SOLUTIONS - Microsimulation Sick-Sicker model with time dependency - Excerise

Includes individual characteristics: age, age dependent mortality, individual treatment effect modifer, state-residency for the sick (S1) state, increasing change of death in the first 6 year of sickness

The DARTH workgroup

This work is developed by the Decision Analysis in R for Technologies in Health (DARTH) workgroup:

* Fernando Alarid-Escudero, PhD
* Eva A. Enns, MS, PhD
* M.G. Myriam Hunink, MD, PhD
* Hawre J. Jalal, MD, PhD
* Eline Krijkamp, PhD
* Petros Pechlivanoglou, PhD
* Alan Yang, MSc

Please acknowledge our work. See details to cite below.

See student template for exercise description.

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

# 01 Load packages

if (!require('pacman')) install.packages('pacman'); library(pacman)   
# load (install if required) packages from CRAN  
p\_load("devtools", "dplyr", "scales", "ellipse", "ggplot2", "lazyeval", "igraph", "truncnorm", "ggraph", "reshape2", "knitr", "markdown", "stringr", "dampack", "matrixStats")  
# load (install if required) packages from GitHub  
# 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 Model input

## General setup   
set.seed(1) # set the seed   
cycle\_length <- 1 # cycle length equal to one year (use 1/12 for monthly)  
n\_cycles <- 30 # time horizon, number of cycles  
n\_i <- 100000 # number of individuals  
  
# the 4 health states of the model:  
v\_names\_states <- c("H", # Healthy (H)  
 "S1", # Sick (S1)  
 "S2", # Sicker (S2)  
 "D") # Dead (D)  
v\_names\_cycles <- paste("cycle", 0:n\_cycles) # cycle names  
n\_states <- length(v\_names\_states) # number of health states   
  
  
### Discounting factors   
d\_c <- 0.03 # annual discount rate for costs   
d\_e <- 0.03 # annual discount rate for QALYs  
  
### Strategies   
v\_names\_str <- c("Standard of care", # store the strategy names  
 "Strategy AB")   
n\_str <- length(v\_names\_str) # number of strategies  
  
### Transition probabilities   
# (all non-probabilities are conditional on survival)  
p\_HS1 <- 0.15 # probability of becoming sick when healthy  
p\_S1H <- 0.5 # probability of recovering to healthy when sick  
p\_S1S2\_SoC <- 0.105 # probability of becoming sicker when sick under standard of care  
p\_S1S2\_trtAB <- 0.05 # probability of becoming sicker when sick under treatment AB  
  
# Annual probabilities of death  
# load age dependent probability  
p\_mort <- read.csv("../data/mortProb\_age.csv")  
# load age distribution  
dist\_Age <- read.csv("../data/MyPopulation-AgeDistribution.csv")   
  
# probability to die in S1 by cycle (is increasing)  
v\_p\_S1D <- c(0.0149, 0.018, 0.021, 0.026, 0.031, rep(0.037, n\_cycles - 5))   
 p\_S2D <- 0.048 # probability to die in S2  
  
### State rewards   
#### Costs   
c\_H <- 2000 # annual cost of being Healthy  
c\_S1 <- 4000 # annual cost of being Sick  
c\_S2 <- 15000 # annual cost of being Sicker  
c\_D <- 0 # annual cost of being dead  
c\_trtAB <- 25000 # annual cost of receiving treatment AB when in Sick  
#### Utilities   
u\_H <- 1 # annual utility of being Healthy  
u\_S1 <- 0.75 # annual utility of being Sick  
u\_S2 <- 0.5 # annual utility of being Sicker  
u\_D <- 0 # annual utility of being dead  
u\_trtAB <- 0.95 # annual utility when receiving treatment AB when in Sick

## 03.2 Calculate internal model parameters

### Discount weight for costs and effects   
v\_dwc <- 1 / ((1 + (d\_e \* cycle\_length)) ^ (0:n\_cycles))  
v\_dwe <- 1 / ((1 + (d\_c \* cycle\_length)) ^ (0:n\_cycles))  
  
