Simple 3-state Partitioned Survival model in R

The DARTH workgroup

Developed by the Decision Analysis in R for Technologies in Health (DARTH) workgroup:

Fernando Alarid-Escudero, PhD (1)

Eva A. Enns, MS, PhD (2)

M.G. Myriam Hunink, MD, PhD (3,4)

Hawre J. Jalal, MD, PhD (5)

Eline M. Krijkamp, MSc (3)

Petros Pechlivanoglou, PhD (6)

Alan Yang, MSc (7)

In collaboration of:

1. Drug Policy Program, Center for Research and Teaching in Economics (CIDE) - CONACyT, Aguascalientes, Mexico
2. University of Minnesota School of Public Health, Minneapolis, MN, USA
3. Erasmus MC, Rotterdam, The Netherlands
4. Harvard T.H. Chan School of Public Health, Boston, USA
5. University of Pittsburgh Graduate School of Public Health, Pittsburgh, PA, USA
6. The Hospital for Sick Children, Toronto and University of Toronto, Toronto ON, Canada
7. The Hospital for Sick Children, Toronto ON, Canada

Please cite our publications when using this code:

- Jalal H, Pechlivanoglou P, Krijkamp E, Alarid-Escudero F, Enns E, Hunink MG. An Overview of R in Health Decision Sciences. Med Decis Making. 2017; 37(3): 735-746. https://journals.sagepub.com/doi/abs/10.1177/0272989X16686559
- Krijkamp EM, Alarid-Escudero F, Enns EA, Jalal HJ, Hunink MGM, Pechlivanoglou P. Microsimulation modeling for health decision sciences using R: A tutorial. Med Decis Making. 2018;38(3):400–22. https://journals.sagepub.com/doi/abs/10.1177/0272989X18754513
- Krijkamp EM, Alarid-Escudero F, Enns E, Pechlivanoglou P, Hunink MM, Jalal H. A Multidimensional Array Representation of State-Transition Model Dynamics. BioRxiv 670612 2019.https://www.biorxiv.org/content/10.1101/670612v1

Change `eval` to `TRUE` if you want to knit this document.

01 Load packages

```
if (!require('pacman')) install.packages('pacman'); library(pacman) # use this package to conveniently
# load (install if required) packages from CRAN
p_load("here", "dplyr", "devtools", "gems", "flexsurv", "flexsurvcure", "survminer", "survHE", "ggplot2
# install_github("DARTH-git/darthtools", force = TRUE) # Uncomment if there is a newer version
p_load_gh("DARTH-git/darthtools")
```

02 Load functions

```
# No function needed
```

03 Input model parameters

```
v_names_states <- c("Healthy", "Sick", "Dead")  # state names

c_l       <- 1 / 12                 # cycle length (a month)
n_t       <- 10                     # number of years (10 years)
set.seed(2020)                      # set the seed
n_sim     <- 100                    # number of simulations

n_states  <- length(v_names_states)  # No of states
times     <- seq(0, n_t, c_l)        # the cycles in years
```

Create a transition probability matrix with all transitions indicated and numbered.

```
tmat <- matrix(NA, n_states, n_states, dimnames = list(v_names_states, v_names_states))
tmat["Healthy", "Sick"]  <- 1
tmat["Healthy", "Dead"]  <- 2
tmat["Sick"   , "Dead"]  <- 3

layout.fig <- c(2,1)
plotmat(t(tmat), t(layout.fig), self.cex = 0.5, curve = 0, arr.pos = 0.76,
        latex = T, arr.type = "curved", relsize = 0.85, box.prop=0.8,
        cex = 0.1, box.cex = 0.7, lwd = 1)
```

Generate data.

```
n_pat       <- 550                      # cohort size
n_years     <- 30                       # number of years
generate    <- gen_data(n_pat, n_years)  # generates true, censored and OS/PFS data
OS_PFS_data <- generate$OS_PFS_data      # store the OS / PFS structured data
head(OS_PFS_data)
```

04 Analysis

Showcasing the use of packages `survival`, `flexsurv`.

```
fit_KM_OS   <- survfit(Surv(time = OS_time, event = OS_status) ~ 1, data = OS_PFS_data)
plot(fit_KM_OS, mark.time = T)

# a prettier way of plotting!!
ggsurvplot(
  fit_KM_OS,
  data = OS_PFS_data,
  size = 1,                 # change line size
  palette = c("blue3"),     # custom color palettes
  conf.int = TRUE,          # Add confidence interval
  pval = TRUE,              # Add p-value
  risk.table = TRUE,        # Add risk table
  risk.table.height = 0.25, # Useful to change when you have multiple groups
  ggtheme = theme_bw(),     # Change ggplot2 theme
  xlab = 'Time in years',   # Change X-axis label
  title    = "Survival curve for Overall Survival (OS)",
  subtitle = "Based on Kaplan-Meier estimates"
)
```

