Markov Sick-Sicker model in R

with dependency for time-since model start AND with state-residency dependency

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 (6,7)

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. 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. Med Decis Making. 2020 Online first. https://doi.org/10.1177/0272989X19893973

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

```
rm(list = ls())      # clear memory (removes all the variables from the workspace)
```

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", "scales", "ellipse", "ggplot2", "lazyeval", "igraph", "truncnorm",
# load (install if required) packages from GitHub
# install_github("DARTH-git/dampack", force = TRUE) Uncomment if there is a newer version
# install_github("DARTH-git/darthtools", force = TRUE) Uncomment if there is a newer version
p_load_gh("DARTH-git/dampack", "DARTH-git/darthtools")
```

02 Load functions

```
# No functions needed
```

03 Input model parameters

```
# Strategy names
v_names_str <- c("No Treatment", "Treatment")

# Number of strategies
n_str <- length(v_names_str)

# Markov model parameters
age     <- 25                     # age at baseline
max_age <- 55                     # maximum age of follow up
n_t     <- max_age - age          # time horizon, number of cycles
v_n     <- c("H", "S1", "S2", "D")  # the 4 states of the model: Healthy (H), Sick (S1),
                                  # Sicker (S2), Dead (D)
n_states <- length(v_n)           # number of health states

# Tunnels
n_tunnel_size <- n_t
# Sick state
v_Sick_tunnels <- paste("S1_", seq(1, n_tunnel_size), "Yr", sep = "")
### Create variables for time-dependent model
v_n_tunnels      <- c("H", v_Sick_tunnels, "S2", "D")  # state names
n_states_tunnels <- length(v_n_tunnels)                # number of states

# Transition probabilities (per cycle) and hazard ratios
# Read age-specific mortality rates from csv file
lt_usa_2005 <- read.csv("HMD_USA_Mx_2015.csv")
```

```
v_r_HD <- lt_usa_2005 %>%
          filter(Age >= age & Age <= (max_age-1)) %>%
          select(Total) %>%
          as.matrix()

p_HD    <- 1 - exp(- v_r_HD)         # probability to die when healthy
p_HS1   <- 0.15                        # probability to become sick when healthy
p_S1H   <- 0.5                         # probability to become healthy when sick

# Weibull parameters
l       <- 0.08 # scale
g       <- 1.1  # shape
# Weibull function
p_S1S2  <- l*g*(1:n_tunnel_size)^{g-1} # probability to become sicker when sick
                                       # (time-dependent)

hr_S1   <- 3                           # hazard ratio of death in sick vs healthy
hr_S2   <- 10                          # hazard ratio of death in sicker vs healthy
r_HD    <- - log(1 - p_HD)           # rate of death in healthy
r_S1D   <- hr_S1 * r_HD                # rate of death in sick
r_S2D   <- hr_S2 * r_HD                # rate of death in sicker
p_S1D   <- 1 - exp(-r_S1D)           # probability to die in sick
p_S2D   <- 1 - exp(-r_S2D)           # probability to die in sicker

# Cost and utility inputs
c_H     <- 2000                      # cost of remaining one cycle in the healthy state
c_S1    <- 4000                      # cost of remaining one cycle in the sick state
c_S2    <- 15000                     # cost of remaining one cycle in the sicker state
c_trt   <- 12000                     # cost of treatment(per cycle)
c_D     <- 0                         # cost of being in the death state
u_H     <- 1                         # utility when healthy
u_S1    <- 0.75                      # utility when sick
u_S2    <- 0.5                       # utility when sicker
u_D     <- 0                         # utility when dead
u_trt   <- 0.95                      # utility when being treated

# Discounting factor
d_r     <- 0.03                      # equal discount of costs and QALYs by 3%
# calculate discount weights for costs for each cycle based on discount rate d_c
v_dwc   <- 1 / (1 + d_r) ^ (0:n_t)
# calculate discount weights for effectiveness for each cycle based on discount rate d_e
v_dwe   <- 1 / (1 + d_r) ^ (0:n_t)
```

04 Define and initialize matrices and vectors

04.1 Cohort trace

