LIFE707 Topic 8: Survival analysis

Liam Dougherty

2024-09-10

Survival analysis helps us understand how the chance of something happening (like a patient passing away) changes over time.

It does this by creating a survival function, which shows the likelihood that the event hasn't happened yet (for example, the patient is still alive) at any given time.

Time-to-event data

In this exercise we will look at a special type of data called **time-to-event** data. Here, each data point represents the time from the start of the study period until some event of interest was observed, usually from a longitudinal study where samples or subjects are observed regularly over time. This type of data is common in clinical studies, where patients are assessed regularly (e.g. once a week) until some event is observed (e.g. death, onset of disease, relapse).

There are two common features that make this type of data unusual:

1. The rate at which the event occurs if often not constant. For example: the risk of death after heart surgery is highest immediately post-op, decreases as the patient recovers, then rises slowly again as the patient ages. So important features of the response may not be captured in a single metric like a mean or median.
2. The event might not be observed in all samples or subjects during the observation period. This is very common in clinical studies, where the study period might end without all patients dying. This could also happen due to the sample/subject dropping out of the study for reasons other than death, or some other loss to follow-up. These data are said to be **censored**, because the time they are lost is likely an underestimate of their true survival.

Instead, we can analyse this type of data using **survival analysis**. Survival analysis specifically estimates how the risk (or **hazard**) of some event happening changes over time. It does this by producing a **survival function**, which estimates the probability that the event of interest does not occur (e.g. a patient is still alive) at each time point.

Survival analysis in R

The core survival analysis functions are in the `survival` package. The survival package is one of the few "core" packages that comes bundled with your basic R installation, so you probably didn't need to `install.packages()` it. But, you'll need to load it like any other library when you want to use it. We'll also want to load the `survminer` package, which provides much nicer Kaplan-Meier plots out-of-the-box than what you get out of base graphics. Make sure these packages are installed before you load them.

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

```
## Warning: package 'survminer' was built under R version 4.4.1
```

```
## Warning: package 'ggpubr' was built under R version 4.4.1
```

The core functions in the survival package are not very intuitive. Basically, before you can make a survival curve, you need to create a "survival object" using `Surv()`. You then use the `survfit()` function to create a survival curve which you can plot, or another function to perform statistical tests using the `Surv()` object.

We're going to be using the built-in lung cancer dataset that ships with the survival package. You can get some more information about the dataset by running `?lung`, then clicking on the `lung{survival}` link in RStudio. The help tells us there are 10 variables in this data:

```
?lung
```

1. `inst`: Institution code
2. `time`: **Survival time in days**
3. `status`: **censoring status 1=censored, 2=dead**
4. `age`: Age in years
5. `sex`: **Male=1 Female=2**
6. `ph.ecog`: ECOG performance score as rated by the physician. 0=asymptomatic, 1= symptomatic but completely ambulatory, 2= in bed <50% of the day, 3= in bed > 50% of the day but not bedbound, 4 = bedbound
7. `ph.karno`: Karnofsky performance score (bad=0-good=100) rated by physician
8. `pat.karno`: Karnofsky performance score as rated by patient
9. `meal.cal`: Calories consumed at meals
10. `wt.loss`: Weight loss in last six months (pounds)

You can access the data just by running `lung`, as if you had read in a dataset and called it lung. You can operate on it just like any other data frame.

We are interested in three columns: `time`, `status`, and `sex`. The status column contains the censoring status of each observation, and is required to create a survival object, and to do any survival analyses. Importantly, the `Surv()` function will accept *either*: TRUE/FALSE (where TRUE is event and FALSE is censored), 1/0 (where 1 is event and 0 is censored), or 2/1 (where 2 is event and 1 is censored). In this case we have the last option, which is the least common. Typically you will see 1=event, 0=censored.

The survival function

Censoring happens when the study ends, or a patient leaves before the event occurs.

First we create a survival object using the `Surv` function

```
s <- Surv(lung$time, lung$status)
```

do not factor the status column

Here, the first argument in the brackets specifies which column has the time-to-event data, and the second specifies which column has the censoring status information. `Surv` will automatically check whether you're using 0/1 or 1/2 to represent censored vs "dead", respectively.

