Basic Kaplan-Meier curves using survminer

Matthew Castelo

June, 2021

ggplot2 is well suited to creating high-quality Kaplan-Meier plots through the **survminer** package. This tutorial will explore KM plots when the full survival analysis may be performed with a different statistical software.

We will need to install the following packages.

```
install.packages(c("tidyverse", "survminer"))
```

As usual, we will load our packages and set the working directory.

```
library(tidyverse)
library(survminer)
library(ggsci)
library(survival)

#The working directory saves the following path when loading files, saving plots, etc
setwd("C:/Users/matth/Documents/R/R code and education/Tutorials/")
```

We will be working with a simulated dataset representing 80 generic cancer patients. The dataset can be loaded directly from GitHub so all the subsequent code will run on your computer.

```
url <-
  "https://raw.githubusercontent.com/mcas-surg/Tutorials/main/Datasets/generic_cancer.csv"

cancer <- read_csv(url)
```

Let's inspect the dataset.

```
glimpse(cancer)
```

```
## Rows: 80
## Columns: 13
## $ male                  <chr> "Male", "Male", "Female", "Male", "Male", "Male"~
## $ pt_id                 <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1~
## $ age                   <dbl> 62, 57, 66, 54, 49, 63, 54, 72, 57, 65, 63, 54, ~
## $ high_grade            <chr> "No", "No", "No", "No", "No", "No", "No", "No", ~
## $ adverse_tumour_marker <chr> "No", "No", "No", "No", "No", "Yes", "No", "No",~
## $ tumour_size           <dbl> 0.41, 2.84, 3.35, 4.63, 1.29, 1.99, 1.86, 1.01, ~
## $ diabetes              <chr> "No", "No", "No", "No", "No", "No", "No", "No", ~
## $ heart_disease         <chr> "No", "No", "No", "Yes", "No", "No", "No", "No",~
```

```
## $ hospital              <chr> "Community hospital", "Community hospital", "Com~
## $ extended_resection    <chr> "Yes", "No", "No", "No", "No", "Yes", "No", "No"~
## $ postop_complication   <chr> "Yes", "No", "Yes", "Yes", "No", "Yes", "Yes", "~
## $ vital_status          <chr> "Alive", "Died", "Died", "Alive", "Alive", "Aliv~
## $ follow_up             <dbl> 58, 52, 33, 58, 56, 41, 64, 32, 30, 40, 29, 29, ~
```

Assuming our full survival analysis has been performed elsewhere, the main outcome variables of interest will be "vital_status" and "follow_up". For this example we will assume the main exposure variable of interest is a "high_grade" tumour.

First, we can explore the data. Using **dplyr** we can `select()` the variables of interest and produce summary statistics using `summary()`. Remember to pipe (%>%) everything together for efficiency.

```
cancer %>%
  select(high_grade, vital_status, follow_up) %>%
  summary()
```

```
##   high_grade        vital_status         follow_up
##  Length:80          Length:80          Min.   :14.00
##  Class :character   Class :character   1st Qu.:33.00
##  Mode  :character   Mode  :character   Median :39.00
##                                        Mean   :40.38
##                                        3rd Qu.:50.25
##                                        Max.   :70.00
```

This didn't give us much useful information, probably because both "vital_status" and "high_grade" are character variables.

```
#A dollar sign after the dataset name allows you to select a single variable
is.character(cancer$high_grade)
```

```
## [1] TRUE
```

We can convert both into factors using `mutate()`. Let's save the new dataset as *cancer_1*.

```
cancer_1 <- cancer %>%
  mutate(high_grade = factor(high_grade),
         vital_status = factor(vital_status))
```

