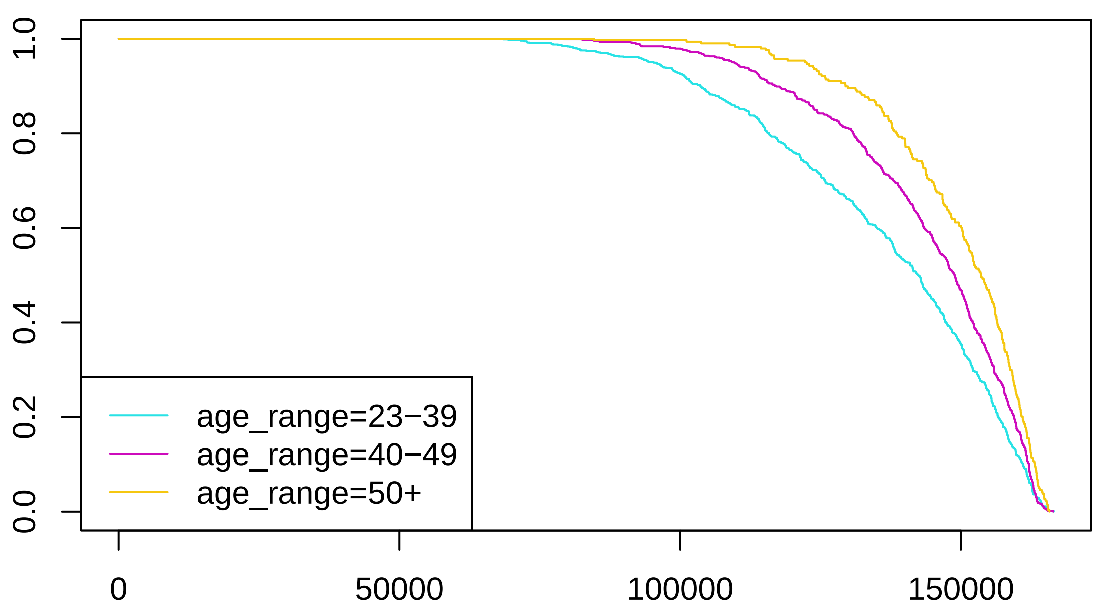
b/ Group by AGE

Kaplan-Meier

```r
fit.KMage <- survfit(Surv(timetoevent, status) ~ age_range, data = data_utmb17)
fit.KMage
```

```
## Call: survfit(formula = Surv(timetoevent, status) ~ age_range, data = data_utmb17)
##
##                     n events median 0.95LCL 0.95UCL
## age_range=23-39   853    645 142474  139378  144144
## age_range=40-49 1144    771 148863  147408  150204
## age_range=50+    535    268 153591  151397  155904
```

```r
plot(fit.KMage, col = 13:16)
legend("bottomleft", lty = 1, col = 13:16, legend = names(fit.KMage$strata))
```



Log rank test

The logrank test is the most widely used method of comparing two or more survival curves

```
diff.KMage <- survdiff(Surv(timetoevent, status) ~ age_range, data = data_utmb17)
diff.KMage
```

```
## Call:
## survdiff(formula = Surv(timetoevent, status) ~ age_range, data = data_utmb17)
##
##                    N Observed Expected (O-E)^2/E (O-E)^2/V
## age_range=23-39  853      645      529    25.374      37.2
## age_range=40-49 1144      771      791     0.526       1.0
## age_range=50+    535      268      363    25.072      32.3
##
##  Chisq= 51.4  on 2 degrees of freedom, p= 7e-12
```

p-value = 7e-12 («0.05), we reject H0 => there exists at least a significant difference between 2 age range reinforcing the visual impression of a trend towards better survival (chance to finish the race) when the age is less advanced.

Semi-parametric Cox regression

```
cox.age<- coxph(Surv(timetoevent, status) ~ age_range, data = data_utmb17)
summary(cox.age)
```

```
## Call:
## coxph(formula = Surv(timetoevent, status) ~ age_range, data = data_utmb17)
##
##   n= 2532, number of events= 1684
##
##                     coef exp(coef) se(coef)      z Pr(>|z|)
## age_range40-49 -0.22543   0.79817  0.05352 -4.212 2.53e-05 ***
## age_range50+   -0.50651   0.60260  0.07293 -6.945 3.78e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                exp(coef) exp(-coef) lower .95 upper .95
## age_range40-49    0.7982      1.253    0.7187    0.8864
## age_range50+      0.6026      1.659    0.5223    0.6952
##
## Concordance= 0.573  (se = 0.007 )
## Likelihood ratio test= 52.24  on 2 df,   p=5e-12
## Wald test            = 50.7  on 2 df,   p=1e-11
## Score (logrank) test = 51.38  on 2 df,   p=7e-12
```

The reference group is the youngest group (23-39). The Cox regression shows that the 2 other age groups are statistically significant compared to the reference (p«0.05). The impact of the age decrease the risk h of finishing the race by 0.8 and 0.6 (respectively for 40-49 and 50+) meaning that the youngest group has, respectively, 1.25 times and 1.66 times more chance to finish the race.