If we display the new object:

```
s
```

```
##   [1]  306   455  1010+  210   883  1022+  310   361   218   166   170   654
##  [13]  728    71   567   144   613   707    61    88   301    81   624   371
##  [25]  394   520   574   118   390    12   473    26   533   107    53   122
##  [37]  814   965+   93   731   460   153   433   145   583    95   303   519
##  [49]  643   765   735   189    53   246   689    65     5   132   687   345
##  [61]  444   223   175    60   163    65   208   821+  428   230   840+  305
##  [73]   11   132   226   426   705   363    11   176   791    95   196+  167
##  [85]  806+  284   641   147   740+  163   655   239    88   245   588+   30
##  [97]  179   310   477   166   559+  450   364   107   177   156   529+   11
## [109]  429   351    15   181   283   201   524    13   212   524   288   363
## [121]  442   199   550    54   558   207    92    60   551+  543+  293   202
## [133]  353   511+  267   511+  371   387   457   337   201   404+  222    62
## [145]  458+  356+  353   163    31   340   229   444+  315+  182   156   329
## [157]  364+  291   179   376+  384+  268   292+  142   413+  266+  194   320
## [169]  181   285   301+  348   197   382+  303+  296+  180   186   145   269+
## [181]  300+  284+  350   272+  292+  332+  285   259+  110   286   270    81
## [193]  131   225+  269   225+  243+  279+  276+  135    79    59   240+  202+
## [205]  235+  105   224+  239   237+  173+  252+  221+  185+   92+   13   222+
## [217]  192+  183   211+  175+  197+  203+  116   188+  191+  105+  174+  177+
```

When you use ~ 1 in the formula, it means that all individuals in the dataset are treated as part of single grp.

If your dataset has a treatment variable with two levels (Control and Treated), the formula would not have 1.

We can see that the object is special type of vector that tells you both how long the subject was tracked for, and whether or not the event occurred or the sample was censored (shown by the +).

Now, let's fit a survival curve with the `survfit()` function. Here we'll create a simple curve that doesn't consider any different groupings, so we'll specify just an intercept (e.g. `~1`) in the formula that `survfit` expects. To make the `Surv()` object we can always run the `Surv()` function separately (like we just did). But from here out we'll nest the `Surv()` call within the `survfit()` call to save one line of code. And we'll use the `data=` argument to specify which data we're using.

```
sfit <- survfit(Surv(time, status)~1, data=lung)
sfit
```

This line of code creates a survival object using the Kaplan-Meier method and generates the overall survival curve for the dataset.

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

Calling the new object gives a very brief summary- e.g. number of subjects, number of events, and the median time until the event occurs across the whole sample.

It's more useful to run `summary` on the object. This will show a life table:

```
summary(sfit)
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     5    228       1   0.9956 0.00438       0.9871        1.000
##    11    227       3   0.9825 0.00869       0.9656        1.000
##    12    224       1   0.9781 0.00970       0.9592        0.997
##    13    223       2   0.9693 0.01142       0.9472        0.992
##    15    221       1   0.9649 0.01219       0.9413        0.989
##    26    220       1   0.9605 0.01290       0.9356        0.986
##    30    219       1   0.9561 0.01356       0.9299        0.983
##    31    218       1   0.9518 0.01419       0.9243        0.980
##    53    217       2   0.9430 0.01536       0.9134        0.974
##    54    215       1   0.9386 0.01590       0.9079        0.970
##    59    214       1   0.9342 0.01642       0.9026        0.967
##    60    213       2   0.9254 0.01740       0.8920        0.960
##    61    211       1   0.9211 0.01786       0.8867        0.957
##    62    210       1   0.9167 0.01830       0.8815        0.953
##    65    209       2   0.9079 0.01915       0.8711        0.946
##    71    207       1   0.9035 0.01955       0.8660        0.943
##    79    206       1   0.8991 0.01995       0.8609        0.939
##    81    205       2   0.8904 0.02069       0.8507        0.932
##    88    203       2   0.8816 0.02140       0.8406        0.925
##    92    201       1   0.8772 0.02174       0.8356        0.921
##    93    199       1   0.8728 0.02207       0.8306        0.917
##    95    198       2   0.8640 0.02271       0.8206        0.910
##   105    196       1   0.8596 0.02302       0.8156        0.906
```

Table Header:

time: The time when something (an event like death or censoring) happened.

n.risk: The number of people still "at risk" (alive or uncensored) right before

n.event: The number of events (e.g., deaths) that happened at that specific time.

survival: The probability that a person survives past that time.

std.err: The standard error, which tells how much uncertainty there is in the survival estimate.

