Decision Analysis in R for Technologies in Health

Appendix B

Microsimulation modeling for health decision sciences using R: A tutorial.

March 2018

Fernando Alarid-Escudero, PhD[1]
Eva A. Enns, MS, PhD[1]
M.G. Myriam Hunink, MD, PhD[2,3]
Hawre J. Jalal, MD, PhD[4]
Eline M. Krijkamp, MSc[2]
Petros Pechlivanoglou, PhD[5]

Acknowledgements and attribution.
Please cite our papers when using any of the material
Jalal H, et al. An Overview of R in Health Decision Sciences. *Med. Dec. Mak*. 2017; 37(3): 735-746.
Krijkamp EM, et al. Microsimulation modeling for health decision sciences using R: A tutorial. Med Decis Making. 2018;38(3):400-22.

In collaboration of:
[1] University of Minnesota School of Public Health, Minneapolis, MN, USA
[2] Erasmus MC, Rotterdam, The Netherlands
[3] Harvard T.H. Chan School of Public Health, Boston, USA
[4] University of Pittsburgh Graduate School of Public Health, Pittsburgh, PA, USA
[5] The Hospital for Sick Children, Toronto and University of Toronto, Toronto ON, Canada

Appendix B - Microsimulation modeling for health decision sciences using R: A tutorial.

Krijkamp EM, Alarid-Escudero F, Enns EA, Jalal HJ, Hunink MGM, Pechlivanoglou P.

2018

This code forms the basis for the microsimulation model of the article: 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.

Please cite the article when using this code.

See GitHub for more information or code updates https://github.com/DARTH-git/Microsimulation-tutorial

To program this tutorial we made use of R: 3.3.0 GUI 1.68 Mavericks build (7202) RStudio: Version 1.0.136 2009-2016 RStudio, Inc.

Code of Appendix B

```
# rm(list = ls())  # remove any variables in R's memory
```

Model input

```
# Model input
n.i   <- 100000                # number of simulated individuals
n.t   <- 30                    # time horizon, 30 cycles
v.n   <- c("H","S1","S2","D")  # the model states: Healthy (H), Sick (S1), Si
cker (S2), Dead (D)
n.s   <- length(v.n)           # the number of health states
v.M_1 <- rep("H", n.i)         # everyone begins in the healthy state
d.c   <- d.e <- 0.03           # equal discounting of costs and QALYs by 3%
v.Trt <- c("No Treatment", "Treatment") # store the strategy names

# Transition probabilities (per cycle)
p.HD    <- 0.005               # 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
p.S1S2  <- 0.105                 # probability to become sicker when sick
rr.S1   <- 3                     # rate ratio of death when sick vs healthy
rr.S2   <- 10                    # rate ratio of death when sicker vs healthy
r.HD    <- -log(1 - p.HD)      # rate of death when healthy
r.S1D   <- rr.S1 * r.HD          # rate of death when sick
r.S2D   <- rr.S2 * r.HD          # rate of death when sicker
p.S1D   <- 1 - exp(- r.S1D)    # probability to die when sick
p.S2D   <- 1 - exp(- r.S2D)    # probability to die when sicker
```

```
rp.S1S2 <- 0.2                  # increase of the mortality rate with every ad
ditional year being sick

# Cost and utility inputs
c.H     <- 2000                 # cost of remaining one cycle healthy
c.S1    <- 4000                 # cost of remaining one cycle sick
c.S2    <- 15000                # cost of remaining one cycle sicker
c.Trt   <- 12000                # cost of treatment (per cycle)

u.H     <- 1                    # utility when healthy
u.S1    <- 0.75                 # utility when sick
u.S2    <- 0.5                  # utility when sicker
u.Trt   <- 0.95                 # utility when sick(er) and being treated
ru.S1S2 <- 0.03                 # decrease in utility of treated sick individu
als with every additional year being sick/sicker
v.x     <- runif(n.i, 0.95, 1.05) # vector capturing individuals' effect modi
fier at baseline
```

