Statistical Computing ~

Survival Analysis - ML | January , 2022

Emmanuelle Rodrigues Nunes

CIDRA-AIPD
African Institute for Professional Development

Survival Analysis

Survival Analysis - Recapping

CIDRA-AIPD
African Institute for Professional Development

Survival analysis utilises Time-To-Event (TTE) data where the event can be any *binary* outcome, not necessarily death, without assuming the rates are constant. It lets you model:

- Time until an event occurs or has not yet occurred due to censoring
- Compare the time-to-event between different groups
- Assess how time-to-event correlates with quantitative or categorical variables

For example:

- Well defined time
  - Time from surgery to death
  - Time from start of treatment to cancer progression
  - Time from HIV infection to development of AIDS
- Unsure time
  - Time to heart attack
  - Time to onset of substance abuse
  - Time to initiation of sexual activity
  - Time to machine malfunction

> **Important feature:** Survival analysis can handle missing data due to censoring

Definitions

The **hazard** is the instantaneous event (death) rate at a particular time point t, conditional on having survived to t.

- Survival analysis **doesn't** assume the hazard is constant over time.
- The cumulative hazard is the total hazard experienced up to time t.

The **survival function**, is the probability an individual will survive (or, the probability that the event of interest does not occur) beyond time t.

- It's the probability that the event hasn't occurred yet.

$$\mathbb{S}(t) = \mathbb{P}(T > t),$$

where T is the time of death

Definitions - Kaplan-Meier

CIDRA-AIPD
African Institute
for Professional
Development

The **Kaplan-Meier** curve illustrates the survival function.

- It estimates $\mathbb{S}(t)$
- It's a step function illustrating the cumulative survival probability over time.
- It's the product of probabilities that the subject did not experience the event in any interval up to time t

$$\hat{\mathbb{S}}(t) = \prod_{t_i < t} \left(1 - \frac{d_i}{n_i}\right),$$

where d_i is the number of subjects that died (or experienced the event of interest) at time t_i, and n_i is the number of subjects at risk at time t_i.

At time 0, the survival probability is 1, i.e., $S(t_0) = 1$

Definitions - Censoring

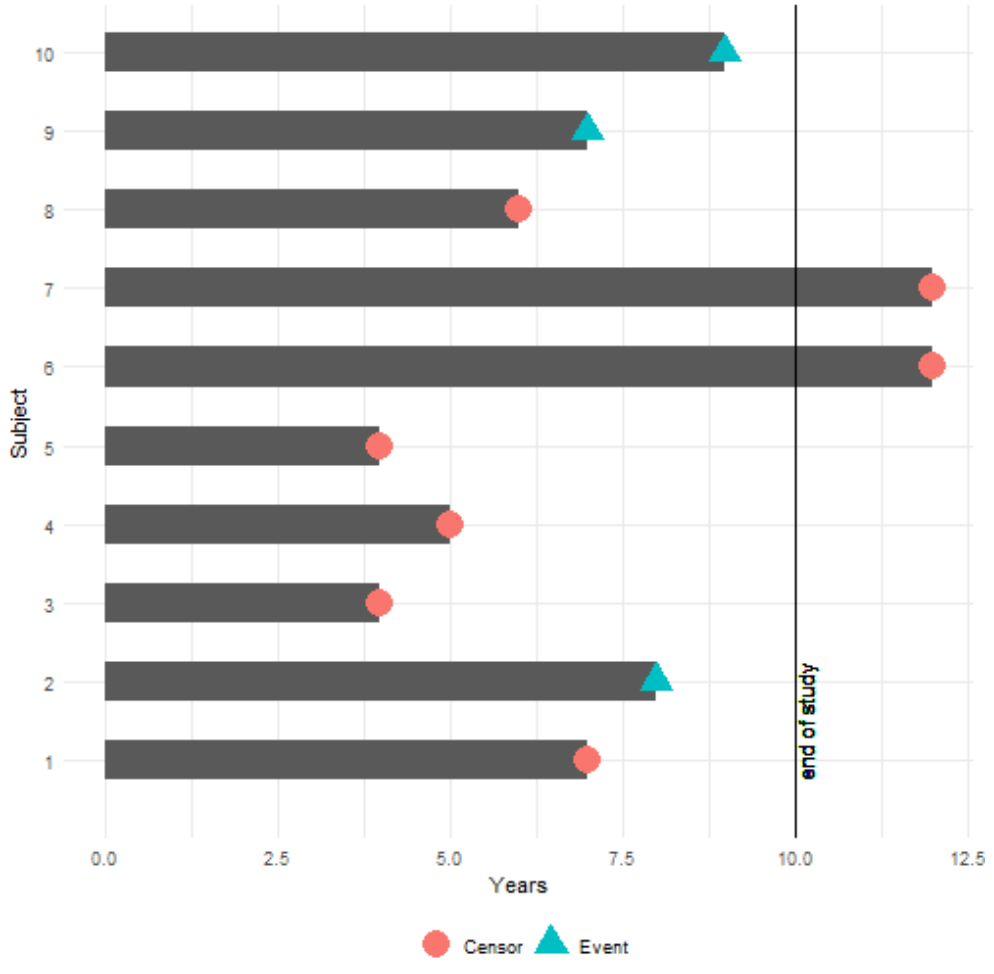
Censoring is a type of missing data problem unique to survival analysis.