Kaplan-Meier Formula:
The survival probability at a given time is calculated as:

S(t) = S(previous) × (1 - (n.event / n.risk))

Where:

S(t): Survival probability at the current time.
S(previous): Survival probability at the previous time point.
n.event: The number of events (e.g., deaths) at the current time.
n.risk: The number of people "at risk" (alive and uncensored) just before the event.

It is worth trying to understand this table. Each time a subject is lost to the event or to censoring, a new row is added. The `time` column specifies the time the event or censor happened, and the `n.event` column shows how many subjects were lost at that time (e.g. multiple subjects might be lost on the same day). The `n.risk` column shows the number of subjects still at risk (i.e. that haven't been lost yet). This goes down at each step according to how many events were observed at this time. Finally, the `survival` column shows the estimated likelihood of survival to this time point. Note that at each time point the risk (or hazard) of the event happening is therefore 1/survival.

One question we may want to ask is: what is the survival probability at a specific time point? You can estimate this roughly by reading the value in the `survival` column for the row closest to the time you want. Or you can specify a specific time directly using the `times` argument. For example, to estimate survival after 1 year:

```
summary(survfit(Surv(time, status) ~ 1, data = lung), times = 365)
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##   365     65     121    0.409  0.0358        0.345        0.486
```

This tells us that the probability of survival at time 365 is 40.9% (and gives 95% confidence intervals).

Alternatively, you can specify times that you want the life table to display. For example, we can specify a sequence of numbers using the `seq` function. Here, you specify the minimum, maximum, and the size of the increments. First, we should check what the range of times is in the dataset:

```
range(lung$time)
```

```
## [1]    5 1022
```

Then we use the `seq` function to display the survival probability in regular intervals. In general this is a much nicer way to view the life table anyway.

```
summary(sfit, times=seq(0, 1100, 100))
```

```
## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     0    228       0   1.0000  0.0000       1.0000        1.000
##   100    196      31   0.8640  0.0227       0.8206        0.910
##   200    144      41   0.6803  0.0311       0.6219        0.744
##   300     92      29   0.5306  0.0346       0.4669        0.603
##   400     57      25   0.3768  0.0358       0.3128        0.454
##   500     41      12   0.2933  0.0351       0.2320        0.371
##   600     24      10   0.2136  0.0335       0.1571        0.290
##   700     16       8   0.1424  0.0303       0.0938        0.216
##   800      8       7   0.0783  0.0246       0.0423        0.145
##   900      3       2   0.0503  0.0228       0.0207        0.123
##  1000      2       0   0.0503  0.0228       0.0207        0.123
```

Lower 95% CI: The lower bound of the confidence interval, meaning the minimum survival probability expected at that time point.

