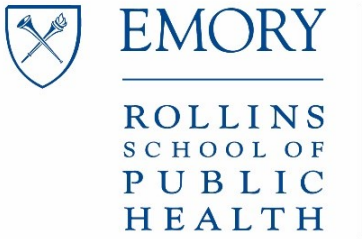


BIOS522: Survival Analysis Methods

Week 3: R Tutorial

Survival data basics in R

Formatting survival data

In this course, we will use the `survival` package in `R`. See documentation: https://cran.r-project.org/web/packages/survival/survival.pdf (https://cran.r-project.org/web/packages/survival/survival.pdf)

Recall that right-censored data cannot be represented by a single variable for follow-up time. We must also indicate whether the time is a failure time or a censoring time. Thus, we must capture both data elements when analyzing right-censored data.

In the `survival` package, right-censored data is formatted using the `Surv` command. For right censored-data, the syntax is:

```
Surv(time,event)
```

where `time` is the variable storing the observed (right-censored) follow-up times T_i^*, and `event` is the variable storing the event indicators δ_i. We will use this `surv` object as the dependent variable in our models.

Data example 1

The survival package includes several built-in data sets. The data set `leukemia` summarizes survival in 23 patients with Acute Myelogeneous Leukemia. The clinical question was whether the standard course of chemotherapy should be extended ("maintenance") for additional cycles.

```
library(survival)
data(leukemia)
leukemia[1:23,]
```

```
##    time status            x
## 1     9      1    Maintained
## 2    13      1    Maintained
## 3    13      0    Maintained
## 4    18      1    Maintained
## 5    23      1    Maintained
## 6    28      0    Maintained
## 7    31      1    Maintained
## 8    34      1    Maintained
## 9    45      0    Maintained
## 10   48      1    Maintained
## 11  161      0    Maintained
## 12    5      1 Nonmaintained
## 13    5      1 Nonmaintained
## 14    8      1 Nonmaintained
## 15    8      1 Nonmaintained
## 16   12      1 Nonmaintained
## 17   16      0 Nonmaintained
## 18   23      1 Nonmaintained
## 19   27      1 Nonmaintained
## 20   30      1 Nonmaintained
## 21   33      1 Nonmaintained
## 22   43      1 Nonmaintained
## 23   45      1 Nonmaintained
```

The variable `time` summarizes survival or censoring time in months. The variable `status` is the event indicator. Thus, our time-to-event variable is `Surv(time=leukemia$time,event=leukemia$status)` or, equivalently, `Surv(leukemia$time,leukemia$status)`. This can be further abbreviated to `Surv(time,status)` if the data set is specified elsewhere in our code.

Data example 2

For small data sets, like the AIDS hemophiliac example used in **Lecture 3**, we can load the data directly into `R`. Note we can give the follow-up time variable and event indicator arbitrary names, as long as we tell `R` which variable is which when we run our analyses.

```
futime <- c(2,3,6,6,8,10,15,15,16,27,30,32, 1,1,2,4,4,5,5,7,14,22)
event <-   c(1,0,1,1,1,0,1,1,1,1,1,1, 1,1,1,1,1,1,0,1,0,1)
agegt40 <- c(0,0,0,0,0,0,0,0,0,0,0,0, 1,1,1,1,1,1,1,1,1,1)
aidshemo <- data.frame(futime=futime,event=event,agegt40=agegt40)
aidshemo[1:22,]
```

```
##    futime event agegt40
## 1       2     1       0
## 2       3     0       0
## 3       6     1       0
## 4       6     1       0
## 5       8     1       0
## 6      10     0       0
## 7      15     1       0
## 8      15     1       0
## 9      16     1       0
## 10     27     1       0
## 11     30     1       0
## 12     32     1       0
## 13      1     1       1
## 14      1     1       1
## 15      2     1       1
## 16      4     1       1
## 17      4     1       1
## 18      5     1       1
## 19      5     0       1
## 20      7     1       1
## 21     14     0       1
## 22     22     1       1
```

Our time-to-event variable is `Surv(time=aidshemo$futime,event=aidshemo$event)`.

Data example 3

Finally, we can read in a data set from an external file. We use data from the AIDS Clinical Trial Group (ACTG) randomized trial ACTG320. The trial assessed the safety and efficacy of the protease inhibitor indinavir in individuals with HIV and low CD4 cell count. (Low CD4 cell count indicates a weak immune system and poor health.) The primary outcome was time until the development of a new AIDS-defining event or death. A secondary outcome was time until death.