- It occurs when you track the sample/subject through the end of the study and the event never occurs, i.e., no event by end of fixed study period
- It could also happen due to the sample/subject dropping out of the study for reasons other than death, for example, stopping treatment due to harmful side effects
- It could also happen due to loss to followup,

If the sample is censored, you only know that the individual survived up to the loss to followup, but you do not know anything about survival after that.

Specifically these are examples of **right censoring**.

Left censoring and interval censoring are also possible.



Survival Analysis in R

CIDRA-AIPD African Institute for Professional Development

Packages we will use for this example:

- **survival**
  - The core functions for this type of analysis are available in this library
  - It is a part of the standard R packages, so you don't need to `install.packages()` before using it
- **dplyr**
  - Data manipulation
- **survminer**
  - Better visualization for Kaplan-Meier plots

For the example we will use the dataset `lung` that is available within the survival package.

```
library(survival)
library(dplyr)
library(survminer)

head(lung, 3)
```

```
##   inst time status age sex ph.ecog ph.karno pat.karno meal.cal wt.loss
## 1    3  306      2  74   1       1       90       100     1175      NA
## 2    3  455      2  68   1       0       90        90     1225      15
## 3    3 1010      1  56   1       0       90        90       NA      15
```

Example - Lung

We can see the dimensions of the dataset by using the function `dim()` from base R. The first value represent the rows, in this example 228 and the second represents the columns, 10 in the example.

```
dim(lung)
```

```
## [1] 228  10
```

The 10 columns represent:

- **inst:** Institution code
- **time:** Survival time in days
- **status:** censoring status 1 = censored, 2 = dead
- **age:** Age in years
- **sex:** Male = 1, Female = 2
- **ph.ecog:** ECOG performance (0 = good, 5 = dead)
- **ph.karno:** Karnofsky performance rated by physician
- **pat.karno:** Karnofsky performance rated by patient
- **meal.cal:** Calories consumed at meals
- **wt.loss:** Weight loss in last six months

```
glimpse(lung)
```

```
## Rows: 228
## Columns: 10
## $ inst      <dbl> 3, 3, 3, 5, 1, 12, 7, 11, 1, 7, 6, 16,
## $ time      <dbl> 306, 455, 1010, 210, 883, 1022, 310, 36
## $ status    <dbl> 2, 2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2,
## $ age       <dbl> 74, 68, 56, 57, 60, 74, 68, 71, 53, 61,
## $ sex       <dbl> 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 2, 2,
## $ ph.ecog   <dbl> 1, 0, 0, 1, 0, 1, 2, 2, 1, 2, 1, 2, 1,
## $ ph.karno  <dbl> 90, 90, 90, 90, 100, 50, 70, 60, 70, 70
## $ pat.karno <dbl> 100, 90, 90, 60, 90, 80, 60, 80, 80, 70
## $ meal.cal  <dbl> 1175, 1225, NA, 1150, NA, 513, 384, 538
## $ wt.loss   <dbl> NA, 15, 15, 11, 0, 0, 10, 1, 16, 34, 27
```

Lung dataset

CIDRA-AIPD African Institute for Professional Development

As all the columns of the dataset are numerical, we need to transform the categorical ones into categorical.