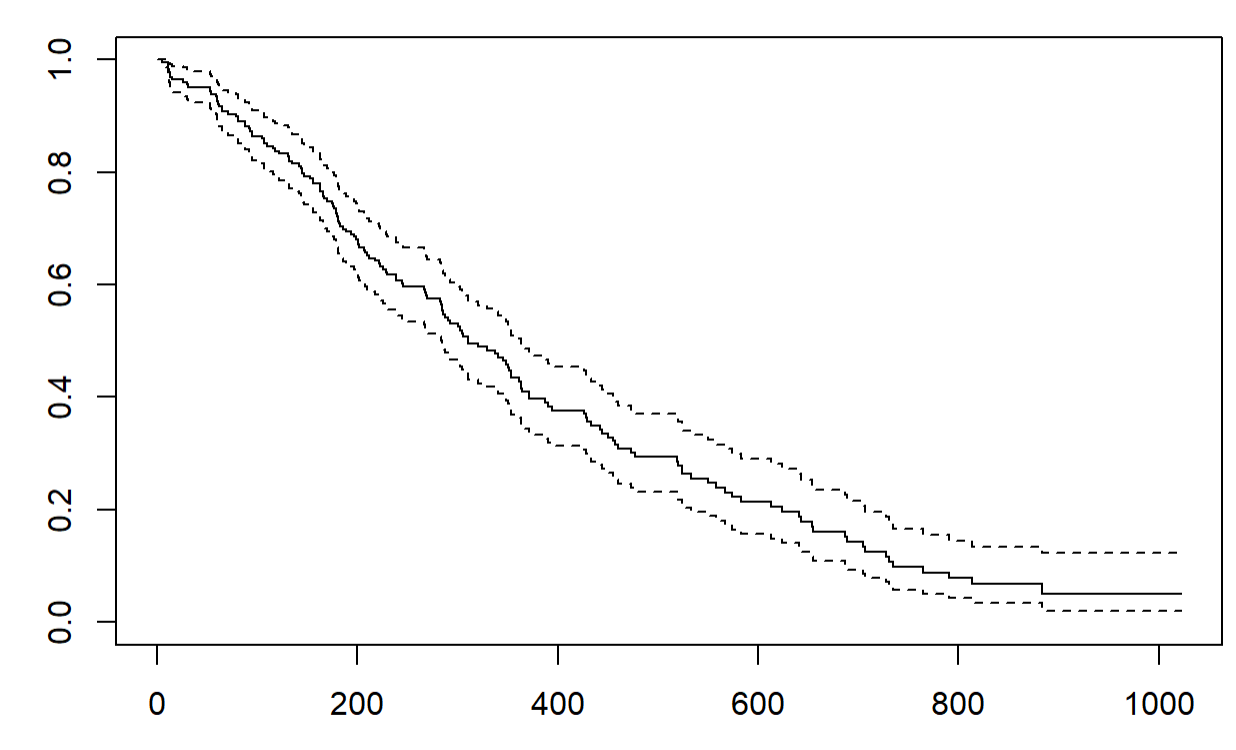
Upper 95% CI: The upper bound of the confidence interval, meaning the maximum survival probability expected at that time point.

Example:
At time = 5, the survival probability is 0.9956, with a 95% confidence interval of [0.9871, 1.000]. This means there is a 95% chance that the true survival probability lies between 0.9871 and 1.000.