Using the drop-down menu, select “Import Dataset” and “From text (readr)” to read in the ACTG320 dataset, which is stored as a comma separated value file. This will generate a statement like this:

```
actg320 <- read.csv("actg320.csv", header = TRUE, sep = ",")
actg320[1:10,]
```

```
##    id time censor time_d censor_d tx txgrp cd4_grp sex raceth ivdrug hemophil
## 1   1  189      0    189        0  0     1       1   1      1      1        0
## 2   2  287      0    287        0  0     1       1   2      2      1        0
## 3   3  242      0    242        0  1     2       0   1      1      1        1
## 4   4  199      0    199        0  0     1       1   1      1      1        0
## 5   5  286      0    286        0  1     2       0   1      1      3        0
## 6   6  285      0    285        0  1     2       0   1      1      1        0
## 7   7  270      0    270        0  0     1       1   1      2      1        0
## 8   8  285      0    285        0  1     2       1   1      2      3        0
## 9   9  276      0    276        0  0     1       1   1      1      1        0
## 10 10  306      0    306        0  0     1       1   1      1      1        0
##    karnof   cd4 priorzdv age
## 1     100 169.0       39  34
## 2      90 149.5       15  34
## 3     100  23.5        9  20
## 4      90  46.0       53  48
## 5      90  10.0       12  46
## 6      70   0.0       24  51
## 7     100  54.5        6  51
## 8      80 117.5       24  40
## 9     100  95.0        7  34
## 10     90  71.0        7  38
```

The variable `time` summarizes survival or censoring time in days for the primary outcome of AIDS-defining event or death. The variable `censor` is the event indicator, where 0 indicates censoring. Thus, our primary time-to-event variable is `Surv(actg320$time,actg320$censor)`.

The variable `time_d` summarizes survival or censoring time in days for the secondary outcome of death. The variable `censor_d` is the event indicator, where 0 indicates censoring. Thus, our secondary time-to-event variable is `Surv(actg320$time_d,actg320$censor_d)`.

We will use the ACTG320 data set throughout the course.

Kaplan-Meier estimation

Fitting a Kaplan-Meier estimate

The basic function for fitting the Kaplan-Meier estimator is `survfit()`. The function includes a model statement, in which we specify the dependent time-to-event variable `Surv(time,event)` on the left and the independent variable(s) on the right. If we want to fit a Kaplan-Meier to all data and so do not have groups defined by independent variables, we write `1` on the right-side instead.

We can specify our data set with the option `data=` which simplifies our `Surv()` statement. Using the `actg320` data set, we store our results with a name of our choosing. We can then examine a brief or detailed summary of the results.

```
actgKM <- survfit(Surv(time,censor) ~ 1, data=actg320)
actgKM
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320)
##
##         n events median 0.95LCL 0.95UCL
## [1,] 1151     96     NA      NA      NA
```

The brief result contains the total sample size (`1151`) and the number of events (`96`). Where calculable, it also reports the Kaplan-Meier estimated median survival time and an associated 95% confidence interval. The median is not calculable for this data example. (**Why is that? Check your understanding.**)

We can also calculate the survival at any specified time using the `summary()` function with the option `times=` . We calculate survival at 6 and 12 months (specified in days, the time scale used in the data set).

```
summary(actgKM,times=c(180,365))
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##   180    851      77    0.929 0.00787        0.913        0.944
```

