Simple 3-state Markov 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. Division of Public Administration, Center for Research and Teaching in Economics (CIDE), 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

- Alarid-Escudero F, Krijkamp EM, Enns EA, Yang A, Hunink MGM Pechlivanoglou P, Jalal H. Cohort State-Transition Models in R: A Tutorial. arXiv:200107824v2. 2020:1-48. http://arxiv.org/abs/2001.07824

- 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("diagram")
# install_github("DARTH-git/darthtools", force = TRUE) Uncomment if there is a newer version
p_load_gh("DARTH-git/darthtools")
```

02 Load functions

```
# No functions needed
```

03 Input model parameters

```
# Strategy names
v_names_str <- c("Standard of Care")

# Markov model parameters
v_n  <- c("Healthy", "Sick", "Dead")    # state names
n_states  <- length(v_n)                # number of states
n_t  <- 60                              # number of cycles

# Transition probabilities
p_HD_min <- 0.003                       # probability of dying when healthy at t = 0
p_HD_max <- 0.01                        # probability of dying when health  at t = n.t
p_HS     <- 0.05                        # probability of becoming sick when healthy, under standard of
p_SD     <- 0.1                         # probability of dying when sick

# Costs and utilities
c_H      <- 400                         # cost of one cycle in healthy state
c_S      <- 1000                        # cost of one cycle in sick state
c_D      <- 0                           # cost of one cycle in dead state
u_H      <- 0.8                         # utility when healthy
u_S      <- 0.5                         # utility when sick
u_D      <- 0                           # utility when dead
d_e      <- d_c <- 0.03                 # equal discount of costs and QALYs by 3%

p_HD <- seq(p_HD_min, p_HD_max, length.out = n_t)  # probabilities of dying when healthy (age-dependent
                                                   # this is now a sequence of numbers, officially v_p_

n_str     <- length(v_names_str)        # Number of strategies
```

```r
n_states  <- length(v_n)                # number of states

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

# Weibull parameters
l <- 0.08
g <- 1.1
p_SD <- l*g*(1:n_tunnel_size)^{g-1}     # probability of dying when sick (time-in-state dependent)

# Discount weights for costs and effects
v_dwc <- 1 / (1 + d_c) ^ (0:n_t)
v_dwe <- 1 / (1 + d_e) ^ (0:n_t)
```

04 Define and initialize matrices and vectors

04.1 Cohort trace

```r
m_M <- matrix(NA,
              nrow = n_t + 1,  # create Markov trace (n_t + 1) because R doesn't understand
                               # Cycle 0)
              ncol = n_states_tunnels,
              dimnames = list(0:n_t, v_n_tunnels))

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

04.2 Transition probability array

```r
# create the transition probability array
a_P <- 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)))
```

Fill in the transition probability array:

```r
# from Healthy
a_P["Healthy", "Healthy", ]  <- (1 - p_HD) * (1 - p_HS)
a_P["Healthy", "Sick_1Yr", ] <- (1 - p_HD) * p_HS
a_P["Healthy", "Dead", ]     <- p_HD

# from Sick
```

```r
for(i in 1:(n_tunnel_size - 1)){
  a_P[v_Sick_tunnels[i], v_Sick_tunnels[i + 1], ] <- 1 - p_SD[i]
  a_P[v_Sick_tunnels[i], "Dead", ] <- p_SD[i]
}

a_P[v_Sick_tunnels[n_tunnel_size], v_Sick_tunnels[n_tunnel_size], ] <- 1 - p_SD[n_tunnel_size]
a_P[v_Sick_tunnels[n_tunnel_size], "Dead", ] <- p_SD[n_tunnel_size]

# from Dead
a_P["Dead", "Dead", ] <- 1
```

04.3 Check if transition array and probabilities are valid

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

05 Run Markov model

```r
for (t in 1:n_t) {                           # loop through the number of cycles
  m_M[t + 1, ] <- m_M[t, ] %*% a_P[, , t]  # estimate the Markov trace for cycle t + 1
                                             # using the t-th matrix from the
                                             # probability array
}
head(m_M)
```

Create aggregated trace.

```r
m_M_tunnels <- cbind(Healthy = m_M[, "Healthy"],
                     Sick = rowSums(m_M[, 2:(n_tunnel_size + 1)]),
                     Dead = m_M[, "Dead"])
head(m_M_tunnels) # show the first rows of the aggregated Markov trace
```

06 Compute and Plot Epidemiological Outcomes

06.1 Cohort trace

```r
# create a plot of the data
matplot(m_M_tunnels, type = 'l',
        ylab = "Probability of state occupancy",
        xlab = "Cycle",
        main = "Cohort Trace", lwd = 2)
# add a legend to the graph
legend("right", v_n, col = c("black", "red", "green"), lty = 1:3, bty = "n")
```

06.2 Overall Survival (OS)

```
v_os <- 1 - m_M_tunnels[, "Dead"]    # calculate the overall survival (OS) probability
v_os <- rowSums(m_M_tunnels[, 1:2])  # alternative way of calculating the OS probability

# create a simple plot showing the OS
plot(v_os, type = 'l',
     ylim = c(0, 1),
     ylab = "Survival probability",
     xlab = "Cycle",
     main = "Overall Survival")
# 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 <- sum(v_os) # summing probablity of OS over time  (i.e. life expectancy)
```

06.3 Disease prevalence

```
v_prev <- m_M_tunnels[, "Sick"]/v_os
plot(v_prev,
     ylim = c(0, 1),
     ylab = "Prevalence",
     xlab = "Cycle",
     main = "Disease prevalence")
```

07 Compute Cost-Effectiveness Outcomes

07.1 Mean Costs and QALYs

```
# per cycle
# calculate expected costs by multiplying m_M with the cost vector for the different
# health states
v_tc <- m_M_tunnels %*% c(c_H, c_S, c_D)
# calculate expected QALYs by multiplying m_M with the utilities for the different
# health states
v_tu <- m_M_tunnels %*% c(u_H, u_S, u_D)
```

07.2 Discounted Mean Costs and QALYs

```r
# Discount costs  by multiplying the cost vector with discount weights (v_dw)
v_tc_d <-  t(v_tc) %*% v_dwc
# Discount QALYS  by multiplying the QALYs vector with discount weights (v_dw)
v_te_d <-  t(v_tu) %*% v_dwe
```

07.3 Store Results

```r
df_ce <- data.frame("Total Discounted Cost"  = v_tc_d,
                    "Life Expectancy"        = v_le,
                    "Total Discounted QALYs" = v_te_d,
                    check.names = F)
df_ce
```