Cohort State-Transition Models 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 (6,7)

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

- 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. Feb;40(2):242-248. https://doi.org/10.1177/0272989X19893973

- Alarid-Escudero, F., Krijkamp, E. M., Enns, E. A., Hunink, M. G. M., Pechlivanoglou, P., & Jalal, H. (2020). Cohort state-transition models in R: From conceptualization to implementation. ArXiv:2001.07824v1, 1–31. http://arxiv.org/abs/2001.07824

Code of Appendix

Implements a time-independent Sick-Sicker cohort state-transition model (cSTM).

- *Standard of Care (SoC)*: current available care for the patients with the disease. This strategy reflects the natural history of the disease progression.
- *Strategy A*: treatment A is given to all sick patients, patients in the Sick and Sicker, but only improves the utility of those being sick.
- *Strategy B*: treatment B reduces disease progression from the Sick to Sicker states. However, it is not possible to distinguish those sick from sicker, and therefore all individuals in one of the two sick states get the treatment.
- *Strategy AB*: combines treatment A and treatment B. The disease progression is reduced, and Sick individuals have an improved utility.

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 install other packages
}
# load (install if required) packages from CRAN
pacman::p_load("dplyr", "data.table", "devtools", "scales", "ellipse", "ggplot2",
               "lazyeval", "igraph", "truncnorm", "ggraph", "reshape2", "knitr",
               "stringr", "gridExtra", "diagram", "dampack", "boot", "doParellel")
# load (install if required) packages from GitHub
# install_github("DARTH-git/darthtools", force = TRUE) Uncomment if there is a newer version
pacman::p_load_gh("DARTH-git/darthtools")
```

02 Load functions

```
source("Functions.R")
```

03 Input model parameters

```
# Strategy names
v_names_str <- c("Standard of care", # store the strategy names
                 "Strategy A",
                 "Strategy B",
                 "Strategy AB")

# Markov model parameters
age       <- 25                      # age at baseline
max_age   <- 100                     # maximum age of follow up
n_cycles <- max_age - age            # time horizon, number of cycles
# the 4 states of the model:
v_names_states <- c("H",  # Healthy (H)
                    "S1", # Sick (S1),
                    "S2", # Sicker (S2)
                    "D")  # Dead (D)
# initial cohort distribution (everyone allocated to the "healthy" state)
v_m_init <- c("H"  = 1,
              "S1" = 0,
              "S2" = 0,
              "D"  = 0)

## Transition probabilities (per cycle), hazard ratios
r_HD      <- 0.002 # constant rate of dying when Healthy (all-cause mortality)
p_HS1     <- 0.15  # probability to become Sick when Healthy conditional on surviving
p_S1H     <- 0.5   # probability to become Healthy when Sick conditional on surviving
p_S1S2    <- 0.105 # probability to become Sicker when Sick conditional on surviving
hr_S1     <- 3     # hazard ratio of death in Sick vs Healthy
```

```r
hr_S2    <- 10    # hazard ratio of death in Sicker vs Healthy

# Effectiveness of treatment B
hr_S1S2_trtB <- 0.6  # hazard ratio of becoming Sicker when Sick under treatment B

## State rewards
# Costs
c_H      <- 2000  # cost of remaining one cycle in Healthy
c_S1     <- 4000  # cost of remaining one cycle in Sick
c_S2     <- 15000 # cost of remaining one cycle in Sicker
c_D      <- 0     # cost of being dead (per cycle)
c_trtA   <- 12000 # cost of treatment A
c_trtB   <- 13000 # cost of treatment B
# Utilities
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_trtA   <- 0.95  # utility when being treated with A

n_str    <- length(v_names_str)     # number of strategies
n_states <- length(v_names_states)  # number of states

# Discounting factors
d_c      <- 0.03                    # discount rate for costs
d_e      <- 0.03                    # discount rate for QA

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

# Within-cycle correction (WCC) using Simpson's 1/3 rule
v_wcc    <- darthtools::gen_wcc(n_cycles = n_cycles,
                                method = "Simpson1/3") # vector of wcc

### Process model inputs
## Transition probabilities to the Dead state
# compute mortality rates
r_S1D    <- r_HD * hr_S1            # Mortality in the Sick state
r_S2D    <- r_HD * hr_S2            # Mortality in the Sick state
# transform rates to probabilities
p_HD     <- rate_to_prob(r_HD)      # Mortality risk in the Healthy state
p_S1D    <- rate_to_prob(r_S1D)     # Mortality risk in the Sick state
p_S2D    <- rate_to_prob(r_S2D)     # Mortality risk in the Sicker state

## Transition probability of becoming Sicker when Sick for treatment B
# transform probability to rate
r_S1S2      <- prob_to_rate(p = p_S1S2)
# apply hazard ratio to rate to obtain transition rate of becoming Sicker when
# Sick for treatment B
r_S1S2_trtB <- r_S1S2 * hr_S1S2_trtB
# transform rate to probability
p_S1S2_trtB <- rate_to_prob(r = r_S1S2_trtB) # probability to become Sicker when Sick
```

```
                                    # under treatment B conditional on surviving
```

Create a state-transition diagram of the cohort model

```
m_P_diag <- matrix(0,
                   nrow = n_states, ncol = n_states,
                   dimnames = list(v_names_states, v_names_states))
m_P_diag["H" , "S1"] = ""
m_P_diag["H" , "D" ] = ""
m_P_diag["H" , "H" ] = ""
m_P_diag["S1", "H" ] = ""
m_P_diag["S1", "S2"] = ""
m_P_diag["S1", "D" ] = ""
m_P_diag["S1", "S1"] = ""
m_P_diag["S2", "D" ] = ""
m_P_diag["S2", "S2"] = ""
m_P_diag["D", "D"  ] = ""
layout.fig <- c(3, 1)

plotmat(t(m_P_diag), t(layout.fig), self.cex = 0.5, curve = 0, arr.pos = 0.7,
        latex = T, arr.type = "curved", relsize = 0.9, box.prop = 0.8,
        cex = 0.8, box.cex = 0.9, lwd = 1)
```

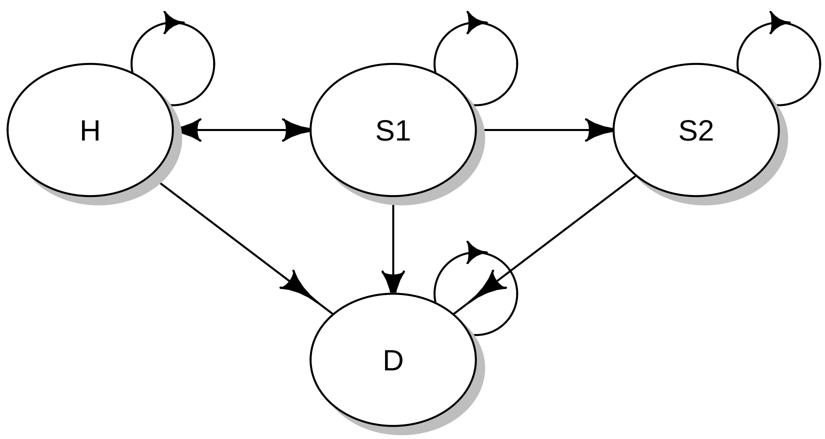

Figure 1: State-transition diagram of the time-independent Sick-Sicker cohort state-transition model.

04 Define and initialize matrices and vectors

04.1 Cohort trace