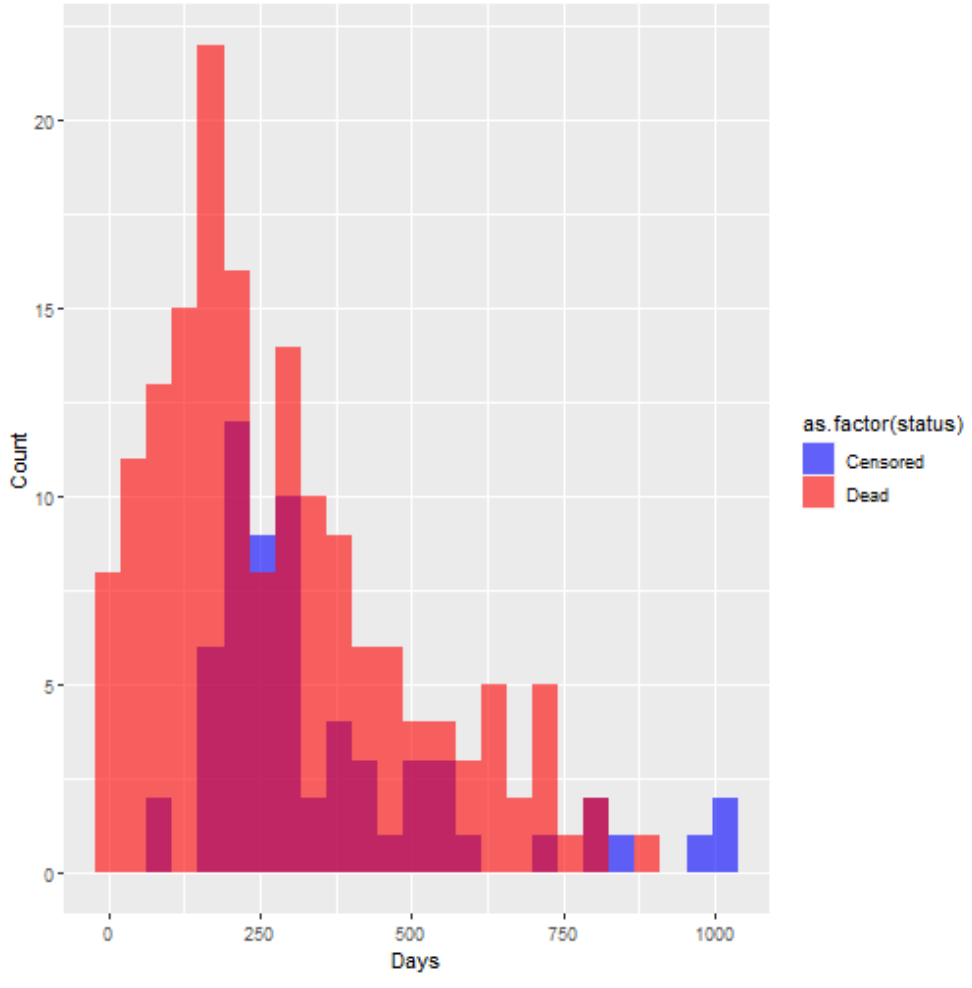
- Use function `mutate_at` from `dplyr` library
  - This function mutates the desired variables to a function
  - The function is `as.factor` to convert the variable to a factor
- We will also convert the `sex` variable to `Male` and `Female` for visualization purposes

```
lung ← lung %>%
  mutate_at(vars("sex", "ph.ecog"), as.factor) %>%
  mutate(sex = forcats::fct_recode(sex, "Male" = "1", "Female" = "2"))
```