Cumulative hazards:

```
# plot cumulative hazards
ggsurvplot(
  fit_KM_OS,
  data = OS_PFS_data,
  size = 1,                 # change line size
  palette = c("blue3"),     # custom color palettes
  conf.int = TRUE,          # Add confidence interval
  pval = FALSE,             # Add p-value
  risk.table = TRUE,        # Add risk table
  risk.table.height = 0.25, # Useful to change when you have multiple groups
  ggtheme = theme_bw(),     # Change ggplot2 theme
  xlab = 'Time in years',   # Change X-axis label
  ylab = 'Cumulative hazard',
  title    = "Cumulative hazard for Overall Survival (OS)",
  subtitle = "Based on Nelson-Aalen estimates",
  fun = "cumhaz"
)
```

04.1 Survival Analysis

Fit different parametric survival models and choose the best fitting ones.

```
# R package flexsurv allows parametric fitting of curves
fit_weib <- flexsurvreg(Surv(time = OS_time, event = OS_status) ~ 1, data = OS_PFS_data,
                        dist = "weibull")
plot(fit_weib)
```

We developed a function `fit.fun` that allows you to fit several parametric survival models at once. The function utilizes functionality from another `R` package - `survHE`.

```
# fit all parametric models to the data and extract the AIC/BIC.
# Select the one with the most appropriate fit
# Repeat for PFS and OS
fit_PFS <- fit.fun(time = "PFS_time", status = "PFS_status", data = OS_PFS_data,
                   extrapolate = TRUE, times = times)
fit_OS  <- fit.fun(time = "OS_time", status  = "OS_status" , data = OS_PFS_data,
                   extrapolate = TRUE, times = times)
```

We also developed a function `fit.fun.cure` that allows you to fit several parametric mixture cure models at once.

```
# fit all parametric mixture cure models to the data and extract the AIC/BIC.
# Select the one with the most appropriate fit
# Repeat for PFS and OS
fit_PFS_cure <- fit.fun.cure(time = "PFS_time", status = "PFS_status", data = OS_PFS_data,
                             extrapolate = TRUE, times = times)
fit_OS_cure  <- fit.fun.cure(time = "OS_time", status  = "OS_status" , data = OS_PFS_data,
                             extrapolate = TRUE, times = times)
```

Select the best-fitting model for OS and PFS based on goodness-of-fit measures like AIC and BIC.

```
# Check AIC of each model to assess goodness-of-fit
GoF_PFS <- data.frame(AIC = c(fit_PFS$AIC, fit_PFS_cure$AIC),
                      BIC = c(fit_PFS$BIC, fit_PFS_cure$BIC))
GoF_OS  <- data.frame(AIC = c( fit_OS$AIC,  fit_OS_cure$AIC),
                      BIC = c( fit_OS$BIC,  fit_OS_cure$BIC))
choose_PFS <- rownames(GoF_PFS)[which.min(GoF_PFS$AIC)]; choose_PFS
choose_OS  <- rownames(GoF_OS)[which.min(GoF_OS$AIC)]; choose_OS
```