```
## Initialize cohort trace for SoC
m_M <- matrix(NA,
              nrow = (n_cycles + 1), ncol = n_states,
              dimnames = list(0:n_cycles, v_names_states))
# Store the initial state vector in the first row of the cohort trace
m_M[1, ] <- v_m_init
## Initialize cohort trace for strategies A, B, and AB
# Structure and initial states are the same as for SoC
```

```r
m_M_strA  <- m_M # Strategy A
m_M_strB  <- m_M # Strategy B
m_M_strAB <- m_M # Strategy AB
```

04.2 Transition probability matrices

```r
## Initialize transition probability matrix for strategy SoC
# all transitions to a non-death state are assumed to be conditional on survival
m_P <- matrix(0,
              nrow = n_states, ncol = n_states,
              dimnames = list(v_names_states,
                              v_names_states)) # define row and column names
m_P
```

```
##    H S1 S2 D
## H  0  0  0 0
## S1 0  0  0 0
## S2 0  0  0 0
## D  0  0  0 0
```

Fill in the transition probability matrix:

```r
# From H
m_P["H", "H"]   <- (1 - p_HD) * (1 - p_HS1)
m_P["H", "S1"]  <- (1 - p_HD) * p_HS1
m_P["H", "D"]   <- p_HD
# From S1
m_P["S1", "H"]  <- (1 - p_S1D) * p_S1H
m_P["S1", "S1"] <- (1 - p_S1D) * (1 - (p_S1H + p_S1S2))
m_P["S1", "S2"] <- (1 - p_S1D) * p_S1S2
m_P["S1", "D"]  <- p_S1D
# From S2
m_P["S2", "S2"] <- 1 - p_S2D
m_P["S2", "D"]  <- p_S2D
# From D
m_P["D", "D"]   <- 1

## Initialize transition probability matrix for strategy A as a copy of SoC's
m_P_strA <- m_P

## Initialize transition probability matrix for strategy B
m_P_strB <- m_P
# Update only transition probabilities from S1 involving p_S1S2
m_P_strB["S1", "S1"] <- (1 - p_S1D) * (1 - (p_S1H + p_S1S2_trtB))
m_P_strB["S1", "S2"] <- (1 - p_S1D) * p_S1S2_trtB

## Initialize transition probability matrix for strategy AB as a copy of B's
m_P_strAB <- m_P_strB

### Check if transition probability matrices are valid
## Check that transition probabilities are [0, 1]
check_transition_probability(m_P,       verbose = TRUE)
check_transition_probability(m_P_strA,  verbose = TRUE)
check_transition_probability(m_P_strB,  verbose = TRUE)
```

```
check_transition_probability(m_P_strAB, verbose = TRUE)
## Check that all rows sum to 1
check_sum_of_transition_array(m_P,       n_states = n_states, verbose = TRUE)
check_sum_of_transition_array(m_P_strA,  n_states = n_states, verbose = TRUE)
check_sum_of_transition_array(m_P_strB,  n_states = n_states, verbose = TRUE)
check_sum_of_transition_array(m_P_strAB, n_states = n_states, verbose = TRUE)
```

05 Run Markov model

```
# Iterative solution of time-independent cSTM
for(t in 1:n_cycles){
  # For SoC
  m_M[t + 1, ] <- m_M[t, ] %*% m_P
  # For strategy A
  m_M_strA[t + 1, ]  <- m_M_strA[t, ]  %*% m_P_strA
  # For strategy B
  m_M_strB[t + 1, ]  <- m_M_strB[t, ]  %*% m_P_strB
  # For strategy AB
  m_M_strAB[t + 1, ] <- m_M_strAB[t, ] %*% m_P_strAB
}

## Store the cohort traces in a list
l_m_M <- list(m_M,
              m_M_strA,
              m_M_strB,
              m_M_strAB)
names(l_m_M) <- v_names_str
```

06 Compute and Plot Epidemiological Outcomes

06.1 Cohort trace

```
# create a plot of the data
plot_trace(l_m_M$`Standard of care`)
```

06.2 Overall Survival (OS)