Functions

The `MicroSim` functions for the extended microsimulation of the 'Sick-Sicker' model keeps track of what happens to each individual during each cycle.

```
MicroSim <- function(v.M_1, n.i, n.t, v.n, X = NULL, d.c, d.e, TR.out = TRUE,
TS.out = TRUE, Trt = FALSE, seed = 1) {
# Arguments:
  # v.M_1:   vector of initial states for individuals
  # n.i:     number of individuals
  # n.t:     total number of cycles to run the model
  # v.n:     vector of health state names
  # X:       vector or matrix of individual characteristics
  # d.c:     discount rate for costs
  # d.e:     discount rate for health outcomes (QALYs)
  # TR.out:  should the output include a microsimulation trace? (default is T
RUE)
  # TS.out:  should the output include a matrix of transitions between states
? (default is TRUE)
  # Trt:     are the n.i individuals receiving treatment? (scalar with a Bool
ean value, default is FALSE)
  # seed:    starting seed number for random number generator (default is 1)
# Makes use of:
  # Probs:   function for the estimation of transition probabilities
  # Costs:   function for the estimation of cost state values
  # Effs:    function for the estimation of state specific health outcomes (Q
ALYs)

  v.dwc <- 1 / ((1 + d.c) ^ (0:n.t))   # calculate the cost discount weight b
ased on the discount rate d.c
  v.dwe <- 1 / ((1 + d.e) ^ (0:n.t))   # calculate the QALY discount weight b
```

```
ased on the discount rate d.e

  # create the matrix capturing the state name/costs/health outcomes for all
individuals at each time point
  m.M <- m.C <- m.E <- matrix(nrow = n.i, ncol = n.t + 1,
                             dimnames = list(paste("ind",   1:n.i, sep =" "
),
                                             paste("cycle", 0:n.t, sep =" "
)))

  m.M[, 1] <- v.M_1             # indicate the initial health state

  for (i in 1:n.i) {
    set.seed(seed + i)          # set the seed for every individual for the r
andom number generator

    # create the dur variable that stores the number of consecutive cycles th
e individual occupies either when sick or sicker
    dur <- 0                            # the individual start without histor
y
    m.C[i, 1] <- Costs(m.M[i, 1], Trt)  # estimate costs per individual for t
he initial health state conditional on treatment
    m.E[i, 1] <- Effs(m.M[i, 1], dur, Trt, X = X[i])  # estimate QALYs per in
dividual for the initial health state conditional on treatment, duration of b
eing sick/sicker and individual characteristics

    for (t in 1:n.t) {
      v.p <- Probs(m.M[i, t], dur)         # calculate the transition probabi
lities at cycle t conditional on the duration of being sick/sicker

      m.M[i, t + 1] <- sample(v.n, prob = v.p, size = 1)  # sample the new he
alth state and store that state in matrix m.M
      m.C[i, t + 1] <- Costs(m.M[i, t + 1], Trt)     # estimate the cost per
individual during cycle t + 1 conditional on treatment
      m.E[i, t + 1] <-  Effs(m.M[i, t + 1], dur, Trt, X = X[i])    # estimate
the utility per individual during cycle t + 1 conditional on treatment, durat
ion of being sick/sicker and individual characteristics

      if (m.M[i, t + 1] == "S1" | m.M[i, t + 1] == "S2") {  # expression to i
dentify sick/sicker individuals
        dur <- dur + 1   # update the duration of being sick/sicker
      } else {
        dur <- 0}        # reset duration variable

    } # close the loop for the time points
    if(i/100 == round(i/100,0)) {    # display the progress of the simulation
      cat('\r', paste(i/n.i * 100, "% done", sep = ""))
    }
  } # close the loop for the individuals
```

```
  tc <- m.C %*% v.dwc       # total (discounted) cost per individual
  te <- m.E %*% v.dwe       # total (discounted) QALYs per individual

  tc_hat <- mean(tc)        # average (discounted) cost
  te_hat <- mean(te)        # average (discounted) QALYs

  if (TS.out == TRUE) {  # create a  matrix of transitions across states
    TS <- paste(m.M, cbind(m.M[, -1], NA), sep = "->")  # transitions from on
e state to the other ###
    TS <- matrix(TS, nrow = n.i)
    rownames(TS) <- paste("Cycle", 0:n.t, sep = " ")    # name the rows of th
e matrix
    colnames(TS) <- paste("Ind",   1:n.s, sep = " ")    # name the columns of
the matrix
  } else {
    TS <- NULL
  }

  if (TR.out == TRUE) {  # create a trace from the individual trajectories
    TR <- t(apply(m.M, 2, function(x) table(factor(x, levels = v.n, ordered =
TRUE))))
    TR <- TR / n.i                                      # create a distribution
trace
    rownames(TR) <- paste("Cycle", 0:n.t, sep = " ")  # name the rows of the
matrix
    colnames(TR) <- v.n                                 # name the columns of t
he matrix
  } else {
    TR <- NULL
  }

  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, TS = TS, TR = TR)   # store the results from the sim
ulation in a list
  return(results)   # return the results
} # end of the MicroSim function
```