```
# create the markov trace matrix M capturing the proportion of the cohort in each state
# at each cycle
m_M_notrt <- m_M_trt <- matrix(NA,
                               nrow = n_t + 1, ncol = n_states_tunnels,
                               dimnames = list(paste("cycle", 0:n_t, sep = " "), v_n_tunnels))

head(m_M_notrt) # show first 6 rows of the matrix

# The cohort starts as healthy
# initialize first cycle of Markov trace accounting for the tunnels
m_M_notrt[1, ] <- m_M_trt[1, ] <- c(1, rep(0, n_tunnel_size), 0, 0)
```

04.2 Transition probability array

```
# create the transition probability array for NO treatment
a_P_notrt <- array(0,                                         # Create 3-D array
                   dim = c(n_states_tunnels, n_states_tunnels, n_t),
                   dimnames = list(v_n_tunnels, v_n_tunnels, 0:(n_t-1))) # name dimensions
```

Fill in the transition probability array:

```
# from Healthy
a_P_notrt["H", "H", ]               <- (1 - p_HD) * (1 - p_HS1)
a_P_notrt["H", v_Sick_tunnels[1], ] <- (1 - p_HD) * p_HS1
a_P_notrt["H", "D", ]               <- p_HD

# from Sick
for(i in 1:(n_tunnel_size - 1)){
  a_P_notrt[v_Sick_tunnels[i], "H", ]  <- (1 - p_S1D) * p_S1H
  a_P_notrt[v_Sick_tunnels[i], v_Sick_tunnels[i + 1], ] <-
    (1 - p_S1D) * (1 - (p_S1H + p_S1S2[i]))
  a_P_notrt[v_Sick_tunnels[i], "S2", ] <- (1 - p_S1D) * p_S1S2[i]
  a_P_notrt[v_Sick_tunnels[i], "D", ]  <- p_S1D
}
a_P_notrt[v_Sick_tunnels[n_tunnel_size], "H", ]  <- (1 - p_S1D) * p_S1H
a_P_notrt[v_Sick_tunnels[n_tunnel_size], v_Sick_tunnels[n_tunnel_size], ] <-
  (1 - p_S1D) * (1 - (p_S1H + p_S1S2[n_tunnel_size]))
a_P_notrt[v_Sick_tunnels[n_tunnel_size], "S2", ] <- (1 - p_S1D) * p_S1S2[n_tunnel_size]
a_P_notrt[v_Sick_tunnels[n_tunnel_size], "D", ] <- p_S1D

# from Sicker
a_P_notrt["S2", "S2", ] <- 1 - p_S2D
a_P_notrt["S2", "D", ]  <- p_S2D

# from Dead
a_P_notrt["D", "D", ] <- 1

# Check that transition probabilities are in [0, 1]
check_transition_probability(a_P_notrt, verbose = TRUE)
# Check that all rows sum to 1
check_sum_of_transition_array(a_P_notrt, n_states = n_states_tunnels, n_t = n_t, verbose = TRUE)

# create transition probability matrix for treatment same as NO treatment
a_P_trt <- a_P_notrt
```

05 Run Markov model

```
for (t in 1:n_t){   # loop through the number of cycles
  m_M_notrt[t + 1, ] <- t(m_M_notrt[t, ]) %*% a_P_notrt[ , , t]  # estimate the Markov
                                                                 # trace for cycle the
                                                                 # next cycle (t + 1)

  m_M_trt[t + 1, ]    <- t(m_M_trt[t, ])    %*% a_P_trt[, , t]   # estimate the Markov
                                                                 # trace for cycle the
                                                                 # next cycle (t + 1)
} # close the loop

head(m_M_notrt)  # show the first 6 lines of the matrix

# create aggregated traces
m_M_td_notrt <- cbind(H  = m_M_notrt[, "H"],
                      S1 = rowSums(m_M_notrt[, 2:(n_tunnel_size +1)]),
                      S2 = m_M_notrt[, "S2"],
                      D  = m_M_notrt[, "D"])
head(m_M_td_notrt)

m_M_td_trt   <- cbind(H = m_M_trt[, "H"],
                      S1 = rowSums(m_M_trt[, 2:(n_tunnel_size +1)]),
                      S2 = m_M_trt[, "S2"],
                      D = m_M_trt[, "D"])
head(m_M_td_trt)
```

06 Compute and Plot Epidemiological Outcomes

06.1 Cohort trace