Distribution of follow-up time

- Censored subjects still provide information so must be appropriately included in the analysis
- Distribution of follow-up times is skewed, and may differ between censored patients and those with events
- Follow-up times are always positive

```
ggplot(lung, aes(x = time,
                 fill = as.factor(status))) +
  geom_histogram(bins = 25,
                 alpha = 0.6,
                 position = "identity") +
  scale_fill_manual(values = c("blue", "red"),
                    labels = c("Censored",
                               "Dead")) +
  labs(x = "Days",
       y = "Count")
```



Kaplan-Meier Analysis

CIDRA-AIPD African Institute for Professional Development

The first thing to do is to use `Surv()` from `survival` to build the standard survival object.

The most important parameters are:

- `time`: Follow-up time for right censored data;
  - For interval data it represents the start time of the interval.
  - The time until event is unkown and could be anywhere within the interval, so we use start time.
- `event`: Normally 0 = alive, 1 = dead
- `time2`: Ending time of the interval for interval censored
- `type`: Type of censoring
  - Options: "right", "left", "counting", "interval", "interval2" or "mstate"
  - Default: "right"

> Note that a “+” after the time in the print out of km indicates censoring. It is representended by "+" because the true survival time is greater than the value as the event has not yet occured.

```
km ← Surv(lung$time, lung$status)
head(km, 10)
```

```
##  [1]  306   455  1010+  210   883  1022+  310   361   218   166
```

Kaplan-Meier Analysis

CIDRA-AIPD African Institute for Professional Development

Now, let's fit a survival curve with the `survfit()` function. We will create a survival curve that doesn't consider any explanatory variables, or groupings.

- We'll need to specify an intercept (represented by `~1` in the formula that survfit expects).
- We can fit the curve in one step by nesting the `Surv()` function within `survfit()`
- Similarly to how we specify data for linear models with `lm()`, we'll use the `data =` argument to specify which data we're using.

```
km_fit ← survfit(Surv(time, status) ~ 1, data = lung)
summary(km_fit, times = c(1, 30, 60, 90))
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     1    228       0    1.000  0.0000        1.000        1.000
##    30    219      10    0.956  0.0136        0.930        0.983
##    60    213       7    0.925  0.0174        0.892        0.960
##    90    201      10    0.882  0.0214        0.841        0.925
```