To examine the entire Kaplan-Meier curve, we can use the `summary()` function.

```
summary(actgKM)
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320)
##
##  time n.risk n.event survival  std.err lower 95% CI upper 95% CI
##     1   1151       1    0.999 0.000868        0.997        1.000
##     2   1148       1    0.998 0.001229        0.996        1.000
##     7   1144       3    0.996 0.001944        0.992        0.999
##     9   1141       1    0.995 0.002129        0.991        0.999
##    13   1138       2    0.993 0.002458        0.988        0.998
##    14   1136       3    0.990 0.002881        0.985        0.996
##    15   1133       1    0.990 0.003008        0.984        0.995
##    16   1132       1    0.989 0.003130        0.983        0.995
##    17   1131       1    0.988 0.003247        0.981        0.994
##    18   1128       3    0.985 0.003575        0.978        0.992
##    20   1124       2    0.983 0.003777        0.976        0.991
##    24   1122       1    0.983 0.003874        0.975        0.990
##    25   1121       2    0.981 0.004061        0.973        0.989
##    26   1117       1    0.980 0.004151        0.972        0.988
##    35   1115       1    0.979 0.004239        0.971        0.987
##    39   1112       1    0.978 0.004326        0.970        0.987
##    42   1105       1    0.977 0.004412        0.969        0.986
##    44   1103       1    0.976 0.004496        0.968        0.985
##    46   1102       2    0.975 0.004659        0.965        0.984
##    47   1097       1    0.974 0.004739        0.964        0.983
##    52   1090       1    0.973 0.004818        0.963        0.982
##    56   1083       1    0.972 0.004896        0.962        0.982
##    58   1081       1    0.971 0.004974        0.961        0.981
##    61   1077       1    0.970 0.005050        0.960        0.980
##    64   1073       1    0.969 0.005126        0.959        0.979
##    65   1072       1    0.968 0.005200        0.958        0.979
##    68   1065       2    0.966 0.005347        0.956        0.977
##    77   1051       1    0.966 0.005420        0.955        0.976
##    81   1048       1    0.965 0.005493        0.954        0.975
##    82   1046       2    0.963 0.005635        0.952        0.974
##    84   1042       1    0.962 0.005705        0.951        0.973
##    85   1039       1    0.961 0.005774        0.950        0.972
##    87   1036       1    0.960 0.005842        0.949        0.972
##    90   1027       1    0.959 0.005911        0.948        0.971
##    91   1025       3    0.956 0.006112        0.944        0.968
##   103   1012       1    0.955 0.006178        0.943        0.968
##   105   1006       1    0.954 0.006245        0.942        0.967
##   108   1001       1    0.953 0.006311        0.941        0.966
##   112    993       1    0.952 0.006377        0.940        0.965
##   113    990       2    0.951 0.006508        0.938        0.963
##   114    988       2    0.949 0.006635        0.936        0.962
##   115    986       1    0.948 0.006698        0.935        0.961
##   117    983       2    0.946 0.006822        0.932        0.959
##   123    972       1    0.945 0.006884        0.931        0.958
##   126    964       1    0.944 0.006946        0.930        0.957
##   127    962       1    0.943 0.007008        0.929        0.957
##   129    961       2    0.941 0.007129        0.927        0.955
##   130    959       1    0.940 0.007189        0.926        0.954
##   135    943       1    0.939 0.007250        0.925        0.953
```

```
##   137    939       1    0.938 0.007311        0.924        0.952
##   138    935       1    0.937 0.007372        0.923        0.951
##   144    925       1    0.936 0.007433        0.921        0.951
##   149    917       1    0.935 0.007495        0.920        0.950
##   151    914       2    0.933 0.007617        0.918        0.948
##   167    882       1    0.932 0.007681        0.917        0.947
##   169    875       1    0.931 0.007746        0.916        0.946
##   171    873       1    0.930 0.007810        0.914        0.945
##   174    865       1    0.929 0.007874        0.913        0.944
##   181    844       1    0.927 0.007941        0.912        0.943
##   184    835       1    0.926 0.008009        0.911        0.942
##   186    833       1    0.925 0.008076        0.909        0.941
##   190    815       1    0.924 0.008146        0.908        0.940
##   194    809       1    0.923 0.008216        0.907        0.939
##   197    796       1    0.922 0.008287        0.906        0.938
##   203    781       1    0.921 0.008360        0.904        0.937
##   206    774       1    0.919 0.008433        0.903        0.936
##   226    717       1    0.918 0.008518        0.902        0.935
##   231    690       1    0.917 0.008609        0.900        0.934
##   233    684       2    0.914 0.008790        0.897        0.931
##   244    639       1    0.913 0.008892        0.895        0.930
##   245    629       1    0.911 0.008996        0.894        0.929
##   248    628       1    0.910 0.009097        0.892        0.928
##   255    596       1    0.908 0.009209        0.890        0.926
##   266    513       1    0.906 0.009360        0.888        0.925
##   288    371       1    0.904 0.009648        0.885        0.923
##   298    312       1    0.901 0.010043        0.882        0.921
```

For each distinct failure time, this reports the number at risk, the number of failures, the estimated survival, standard error, and a 95% confidence interval. By default, the confidence intervals are calculated using a log transformation.

We can specify other options using the `conf.type` command. We compare Wald (`plain'), log-transformed, and log-log transformed intervals for estimated survival at 3 months. Note the same point estimate but differing confidence intervals.

```
actgKM.plain <- survfit(Surv(time,censor) ~ 1, data=actg320, conf.type="plain")
summary(actgKM.plain,times=90)
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320, conf.type = "plain")
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##    90   1027      46    0.959 0.00591        0.947        0.971
```

```
actgKM.log <- survfit(Surv(time,censor) ~ 1, data=actg320, conf.type="log")
summary(actgKM.log,times=90)
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320, conf.type = "log")
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##    90   1027      46    0.959 0.00591        0.948        0.971
```