```
# calculate the overall survival (OS) probability for no treatment
v_os_SoC <- 1 - l_m_M$`Standard of care`[, "D"]
# alternative way of calculating the OS probability
v_os_SoC <- rowSums(l_m_M$`Standard of care`[, 1:3])
# create a simple plot showing the OS
plot(0:n_cycles, v_os_SoC, type = 'l',
     ylim = c(0, 1),
     ylab = "Survival probability",
     xlab = "Cycle",
     main = "Overall Survival")
# add grid
grid(nx = n_cycles, ny = 10, col = "lightgray", lty = "dotted", lwd = par("lwd"),
     equilogs = TRUE)
```

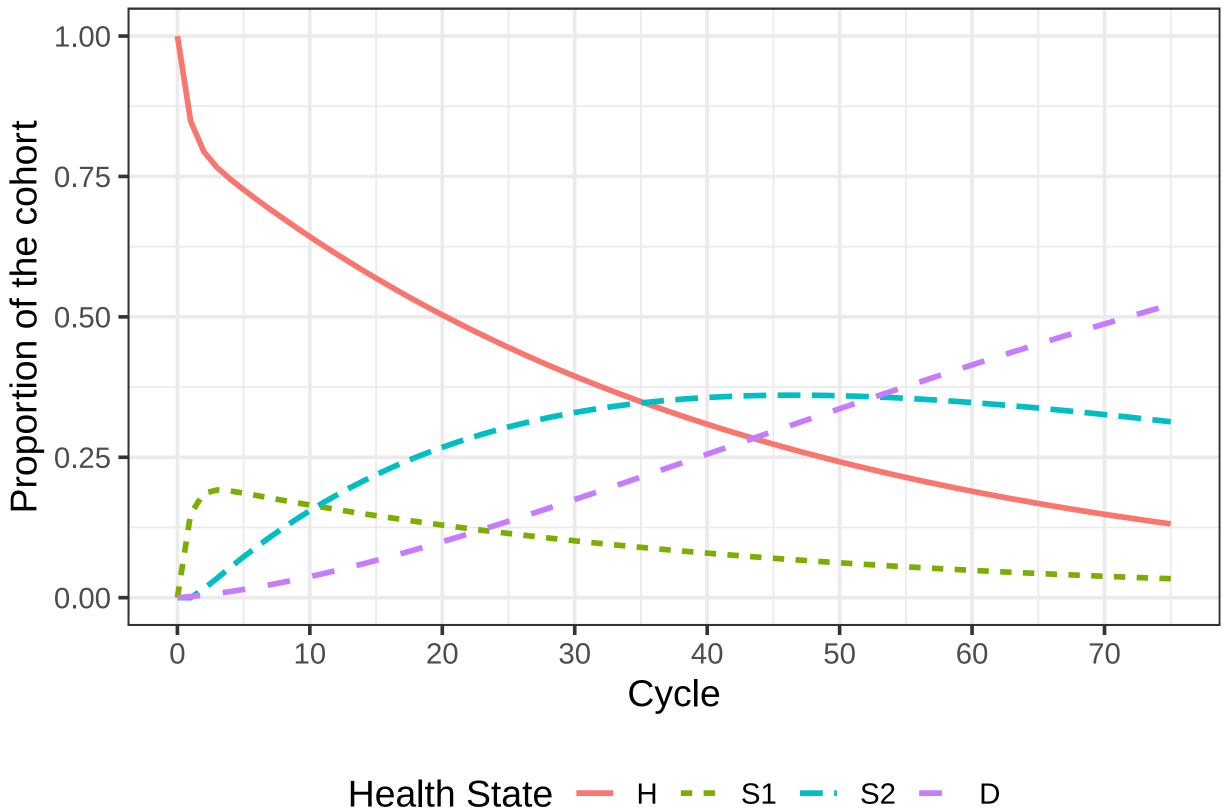

Figure 2: Cohort trace of the time-independent cSTM under standard of care

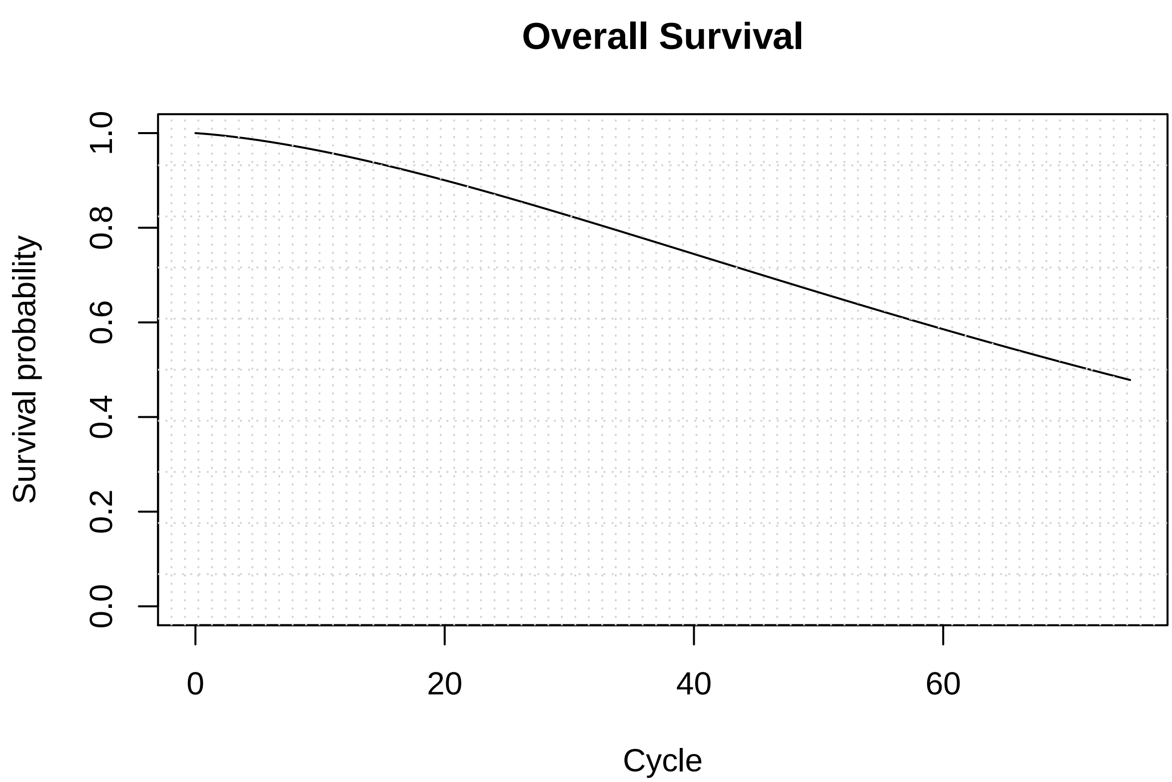

Figure 3: Overall survival of the time-independent cSTM under standard of care.

06.2.1 Life Expectancy (LE)

```
v_le <- sum(v_os_SoC)  # summing probability of OS over time  (i.e. life expectancy)
```

06.3 Disease prevalence