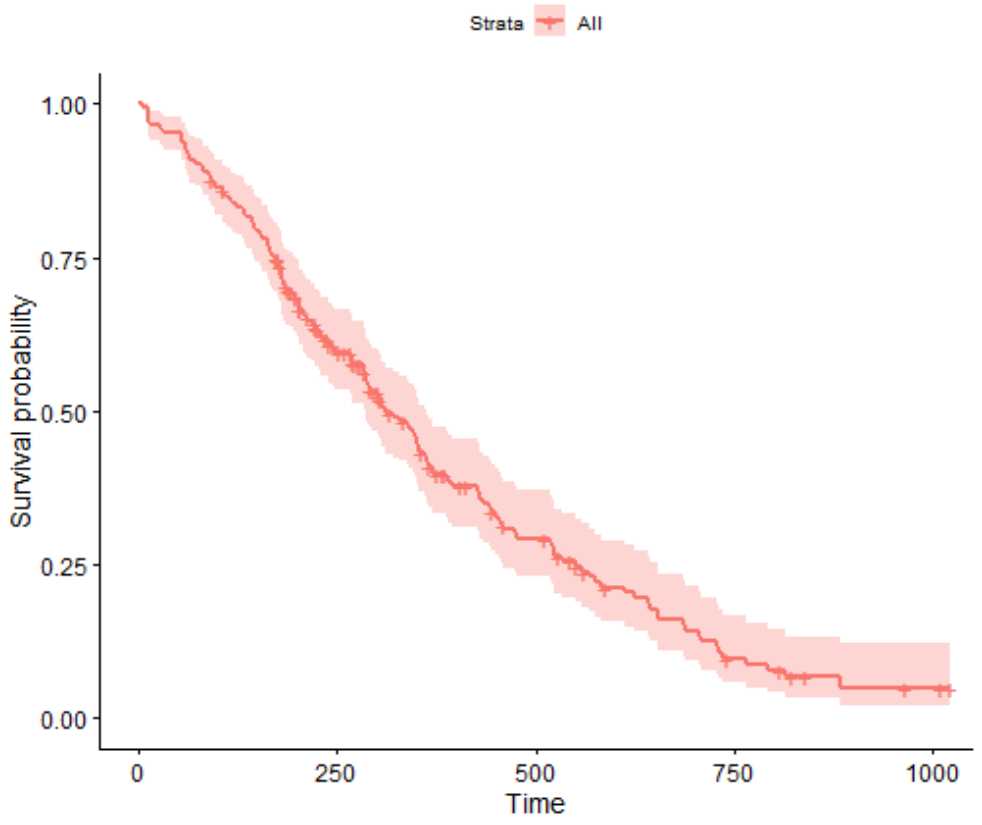
Kaplan-Meier Plot

- We can plot the `survfit` object using the function `plot` from base

```
plot(km_fit)
```

- The `ggsurvplot` function from the `survminer` package is built on ggplot2
- The default plot shows the step function (solid line) with associated confidence bands (shaded area).
- The tick marks representing censored patients are shown by default
  - Can be suppressed using the option `censor = FALSE`

```
ggsurvplot(km_fit)
```



Kaplan-Meier plot

We can also plot the KM curve upside down; this shows the probability of the event.

- It shows death (or the event of interest) instead of survival
- We can do this by adding the option `fun = event` in both plot and ggsurvplot

```
plot(km_fit, fun = 'event')
```

```
ggsurvplot(km_fit, fun = 'event')
```

Estimating x-year survival

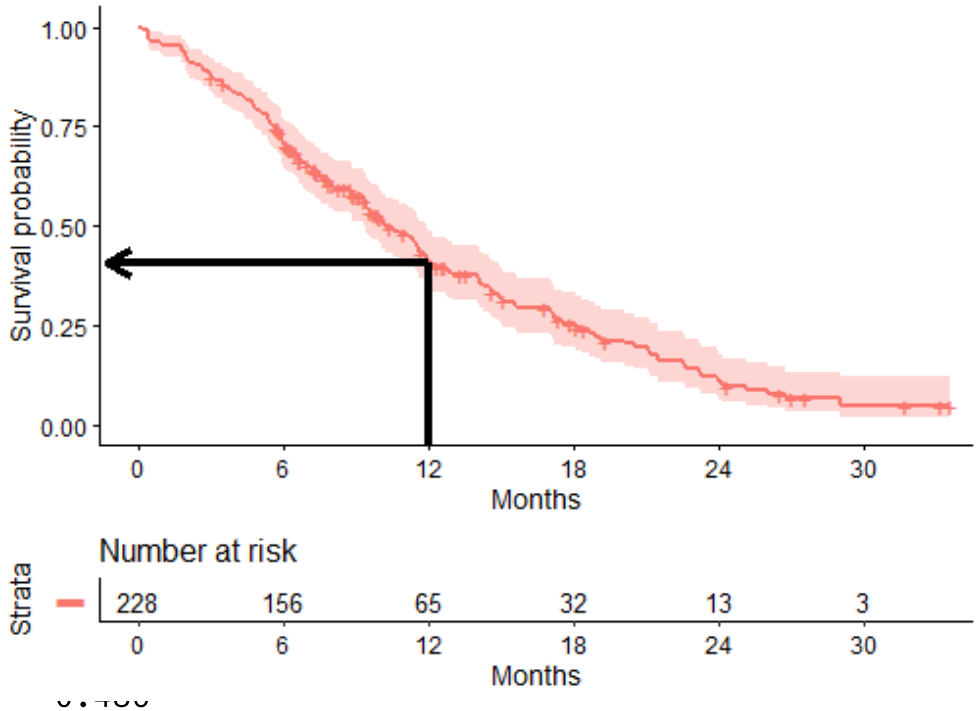
Often of interest is the probability of surviving beyond a certain number x of years.