```
# create a plot of the data
matplot(m_M_td_notrt, type = 'l',
        ylab = "Probability of state occupancy",
        xlab = "Cycle",
        main = "Cohort Trace")
# add a legend to the graph
legend("topright", v_n, col = 1:n_states,lty = 1:n_states, bty = "n")
```

06.2 Overall Survival (OS)

```
# calculate the overall survival (OS) probability for no treatment
v_os_notrt_tunnels <- 1 - m_M_notrt[, "D"]
# alternative way of calculating the OS probability
v_os_notrt_tunnels <- rowSums(m_M_notrt[, 1:3])
# create a simple plot showing the OS
plot(age:max_age, v_os_notrt_tunnels, type = 'l',
     ylim = c(0, 1),
     ylab = "Survival probability",
     xlab = "Age",
     main = "Overall Survival Age-dependent with tunnels")
# add grid
grid(nx = n_t, ny = 10, col = "lightgray", lty = "dotted", lwd = par("lwd"),
     equilogs = TRUE)
```

06.2.1 Life Expectancy (LE)

```
v_le_tunnels <- sum(v_os_notrt_tunnels)  # summing probability of OS over time
                                         # (i_e_ life expectancy)
```

06.3 Disease prevalence

```
v_prev_tunnels <- rowSums(m_M_td_notrt[, c("S1", "S2")]) / v_os_notrt_tunnels
plot(v_prev_tunnels,
     ylim = c(0, 1),
     ylab = "Prevalence",
     xlab = "Cycle",
     main = "Disease prevalence")
```

06.4 ratio of sick(S1) vs sicker(S2)

```
v_ratio_S1S2_tunnels <- m_M_td_notrt[, "S1"] / m_M_td_notrt[, "S2"]
plot(0:n_t, v_ratio_S1S2_tunnels,
     xlab = "Cycle",
     ylab = "Ratio S1 vs S2",
     main = "Ratio of sick and sicker",
     col  = "black", type = "l")
```

07 Compute Cost-Effectiveness Outcomes

```
# Vectors with costs and utilities by treatment
v_u_notrt   <- c(u_H, u_S1, u_S2, u_D)
v_u_trt     <- c(u_H, u_trt, u_S2, u_D)

v_c_notrt   <- c(c_H, c_S1, c_S2, c_D)
v_c_trt     <- c(c_H, c_S1 + c_trt, c_S2 + c_trt, c_D)
```

07.1 Mean Costs and QALYs for Treatment and NO Treatment

```r
v_tu_notrt  <- m_M_td_notrt   %*%  v_u_notrt
v_tu_trt    <- m_M_td_trt     %*%  v_u_trt

v_tc_notrt  <- m_M_td_notrt   %*%  v_c_notrt
v_tc_trt    <- m_M_td_trt     %*%  v_c_trt
```

07.2 Discounted Mean Costs and QALYs

```r
tu_d_notrt  <- t(v_tu_notrt)  %*%  v_dwe
tu_d_trt    <- t(v_tu_trt)    %*%  v_dwe

tc_d_notrt  <- t(v_tc_notrt)  %*%  v_dwc
tc_d_trt    <- t(v_tc_trt)    %*%  v_dwc


# store them into a vector
v_tc_d      <- c(tc_d_notrt, tc_d_trt)
v_tu_d      <- c(tu_d_notrt, tu_d_trt)

# Store discounted costs and effectiveness for each strategy in a data frame
df_ce       <- data.frame(Strategy = v_names_str,
                          Cost     = v_tc_d,
                          Effect   = v_tu_d
                          )
df_ce
```

07.3 Compute ICERs of the Markov model

```r
# Use the function calculate_icers() from the dampack package
df_cea <- calculate_icers(cost       = df_ce$Cost,
                          effect     = df_ce$Effect,
                          strategies = df_ce$Strategy
                          )
df_cea
```

07.4 Plot frontier of the Markov model

```r
plot(df_cea, effect_units = "QALYs", xlim=c(17,18))
```