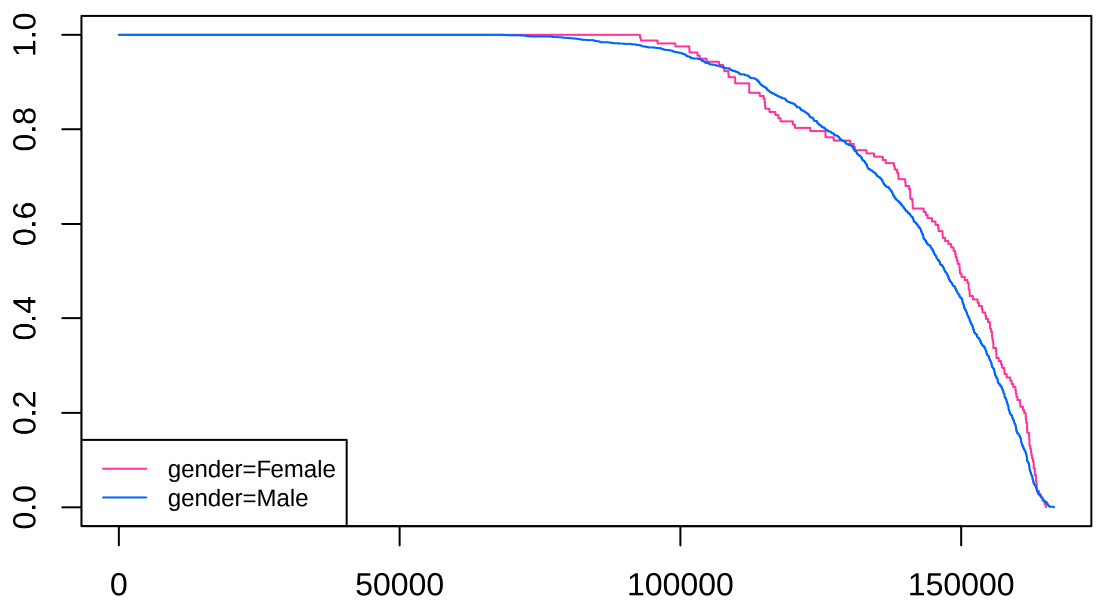
c/ Group by GENDER

Kaplan-Meier

```r
fit.KMgender <- survfit(Surv(timetoevent, status) ~ gender, data = data_utmb17)
fit.KMgender
```

```
## Call: survfit(formula = Surv(timetoevent, status) ~ gender, data = data_utmb17)
##
##                  n events median 0.95LCL 0.95UCL
## gender=Female  242    147 149784  146716  154309
## gender=Male   2290   1537 147135  145792  148341
```

```r
plot(fit.KMgender, col = c("#FF3399", "#0066FF"),pch = 19)
legend("bottomleft", lty = 1, col = c("#FF3399", "#0066FF"), cex= 0.75, legend = names(fit.KMgender$stra
```



Log rank test by gender

```r
diff.KMgender <- survdiff(Surv(timetoevent, status) ~ gender, data = data_utmb17)
diff.KMgender
```

```
## Call:
## survdiff(formula = Surv(timetoevent, status) ~ gender, data = data_utmb17)
##
```

```
##                    N Observed Expected (O-E)^2/E (O-E)^2/V
## gender=Female  242      147      167      2.45      2.73
## gender=Male   2290     1537     1517      0.27      2.73
##
##  Chisq= 2.7  on 1 degrees of freedom, p= 0.1
```

The p-value is large (p=0.1): the difference *is not* statistically significant.