```
v_prev <- rowSums(l_m_M$`Standard of care`[, c("S1", "S2")]) / v_os_SoC
plot(v_prev,
     ylim = c(0, 1),
     ylab = "Prevalence",
     xlab = "Cycle",
     main = "Disease prevalence")
```

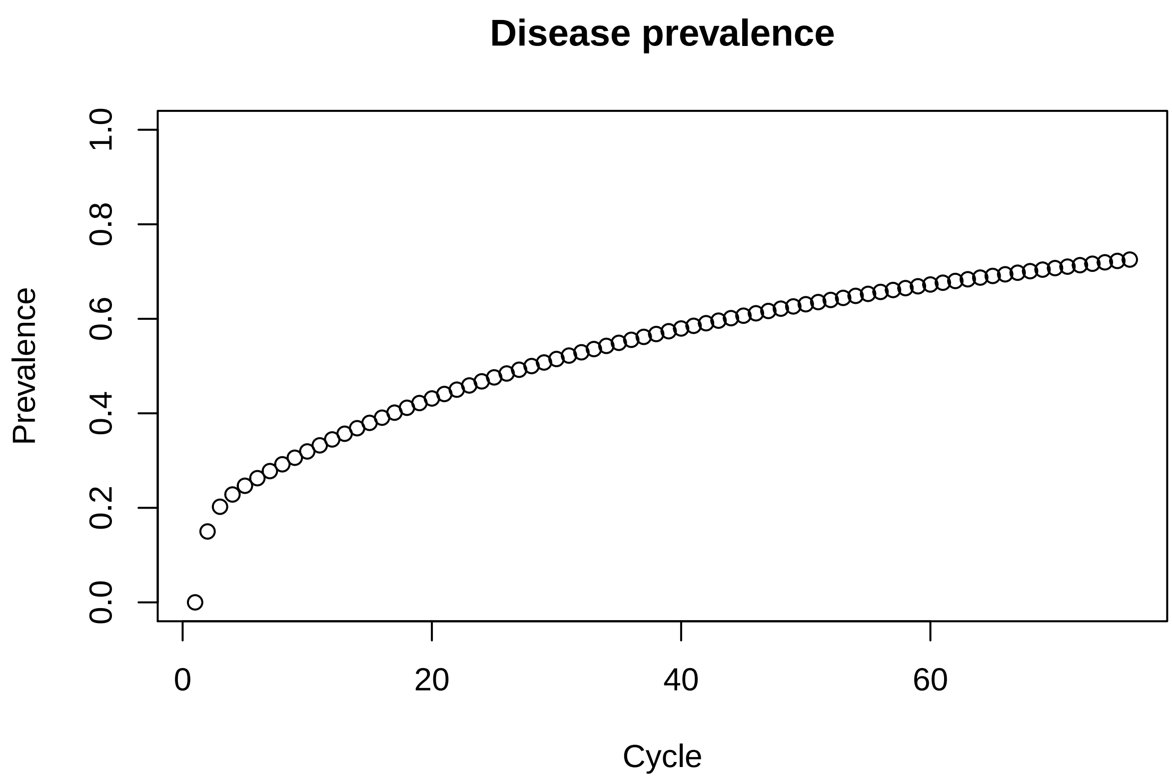

Figure 4: Disease prevalence of the time-independent cSTM under standard of care.

06.4 Proportion of sick in S1 state

```
v_prop_S1 <- l_m_M$`Standard of care`[, "S1"] / v_prev
plot(0:n_cycles, v_prop_S1,
     xlab = "Cycle",
     ylab = "Proportion",
     main = "Proportion of sick in S1 state",
     col  = "black", type = "l")
```

07 Compute Cost-Effectiveness Outcomes

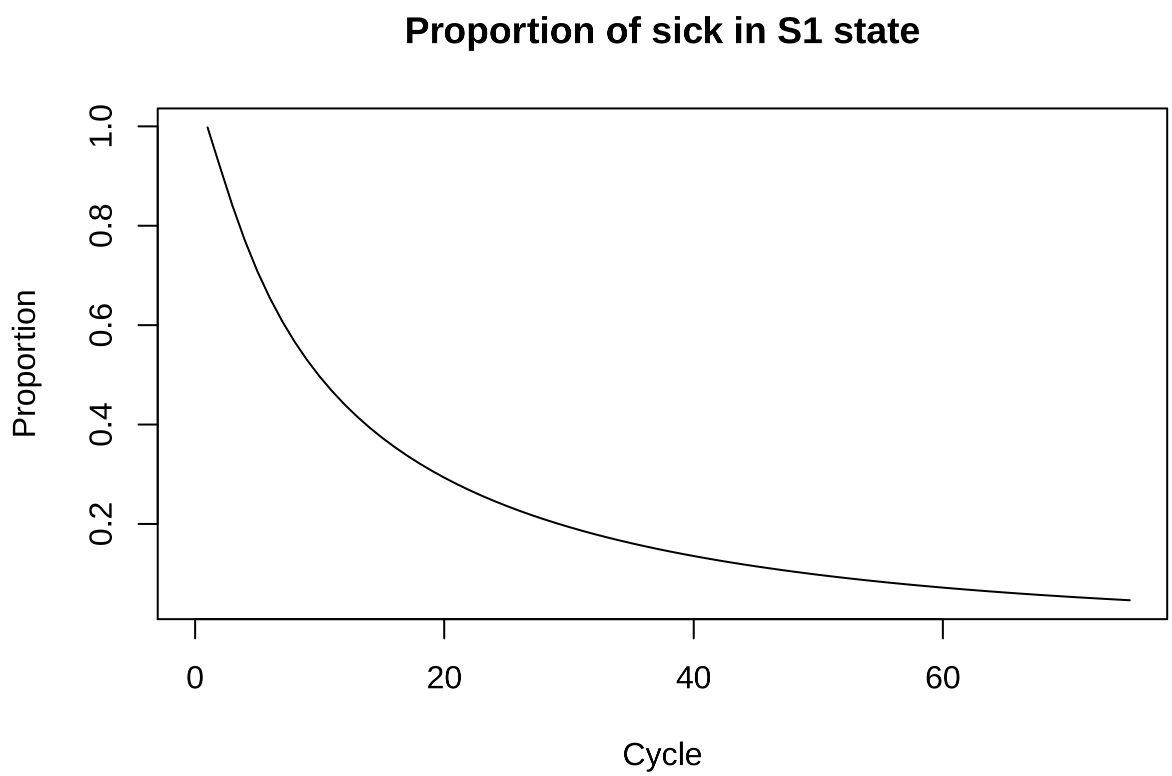

Figure 5: Proportion of sick in S1 state of the time-independent cSTM under standard of care.

07.1 State rewards for each strategy

```
## Vector of state utilities under strategy SoC
v_u_SoC    <- c(H  = u_H,
                S1 = u_S1,
                S2 = u_S2,
                D  = u_D)
## Vector of state costs under strategy SoC
v_c_SoC    <- c(H  = c_H,
                S1 = c_S1,
                S2 = c_S2,
                D  = c_D)
## Vector of state utilities under strategy A
v_u_strA   <- c(H  = u_H,
                S1 = u_trtA,
                S2 = u_S2,
                D  = u_D)
## Vector of state costs under strategy A
v_c_strA   <- c(H  = c_H,
                S1 = c_S1 + c_trtA,
                S2 = c_S2 + c_trtA,
                D  = c_D)
## Vector of state utilities under strategy B
v_u_strB   <- c(H  = u_H,
                S1 = u_S1,
                S2 = u_S2,
```

```
                D  = u_D)
## Vector of state costs under strategy B
v_c_strB   <- c(H  = c_H,
                S1 = c_S1 + c_trtB,
                S2 = c_S2 + c_trtB,
                D  = c_D)
## Vector of state utilities under strategy AB
v_u_strAB  <- c(H  = u_H,
                S1 = u_trtA,
                S2 = u_S2,
                D  = u_D)
## Vector of state costs under strategy AB
v_c_strAB  <- c(H  = c_H,
                S1 = c_S1 + (c_trtA + c_trtB),
                S2 = c_S2 + (c_trtA + c_trtB),
                D  = c_D)