For example, to estimate the probability of surviving to 1 year:

- Use `summary` with the `times` argument
  - *Note* the `time` variable in the `lung` data is actually in days

```
summary(km_fit, times = 365)
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data =
##
##  time n.risk n.event survival std.err lower 95% CI uppe
##   365     65     121    0.409  0.0358        0.345
```



The 1-year probability of survival in this study is 40.92%.

Estimating median survival time

Another quantity often of interest is the **median survival time**.

- We can obtain this directly from the `survfit` object

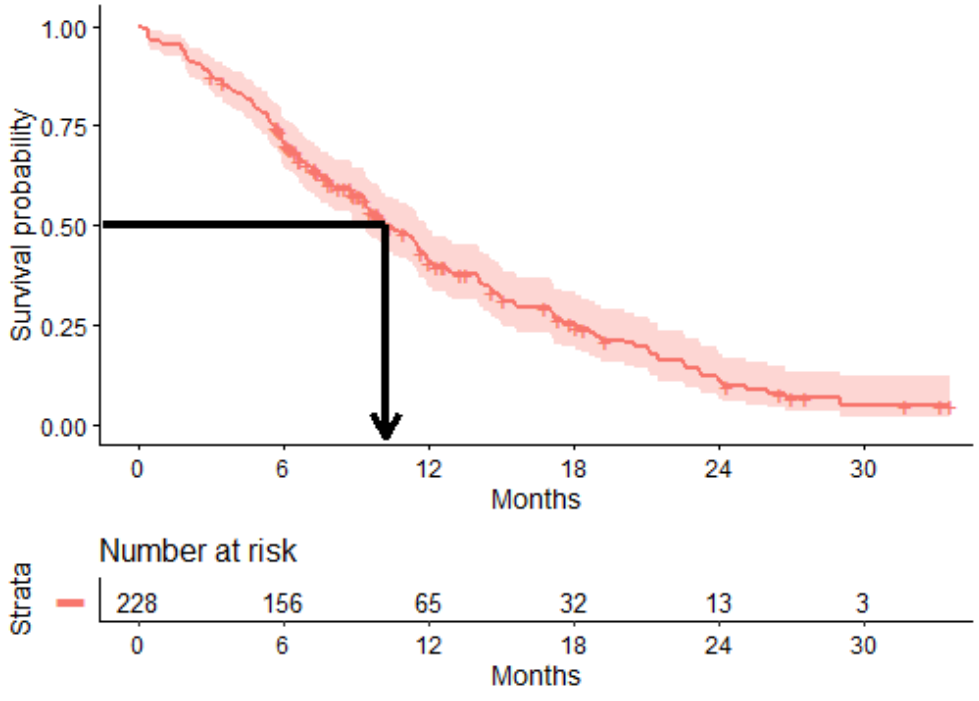
```
km_fit
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data =
##
##         n events median 0.95LCL 0.95UCL
## [1,] 228    165    310     285     363
```

We see the median survival time is 310 days

- The 95% confidence interval is also displayed.