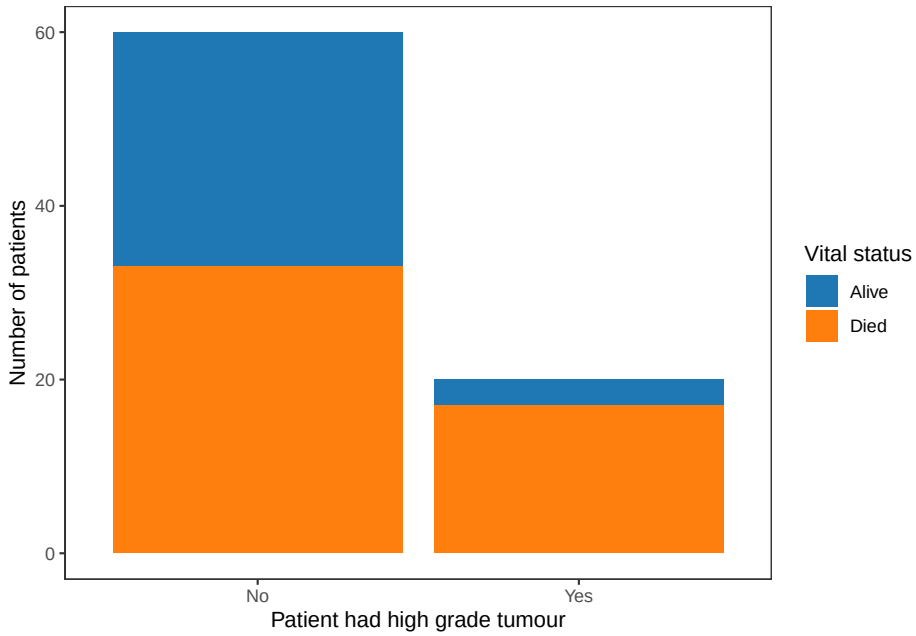
And we will try again.

```
cancer_1 %>%
  select(high_grade, vital_status, follow_up) %>%
  summary()
```

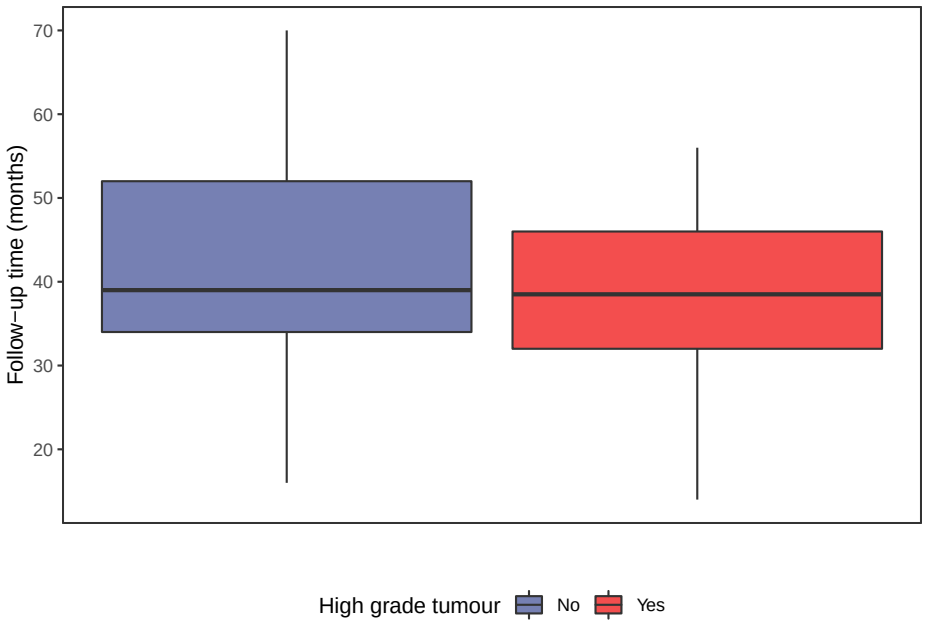
```
##  high_grade vital_status   follow_up
##  No :60     Alive:30     Min.   :14.00
##  Yes:20     Died :50     1st Qu.:33.00
##                          Median :39.00
##                          Mean   :40.38
##                          3rd Qu.:50.25
##                          Max.   :70.00
```

We can visualize these results using bar charts and boxplots.

```
cancer_1 %>%
  ggplot(aes(y = high_grade, fill = vital_status))+
  geom_bar()+
  labs(x = "Number of patients",
       y = "Patient had high grade tumour",
       fill = "Vital status")+
  coord_flip()+
  theme_bw()+
  scale_fill_d3()+
  theme(panel.grid = element_blank())
```




```
cancer_1 %>%
  ggplot(aes(y = follow_up, fill = high_grade))+
  geom_boxplot(alpha = 0.7)+
  labs(x = " ",
       y = "Follow-up time (months)",
       fill = "High grade tumour")+
  theme_bw()+
  scale_fill_aaas()+
  theme(panel.grid = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        legend.position = "bottom")
```



It appears patients with high grade tumours have worse survival. Let's create our KM curve. First we need to create a survival curve. The equation has two sides, separated by a ~. On the left side we provide our variable indicating follow-up time, and the status variable (death or censored). On the right we provide the stratification variable.

