Survival Analysis in R

Wang Ye

Authors:

- Wang Ye 1589936809@qq.com
- Wang Hao

In collaboration of:

1. Nanjing Drum Tower Hospital, China Pharmaceutical University, Nanjing, China
2. Department of Pharmacy, Drum Tower Hospital Affiliated to Medical School of Nanjing University, Nanjing, China

This Code adaptation Developed by the Decision Analysis in R for Technologies in Health (DARTH) workgroup.

Preparation

Below we load the R packages required to conduct the analysis

We have created an R file with the custom functions created by our group that are specific to survival analysis in decision modeling. Briefly there are two main functions. `fit.fun` is the function that fits a number of parametric survival models to the data, estimates the Akaike and bayesian information criterion and stores all the output on a list of survival models and their goodness of fit. The code below reads in the function file with survival fitting, partitioned survival and microsimulation functions. Information on using the functions in the `SurvFunctions_final.R` file can be found in each function.

```r
source("SurvFunctions_final.R")
```

Next we read in the simulated dataset. This dataset contains 140 patients with their overall survival time (variable 'os') and progression free survival time (variable 'pfs') as well as whether they experienced mortality (variable 'event.os') or progression (variable 'event.pfs') with administrative censoring at 5 years. The first 6 observations of the dataset are displayed below:

The patient is censored for a particular event if the patient's event variable ('event.os' or 'event.pfs') = 0.

```r
dataA      <- read.csv('data.treat_A.csv', header = T)
dataB      <- read.csv('data.treat_B.csv', header = T)
head_dataA <- head(dataA)
head_dataB <- head(dataB)
kable(head_dataA)
```

id	os	pfs	event.os	event.pfs
Patient 1	0.185	0.572	1	1
Patient 2	0.185	0.572	1	1
Patient 3	1.229	1.287	1	1
Patient 4	3.018	1.602	1	1
Patient 5	4.013	1.602	1	1
Patient 6	4.013	1.602	1	1

id	os	pfs	event.os	event.pfs

```
kable(head_dataB)
```

id	os	pfs	event.os	event.pfs
Patient 1	0.185	1.287	1	1
Patient 2	2.670	1.287	1	1
Patient 3	3.018	1.287	1	1
Patient 4	3.018	1.602	1	1
Patient 5	4.013	1.602	1	1
Patient 6	4.013	1.602	1	1

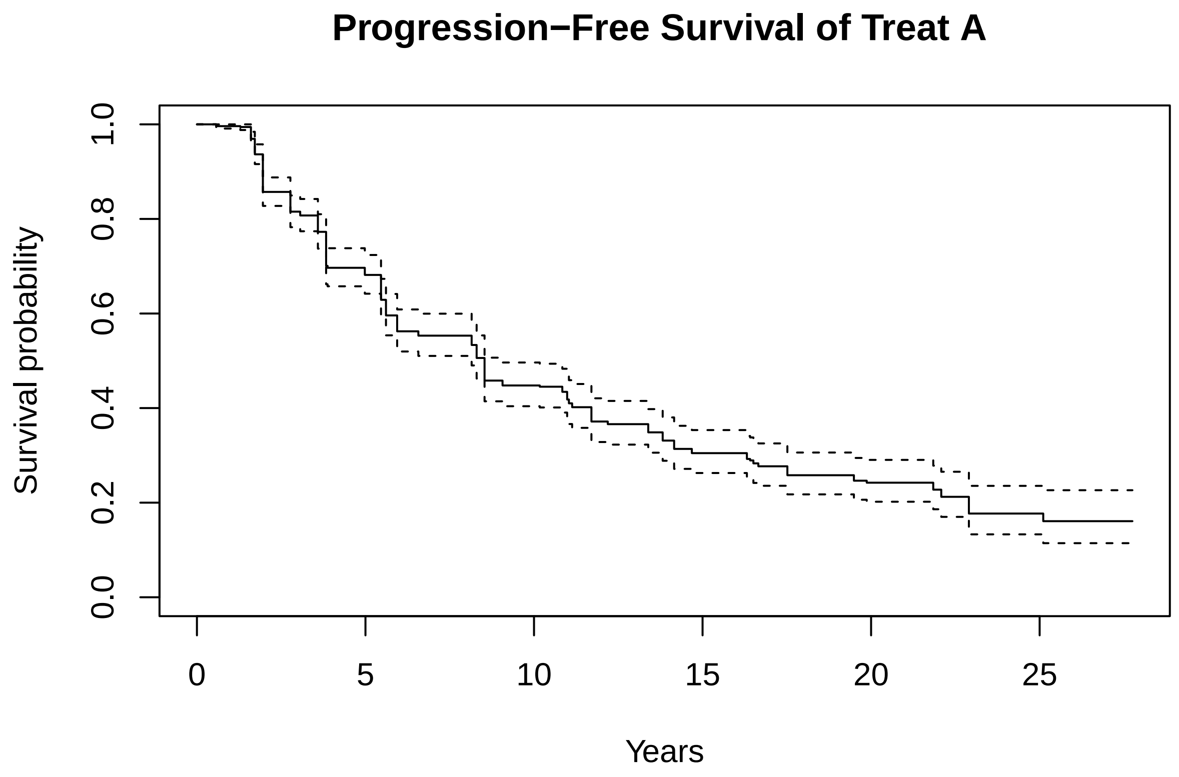
In this report, for simplicity, the state names ‘Progression-free’ and ‘stable’ are interchangeable, ‘Progression’ and ‘progressed’ are interchangeable, and ‘Death’ and ‘Dead’ are interchangeable.