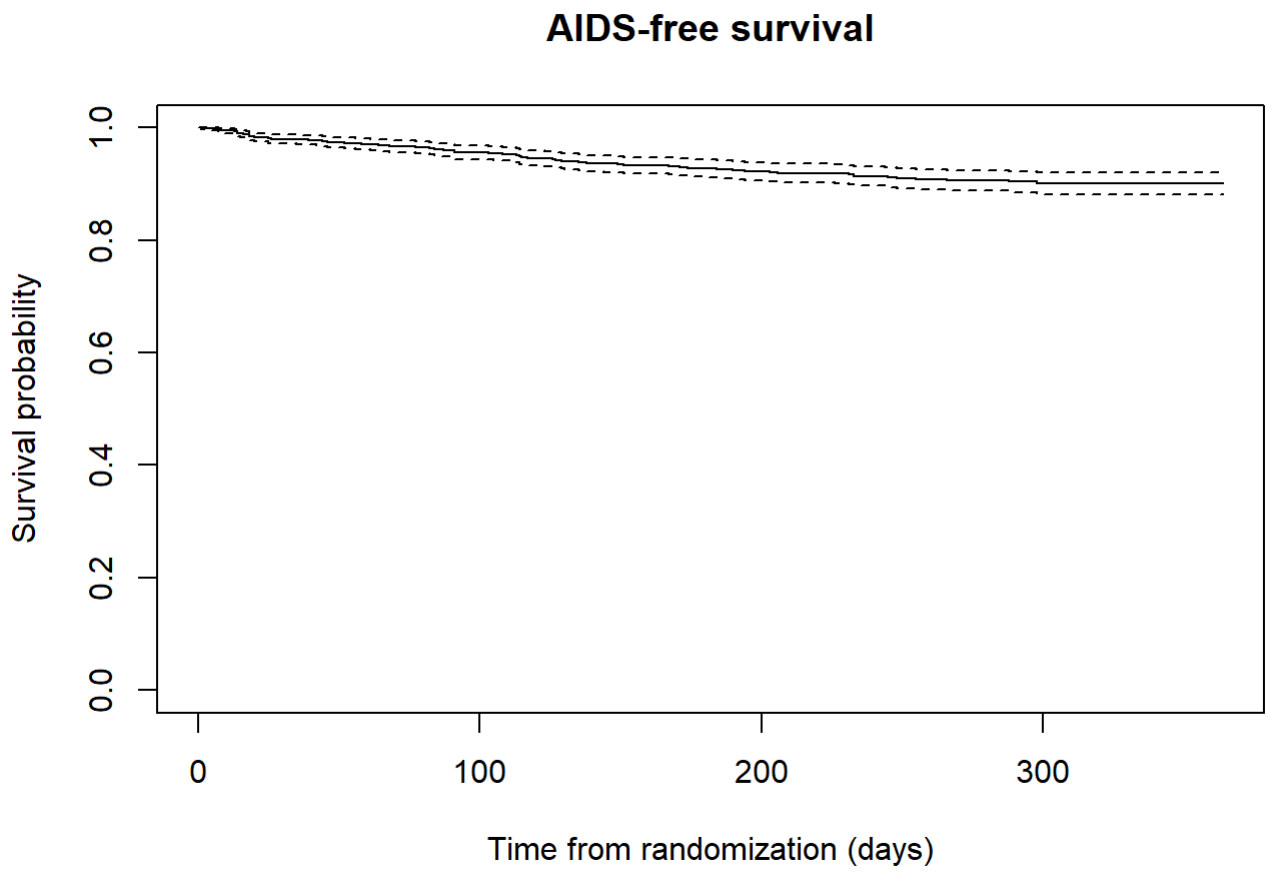
```
actgKM.loglog <- survfit(Surv(time,censor) ~ 1, data=actg320, conf.type="log-log")
summary(actgKM.loglog,times=90)
```

```
## Call: survfit(formula = Surv(time, censor) ~ 1, data = actg320, conf.type = "log-log")
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##    90   1027      46    0.959 0.00591        0.946        0.969
```

Plotting the Kaplan-Meier estimate

Typically, we are most interested in plotting our output. We can achieve this using the `plot()` function. This is shown in base `R`, but `ggplot()` also has nice options for plotting survival objects. We add informative labels.

```
plot(actgKM,xlab = "Time from randomization (days)", ylab = "Survival probability",
     main = "AIDS-free survival")
```



By default, `R` includes confidence limits when we plot a single Kaplan-Meier curve. We can turn these off by adding the option `conf.int=FALSE` to the plot command.

Fitting multiple Kaplan-Meier curves

The `actg320` data includes two groups identified by the covariate `tx`, indicating that patients received a regimen with indinavir (`tx==1`) or without indinavir (`tx==0`). By adding the covariate to the right side of our model statement in `survfit()`, we estimate a separate Kaplan-Meier curve for each level of the covariate.

```
actgKM.tx <- survfit(Surv(time,censor) ~ tx, data=actg320)
summary(actgKM.tx)
```