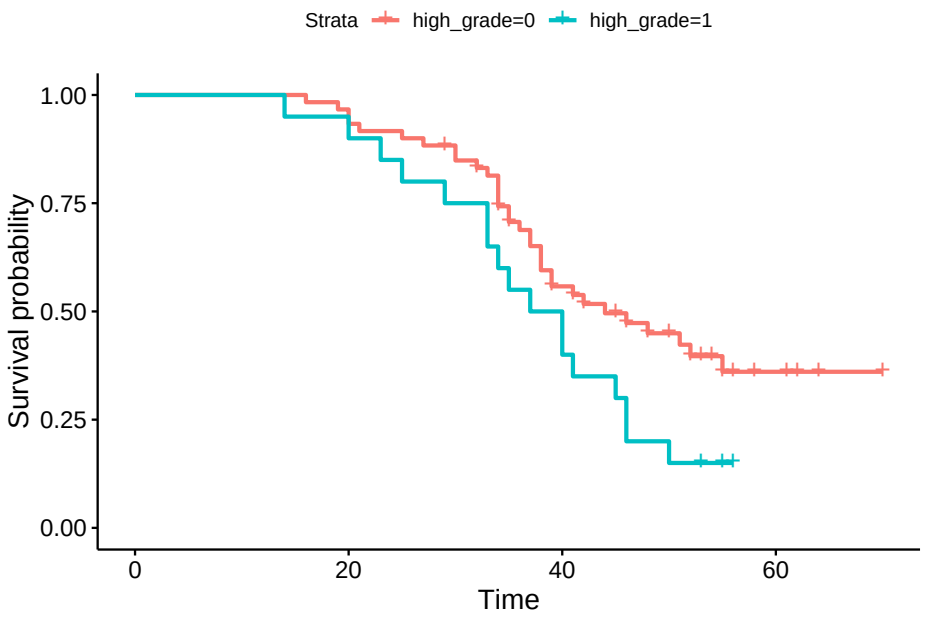
First, **survminer** does not like factors, we need to change our variables into binary 1's and 0's. We will call this new dataset *cancer_surv*.

```
cancer_surv <- cancer %>%
  mutate(high_grade = ifelse(high_grade == "Yes",
                             1, 0),
         vital_status = ifelse(vital_status == "Died",
                               1, 0))

survival_curve <- survfit(Surv(follow_up, vital_status) ~ high_grade,
                          data = cancer_surv)
```

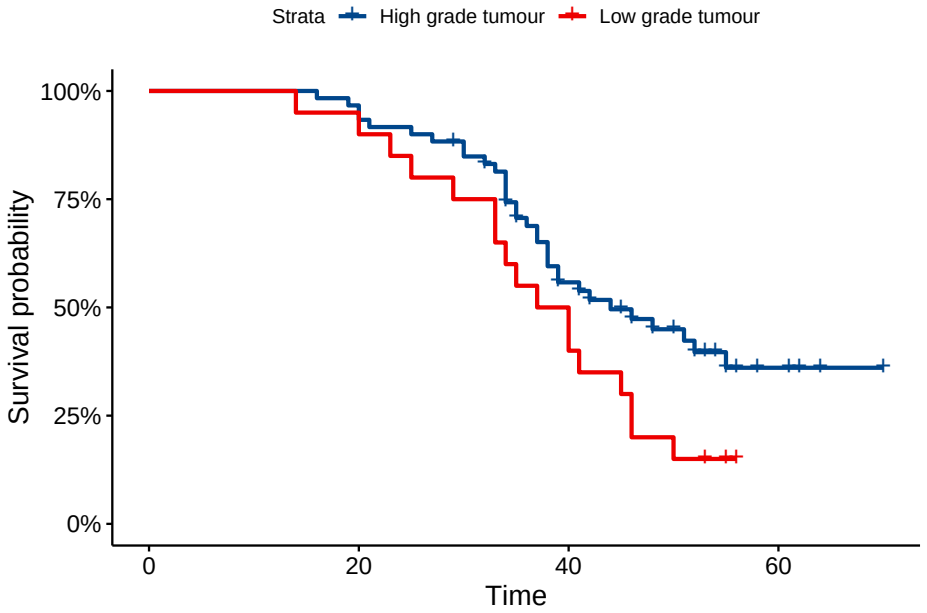
Now let's create a basic KM plot.

```
ggsurvplot(survival_curve,
           data = cancer_surv)
```