Method No.1: Partitioned Survival Model

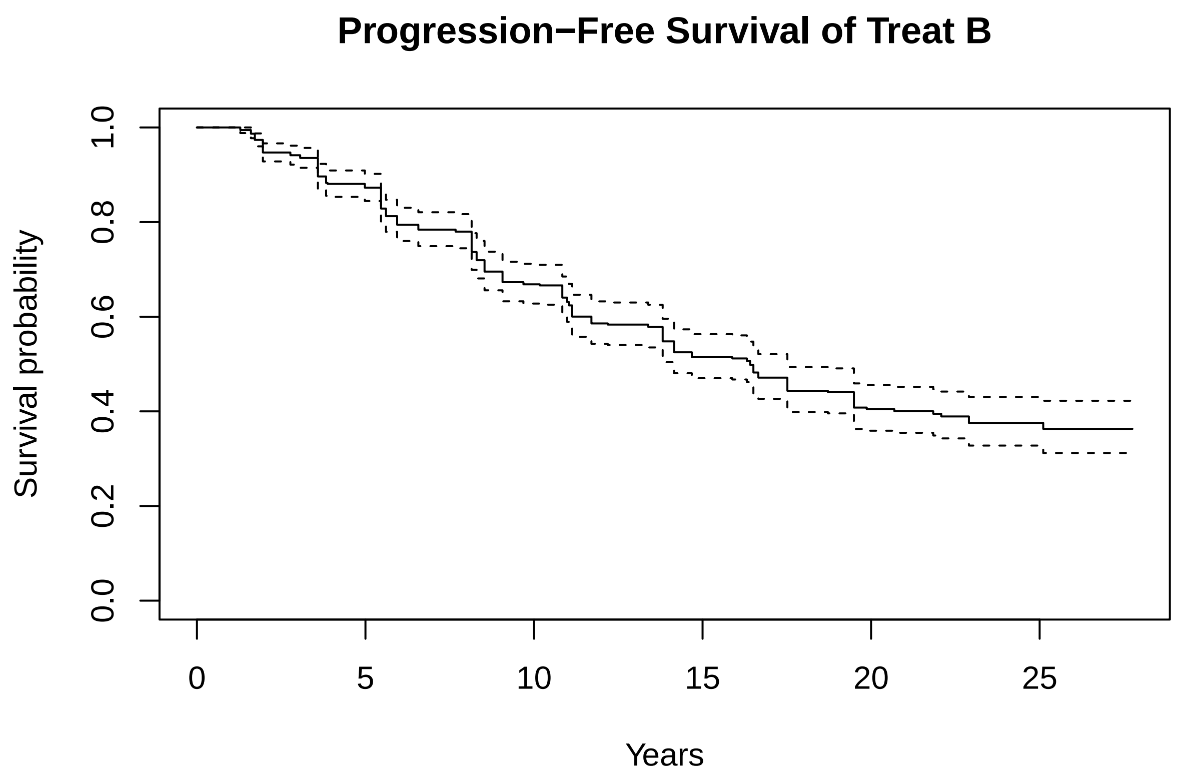
Step 1 : Descriptive statistics to the survival data - treatment arm

The function below draws a Kaplan-Meier plot for progression-free survival (PFS):

```
KM.pfsA <- survfit(Surv(time = pfs, event = event.pfs) ~ 1, data = dataA)
KM.pfsB <- survfit(Surv(time = pfs, event = event.pfs) ~ 1, data = dataB)
# Kaplan-Meier fit for PFS
plot(KM.pfsA, main = paste("Progression-Free Survival of Treat A "), ylab = "Survival probability", xlal
```

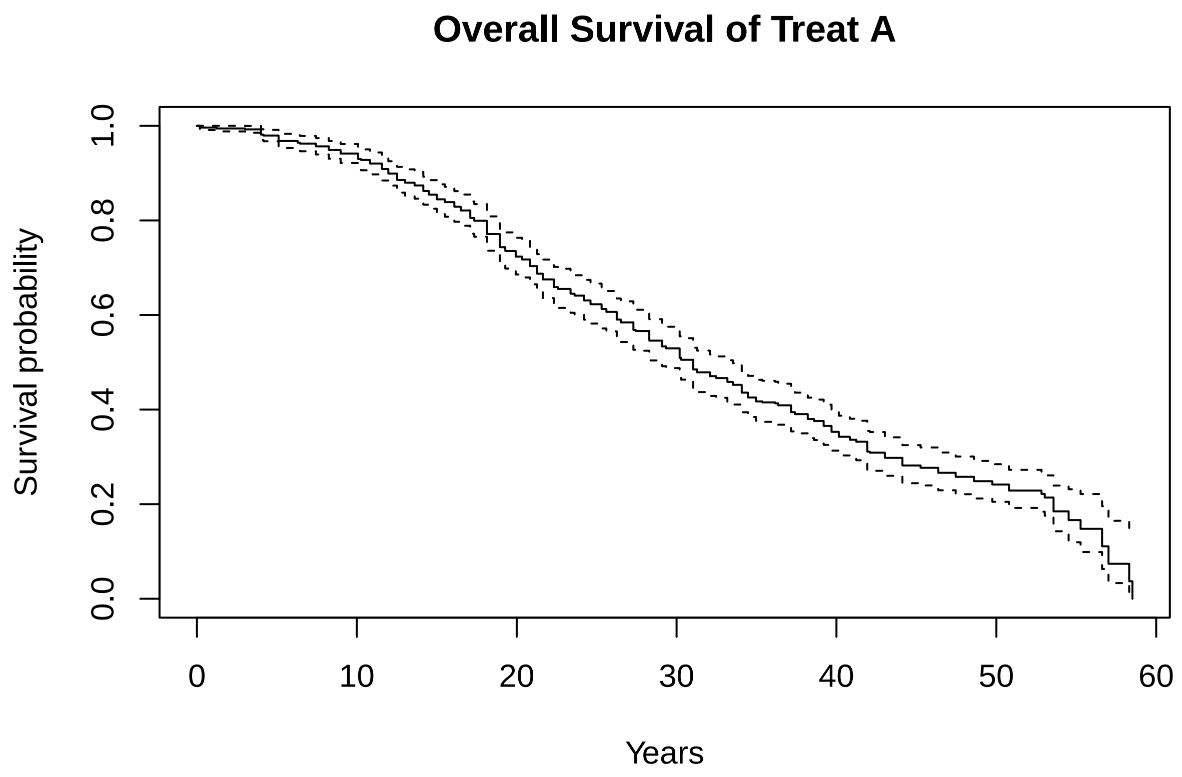


```
plot(KM.pfsB, main = paste("Progression-Free Survival of Treat B"), ylab = "Survival probability", xlab
```