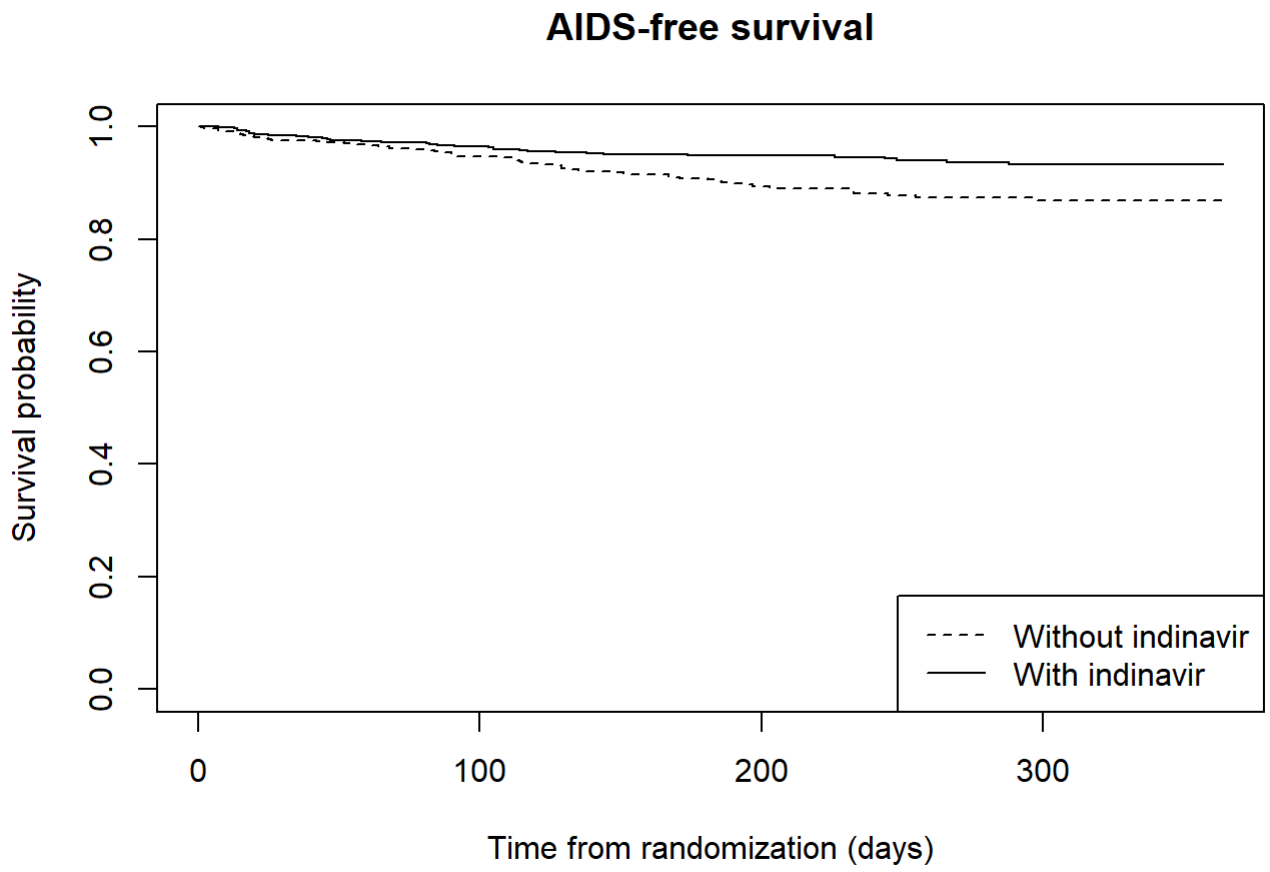
```
## Call: survfit(formula = Surv(time, censor) ~ tx, data = actg320)
##
##                 tx=0
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     1    577       1    0.998 0.00173        0.995        1.000
##     2    575       1    0.997 0.00245        0.992        1.000
##     7    572       2    0.993 0.00346        0.986        1.000
##     9    570       1    0.991 0.00387        0.984        0.999
##    13    567       1    0.990 0.00424        0.981        0.998
##    14    566       1    0.988 0.00458        0.979        0.997
##    15    565       1    0.986 0.00489        0.977        0.996
##    16    564       1    0.984 0.00519        0.974        0.995
##    18    562       1    0.983 0.00547        0.972        0.993
##    20    561       1    0.981 0.00573        0.970        0.992
##    24    560       1    0.979 0.00598        0.967        0.991
##    25    559       1    0.977 0.00622        0.965        0.990
##    26    556       1    0.976 0.00646        0.963        0.988
##    42    551       1    0.974 0.00668        0.961        0.987
##    46    549       1    0.972 0.00690        0.959        0.986
##    52    545       1    0.970 0.00712        0.956        0.984
##    56    539       1    0.968 0.00733        0.954        0.983
##    61    536       1    0.967 0.00753        0.952        0.981
##    64    534       1    0.965 0.00773        0.950        0.980
##    68    530       2    0.961 0.00812        0.945        0.977
##    77    522       1    0.959 0.00831        0.943        0.976
##    82    520       1    0.957 0.00850        0.941        0.974
##    84    517       1    0.956 0.00868        0.939        0.973
##    87    514       1    0.954 0.00886        0.937        0.971
##    90    508       1    0.952 0.00904        0.934        0.970
##    91    506       2    0.948 0.00939        0.930        0.967
##   108    495       1    0.946 0.00956        0.928        0.965
##   112    489       1    0.944 0.00974        0.925        0.964
##   113    486       2    0.940 0.01008        0.921        0.960
##   114    484       1    0.938 0.01024        0.919        0.959
##   115    483       1    0.937 0.01040        0.916        0.957
##   117    482       1    0.935 0.01056        0.914        0.955
##   123    475       1    0.933 0.01072        0.912        0.954
##   126    471       1    0.931 0.01088        0.910        0.952
##   129    470       2    0.927 0.01119        0.905        0.949
##   130    468       1    0.925 0.01134        0.903        0.947
##   135    458       1    0.923 0.01149        0.900        0.945
##   137    456       1    0.921 0.01164        0.898        0.944
##   149    446       1    0.919 0.01180        0.896        0.942
##   151    444       2    0.914 0.01210        0.891        0.938
##   167    425       1    0.912 0.01226        0.889        0.937
##   169    424       1    0.910 0.01242        0.886        0.935
##   171    422       1    0.908 0.01258        0.884        0.933
##   181    408       1    0.906 0.01274        0.881        0.931
##   184    403       1    0.903 0.01291        0.879        0.929
##   186    401       1    0.901 0.01307        0.876        0.927
##   190    394       1    0.899 0.01324        0.873        0.925
##   194    390       1    0.897 0.01340        0.871        0.923
```

```
##   197    382       1    0.894 0.01357        0.868        0.921
##   203    374       1    0.892 0.01374        0.865        0.919
##   206    369       1    0.889 0.01392        0.863        0.917
##   231    329       1    0.887 0.01413        0.860        0.915
##   233    325       2    0.881 0.01456        0.853        0.910
##   245    295       1    0.878 0.01482        0.850        0.908
##   255    279       1    0.875 0.01510        0.846        0.905
##   298    146       1    0.869 0.01614        0.838        0.901
##
##                tx=1
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     7    572       1    0.998 0.00175        0.995        1.000
##    13    571       1    0.997 0.00247        0.992        1.000
##    14    570       2    0.993 0.00348        0.986        1.000
##    17    568       1    0.991 0.00389        0.984        0.999
##    18    566       2    0.988 0.00460        0.979        0.997
##    20    563       1    0.986 0.00491        0.976        0.996
##    25    562       1    0.984 0.00521        0.974        0.995
##    35    561       1    0.982 0.00549        0.972        0.993
##    39    559       1    0.981 0.00575        0.970        0.992
##    44    554       1    0.979 0.00601        0.967        0.991
##    46    553       1    0.977 0.00625        0.965        0.990
##    47    550       1    0.975 0.00649        0.963        0.988
##    58    543       1    0.974 0.00672        0.961        0.987
##    65    539       1    0.972 0.00695        0.958        0.986
##    81    528       1    0.970 0.00717        0.956        0.984
##    82    526       1    0.968 0.00739        0.954        0.983
##    85    524       1    0.966 0.00761        0.951        0.981
##    91    519       1    0.964 0.00782        0.949        0.980
##   103    514       1    0.963 0.00802        0.947        0.978
##   105    509       1    0.961 0.00823        0.945        0.977
##   114    504       1    0.959 0.00843        0.942        0.975
##   117    501       1    0.957 0.00863        0.940        0.974
##   127    492       1    0.955 0.00883        0.938        0.972
##   138    482       1    0.953 0.00903        0.935        0.971
##   144    477       1    0.951 0.00923        0.933        0.969
##   174    447       1    0.949 0.00945        0.930        0.967
##   226    375       1    0.946 0.00976        0.927        0.966
##   244    338       1    0.943 0.01012        0.924        0.963
##   248    334       1    0.941 0.01048        0.920        0.961
##   266    272       1    0.937 0.01099        0.916        0.959
##   288    196       1    0.932 0.01193        0.909        0.956
```