As we can see on the KM curve, both curves are crossing twice. We can suspect an influence of the age. Let's now stratify on the age to see of we can observe a difference between gender

```
diff.KMgender2 <- survdiff(Surv(timetoevent, status) ~ gender + strata(age_range), data = data_utmb17)
diff.KMgender2
```

```
## Call:
## survdiff(formula = Surv(timetoevent, status) ~ gender + strata(age_range),
##     data = data_utmb17)
##
##                    N Observed Expected (O-E)^2/E (O-E)^2/V
## gender=Female  242      147      169     2.929      3.28
## gender=Male   2290     1537     1515     0.327      3.28
##
##  Chisq= 3.3  on 1 degrees of freedom, p= 0.07
```

Semi-parametric Cox regression

```
cox.gender<- coxph(Surv(timetoevent, status) ~ gender + strata(age_range), data = data_utmb17)
summary(cox.gender)
```

```
## Call:
## coxph(formula = Surv(timetoevent, status) ~ gender + strata(age_range),
##     data = data_utmb17)
##
##   n= 2532, number of events= 1684
##
##               coef exp(coef) se(coef)     z Pr(>|z|)
## genderMale 0.15658   1.16951  0.08659 1.808   0.0706 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##            exp(coef) exp(-coef) lower .95 upper .95
## genderMale      1.17     0.8551     0.987     1.386
##
## Concordance= 0.508  (se = 0.004 )
## Likelihood ratio test= 3.41  on 1 df,   p=0.06
## Wald test            = 3.27  on 1 df,   p=0.07
## Score (logrank) test = 3.28  on 1 df,   p=0.07
```

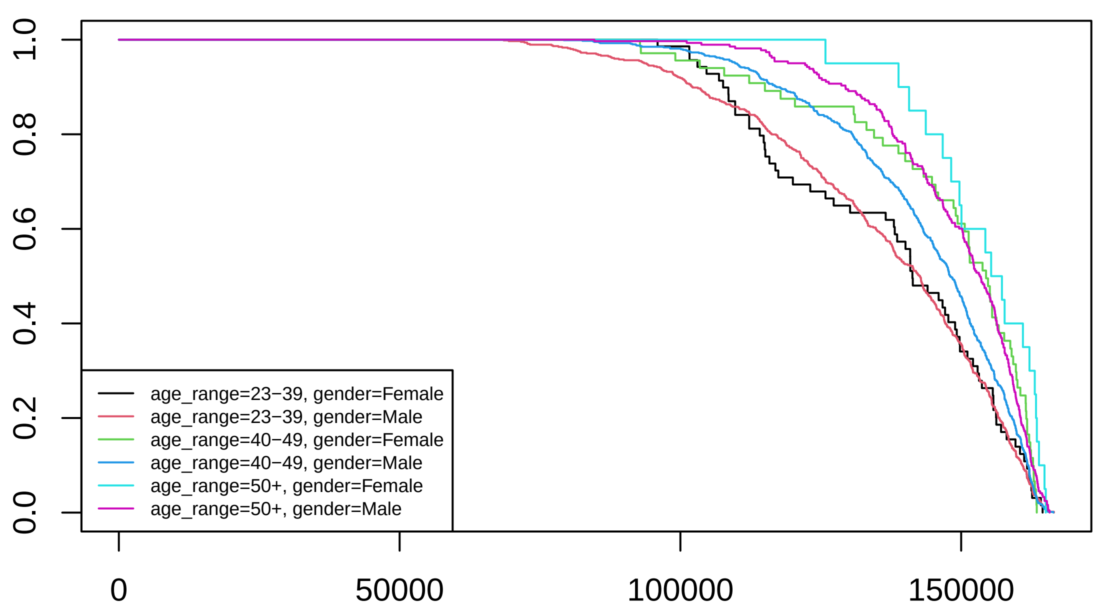
d/ Group by Age AND Gender

Kaplan-Meier

```r
fit.KMage_gender <- survfit(Surv(timetoevent, status) ~ age_range + gender, data = data_utmb17)
fit.KMage_gender
```

```
## Call: survfit(formula = Surv(timetoevent, status) ~ age_range + gender,
##     data = data_utmb17)
##
##                                      n events median 0.95LCL 0.95UCL
## age_range=23-39, gender=Female      95     66 141317  138019  149719
## age_range=23-39, gender=Male       758    579 142665  138688  144246
## age_range=40-49, gender=Female     108     61 154429  150636  158749
## age_range=40-49, gender=Male      1036    710 148005  146380  149735
## age_range=50+, gender=Female        39     20 156299  149696  163332
## age_range=50+, gender=Male         496    248 153310  151395  155904
```

```r
plot(fit.KMage_gender, col = 1:9)
legend("bottomleft",lty = 1, col = 1:9, legend = names(fit.KMage_gender$strata), cex= 0.6, box.lty=1)
```