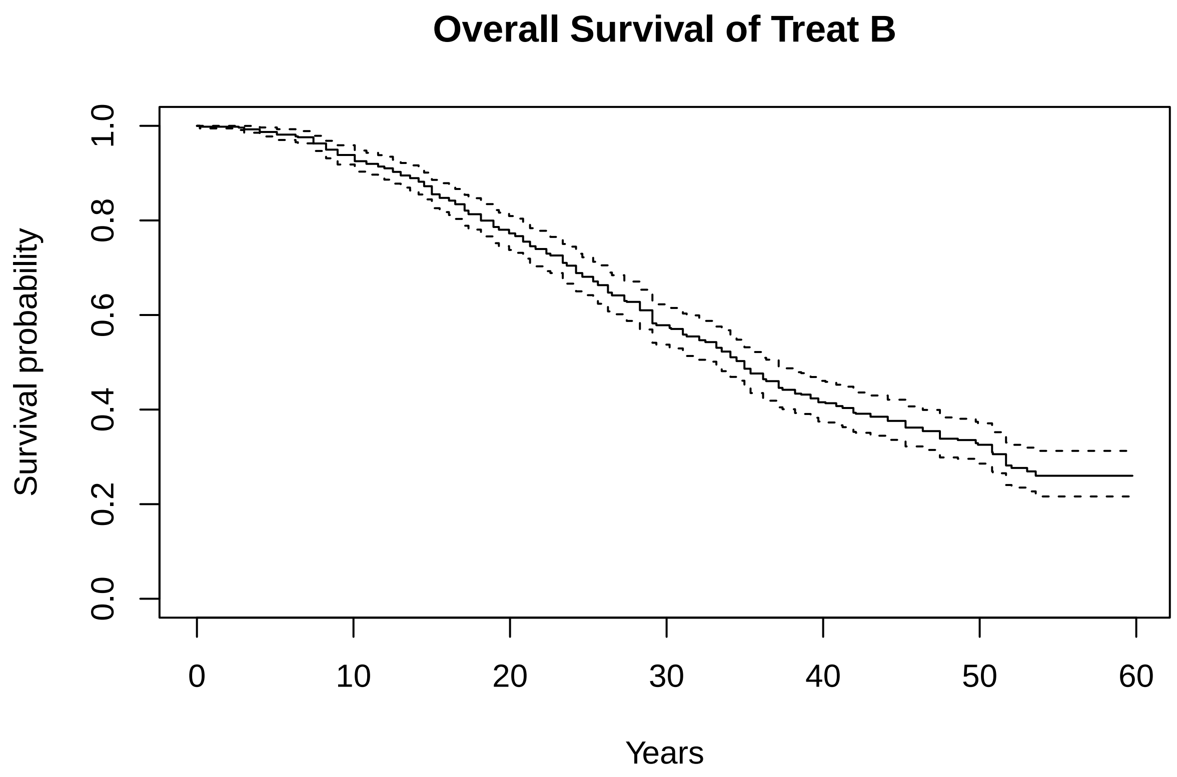


Similarly we can draw a Kaplan-Meier plot for overall survival (OS):

```
KM.osA  <- survfit(Surv(time = os,  event = event.os)  ~ 1, data = dataA) #
KM.osB  <- survfit(Surv(time = os,  event = event.os)  ~ 1, data = dataB) #Kaplan-Meier fit for OS
plot( KM.osA,  main = paste("Overall Survival of Treat A"),   ylab = "Survival probability", xlab = "Ye
```



```
plot( KM.osB,  main = paste("Overall Survival of Treat B"),   ylab = "Survival probability", xlab = "Yea
```



Step 2 : Fit different parametric survivor functions to the data

First the following parameters that relate to the partitioned survival model (PSM) are defined:

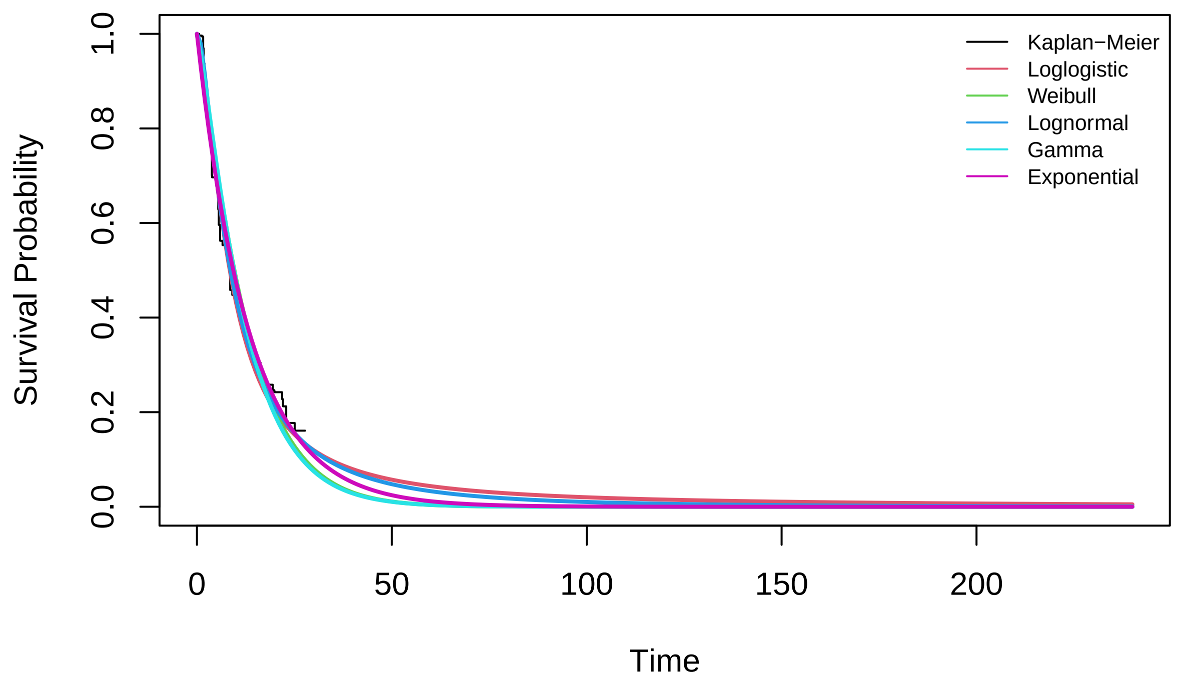
```
### Strategies
v_names_str     <- c("Treatment A",
                     "Treatment B")
n_str           <- length(v_names_str)           # number of strategies

v.n            <- c("Stable","Progressed","Dead") # state names
n.s            <- length(v.n)     # number of states
n.t            <- 240              # number of cycles to run
c.l            <- 1        # cycle length (a month)
times  <- seq(from = 0, to = n.t, by = c.l)  # sequence of times to be considered in the model
```