Probability function

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

```
Probs <- function(M_it, dur) {
  # M_it:   health state occupied by individual i at cycle t (character varia
ble)
  # dur:    the duration of being sick (sick/sicker)

  v.p.it <- rep(NA, n.s)     # create vector of state transition probabilitie
s
```

```
  names(v.p.it) <- v.n       # name the vector

  # update probabilities of death after first converting them to rates and ap
plying the rate ratio
  r.S1D <-  - log(1 - p.S1D)
  r.S2D <-  - log(1 - p.S2D)
  p.S1D <- 1 - exp(- r.S1D * (1 + dur * rp.S1S2)) # calculate p.S1D condition
al on duration of being sick/sicker
  p.S2D <- 1 - exp(- r.S2D * (1 + dur * rp.S1S2)) # calculate p.S2D condition
al on duration of being sick/sicker

  # update v.p.it with the appropriate probabilities
  v.p.it[M_it ==  "H"] <- c(1 - p.HS1 - p.HD, p.HS1, 0, p.HD)
# transition probabilities when healthy
  v.p.it[M_it == "S1"] <- c(p.S1H, 1- p.S1H - p.S1S2 - p.S1D, p.S1S2, p.S1D)
# transition probabilities when sick
  v.p.it[M_it == "S2"] <- c(0, 0, 1 - p.S2D, p.S2D)
# transition probabilities when sicker
  v.p.it[M_it ==  "D"] <- c(0, 0, 0, 1)
# transition probabilities when dead
  ifelse(sum(v.p.it) == 1, return(v.p.it), print("Probabilities do not sum to
1")) # return the transition probabilities or produce an error
}
```

Costs function

The `Costs` function estimates the costs at every cycle for the extended microsimulaton.

```
Costs <- function (M_it, Trt = FALSE) {
  # M_it: health state occupied by individual i at cycle t (character variabl
e)
  # Trt:  is the individual being treated? (default is FALSE)

  c.it <- 0                                  # by default the cost for every
one is zero
  c.it[M_it == "H"]  <- c.H                  # update the cost if healthy
  c.it[M_it == "S1"] <- c.S1 + c.Trt * Trt   # update the cost if sick condi
tional on treatment
  c.it[M_it == "S2"] <- c.S2 + c.Trt * Trt   # update the cost if sicker con
ditional on treatment
  return(c.it)                               # return the costs
}
```