Log rank test

```
diff.KMage_gender1 <- survdiff(Surv(timetoevent, status) ~ gender + age_range , data = data_utmb17)
diff.KMage_gender1
```

```
## Call:
## survdiff(formula = Surv(timetoevent, status) ~ gender + age_range,
##     data = data_utmb17)
##
##                                     N Observed Expected (O-E)^2/E (O-E)^2/V
## gender=Female, age_range=23-39   95       66     54.2    2.5854     2.675
## gender=Female, age_range=40-49  108       61     76.9    3.2996     3.472
## gender=Female, age_range=50+     39       20     36.1    7.2044     7.417
## gender=Male, age_range=23-39    758      579    475.0   22.7890    31.954
## gender=Male, age_range=40-49   1036      710    714.5    0.0281     0.049
## gender=Male, age_range=50+      496      248    327.3   19.2240    24.011
##
##  Chisq= 55.7  on 5 degrees of freedom, p= 1e-10
```

Semi-parametric Cox regression

without interaction btw age and sex

```
cox.age_gender1<- coxph(Surv(timetoevent, status) ~ gender + age_range, data = data_utmb17)
summary(cox.age_gender1)
```

```
## Call:
## coxph(formula = Surv(timetoevent, status) ~ gender + age_range,
##     data = data_utmb17)
##
##   n= 2532, number of events= 1684
##
##                    coef exp(coef) se(coef)      z Pr(>|z|)
## genderMale      0.14695   1.15830  0.08647  1.699   0.0893 .
## age_range40-49 -0.22643   0.79737  0.05352 -4.231 2.32e-05 ***
## age_range50+   -0.50715   0.60221  0.07292 -6.955 3.53e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                exp(coef) exp(-coef) lower .95 upper .95
## genderMale        1.1583     0.8633    0.9777    1.3722
## age_range40-49    0.7974     1.2541    0.7180    0.8856
## age_range50+      0.6022     1.6606    0.5220    0.6947
##
## Concordance= 0.576  (se = 0.008 )
## Likelihood ratio test= 55.25  on 3 df,   p=6e-12
## Wald test            = 53.61  on 3 df,   p=1e-11
## Score (logrank) test = 54.29  on 3 df,   p=1e-11
```

with interaction btw age and sex age:gender

```r
cox.age_gender2<- coxph(Surv(timetoevent, status) ~ gender + age_range + age_range:gender, data = data_u
summary(cox.age_gender2)
```

```
## Call:
## coxph(formula = Surv(timetoevent, status) ~ gender + age_range +
##     age_range:gender, data = data_utmb17)
##
##   n= 2532, number of events= 1684
##
##                                coef exp(coef)  se(coef)      z Pr(>|z|)
## genderMale                 0.001039  1.001040  0.130026  0.008   0.9936
## age_range40-49            -0.432995  0.648564  0.177780 -2.436   0.0149 *
## age_range50+              -0.798129  0.450170  0.255725 -3.121   0.0018 **
## genderMale:age_range40-49  0.227239  1.255129  0.186348  1.219   0.2227
## genderMale:age_range50+    0.318719  1.375365  0.266579  1.196   0.2319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                           exp(coef) exp(-coef) lower .95 upper .95
## genderMale                   1.0010     0.9990    0.7758    1.2916
## age_range40-49               0.6486     1.5419    0.4577    0.9189
## age_range50+                 0.4502     2.2214    0.2727    0.7431
## genderMale:age_range40-49    1.2551     0.7967    0.8711    1.8085
## genderMale:age_range50+      1.3754     0.7271    0.8157    2.3192
##
## Concordance= 0.576  (se = 0.008 )
## Likelihood ratio test= 57.42  on 5 df,   p=4e-11
## Wald test            = 54.75  on 5 df,   p=1e-10
## Score (logrank) test = 55.66  on 5 df,   p=1e-10
```

d/Comparison of the Cox models

Let's compare the different Cox models and see which is the "best" one using AIC: cox.full, cox.age, cox.gender and cox.age_gender

```r
fits<- list(M0 = cox.full, MA = cox.age, MB = cox.gender, MC1 = cox.age_gender1, MC2=cox.age_gender2)
sapply(fits, AIC)
```

```
##       M0       MA       MB      MC1      MC2
## 21708.30 21660.06 18287.60 21659.05 21660.88
```

We can see that the best model (with lowest AIC) is the model considering only 1 covariate: gender!