The parametric survival functions fitted are:

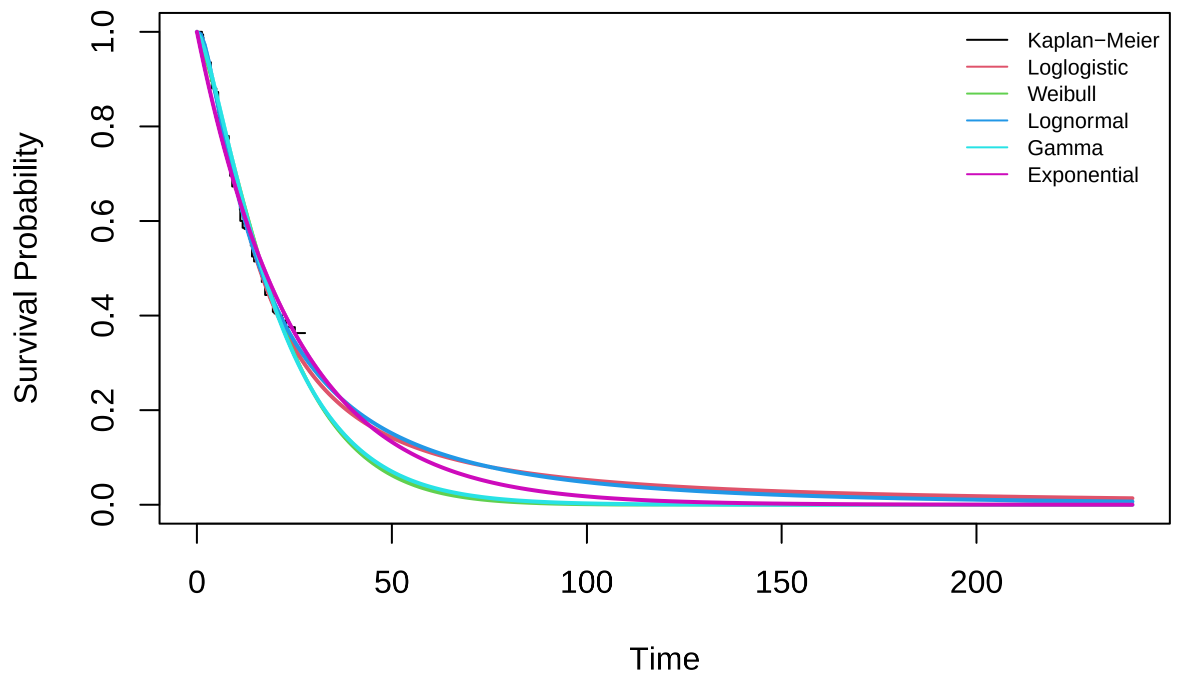
- the log-logistic survivor function
- the Weibull survivor function
- the log-normal survivor function
- the Gamma survivor function
- the Exponential survivor function

For PFS:

```
fit.pfsA  <- fit.fun(time = "pfs", event = "event.pfs", data = dataA)
```

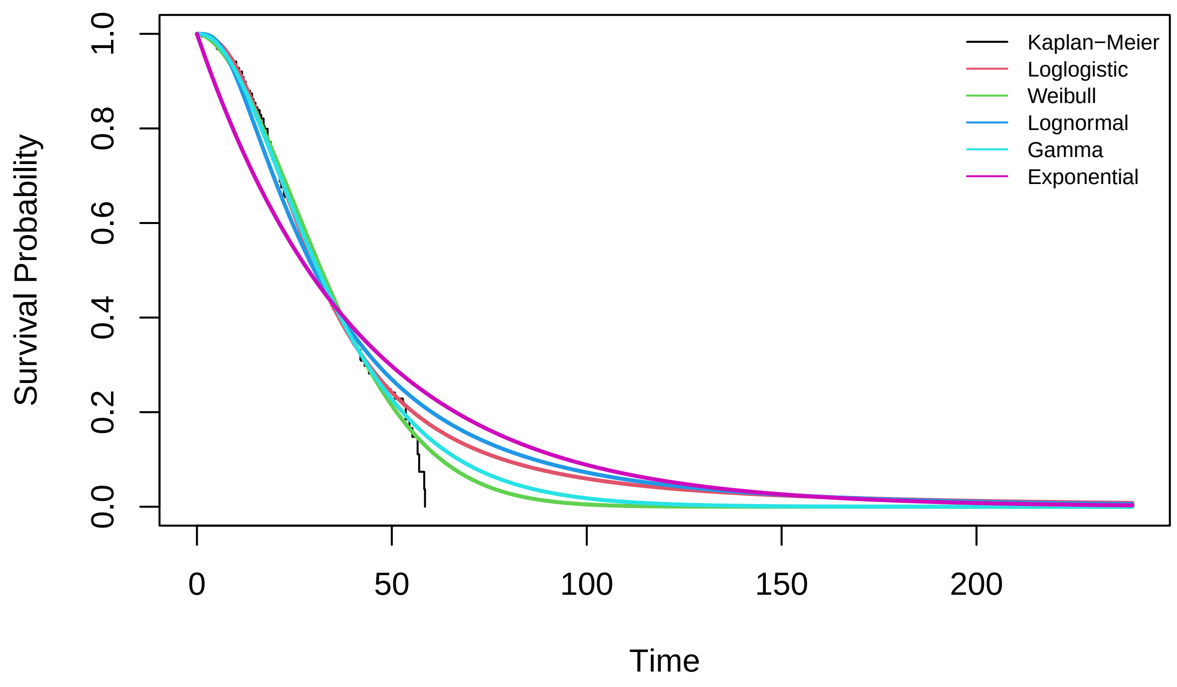


```
fit.pfsB  <- fit.fun(time = "pfs", event = "event.pfs", data = dataB)
```