The separate Kaplan-Meier curves for each treatment group are reported one after the other. Note that the time intervals for each group are different, since each curve is fit using only the data (and intervals) from that group.

Again, we can use the `plot()` function. Since the default is for both lines to be solid and black, we can further customize it, to color each line separately, add informative labels, and provide a legend. *For the purposes of labeling the legend, note that the groups are in the same order as they appear in* `summary(actgKM.groups)`.

```
plot(actgKM.tx, lty=c(2,1),
     xlab = "Time from randomization (days)", ylab = "Survival probability",
     main = "AIDS-free survival")
legend("bottomright", legend=c("Without indinavir","With indinavir"),
      lty=c(2,1))
```



Log-rank testing

Standard log-rank test

The basic function for performing the log-rank test is `survdiff()`. Just like `survfit()`, we specify the censored failure time variable on the left of the model statement with the syntax `Surv(time,event)`. On the right of the model statement, we specify the binary covariate that defines the two groups we want to compare. We can use the `summary()` function to examine our output.

We continue with our `actg320` example, comparing the two treatment groups (with and without indinavir).

```
survdiff(Surv(time, censor) ~ tx, data=actg320)
```

```
## Call:
## survdiff(formula = Surv(time, censor) ~ tx, data = actg320)
##
##        N Observed Expected (O-E)^2/E (O-E)^2/V
## tx=0 577       63     47.1      5.37      10.5
## tx=1 574       33     48.9      5.17      10.5
##
##  Chisq= 10.5  on 1 degrees of freedom, p= 0.001
```

In this analysis, 577 patients received standard treatment, and there were 63 primary events observed. In contrast, 574 patients received treatment with indinavir, and there were 33 primary events observed. Conditioning on the total number of events (96), under the null hypothesis, the expected numbers of events are similar across the two groups because of the similar sample sizes.

The chi-squared test statistic is 10.5 with associated p-value of 0.001. Thus, there is a statistically significant difference in AIDS-free survival across the two groups. Patients receiving indinavir had longer AIDS-free survival than patients receiving standard treatment.

Weighted log-rank test

The `survdiff()` function allows one to specify a parameter `rho` that controls the weighting of the test. The default `rho=0` returns the standard log-rank test. Setting `rho=1` returns the Peto-Prentice test. This uses a slightly different weighting scheme than the generalized Wilcoxon test, but it operates similarly in that it places more weight on earlier observations.

```
survdiff(Surv(time, censor) ~ tx, data=actg320, rho=1)
```

```
## Call:
## survdiff(formula = Surv(time, censor) ~ tx, data = actg320, rho = 1)
##
##        N Observed Expected (O-E)^2/E (O-E)^2/V
## tx=0 577     60.1     45.1      5.02      10.3
## tx=1 574     31.7     46.8      4.84      10.3
##
##  Chisq= 10.3  on 1 degrees of freedom, p= 0.001
```

The results are very similar. The chi-squared test statistic is slightly lower (10.3), but the associated p-value is the same.