## Store the vectors of state utilities for each strategy in a list
l_u   <- list(SQ = v_u_SoC,
              A  = v_u_strA,
              B  = v_u_strB,
              AB = v_u_strAB)
## Store the vectors of state cost for each strategy in a list
l_c   <- list(SQ = v_c_SoC,
              A  = v_c_strA,
              B  = v_c_strB,
              AB = v_c_strAB)

# assign strategy names to matching items in the lists
names(l_u) <- names(l_c) <- v_names_str
```

07.2 Mean Costs and QALYs for each strategy

```
# create empty vectors to store total utilities and costs
v_tot_qaly <- v_tot_cost <- vector(mode = "numeric", length = n_str)
names(v_tot_qaly) <- names(v_tot_cost) <- v_names_str

#### Loop through each strategy and calculate total utilities and costs ####
for (i in 1:n_str) {
  v_u_str <- l_u[[i]] # select the vector of state utilities for the i-th strategy
  v_c_str <- l_c[[i]] # select the vector of state costs for the i-th strategy

  #### Expected QALYs and costs per cycle ####
  ### Vector of QALYs and Costs
  ## Apply state rewards ###
  v_qaly_str <- l_m_M[[i]] %*% v_u_str # sum the utilities of all states for each cycle
  v_cost_str <- l_m_M[[i]] %*% v_c_str # sum the costs of all states for each cycle

  #### Discounted total expected QALYs and Costs per strategy and apply half-cycle correction ####
  ## QALYs
  v_tot_qaly[i] <- t(v_qaly_str) %*% (v_dwe * v_wcc)
  ## Costs
  v_tot_cost[i] <- t(v_cost_str) %*% (v_dwc * v_wcc)
```

```
}
```

07.3 Compute ICERs of the Markov model

```
# Calculate incremental cost-effectiveness ratios (ICERs)
df_cea <- calculate_icers(cost       = v_tot_cost,
                          effect     = v_tot_qaly,
                          strategies = v_names_str)
df_cea
```

```
##                          Strategy     Cost   Effect  Inc_Cost Inc_Effect
## Standard of care Standard of care 148657.5 20.99026        NA         NA
## Strategy B             Strategy B 248570.6 22.48240  99913.15  1.4921371
## Strategy AB           Strategy AB 361341.5 23.35420 112770.85  0.8718013
## Strategy A             Strategy A 275936.5 21.71749        NA         NA
##                       ICER Status
## Standard of care        NA     ND
## Strategy B        66959.76     ND
## Strategy AB      129353.84     ND
## Strategy A              NA      D
```

07.4 CEA results

```
# Create CEA table in proper format
table_cea <- format_table_cea(df_cea)
table_cea
```

```
##                          Strategy Costs ($) QALYs Incremental Costs ($)
## Standard of care Standard of care   148,657 20.99                  <NA>
## Strategy B             Strategy B   248,571 22.48                99,913
## Strategy AB           Strategy AB   361,341 23.35               112,771
## Strategy A             Strategy A   275,937 21.72                  <NA>
##                  Incremental QALYs ICER ($/QALY) Status
## Standard of care                NA          <NA>     ND
## Strategy B                    1.49        66,960     ND
## Strategy AB                   0.87       129,354     ND
## Strategy A                      NA          <NA>      D
```

07.5 Plot frontier of the Markov model

```
plot(df_cea, label = "all") +
  expand_limits(x = max(table_cea$QALYs) + 0.5)
```

08 Deterministic Sensitivity Analysis

08.1 List of input parameters

Create list `l_params_all` with all input probabilities, cost and utilities.

```
l_params_all <- list(
  # Transition probabilities (per cycle), hazard ratios
  r_HD   = 0.002, # constant rate of dying when Healthy (all-cause mortality)
  p_HS1  = 0.15,  # probability to become Sick when Healthy conditional on surviving
```

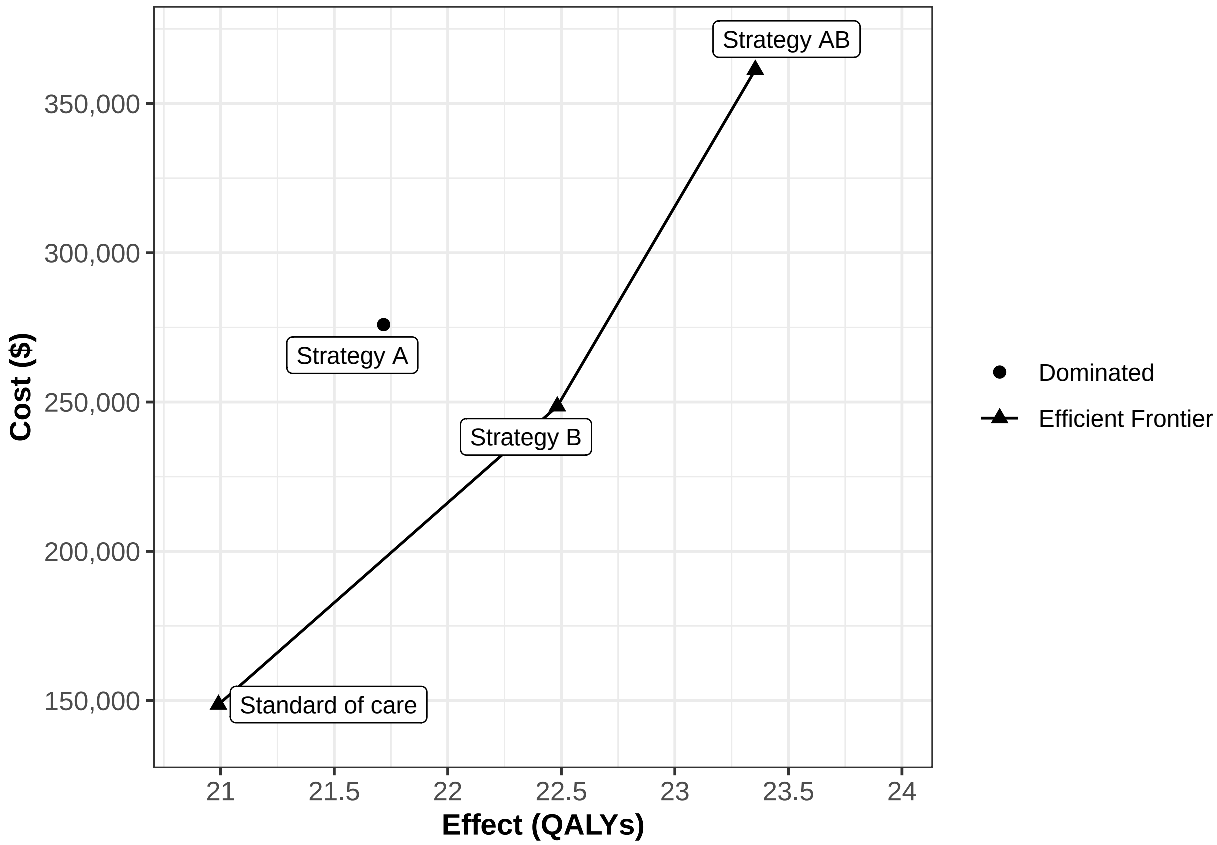

Figure 6: Cost-effectiveness efficient frontier for the time-independent Sick-Sicker model.