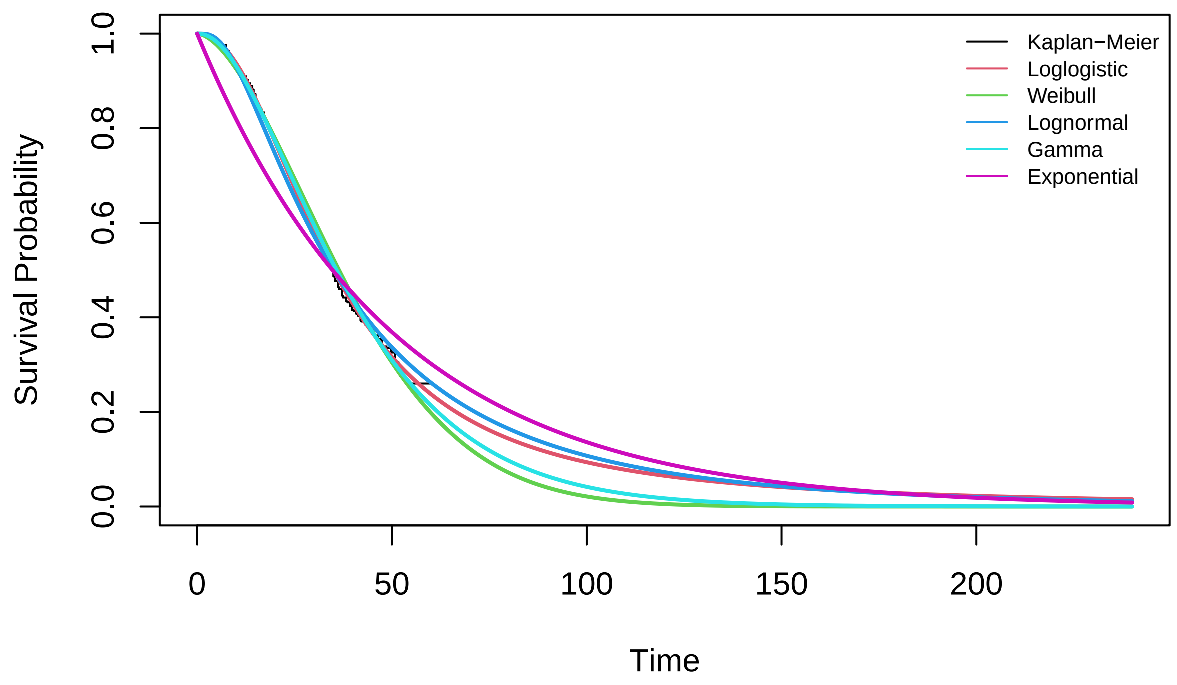


For OS:

```
fit.osA  <- fit.fun( time = "os", event = "event.os", data = dataA)
```



```
fit.osB  <- fit.fun( time = "os", event = "event.os", data = dataB)
```