Kaplan-Meier Plots

Now that we've fit a survival curve to the data it's pretty easy to visualize it with a **Kaplan-Meier plot**. Create the survival object if you don't have it yet, and instead of using `summary()`, use `plot()`.

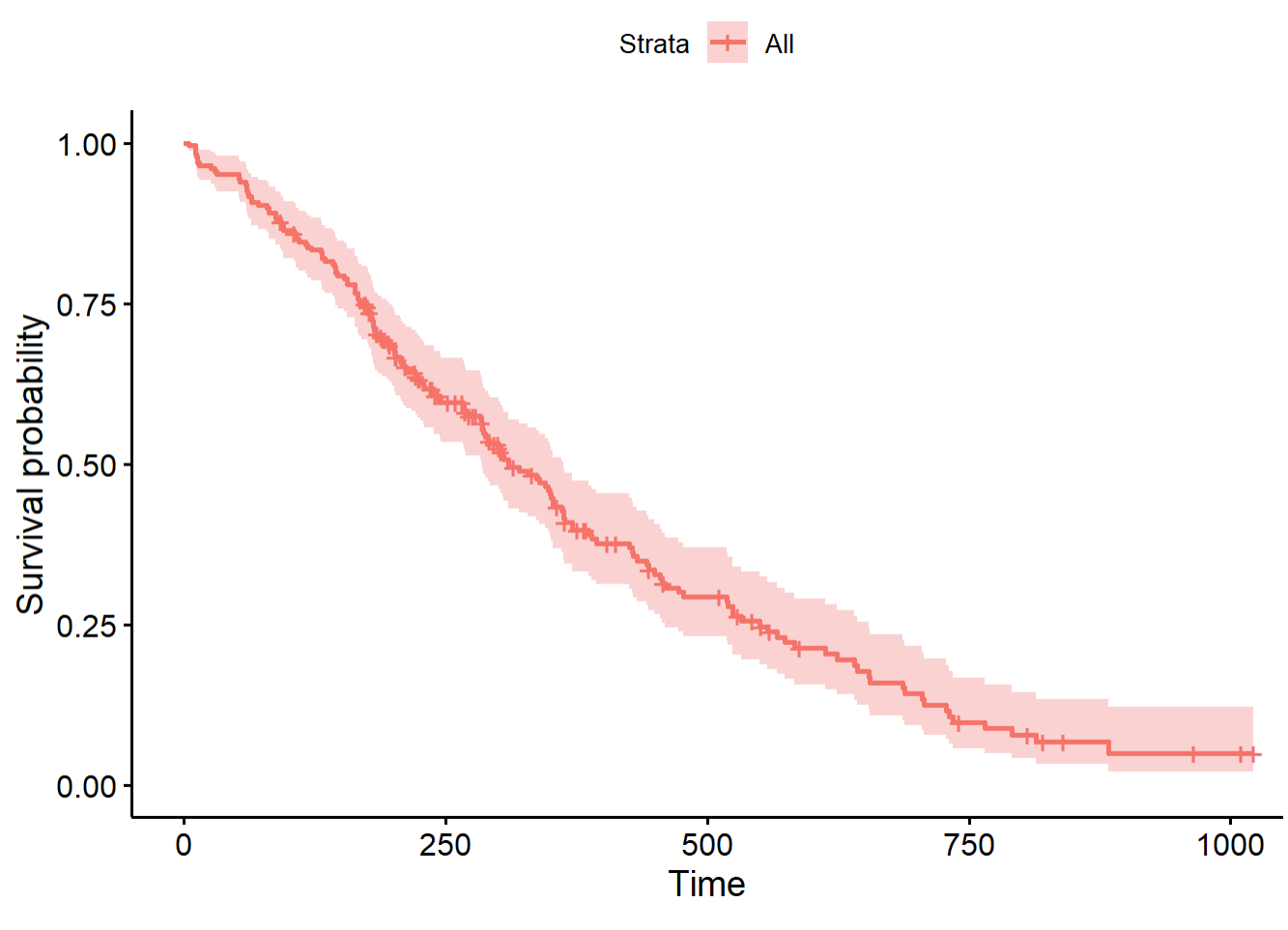
```
sfit <- survfit(Surv(time, status)~1, data=lung)
plot(sfit)
```