# Within-cycle correction (WCC) - method options Simpson's 1/3 rule, "half-cycle" or "none"   
v\_wcc <- darthtools::gen\_wcc(n\_cycles = n\_cycles,   
 method = "Simpson1/3") # vector of wcc

# 04 Sample individual level characteristics

## 04.1 Static characteristics

# sample the treatment effect modifier at baseline   
v\_x <- runif(n\_i, min = 0.95, max = 1.05)   
# sample from the age distribution the initial age for every individual  
v\_age0 <- sample(x = dist\_Age$age, prob = dist\_Age$prop, size = n\_i, replace = TRUE)

## 04.2 Dynamic characteristics

# Specify the initial health state of the individuals   
# everyone begins in the healthy state (in this example)  
v\_M\_init <- rep("H", times = n\_i)   
v\_n\_cycles\_s\_init <- rep(0, n\_i) # a vector with the time of being sick at the start of the model

## 04.3 Create a dataframe with the individual characteristics

# create a data frame with each individual's   
# ID number, treatment effect modifier, age and initial time in sick state   
df\_X <- data.frame(ID = 1:n\_i, x = v\_x, Age = v\_age0, n\_cycles\_s = v\_n\_cycles\_s\_init, M\_init = v\_M\_init)   
head(df\_X) # print the first rows of the dataframe

## ID x Age n\_cycles\_s M\_init  
## 1 1 0.9765509 43 0 H  
## 2 2 0.9872124 43 0 H  
## 3 3 1.0072853 32 0 H  
## 4 4 1.0408208 41 0 H  
## 5 5 0.9701682 46 0 H  
## 6 6 1.0398390 27 0 H

# 05 Define Simulation Functions

## 05.1 Probability function

The Probs function updates the transition probabilities of every cycle is shown below.

Probs <- function(M\_t, df\_X, Trt = "Standard of care") {   
 # Arguments:  
 # M\_t: health state occupied by individual i at cycle t (character variable)  
 # df\_X: data frame with individual characteristics data   
 # Trt: treatment  
 # Returns:   
 # transition probabilities for that cycle  
   
 # Treatment specific transition probabilities  
 if (Trt == "Standard of care") {  
 p\_S1S2 <- p\_S1S2\_SoC  
 } else if (Trt == "Strategy AB") {  
 p\_S1S2 <- p\_S1S2\_trtAB  
 }  
   
 # create matrix of state transition probabilities   
 m\_p\_t <- matrix(data = 0, nrow = n\_states, ncol = n\_i)   
 rownames(m\_p\_t) <- v\_names\_states # give the state names to the rows  
   
 # lookup baseline probability and rate of dying based on individual characteristics age  
 p\_HD\_all <- inner\_join(x = df\_X, y = p\_mort, by = c("Age"))  
 p\_HD <- p\_HD\_all[M\_t == "H", "p\_HD"]  
   
   
 # update the m\_p with the appropriate probabilities   
 # (all non-death probabilities are conditional on survival)   
 # transition probabilities when Healthy   
 m\_p\_t["H", M\_t == "H"] <- (1 - p\_HD) \* (1 - p\_HS1)  
 m\_p\_t["S1", M\_t == "H"] <- (1 - p\_HD) \* p\_HS1   
 m\_p\_t["S2", M\_t == "H"] <- 0  
 m\_p\_t["D", M\_t == "H"] <- p\_HD   
   
 # transition probabilities when Sick   
 m\_p\_t["H", M\_t == "S1"] <- (1 - v\_p\_S1D[df\_X$n\_cycles\_s]) \* p\_S1H   
 m\_p\_t["S1", M\_t == "S1"] <- (1 - v\_p\_S1D[df\_X$n\_cycles\_s]) \* (1 - p\_S1H - p\_S1S2)  
 m\_p\_t["S2", M\_t == "S1"] <- (1 - v\_p\_S1D[df\_X$n\_cycles\_s]) \* p\_S1S2  
 m\_p\_t["D", M\_t == "S1"] <- v\_p\_S1D[df\_X$n\_cycles\_s]   
   