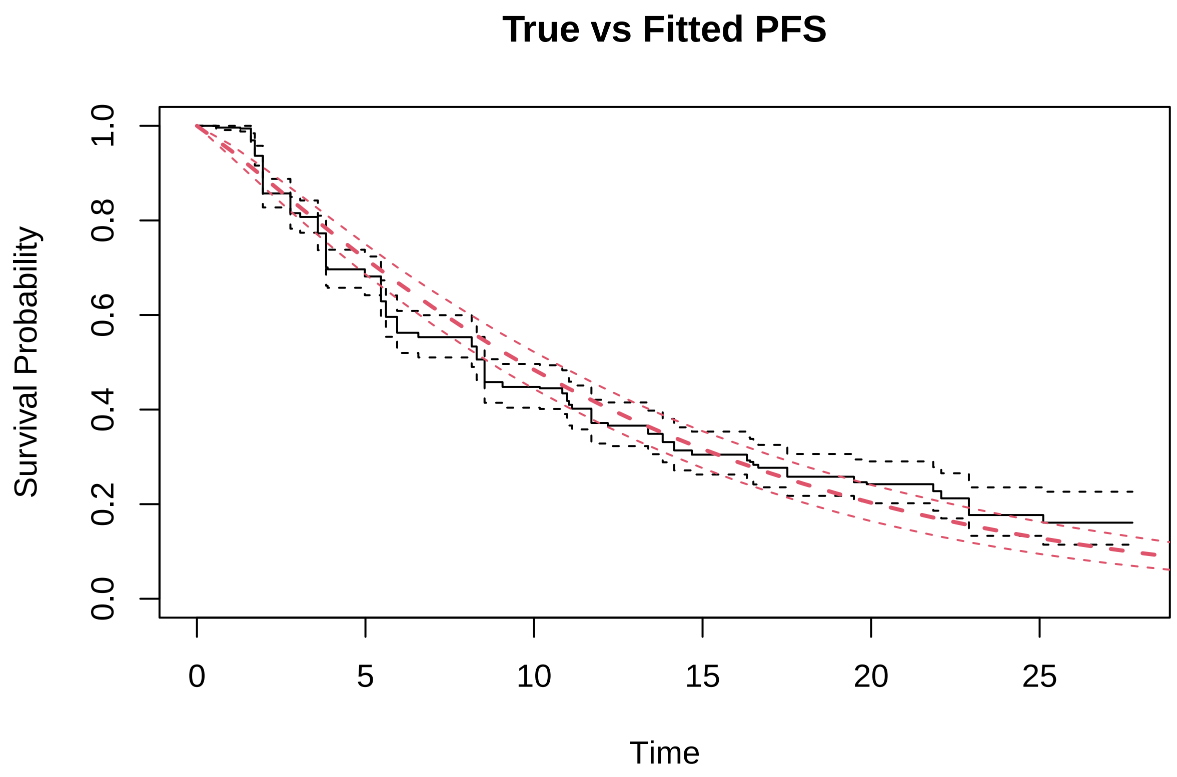


Step 3: Select the best fitting survivor function and compare it against the Kaplan-Meier curve

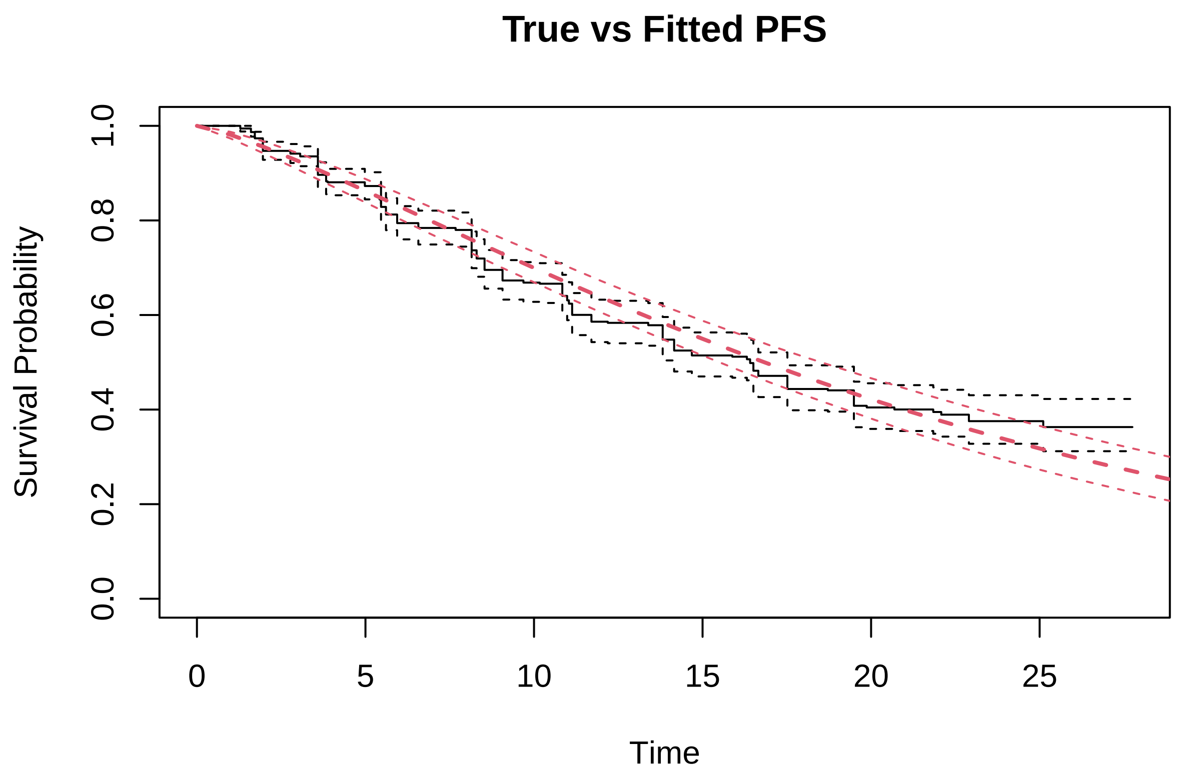
For PFS:

In this example the Weibull survivor function hsa been identified as the best-fitting function.

```
best.pfsA <- fit.pfsA$Weibull
plot(KM.pfsA, ylab = "Survival Probability", xlab = "Time",  main = paste ("True vs Fitted PFS"))
lines(best.pfsA,  col = 2, t = times, lty = 2)
```