The plot shows time on the x axis, and the estimated survival probability on the y axis. The solid line is the Kaplan-Meier survival function. This is a step function illustrating the *cumulative* survival probability over time. The curve is horizontal over periods where no event occurs, then drops vertically corresponding to a change in the survival function at each time an event (or censor) occurs. If several events or censors occur at the same time point, the curve will drop by a larger amount. Finally, the dotted lines show the upper and lower 95% confidence intervals for survival probability, estimated from the `survfit()` function.

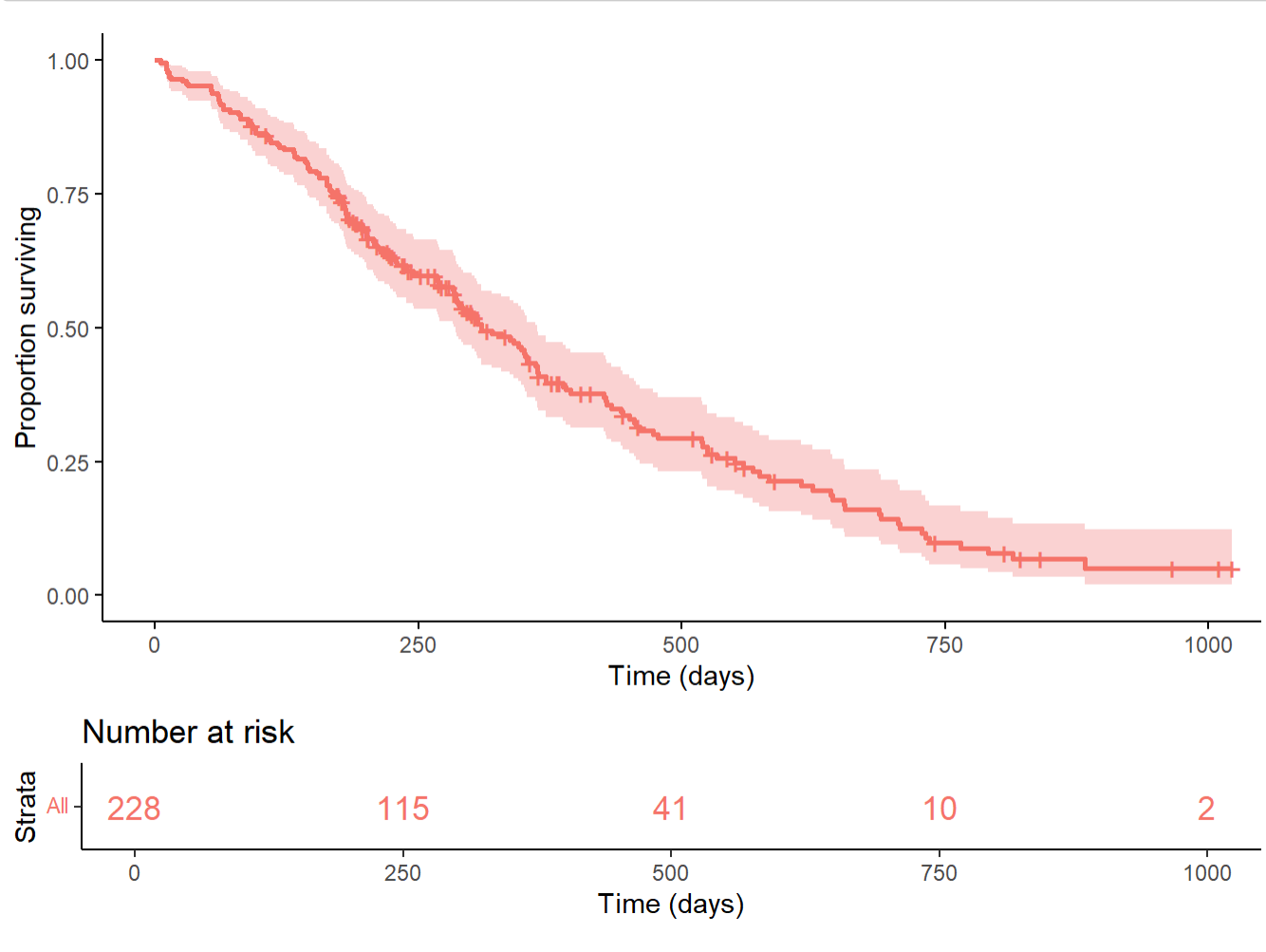
There are lots of ways to modify the plot produced by base R's `plot()` function. You can see more options with the help for `?plot.survfit`. We're not going to go into any more detail here, because there's another package called **survminer** that provides a function called `ggsurvplot()` that makes it much easier to produce publication-ready survival plots, and if you're familiar with **ggplot2** syntax it's pretty easy to modify. So, let's load the package and try it out.

```
library(survminer)
ggsurvplot(sfit)
```



This plot is substantially more informative by default, because it automatically colour codes different groups, adds axis labels, and creates an automatic legend. It also colours in the 95% confidence interval, which looks nicer. You can easily extend the plot to add a table at the bottom showing the number of survivors at different time increments:

```
ggsurvplot(sfit,
           risk.table = TRUE, # Add risk table
           risk.table.col = "strata", # Change risk table color by groups
           xlab = "Time (days)",
           ylab = "Proportion surviving",
           legend="none",
           ggtheme = theme_classic())
```



Comparing two curves

So far we've looked at how we make the survival function itself. But an important use of survival analysis is to test whether two or more survival functions are significantly different from each other. The `lung` data set contains observations for men and women. So one question we can ask is: does survival differ between the sexes?

We can model this by replacing the intercept in the model (`~1`) with some fixed factor. First, let's add sex as a categorical factor. The `survfit()` function will now estimate two survival functions- one for males and one for females.

```
sfit_sex <- survfit(Surv(time, status)~sex, data=lung)
sfit_sex
```

```
## Call: survfit(formula = Surv(time, status) ~ sex, data = lung)
##
##         n events median 0.95LCL 0.95UCL
## sex=1 138    112    270     212     310
## sex=2  90     53    426     348     550
```

Now, calling the `surv` object gives a summary for males and a summary for females. Remember: sex 1= male, sex2= female.