Superimpose the best-fitting survival curves on the Kaplan-Meier curves.

```
# PFS
fit_KM_PFS   <- survfit(Surv(time = PFS_time, event = PFS_status) ~ 1, data = OS_PFS_data)
# Select the best fitted survivor function based on the smallest AIC
best_PFS <- fit_PFS$model.objects$models[[choose_PFS]]
# plot KM for PFS
PFS_superimpose <- ggsurvplot(
  fit_KM_PFS,
  data = OS_PFS_data,
  size = 1,                 # change line size
  palette = c("orange2"),    #  custom color palettes
  conf.int = TRUE,          # Add confidence interval
  pval = TRUE,              # Add p-value
  risk.table = TRUE,        # Add risk table
  risk.table.height = 0.25,  # Useful to change when you have multiple groups
  ggtheme = theme_bw(),      # Change ggplot2 theme
  xlab = 'Time in years',    # Change X-axis label
  title    = "Kaplan-Meier survival curve for Progression-Free Survival (PFS)",
  subtitle = paste0("Superimposed by ", choose_PFS, " survival curve")
)
```

```
# extract the estimated survival probabilities and the confidence intervals
summary_best_PFS <- as.data.frame(summary(best_PFS))
# superimpose the survival probabilities
PFS_superimpose$plot  <- PFS_superimpose$plot +
  geom_line(aes(x=time, y=est), size=0.55, alpha=0.65, data=summary_best_PFS)
# superimpose the lower bound of the confidence interval
PFS_superimpose$plot  <- PFS_superimpose$plot +
  geom_line(aes(x=time, y=lcl), size=0.55, alpha=0.65, linetype='dashed', data=summary_best_PFS)
# superimpose the upper bound of the confidence interval
PFS_superimpose$plot  <- PFS_superimpose$plot +
  geom_line(aes(x=time, y=ucl), size=0.55, alpha=0.65, linetype='dashed', data=summary_best_PFS)
PFS_superimpose

# OS
# Select the best fitted survivor function based on the smallest AIC
best_OS <- fit_OS$model.objects$models[[choose_OS]]
# plot KM for OS
OS_superimpose <- ggsurvplot(
  fit_KM_OS,
  data = OS_PFS_data,
  size = 1,                 # change line size
  palette = c("blue3"),     #  custom color palettes
  conf.int = TRUE,          # Add confidence interval
  pval = TRUE,              # Add p-value
  risk.table = TRUE,        # Add risk table
  risk.table.height = 0.25, # Useful to change when you have multiple groups
  ggtheme = theme_bw(),     # Change ggplot2 theme
  xlab = 'Time in years',   # Change X-axis label
  title    = "Kaplan-Meier survival curve for Overall Survival (OS)",
  subtitle = paste0("Superimposed by ", choose_OS, " survival curve")
)
# extract the estimated survival probabilities and the confidence intervals
summary_best_OS <- as.data.frame(summary(best_OS))
# superimpose the survival probabilities
OS_superimpose$plot  <- OS_superimpose$plot +
  geom_line(aes(x=time, y=est), size=0.55, alpha=0.65, data=summary_best_OS)
# superimpose the lower bound of the confidence interval
OS_superimpose$plot  <- OS_superimpose$plot +
  geom_line(aes(x=time, y=lcl), size=0.55, alpha=0.65, linetype='dashed', data=summary_best_OS)
# superimpose the upper bound of the confidence interval
OS_superimpose$plot  <- OS_superimpose$plot +
  geom_line(aes(x=time, y=ucl), size=0.55, alpha=0.65, linetype='dashed', data=summary_best_OS)
OS_superimpose
```

We can plot the hazard ratio of OS vs. PFS using the best-fitting models.

```
df_HR <- boot_hr(surv_model1 = best_PFS, surv_model2 = best_OS, times = times, B = 100)
df_HR %>% filter(time > 1) %>%
  ggplot(aes(x = time, y = med)) +
         geom_ribbon(aes(ymin=lcl, ymax=ucl), colour = NA, fill = "darkgreen", alpha=0.1) +
         geom_line(size = 0.5, color = "darkgreen") +
         scale_x_continuous(breaks = c(1:10)) +
         labs(title = "Hazard ratio of PFS vs. OS",
```

```
              subtitle = "Median [2.5%, 97.5%]",
              x = "time (years)",
              y = "hazard ratio") +
         theme_bw()
```

We developed the function `surv_prob` to extract survival probabilities from a partitioned survival model obtained by `partsurv`. We also developed the function `trans_prob` to convert survival probabilities into transition probabilities.

```
surv_prob(best_OS) # extract survival probabilities
trans_prob(surv_prob(best_OS)) # convert survival probabilities to transition probabilities
```

04.2 Partitioned Survival model

```
# construct a partitioned survival model out of the chosen models
m_M_PSM <- partsurv(pfs_survHE = fit_PFS,
                    os_survHE  = fit_OS,
                    choose_PFS = choose_PFS,
                    choose_OS  = choose_OS,
                    time = times,
                    v_names_states = v_names_states)

# plot the results of PSM and the true data
plot_trace_PSM(time = times, partsurv.model = m_M_PSM, v_names_states = v_names_states)

# construct a PSA  partitioned survival model out of the fitted models
m_M_PSM_PSA <- partsurv(pfs_survHE = fit_PFS,
                        os_survHE  = fit_OS,
                        choose_PFS = choose_PFS,
                        choose_OS  = choose_OS,
                        time = times,
                        v_names_states = v_names_states,
                        PA = T, n_sim = 1000)

# plot the results of PSM and the trace
plot_trace_PSM(time = times, partsurv.model = m_M_PSM_PSA, PA = T, v_names_states = v_names_states)
```

Other outputs from the partitioned survival model could be explored:

```
# Expected survival
m_M_PSM$pfs.expected.surv
m_M_PSM$os.expected.surv
```