Health outcome function

The `Effs` function to update the utilities at every cycle for the extended microsimulaton.

```
Effs <- function (M_it, dur, Trt = FALSE, cl = 1, X = NULL) {
  # M_it: health state occupied by individual i at cycle t (character variabl
e)
```

```
  # dur:  the duration of being sick/sicker
  # Trt:  is the individual being treated? (default is FALSE)
  # cl:   the cycle length (default = 1 )
  # X:    the vector or matrix of individual characteristics (optional)

  u.it               <- 0        # by default the utility for everyone is zer
o
  u.it[M_it == "H"]  <- u.H      # update the utility if healthy
  u.it[M_it == "S1"] <- X * Trt * (u.Trt - dur * ru.S1S2) + (1 - Trt) * u.S1
# update the utility if sick conditional on treatment and duration of being s
ick/sicker
  u.it[M_it == "S2"] <- u.S2     # update the utility if sicker
  QALYs <- u.it * cl             # calculate the QALYs during cycle t
  return(QALYs)                  # return the results
}
```

Run the simulation

```
sim_no_trt <- MicroSim(v.M_1, n.i, n.t, v.n, X = v.x, d.c, d.e, TS.out = FALS
E, TR.out = TRUE, Trt = FALSE) # run for no treatment
sim_trt    <- MicroSim(v.M_1, n.i, n.t, v.n, X = v.x, d.c, d.e, TS.out = FALS
E, TR.out = TRUE, Trt = TRUE) # run for treatment
```

Cost-effectiveness analysis

```
# store the mean costs (and the MCSE)of each strategy in a new variable C (ve
ctor costs)
v.C <- c(sim_no_trt$tc_hat, sim_trt$tc_hat)
se.C<- c(sd(sim_no_trt$tc), sd(sim_trt$tc)) / sqrt(n.i)
# store the mean QALYs (and the MCSE) of each strategy in a new variable E (v
ector health outcomes)
v.E <- c(sim_no_trt$te_hat, sim_trt$te_hat)
se.E<- c(sd(sim_no_trt$te), sd(sim_trt$te)) / sqrt(n.i)

delta.C <- v.C[2] - v.C[1]                    # calculate incremental costs
delta.E <- v.E[2] - v.E[1]                    # calculate incremental QALYs
se.delta.E <- sd(sim_trt$te - sim_no_trt$te) / sqrt(n.i) # Monte Carlo square
d error (MCSE) of incremental QALYS
se.delta.C <- sd(sim_trt$tc - sim_no_trt$tc) / sqrt(n.i) # Monte Carlo square
d error (MCSE) of incremental costs
ICER <- delta.C / delta.E                     # calculate the ICER
results <- c(delta.C, delta.E, ICER)          # store the values in a new vari
able
```

Create full incremental cost-effectiveness analysis table

```
table_micro <- data.frame(
  c(round(v.C, 0),  ""),           # costs per arm
  c(round(se.C, 0), ""),           # MCSE for costs
```

```
  c(round(v.E, 3),  ""),           # health outcomes per arm
  c(round(se.E, 3), ""),           # MCSE for health outcomes
  c("", round(delta.C, 0),   ""),  # incremental costs
  c("", round(se.delta.C, 0),""),  # MCSE for incremental costs
  c("", round(delta.E, 3),   ""),  # incremental QALYs
  c("", round(se.delta.E, 3),""),  # MCSE for health outcomes (QALYs) gained
  c("", round(ICER, 0),      "")   # ICER
)

rownames(table_micro) = c(v.Trt, "* are MCSE values")  # name the rows
colnames(table_micro) = c("Costs", "*",  "QALYs", "*", "Incremental Costs", "
*", "QALYs Gained", "*", "ICER") # name the columns

kable(table_micro)  # print the table
```

	Costs	*	QALYs	*	Incremental Costs	*	QALYs Gained	*	ICER
No Treatment	62667	120	15.279	0.017					
Treatment	117455	231	15.786	0.017	54787	117	0.508	0.001	107873

* are MCSE values