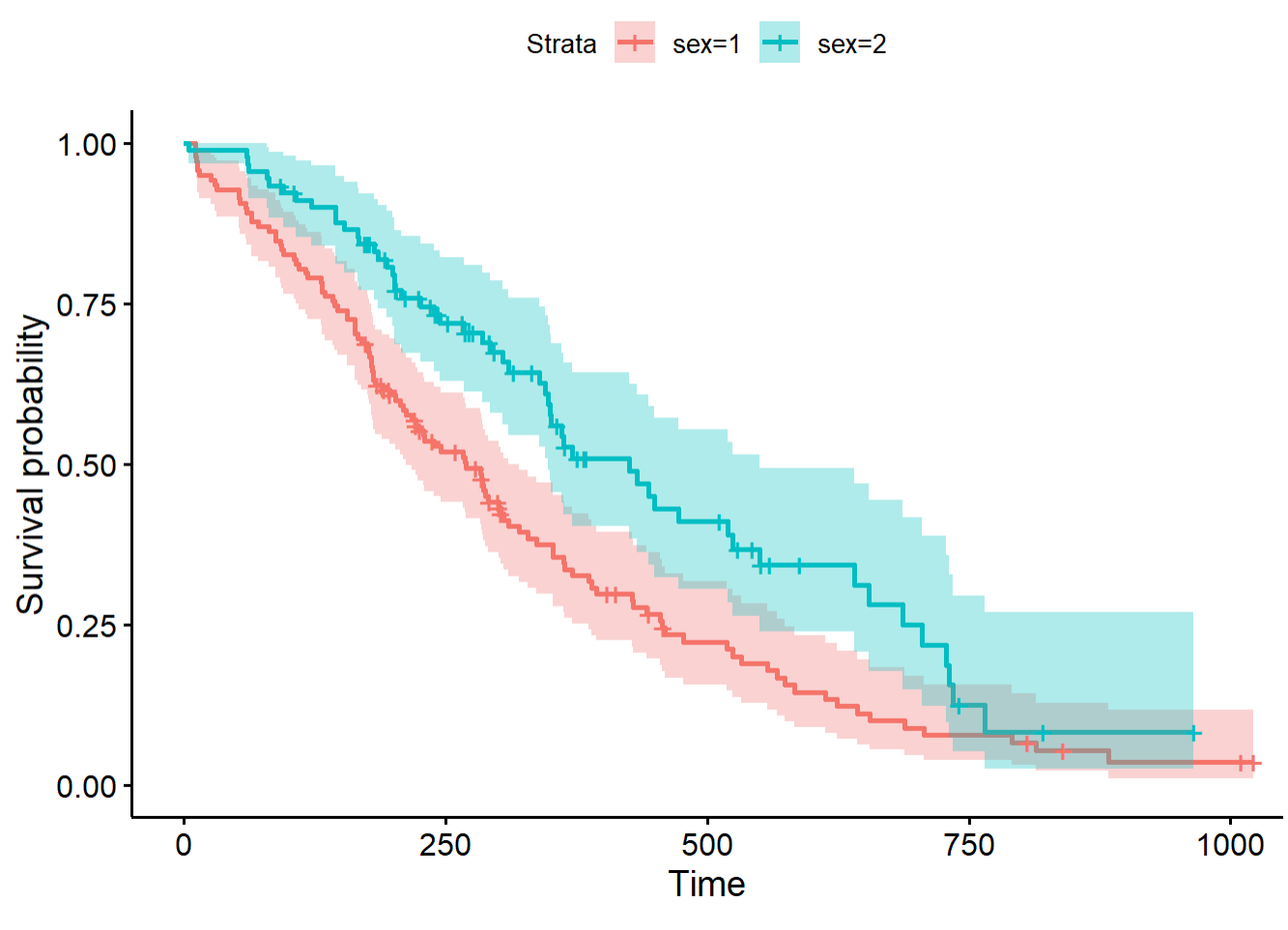
We can also produce a life table separately for males and females. From these tables we can see that males tend to have worse survival than females. For example, survival at time 500 is about half as likely for males (0.22) as for females (0.41).

```
summary(sfit_sex, times=seq(0, 1000, 100))
```

```
## Call: survfit(formula = Surv(time, status) ~ sex, data = lung)
##
##                 sex=1
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     0    138       0   1.0000  0.0000       1.0000        1.000
##   100    114      24   0.8261  0.0323       0.7652        0.892
##   200     78      30   0.6073  0.0417       0.5309        0.695
##   300     49      20   0.4411  0.0439       0.3629        0.536
##   400     31      15   0.2977  0.0425       0.2250        0.394
##   500     20       7   0.2232  0.0402       0.1569        0.318
##   600     13       7   0.1451  0.0353       0.0900        0.234
##   700      8       5   0.0893  0.0293       0.0470        0.170
##   800      6       2   0.0670  0.0259       0.0314        0.143
##   900      2       2   0.0357  0.0216       0.0109        0.117
##  1000      2       0   0.0357  0.0216       0.0109        0.117
##
##                 sex=2
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     0     90       0   1.0000  0.0000       1.0000        1.000
##   100     82       7   0.9221  0.0283       0.8683        0.979
##   200     66      11   0.7946  0.0432       0.7142        0.884
##   300     43       9   0.6742  0.0523       0.5791        0.785
##   400     26      10   0.5089  0.0603       0.4035        0.642
##   500     21       5   0.4110  0.0626       0.3050        0.554
##   600     11       3   0.3433  0.0634       0.2390        0.493
##   700      8       3   0.2496  0.0652       0.1496        0.417
```