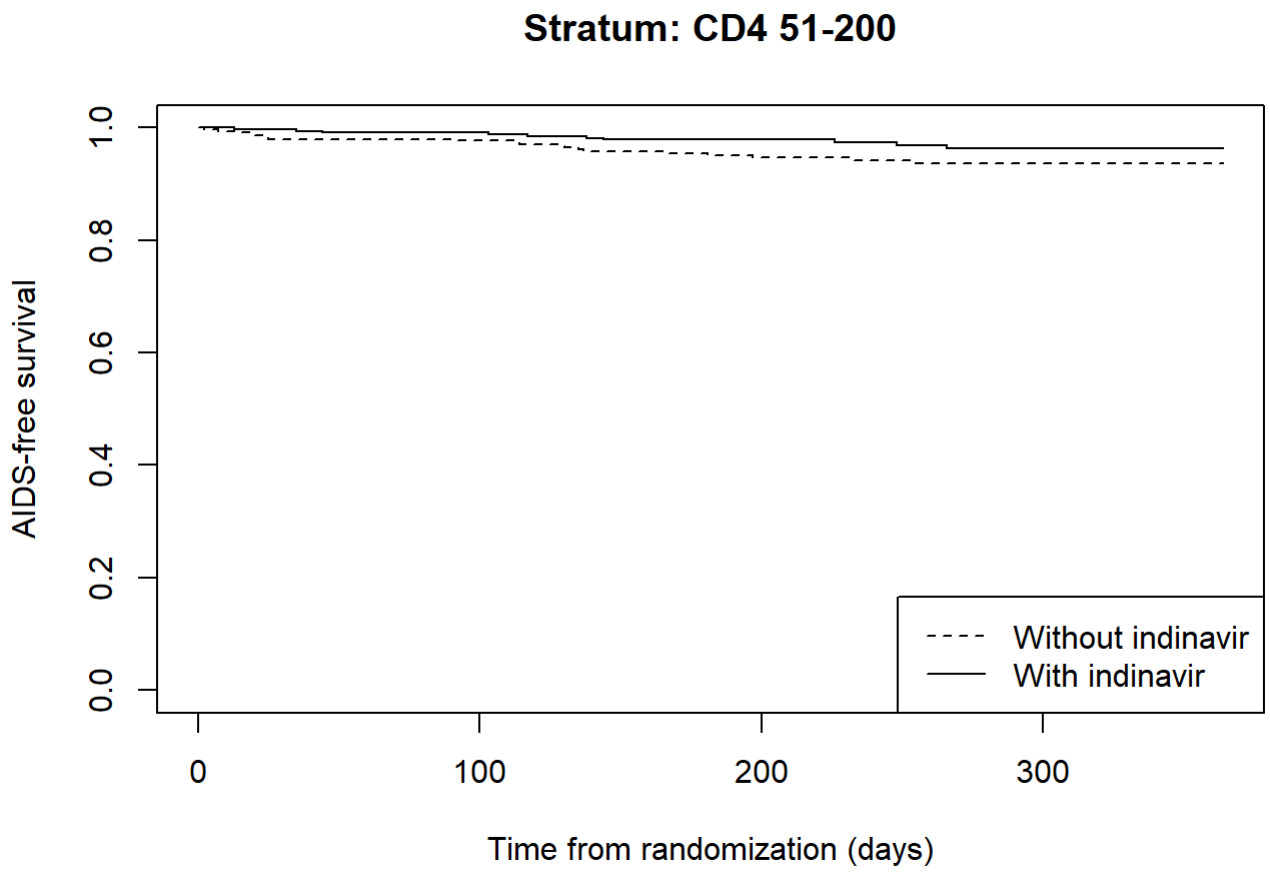
For those interested, the package `survmisc` includes a function `comp()` (for comparing two survival curves) that allows users to specify any of the available weighting schemes, including the generalized Wilcoxon weights.