There are a few improvements we can make immediately. Let's rename the Strata groups, and change the y-axis to percent rather than probability. We will also use the Lancet colour scheme.

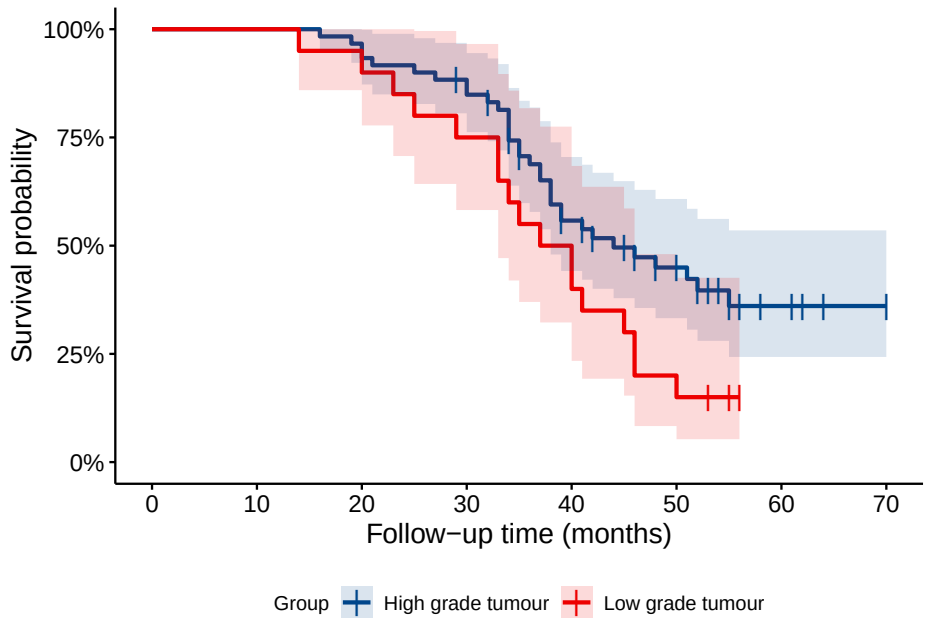
```
ggsurvplot(survival_curve,
           data = cancer_surv,
           legend.labs = c("High grade tumour",
                           "Low grade tumour"),
           surv.scale = "percent",
           palette = "lancet")
```



Now, let's improve the x-axis breaks and rename the x-axis. We will also change the censor markings to

vertical lines and add confidence bands. Finally, the legend will be positioned at the bottom of the plot and renamed.