This difference is also clear when we plot the curves:

```
library(survminer)
ggsurvplot(sfit_sex, conf.int = TRUE)
```



From the plot, it looks like the estimated male survival probability is lower than female survival probability up until around 750 days (when the confidence intervals start overlapping).

We can then formally test for a difference in the survival function between males and females using a **Cox proportional hazards (Cox ph) regression model**. This statistical tests for a difference in the shape of the survival curves. This is a non-parametric model which compares the entire survival curve, and so can account for complex changes in the shapes of the curves.

We can see the z score and P value for this comparison using the `summary` function.

```
sfit_sex2 <- coxph(Surv(time, status)~sex, data = lung)
summary(sfit_sex2)
```

The model examines the relationship between sex (a predictor variable) and survival time (the response variable) in the lung dataset.

```
## Call:
## coxph(formula = Surv(time, status) ~ sex, data = lung)
##
##   n= 228, number of events= 165
##
##         coef exp(coef) se(coef)      z Pr(>|z|)
## sex -0.5310    0.5880   0.1672 -3.176  0.00149 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##     exp(coef) exp(-coef) lower .95 upper .95
## sex     0.588      1.701    0.4237     0.816
##
## Concordance= 0.579  (se = 0.021 )
## Likelihood ratio test= 10.63  on 1 df,   p=0.001
## Wald test            = 10.09  on 1 df,   p=0.001
## Score (logrank) test = 10.33  on 1 df,   p=0.001
```

From the first row, we know the sex variable is a significant predictor of survival (p = 0.00149).

From the second row, we see that females have a 41.2% lower hazard (HR = 0.588) than males, with a 95% confidence interval of (0.4237, 0.816).

The Cox proportional hazards analysis revealed a statistically significant difference in survival between males and females (p < 0.001).

The hazard ratio of 0.588 indicates that females had a 41.2% lower risk of death compared to males over the study duration. This suggests that survival probability was significantly higher for females than males.

The findings are consistent with visual inspection of the survival curves

The P value of 0.00149 tells us that survival differs significantly between males and females. How can we tell which sex has lower survival? Well, we can see clearly in the above figures that male survival is lower than female survival. But we can also tell from the model output if we know where to look. The second reported coefficient (`exp(coef)` in the output= 0.588) is the hazard ratio- the difference in the **risk of dying** between males and females. And because we haven't specified, R by default sets the first treatment it finds in the data set as the 'reference' treatment. So because males= 1 in this data set, they are taken as the reference level. A value less than one therefore means females have a lower *risk of dying* than males (and so a *higher* chance of survival). In absolute terms, 0.588 times as many females died as males across the whole study duration.

We won't cover this, but you should know that Cox PH regression can also assess the effect of continuous variables, and can model the effect of multiple variables at once.

Exercise

> For this exercise, you will need to use the 'colon' data found in the survival package. This dataset has survival and recurrence information on 929 people from a clinical trial on colon cancer chemotherapy. There are two rows per person, indicated by the event type (`etype`) variable – `etype==1` indicates that row corresponds to recurrence; `etype==2` indicates death. You will have to think about which of these 'etypes' you need to work with for the survival analysis.

Colon Data:

Sex:
1 = male
0 = female

Status:
1=event
0=censored

Lung Data:
Status: 2 = event, 1 = censored; Sex: 1 = male, 2 = female

For this exercise, please complete the following questions:

1. Look at the help for `?colon` again. How are sex and status coded? How is this different from the lung data?
2. Create a survival curve separately for males versus females. Run a `summary()` on this object, showing time points 0, 500, 1000, 1500, and 2000. Do males or females appear to fair better over this time period?
3. Plot a Kaplan-Meier curve for both sexes. Using an appropriate statistical test, determine whether there is a significant difference in survival between males and females.