 # transition probabilities when Sicker  
 m\_p\_t["H", M\_t == "S2"] <- 0  
 m\_p\_t["S1", M\_t == "S2"] <- 0  
 m\_p\_t["S2", M\_t == "S2"] <- 1 - p\_S2D  
 m\_p\_t["D", M\_t == "S2"] <- p\_S2D   
   
 # transition probabilities when Dead  
 m\_p\_t["H", M\_t == "D"] <- 0  
 m\_p\_t["S1", M\_t == "D"] <- 0  
 m\_p\_t["S2", M\_t == "D"] <- 0   
 m\_p\_t["D", M\_t == "D"] <- 1   
   
 return(t(m\_p\_t))  
}

## 05.2 Cost function

The Costs function estimates the costs at every cycle.

Costs <- function (M\_t, Trt = "Standard of care") {  
 # Arguments:  
 # M\_t: health state occupied by individual i at cycle t (character variable)  
 # Trt: Treatment   
 # Returns:  
 # costs accrued in this cycle  
   
 # Treatment specific costs  
 if (Trt == "Standard of care") {  
 c\_Trt <- 0  
 } else if (Trt == "Strategy AB") {  
 c\_Trt <- c\_trtAB  
 }  
   
 c\_t <- c() # initiate the cost parameter  
 c\_t[M\_t == "H"] <- c\_H # update the cost if healthy  
 c\_t[M\_t == "S1"] <- c\_S1 + c\_Trt # update the cost if sick conditional on treatment  
 c\_t[M\_t == "S2"] <- c\_S2 + c\_Trt # update the cost if sicker conditional on treatment  
 c\_t[M\_t == "D"] <- c\_D # update the cost if dead  
   
 return(c\_t) # return costs accrued this cycle   
}

## 05.3 Health outcome function

The Effs function to update the utilities at every cycle.

Effs <- function (M\_t, df\_X, Trt = "Standard of care", cycle\_length = 1) {  
 # Arguments:  
 # M\_t: health state occupied by individual i at cycle t (character variable)  
 # df\_X: data frame with individual characteristics data   
 # Trt: Treatment  
 # cycle\_length: cycle length (default is 1)  
 # Returns:  
 # QALYs accrued this cycle  
   
 u\_t <- c() # initialize the utility parameter  
 u\_t[M\_t == "H"] <- u\_H # update the utility if healthy  
   
   
 if (Trt == "Standard of care") { # update the utility if sick under standard of care  
 u\_t[M\_t == "S1"] <- u\_S1  
 } else if (Trt == "Strategy AB") {  
 # update the utility if sick but on treatment AB (adjust for individual effect modifier)   
 u\_t[M\_t == "S1"] <- u\_trtAB \* df\_X$x[M\_t == "S1"]   
 }  
   
 u\_t[M\_t == "S2"] <- u\_S2 # update the utility if sicker  
 u\_t[M\_t == "D"] <- u\_D # update the utility if dead  
   
 QALYs <- u\_t \* cycle\_length # calculate the QALYs during cycle t  
 return(QALYs) # return the QALYs accrued this cycle  
}

## 05.4 The Microsimulation function

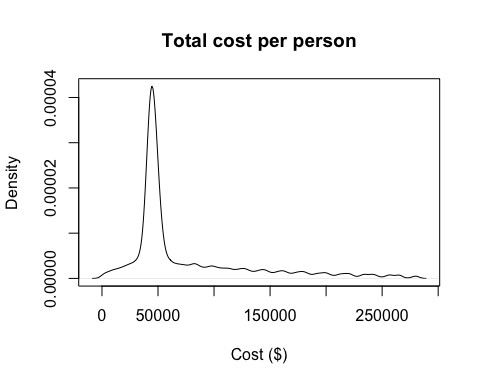
MicroSim <- function(n\_i, df\_X, Trt = "Standard of care", seed = 1, cycle\_length = 1) {  
 # Arguments:   
 # n\_i : number of individuals  
 # df\_X: data frame with individual characteristics data   
 # Trt : treatment  
 # seed: seed for the random number generator, default is 1  
 # cycle\_length : cycle length  
 # Returns:  
 # results: data frame with total cost and QALYs  
   