Median survival is the time corresponding to a survival probability of 0.5:



Comparing survival times between groups

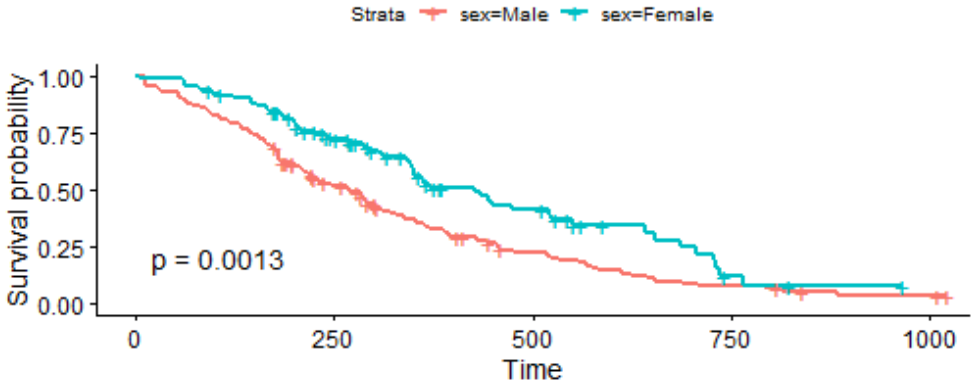
- The **log-rank test** equally weights observations over the entire follow-up time
  - Most common way to compare survival times between groups

We get the log-rank p-value using the `survdiff` function. For example, we can test whether there was a difference in survival time according to sex.

```
survdiff(Surv(time, status) ~ sex, data = lung)
```

```
## Call:
## survdiff(formula = Surv(time, status) ~ sex, data = lun
##
##              N Observed Expected (O-E)^2/E (O-E)^2/V
## sex=Male   138      112     91.6      4.55      10.3
## sex=Female  90       53     73.4      5.68      10.3
##
##  Chisq= 10.3  on 1 degrees of freedom, p= 0.001
```

```
ggsurvplot(survfit(Surv(time, status) ~ sex,
                   data = lung),
           pval = TRUE)
```



Cox-Regression

Kaplan-Meier curves are good for visualizing differences in survival between two categorical groups.

- Does not generalize well for assessing the effect of **quantitative** variables.

We may want to quantify an effect size for a single variable, or include more than one variable into a regression model to account for the effects of multiple variables.

The Cox regression model can be used to fit univariable and multivariable regression models that have survival outcomes.

$$h(t|X_i) = h_0(t)e^{(\beta_1 X_{i1} + \ldots + \beta_p X_{ip})},$$

where $h(t)$ is the hazard and $h_0(t)$ is the baseline hazard

The Cox model relies on the assumption of proportional hazards (PH), i.e., the hazard ratio (HR) is assumed to be constant over time for each covariate, even though the hazards vary. For example, the hazards for males and females may be changing over time, but their ratio stays the same.

Cox Regression - R

Continuing with the `lung` dataset. We can fit regression models for survival data using the `coxph` function from `survival` package.

- It requires a `Surv` object.

```
cox_fit ← coxph(Surv(time, status) ~ sex, data = lung)
cox_fit %>% broom::tidy()
```

```
## # A tibble: 1 x 5
##   term      estimate std.error statistic p.value
##   <chr>        <dbl>     <dbl>     <dbl>   <dbl>
## 1 sexFemale   -0.531     0.167     -3.18 0.00149
```

The quantity of interest from a Cox regression model is the **hazard ratio (HR)**

- Represents the ratio of hazards between two groups at any particular point in time
- Interpreted as the instantaneous rate of occurrence of the event of interest in those who are still at risk for the event
- A HR < 1 indicates reduced hazard of death whereas a HR > 1 indicates an increased hazard of death.

From R output, HR is represented by $\exp(\text{estimate}) = e^{-0.5310} = 0.588$

Check Cox assumption

To test for the proportional-hazards (PH) assumption we are going to use the function `cox.zph` from `survival` package.

```
cox.zph(cox_fit)
```

```
##         chisq df     p
## sex      2.86  1 0.091
## GLOBAL   2.86  1 0.091
```

From the output above, the test is not statistically significant using an α of 5% for the covariate, and the global test is also not statistically significant.

- Therefore, we **can** assume the proportional hazards.

Mutivariate Anaylis - Lung dataset

Most of the time we will have more than one explanatory variable of interest.

For this example, we will use the patient age, sex, ECOG status and the amount of weight loss as covariates.