```r
  p_S1H   = 0.5,   # probability to become Healthy when Sick conditional on surviving
  p_S1S2 = 0.105, # probability to become Sicker when Sick conditional on surviving
  hr_S1   = 3,     # hazard ratio of death in Sick vs Healthy
  hr_S2   = 10,    # hazard ratio of death in Sicker vs Healthy
  # Effectiveness of treatment B
  hr_S1S2_trtB = 0.6, # hazard ratio of becoming Sicker when Sick under B under treatment B
  ## State rewards
  # Costs
  c_H     = 2000,  # cost of remaining one cycle in Healthy
  c_S1    = 4000,  # cost of remaining one cycle in Sick
  c_S2    = 15000, # cost of remaining one cycle in Sicker
  c_D     = 0,     # cost of being dead (per cycle)
  c_trtA = 12000, # cost of treatment A
  c_trtB = 13000, # cost of treatment B
  # Utilities
  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_trtA = 0.95    # utility when being treated with A
)

# store the parameter names into a vector
v_names_params <- names(l_params_all)
```

08.2 Load Sick-Sicker Markov model function

```r
source("Functions_markov_sick-sicker_intro_tutorial.R")
# Test function to compute CE outcomes
calculate_ce_out(l_params_all)
```

```
##                            Strategy     Cost   Effect     NMB
## Standard of care Standard of care 148657.5 20.99026 1950369
## Strategy A               Strategy A 275936.5 21.71749 1895812
## Strategy B               Strategy B 248570.6 22.48240 1999669
## Strategy AB             Strategy AB 361341.5 23.35420 1974079
```

08.3 One-way sensitivity analysis (OWSA)

```r
options(scipen = 999) # disabling scientific notation in R
# data.frame containing all parameters, their base-case values, and the min and
# max values of the parameters of interest
df_params_owsa <- data.frame(pars = c("hr_S1S2_trtB", "c_trtA", "u_S1", "u_trtA"),
                             min  = c(0.10,  6000 , 0.65, 0.80), # min parameter values
                             max  = c(1.00, 18000 , 0.85, 0.98)  # max parameter values
                             )

owsa_nmb  <- run_owsa_det(params_range     = df_params_owsa,   # data.frame with parameters for OWSA
                          params_basecase = l_params_all,      # list with all parameters
                          nsamp            = 100,              # number of parameter values
                          FUN              = calculate_ce_out, # function to compute outputs
                          outcomes         = c("NMB"),         # output to do the OWSA on
                          strategies       = v_names_str,      # names of the strategies
```

```
                            n_wtp          = 120000)          # extra argument to pass to FUN
```

```
##    |                                                                                  |
```

```
plot(owsa_nmb, txtsize = 10, n_x_ticks = 4,
     facet_scales = "free") +
     theme(legend.position = "bottom")
```

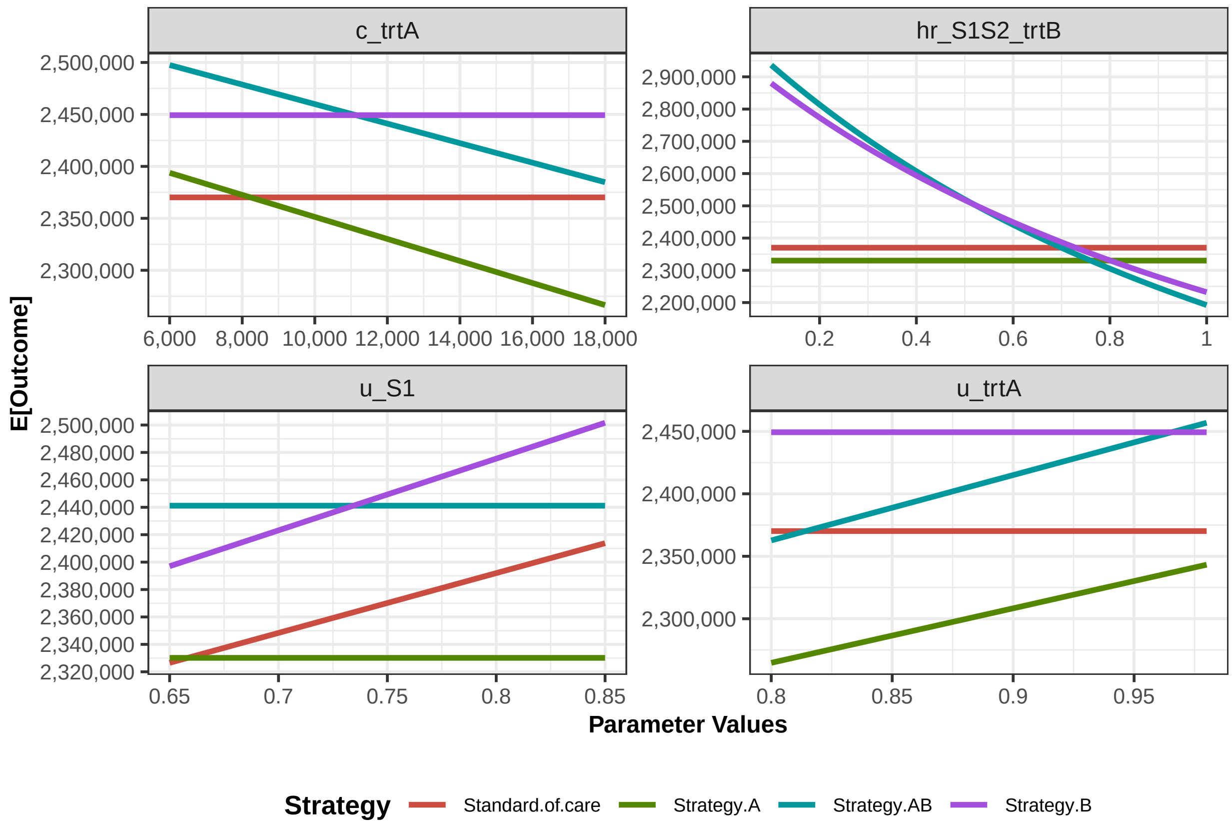

Figure 7: One-way sensitivity analysis for the time-independent Sick-Sicker model.

08.3.2 Optimal strategy with OWSA