 set.seed(seed) # set a seed to be able to reproduce the same results  
   
 # create three matrices called m\_M, m\_C and m\_E  
 # number of rows is equal to the n\_i, the number of columns is equal to n\_cycles   
 # (the initial state and all the n\_cycles cycles)  
 # m\_M is used to store the health state information over time for every individual  
 # m\_C is used to store the costs information over time for every individual  
 # m\_E is used to store the effects information over time for every individual  
   
 m\_M <- m\_C <- m\_E <- matrix(nrow = n\_i, ncol = n\_cycles + 1,   
 dimnames = list(paste("ind" , 1:n\_i, sep = " "),   
 paste("cycle", 0:n\_cycles, sep = " ")))   
  
 m\_M[, 1] <- as.character(df\_X$M\_init) # initial health state at cycle 0 for individual i  
 m\_C[, 1] <- Costs(m\_M[, 1], Trt = Trt) # calculate costs per individual during cycle 0   
 m\_E[, 1] <- Effs (m\_M[, 1], df\_X, Trt = Trt, cycle\_length = cycle\_length) # calculate QALYs per individual during cycle 0  
   
 # open a loop for time running cycles 1 to n\_cycles   
 for (t in 1:n\_cycles) {  
 # calculate the transition probabilities for the cycle based on health state t  
 m\_P <- Probs(m\_M[, t], df\_X, Trt = Trt)   
 # check if transition probabilities are between 0 and 1  
 check\_transition\_probability(m\_P, verbose = TRUE)  
 # check if checks if each of the rows of the transition probabilities matrix sum to one  
 ## NOTE: to make this function work n\_states = n\_i in a Microsimulation  
 check\_sum\_of\_transition\_array(m\_P, n\_rows = n\_i, n\_cycles = n\_cycles, verbose = TRUE) ##  
 # sample the current health state and store that state in matrix m\_M   
 m\_M[, t + 1] <- samplev(m\_P, 1)   
 # calculate costs per individual during cycle t + 1  
 m\_C[, t + 1] <- Costs(m\_M[, t + 1], Trt)   
 # calculate QALYs per individual during cycle t + 1  
 m\_E[, t + 1] <- Effs(m\_M[, t + 1], df\_X, Trt, cycle\_length = cycle\_length)   
   
 # update time since illness onset for t + 1   
 df\_X$n\_cycles\_s <- if\_else(m\_M[, t + 1] == "S1", df\_X$n\_cycles\_s + 1, 0)   
 # update the age of individuals that are alive  
 df\_X$Age[m\_M[, t + 1] != "D"] <- df\_X$Age[m\_M[, t + 1] != "D"] + 1  
   
 # Display simulation progress  
 if(t/(n\_cycles/10) == round(t/(n\_cycles/10), 0)) { # display progress every 10%  
 cat('\r', paste(t/n\_cycles \* 100, "% done", sep = " "))  
 }  
   
 } # close the loop for the time points   
   
  
 # Discounted total expected QALYs and Costs per strategy and apply cycle correction ####  
 tc <- m\_C %\*% (v\_dwc \* v\_wcc) # total (discounted and cycle corrected) cost per individual  
 te <- m\_E %\*% (v\_dwe \* v\_wcc) # total (discounted and cycle corrected) QALYs per individual   
   
 tc\_hat <- mean(tc) # average (discounted and cycle corrected) cost   
 te\_hat <- mean(te) # average (discounted and cycle corrected) QALY   
   
 # store the results from the simulation in a list  
 results <- list(m\_M = m\_M, m\_C = m\_C, m\_E = m\_E, tc = tc , te = te, tc\_hat = tc\_hat,   
 te\_hat = te\_hat)   
   
 return(results) # return the results  
  
} # end of the MicroSim function   
  
# By specifying all the arguments in the `MicroSim()` the simulation can be started  
# In this example the outcomes are of the simulation are stored in the variables `outcomes\_SoC` and `outcomes\_trtAB`.