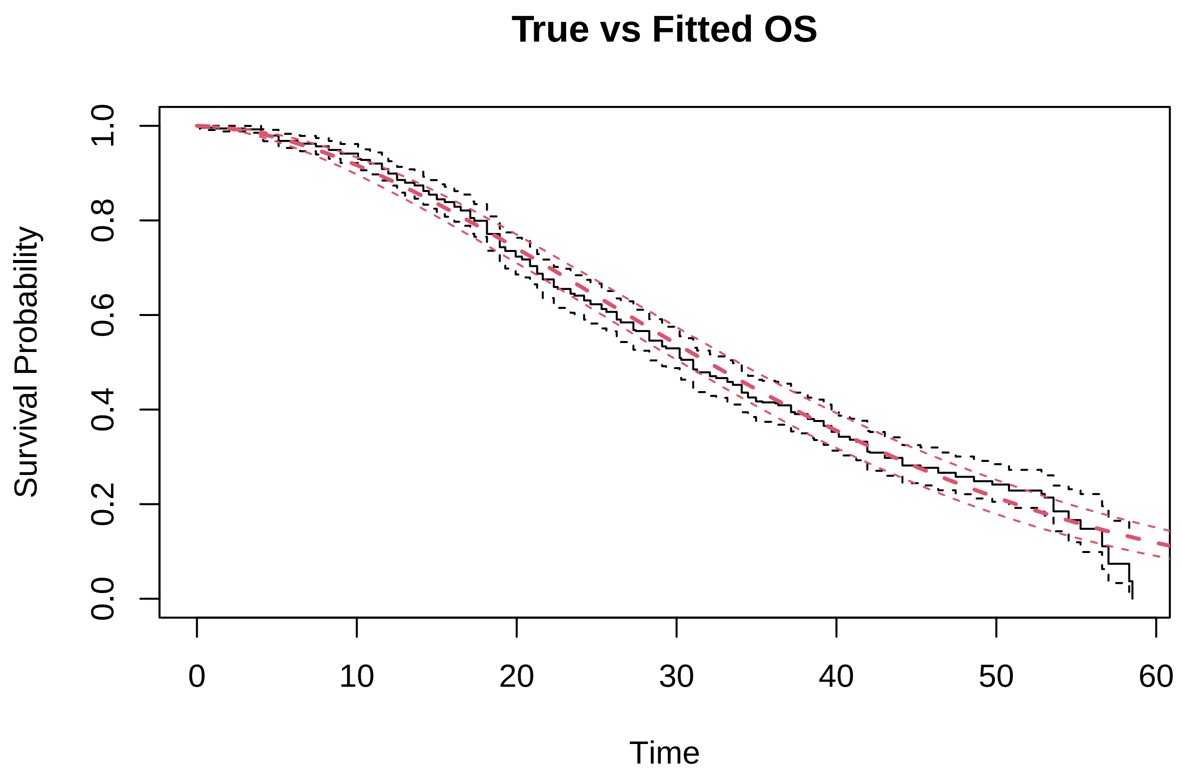
```
best.pfsB <- fit.pfsB$Weibull
plot(KM.pfsB, ylab = "Survival Probability", xlab = "Time",  main = paste ("True vs Fitted PFS"))
lines(best.pfsB,  col = 2, t = times, lty = 2)
```



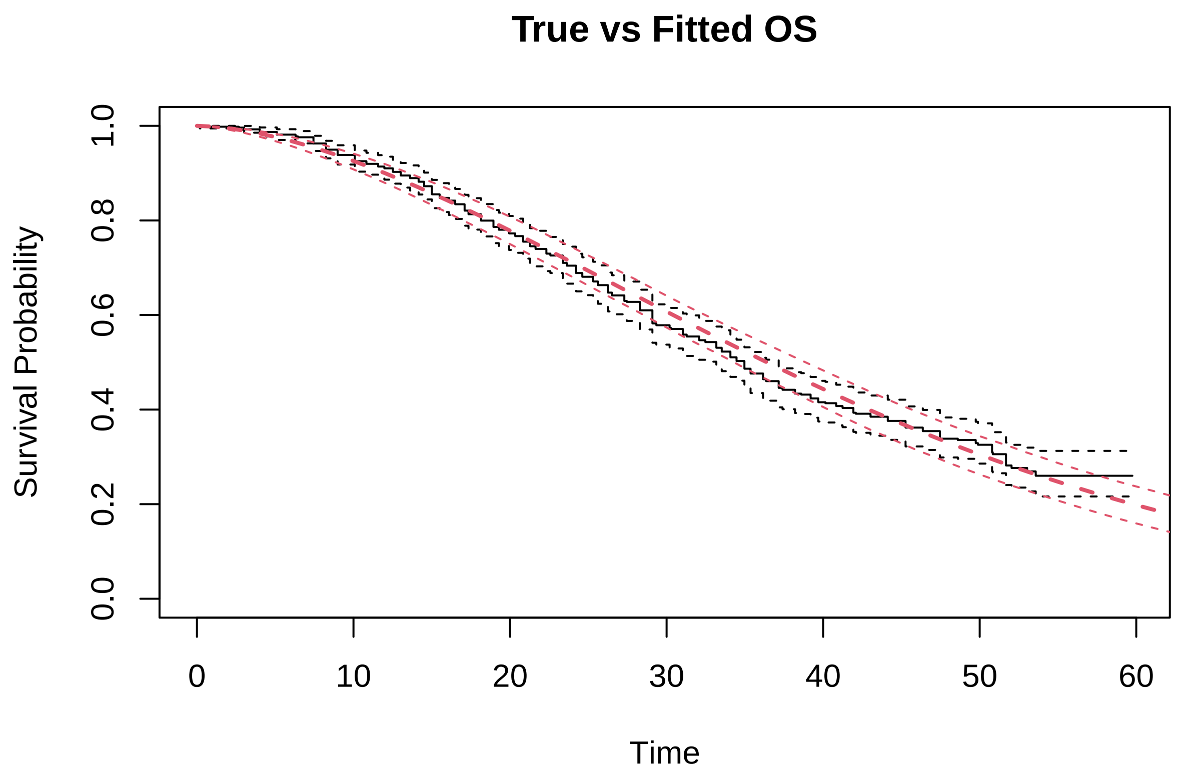
For OS:

In this example The Weibull survivor function was identified as the best-fitting function.

```
best.osA <- fit.osA$Weibull
plot(KM.osA, ylab = "Survival Probability", xlab = "Time",  main = c("True vs Fitted OS"))
lines(best.osA,  col = 2, t = times, lty = 2)
```