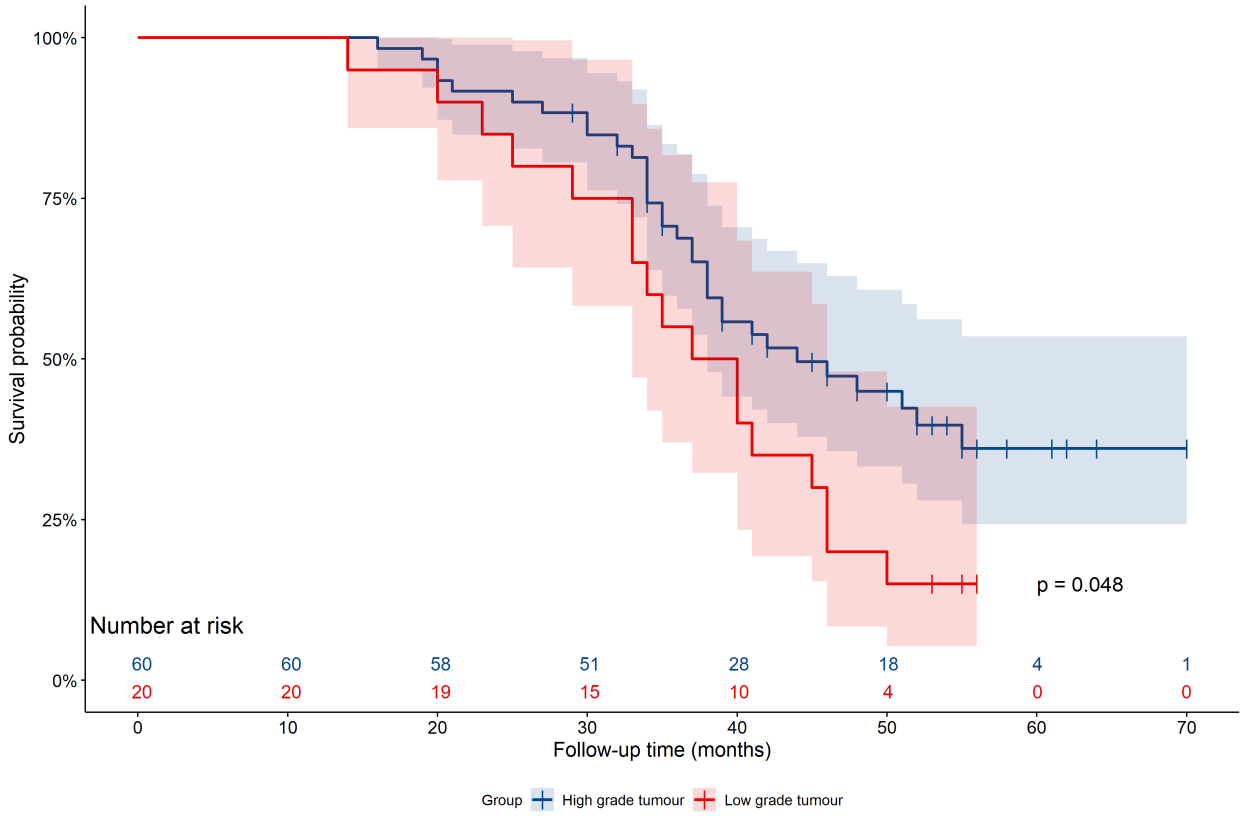
```
ggsurvplot(survival_curve,
           data = cancer_surv,
           legend.labs = c("High grade tumour",
                           "Low grade tumour"),
           surv.scale = "percent",
           palette = "lancet",
           break.x.by = 10,
           xlab = "Follow-up time (months)",
           censor.shape = "|",
           conf.int = T,
           conf.int.alpha = 0.15,
           legend = "bottom",
           legend.title = "Group")
```



We can add a p-value to the plot and customize the position. It is also advisable to add an at-risk table to the KM curve. Here I have chosen to place the table inside the figure, but it can also be placed outside with the argument `risk.table.pos = "out"`.

```
final_km <- ggsurvplot(survival_curve,
           data = cancer_surv,
           legend.labs = c("High grade tumour",
                           "Low grade tumour"),
           surv.scale = "percent",
           palette = "lancet",
           break.x.by = 10,
           xlab = "Follow-up time (months)",
           censor.shape = "|",
           conf.int = T,
           conf.int.alpha = 0.15,
```

```
          legend = "bottom",
          legend.title = "Group",
          pval = TRUE,
          pval.coord = c(60, .15),
          risk.table = TRUE,
          risk.table.pos = "in")
```



Remember to save a publication-quality version of your figure. Note there is a requirement to wrap the name of the plot in `print()` for it to save correctly.

```
#If you have not set the working directory, include the full file path
ggsave("my_plot.png", plot = print(final_km),
       dpi = 300, dev = "png",
       height = 20, width = 30, units = "cm")
```

This was a brief overview of creating KM plots using the **survminer** package in R. Although these plots can be created in other software, if you are using **ggplot2** for other figures, this method may help your manuscript maintain a consistent style, colour palette, and resolution across all figures.

One limitation is the need to start from a survival curve using the `survfit()` function. If you have undertaken more complicated analyses in other software and have the ability to produce the raw coordinates, survival curves can still be created in **ggplot2**, including at-risk tables using more complex code. This will be covered in later tutorials.

https://github.com/mcas-surg/Tutorials