# 06 Run Microsimulation

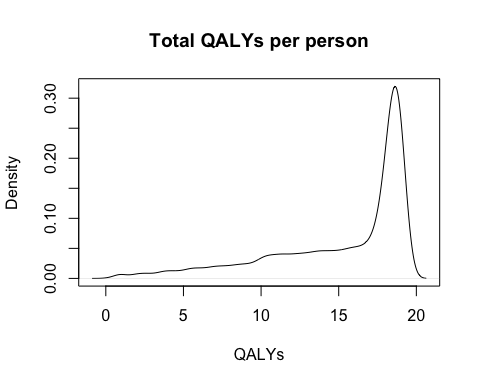
# Run the simulation for both no treatment and treatment options   
outcomes\_SoC <- MicroSim(n\_i = n\_i, df\_X = df\_X, Trt = "Standard of care", seed = 1, cycle\_length = cycle\_length )  
outcomes\_trtAB <- MicroSim(n\_i = n\_i, df\_X = df\_X, Trt = "Strategy AB", seed = 1, cycle\_length = cycle\_length)

# 07 Visualize results

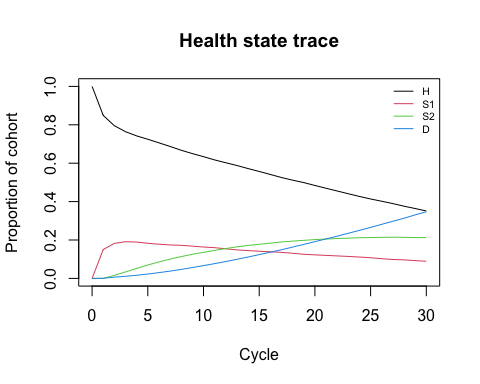
# Standard of care  
plot(density(outcomes\_SoC$tc), main = paste("Total cost per person"), xlab = "Cost ($)")



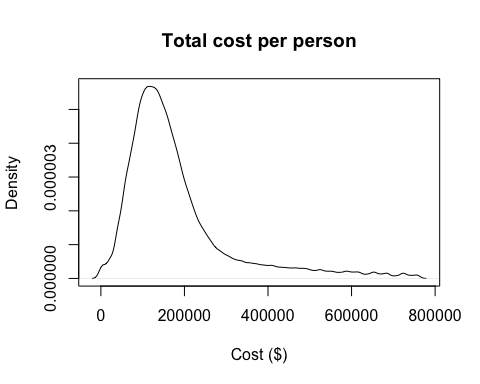
plot(density(outcomes\_SoC$te), main = paste("Total QALYs per person"), xlab = "QALYs")



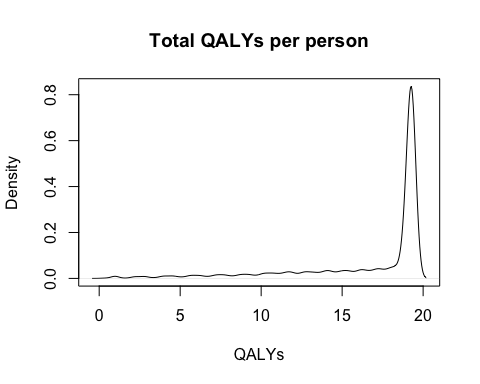
plot\_trace\_microsim(outcomes\_SoC$m\_M) # health state trace



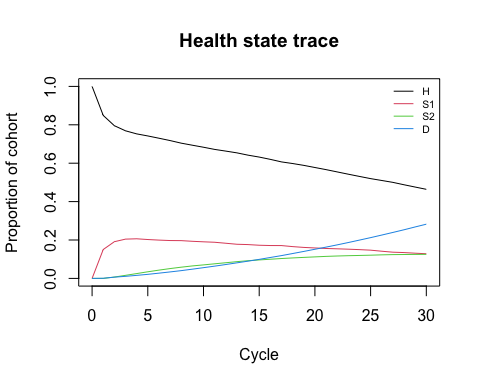
# Strategy AB  
plot(density(outcomes\_trtAB$tc), main = paste("Total cost per person"), xlab = "Cost ($)")