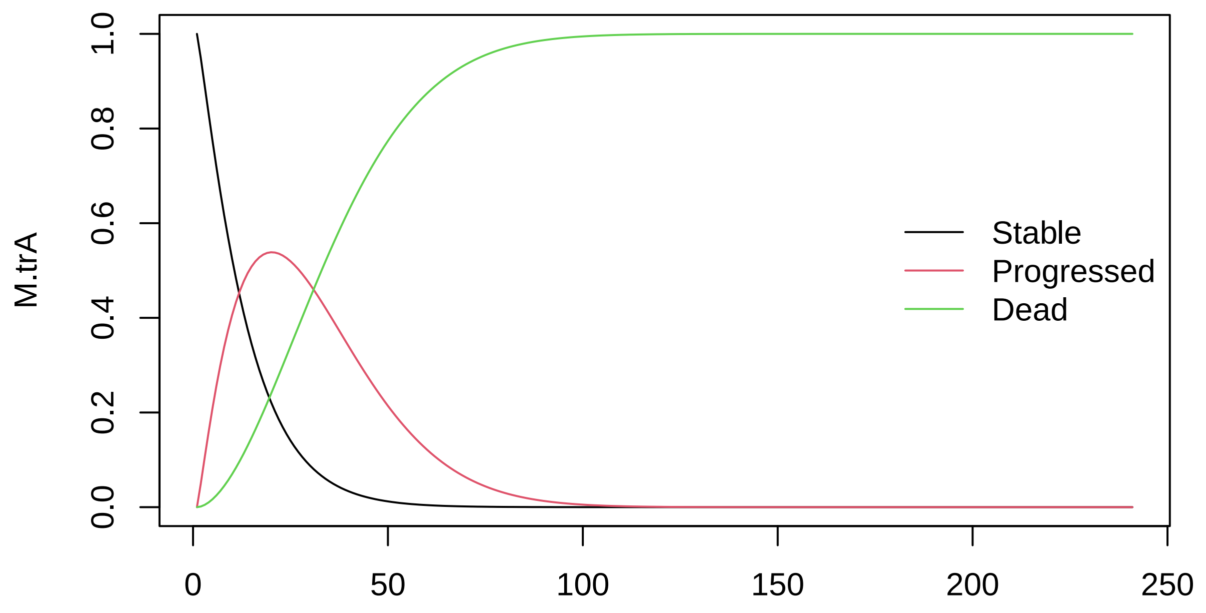
```
best.osB <- fit.osB$Weibull
plot(KM.osB, ylab = "Survival Probability", xlab = "Time",  main = c("True vs Fitted OS"))
lines(best.osB,  col = 2, t = times, lty = 2)
```



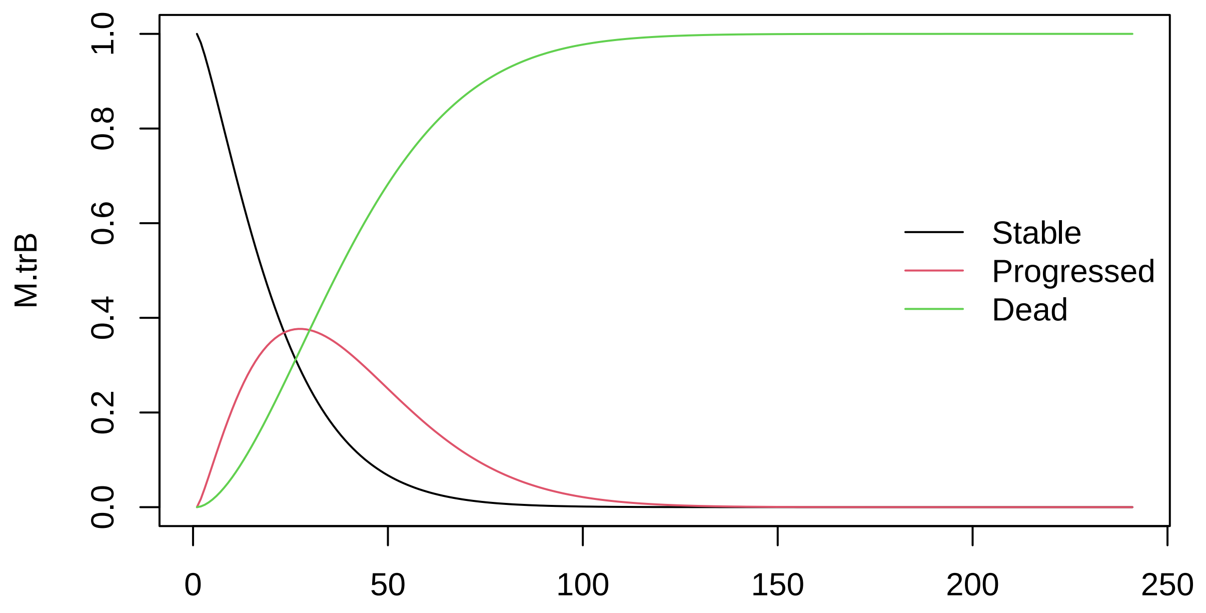
Step 4: Fit partitioned survival model and plot the Markov trace

Once the best fitting models for OS and PFS have been identified we can use the function partsurv which uses these two models as input, extracts the coefficients from these models and calculates the survival probability for a sepcified time horizon. Subsequently, it allocates the cohort across the three states on the basis of the probabilities of occupying each state. The output of the function is a markov trace, whcih we plot in the section below.

```
survA <-partsurv(best.pfsA, best.osA, title = "all", time= times)

M.trA <- as.matrix(survA$trace)
matplot(M.trA, type = 'l', lty=1)
legend("right", v.n, col=1:n.s, lty=rep(1,n.s), bty='n')
```



```
survB <-partsurv(best.pfsB, best.osB, title = "all", time= times)

M.trB <- as.matrix(survB$trace)
matplot(M.trB, type = 'l', lty=1)
legend("right", v.n, col=1:n.s, lty=rep(1,n.s), bty='n')
```



```
## Store the cohort traces in a list
l_m_M <- list(A   =  M.trA,
              B   =  M.trB)
names(l_m_M) <- v_names_str
write.csv(M.trA, file = "M.trA.csv",row.names = F)
write.csv(M.trB, file = "M.trB.csv",row.names = F)
```