```
owsa_opt_strat(owsa = owsa_nmb, txtsize = 10)
```

08.3.3 Tornado plot

```
owsa_tornado(owsa = owsa_nmb, txtsize = 10)
```

08.4 Two-way sensitivity analysis (TWSA)

```
# dataframe containing all parameters, their basecase values, and the min and
# max values of the parameters of interest
df_params_twsa <- data.frame(pars = c("hr_S1S2_trtB", "u_trtA"),
                             min  = c(0.10, 0.80),  # min parameter values
                             max  = c(1.00, 0.98)   # max parameter values
                             )
```

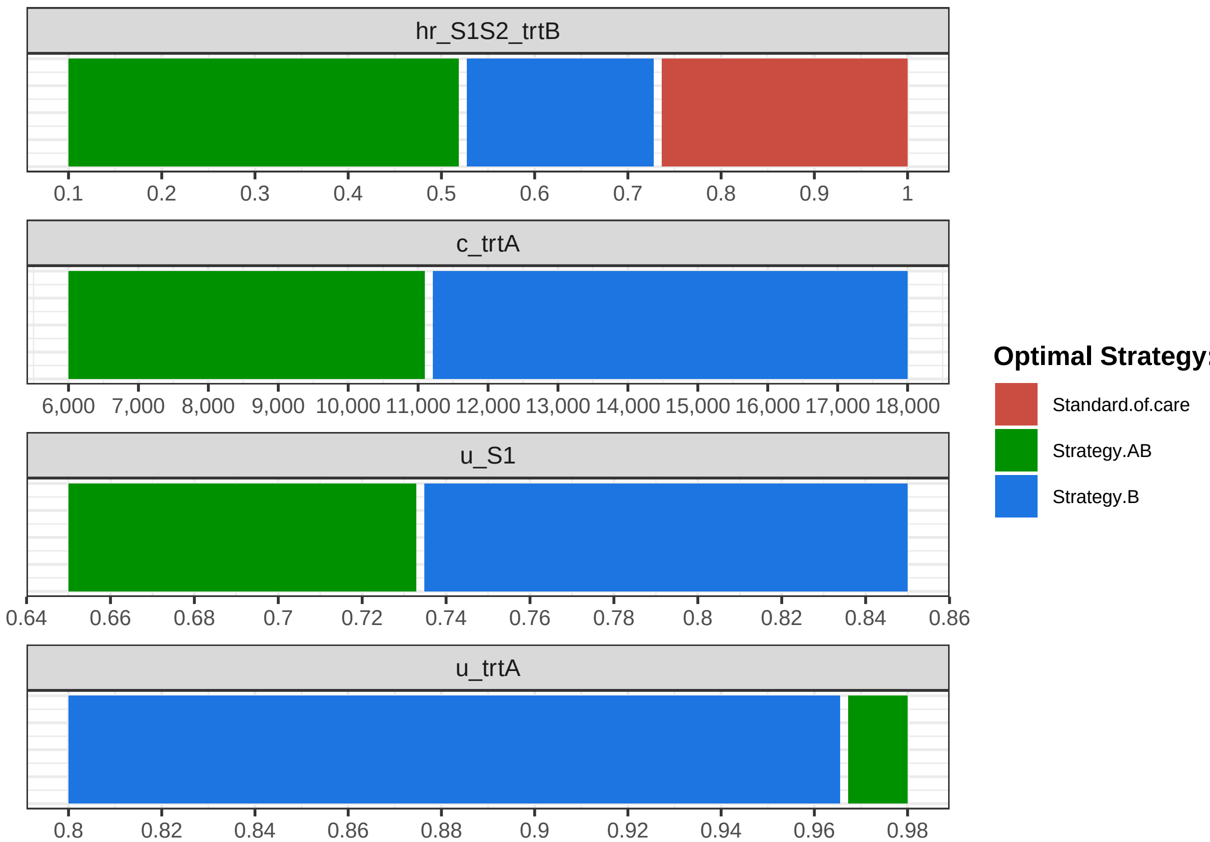

Figure 8: Optimal strategy with one-way sensitivity analysis for the time-independent Sick-Sicker model.

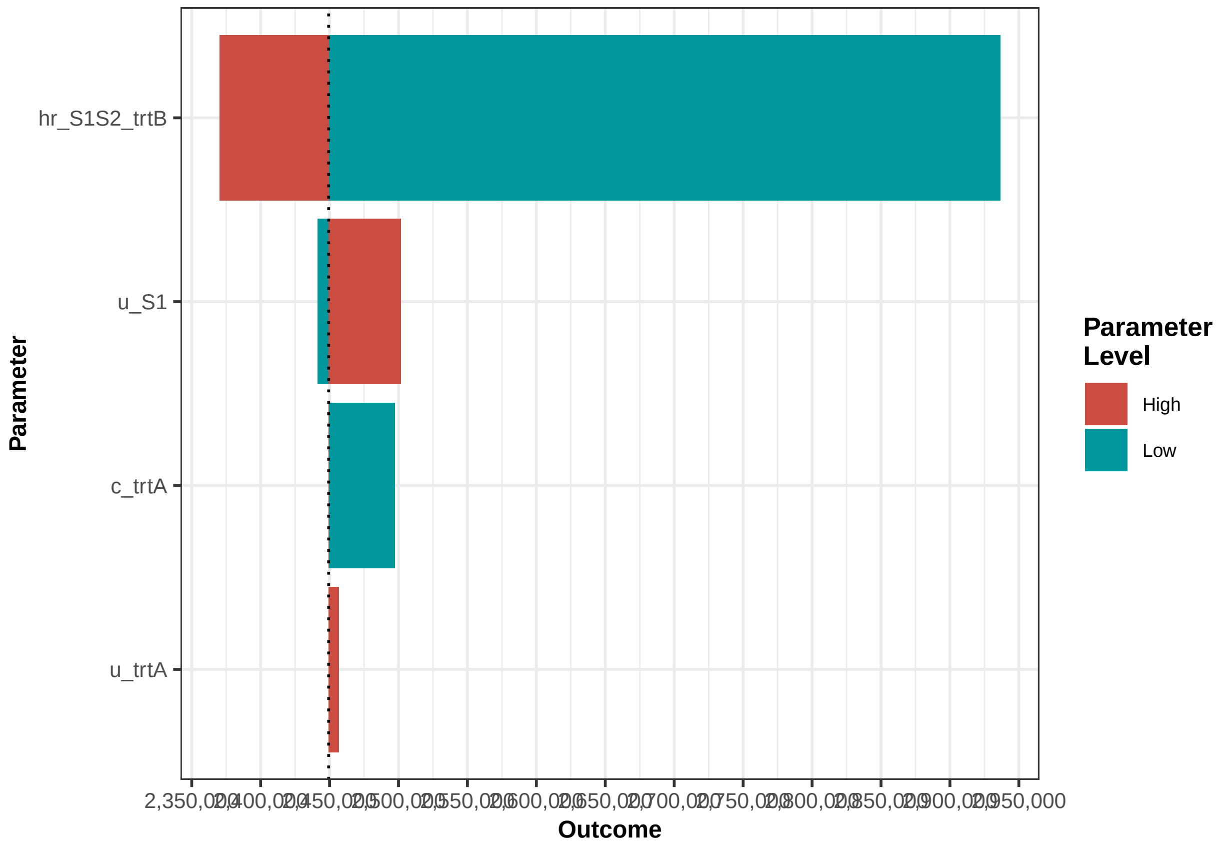

Figure 9: Tornado plot for the time-independent Sick-Sicker model.