plot(density(outcomes\_trtAB$te), main = paste("Total QALYs per person"), xlab = "QALYs")



plot\_trace\_microsim(outcomes\_trtAB$m\_M) # health state trace



# 08 Cost-effectiveness analysis (CEA)

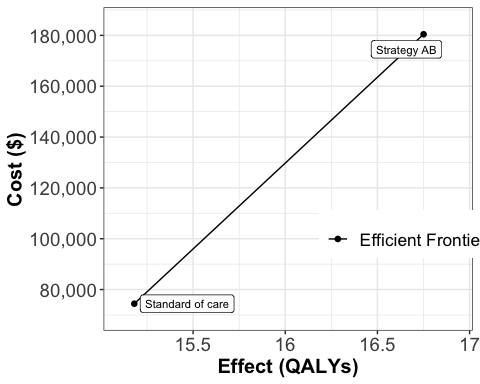
# store the mean costs of each strategy in a new variable C (vector of costs)  
v\_C <- c(outcomes\_SoC$tc\_hat, outcomes\_trtAB$tc\_hat)  
# store the mean QALYs of each strategy in a new variable E (vector of effects)  
v\_E <- c(outcomes\_SoC$te\_hat, outcomes\_trtAB$te\_hat)  
  
# use dampack to calculate the ICER  
df\_cea <- calculate\_icers(cost = v\_C,  
 effect = v\_E,  
 strategies = v\_names\_str)  
df\_cea

## Strategy Cost Effect Inc\_Cost Inc\_Effect ICER Status  
## 1 Standard of care 74428.87 15.18098 NA NA NA ND  
## 2 Strategy AB 180445.95 16.75045 106017.1 1.569477 67549.32 ND

## CEA table in proper format   
table\_cea <- format\_table\_cea(df\_cea)   
table\_cea

## Strategy Costs ($) QALYs Incremental Costs ($) Incremental QALYs  
## 1 Standard of care 74,429 15.18 <NA> NA  
## 2 Strategy AB 180,446 16.75 106,017 1.57  
## ICER ($/QALY) Status  
## 1 <NA> ND  
## 2 67,549 ND

## CEA frontier   
plot(df\_cea, label = "all", txtsize = 16) +  
 expand\_limits(x = max(table\_cea$QALYs) + 0.1) +  
 theme(legend.position = c(0.82, 0.3))



## CEA table in proper format   
table\_cea <- format\_table\_cea(df\_cea)  
table\_cea

## Strategy Costs ($) QALYs Incremental Costs ($) Incremental QALYs  
## 1 Standard of care 74,429 15.18 <NA> NA  
## 2 Strategy AB 180,446 16.75 106,017 1.57  
## ICER ($/QALY) Status  
## 1 <NA> ND  
## 2 67,549 ND

We kindly request you to add the following Acknowledgement paragraph to your further work where DARTH code formed the basis. We also like to remind you that you can add other sources of reference to this paragraph to acknowledge code you got from others.

# Acknowlegdement

For this work we made use of the template developed by the Decision Analysis in R for Technologies in Health (DARTH) workgroup: <http://darthworkgroup.com>.

The notation of our code is based on the following provided framework and coding convention: Alarid-Escudero, F., Krijkamp, E., Pechlivanoglou, P. et al. A Need for Change! A Coding Framework for Improving Transparency in Decision Modeling. PharmacoEconomics 37, 1329–1339 (2019). <https://doi.org/10.1007/s40273-019-00837-x>.

Other work from DARTH can be found on the website: <http://darthworkgroup.com/publications/>

# Copyright for assignment work

Copyright 2017, THE HOSPITAL FOR SICK CHILDREN AND THE COLLABORATING INSTITUTIONS.All rights reserved in Canada, the United States and worldwide. Copyright, trademarks, trade names and any and all associated intellectual property are exclusively owned by THE HOSPITAL FOR Sick CHILDREN and the collaborating institutions. These materials may be used, reproduced, modified, distributed and adapted with proper attribution.