- Sex is a factor with two categories
- ECOG status is a factor with 6 categories

With three or more categories, we **should** choose one level to be the **reference level** (with defined hazard ratio 1.0).

- R will automatically pick one to be a reference
- For n levels, there will be n-1 variables representing the hazard ratio compared to subjects in the reference level.

We will use the function `analyse_multivariate` from `survivalAnalysis`.

Multivariate Analysis Example - Lung dataset

CIDRA-AIPD African Institute for Professional Development

```
library(tidytidbits)
library(survivalAnalysis)

multivar_fit ← lung %>%
  # variables that represent time and status within vars() function
  analyse_multivariate(vars(time, status),
                       # select covariates using the vars() function
                       covariates = vars(age, sex, ph.ecog, wt.loss))

multivar_fit$summaryAsFrame
```

```
##    factor.id factor.name factor.value        HR  Lower_CI   Upper_CI    Inv_HR
## 1 sex:Female         sex       Female 0.5584717 0.3955858  0.7884273 1.7906010
## 2    wt.loss     wt.loss <continuous> 0.9909481 0.9780141  1.0040532 1.0091345
## 3        age         age <continuous> 1.0131906 0.9941613  1.0325842 0.9869811
## 4  ph.ecog:1     ph.ecog            1 1.5835452 1.0528108  2.3818292 0.6314944
## 5  ph.ecog:2     ph.ecog            2 2.7309998 1.6638158  4.4826836 0.3661663
## 6  ph.ecog:3     ph.ecog            3 8.6863048 1.1353408 66.4574800 0.1151238
##   Inv_Lower_CI Inv_Upper_CI            p
## 1   1.26834783    2.5278964 9.293721e-04
## 2   0.99596313    1.0224801 1.749341e-01
## 3   0.96844403    1.0058730 1.755352e-01
## 4   0.41984539    0.9498383 2.730962e-02
## 5   0.22308066    0.6010281 7.080711e-05
```

Multivariate Analysis Example - Lung dataset

The summary on the previous slide shows:

- The HR for all explanatory variables with their CI
- The Inverted HR with their CI
- The pvalue of the test, using Wald test, that shows that wt.loss and age are not significant

We can output that table in a more visual way by using the function `forest_plot` from `survivalAnalysis`.

```
forest_plot(multivar_fit,
            orderer = ~order(HR),
            HR_x_breaks = c(0.25, 0.5, 1, 1.5, 2))
```

Multivariate Analysis Example - Lung dataset

CIDRA-AIPD African Institute for Professional Development

We can do a likelihood test by using the function `drop1`. Also, for the likelihood test, we need to remove the `NA` values by using the function `na.omit` and `update` function to update the `coxph` result.

```
(l_test ← update(multivar_fit$coxph, data =lung %>% na.omit()) %>%
  drop1(test = "Chi"))
```

```
## Single term deletions
##
## Model:
## Surv(time, status) ~ age + sex + ph.ecog + wt.loss
##         Df    AIC     LRT Pr(>Chi)
## <none>     1004.8
## age      1 1003.1  0.2543 0.614078
## sex      1 1010.5  7.7040 0.005510 **
## ph.ecog  3 1013.3 14.5062 0.002291 **
## wt.loss  1 1005.5  2.6336 0.104626
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The likelihood test gave the same conclusion as the Wald test.

Multivariate Analysis Example - Lung dataset

Another important metric is the Concordance (C) statistic:

- Used to assess the ability of a risk factor to predict outcome, i.e., gives the probability a randomly selected patient who experienced an event had a higher risk score than a patient who had not experienced the event.
- It helps us understand the **accuracy** of the model.
- It ranges from 0.5 (poor model) to 1 (perfect fit)

```
multivar_fit$summary$concordance
```

```
##           C      se(C)
## 0.64705882 0.02633085
```

References

Dirk F. Moore, Applied Survival Analysis Using R (Springer, 2016)

David G. Kleinbaum, Mitchel Klein, Survival Analysis: A Self-Learning Text, Third Edition (Springer, 2011)