Stratified log-rank test

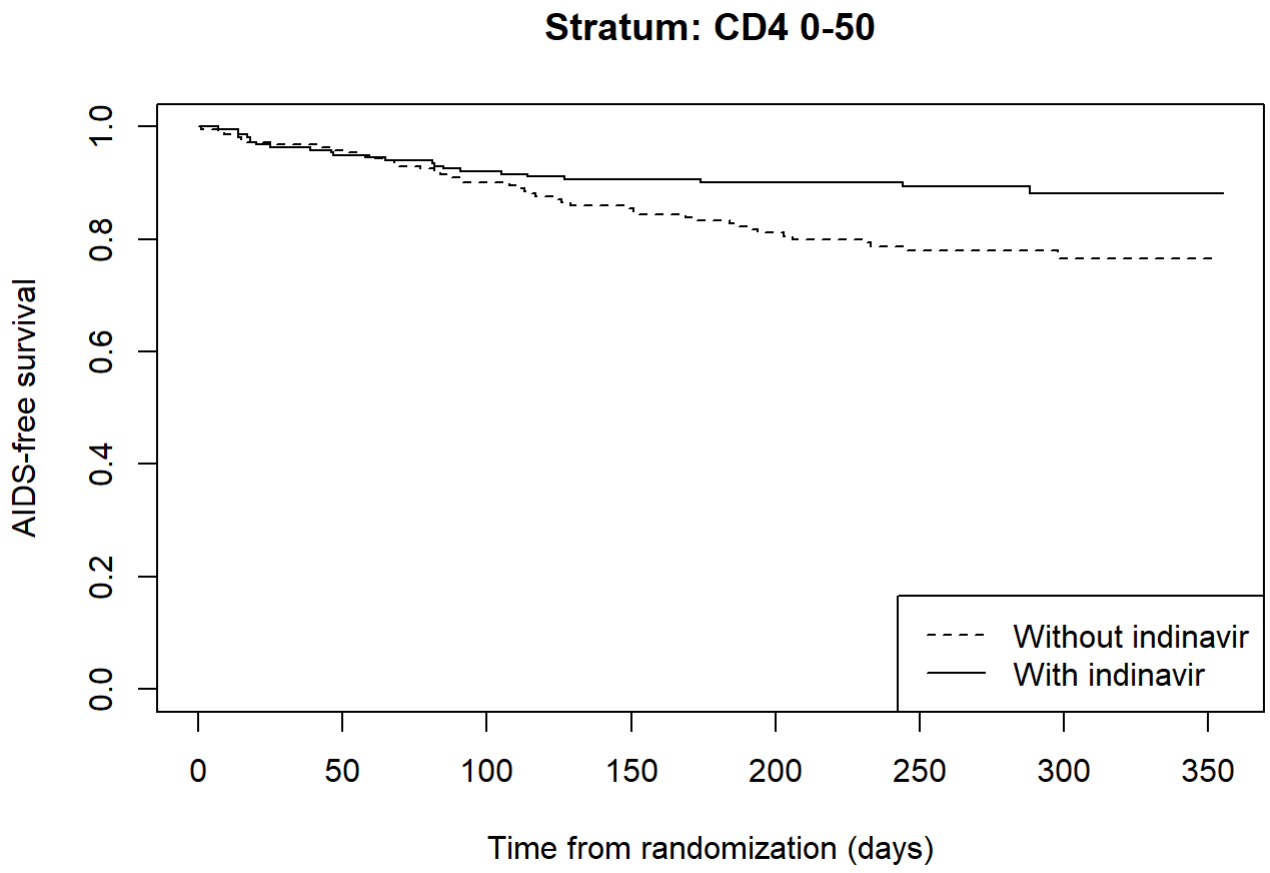
A key prognostic variable for patients living with HIV is CD4 cell count. CD4 cell count is a measure of immune functioning, with lower count indicating poorer immune status. The CD4 cell count of a person who does not have HIV can be anything between 500 and 1500. People living with HIV who have a CD4 count over 500 are usually in pretty good health. People living with HIV who have a CD4 cell count below 200 are at high risk of developing serious illnesses.

Two strata are defined based on CD4 cell count at baseline: `cd4_grp==1` is for individuals with CD4 cell count of 51-200; `cd4_grp==0` is for individuals with CD4 cell count of 50 or lower. Using the `subset()` function, we fit Kaplan-Meier curves within each CD4 stratum.

```
plot(survfit(Surv(time,censor) ~ tx, data=subset(actg320,cd4_grp==1)), lty=c(2,1),
     xlab="Time from randomization (days)", ylab="AIDS-free survival", main="Stratum: CD4 51-20
0")
legend("bottomright",legend=c("Without indinavir","With indinavir"),lty=c(2,1))
```




```
plot(survfit(Surv(time,censor) ~ tx, data=subset(actg320,cd4_grp==0)), lty=c(2,1),
     xlab="Time from randomization (days)", ylab="AIDS-free survival", main="Stratum: CD4 0-50")
legend("bottomright",legend=c("Without indinavir","With indinavir"),lty=c(2,1))
```



Examining the plots, we see better overall survival for the group with higher CD4, confirming the prognostic value of CD4. Within strata, we see improved survival for the group receiving indinavir. This difference is more pronounced for the group with CD4 ≤ 50.

To fit a log-rank test stratified on CD4 group, we use the `survdiff()` function and specify stratification by adding `+ strata(cd4_grp)` to the model statement after the `tx` variable.

```
survdiff(Surv(time, censor) ~ tx + strata(cd4_grp), data=actg320)
```

```
## Call:
## survdiff(formula = Surv(time, censor) ~ tx + strata(cd4_grp),
##     data = actg320)
##
##         N Observed Expected (O-E)^2/E (O-E)^2/V
## tx=0 577       63     46.9      5.55      10.9
## tx=1 574       33     49.1      5.29      10.9
##
##  Chisq= 10.9  on 1 degrees of freedom, p= 0.001
```

The chi-squared test statistic is 10.9, which is very similar to the standard test. Recall that this is a randomized trial, so baseline CD4 cell count is well-balanced across the two treatment groups. The test statistic is slightly higher because some of the variability in patient prognosis is explained by CD4 subgroup, slightly improving precision.

End of tutorial