```
twsa_nmb <- run_twsa_det(params_range = df_params_twsa,       # data.frame with parameters for TWSA
                         params_basecase = l_params_all,      # list with all parameters
                         nsamp           = 40,                # number of parameter values
                         FUN             = calculate_ce_out, # function to compute outputs
                         outcomes        = c("NMB"),          # output to do the TWSA on
                         strategies      = v_names_str,       # names of the strategies
                         n_wtp           = 120000)            # extra argument to pass to FUN
```

```
##   |                                                                                |
```

```
plot(twsa_nmb)
```

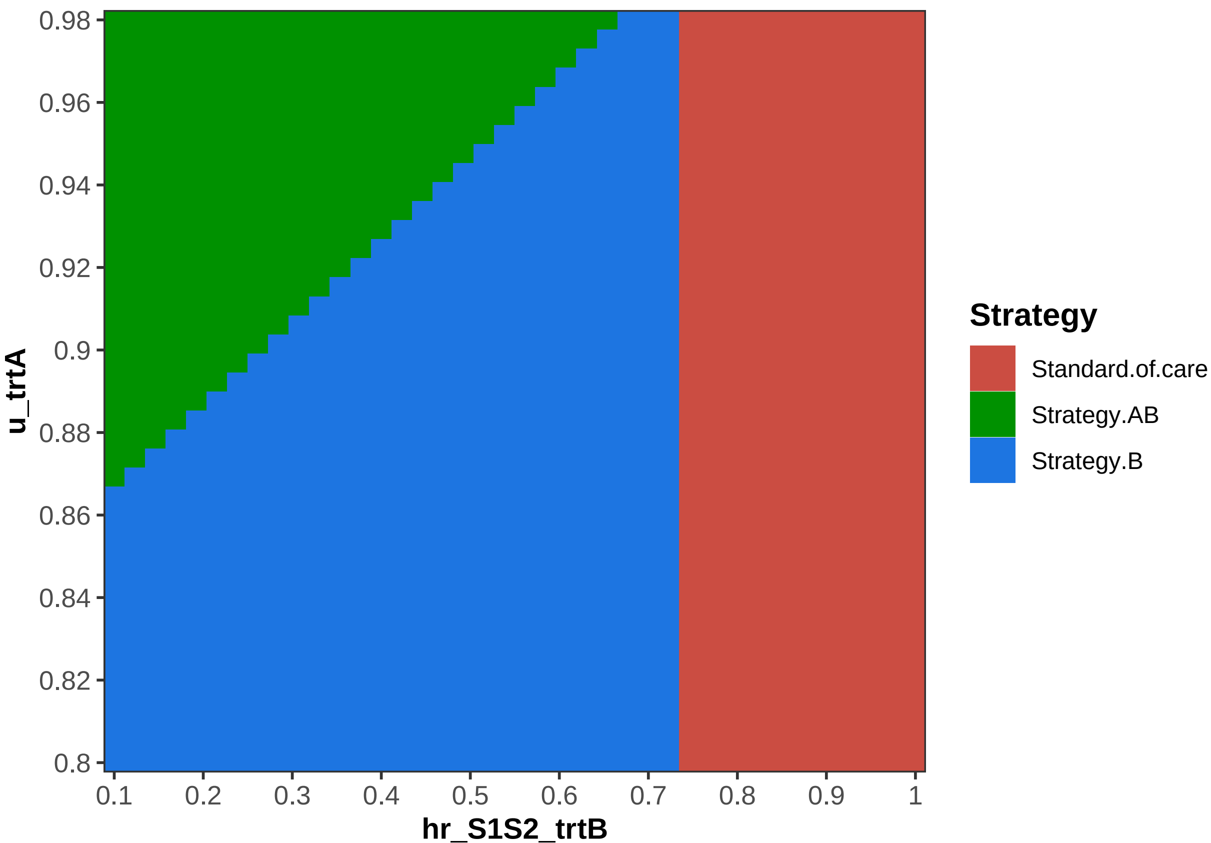

Figure 10: Two-way sensitivity analysis for the time-independent Sick-Sicker model.

09 Probabilistic Sensitivity Analysis (PSA)

```
# Function to generate PSA input dataset
generate_psa_params <- function(n_sim = 1000, seed = 071818){
  set.seed(seed) # set a seed to be able to reproduce the same results
  df_psa <- data.frame(
    # Transition probabilities (per cycle), hazard ratios
    r_HD    = rlnorm(n_sim, meanlog = log(0.002), sdlog = 0.01), # constant rate of dying when Healthy
```

```
    p_HS1    = rbeta(n_sim, shape1 = 30, shape2 = 170),        # probability to become Sick when Healthy
    p_S1H    = rbeta(n_sim, shape1 = 60, shape2 = 60) ,        # probability to become Healthy when Sick
    p_S1S2   = rbeta(n_sim, shape1 = 84, shape2 = 716),        # probability to become Sicker when Sick
    hr_S1    = rlnorm(n_sim, meanlog = log(3),  sdlog = 0.01), # hazard ratio of death in Sick vs Healthy
    hr_S2    = rlnorm(n_sim, meanlog = log(10), sdlog = 0.02), # hazard ratio of death in Sicker vs Heal

    # Effectiveness of treatment B
    hr_S1S2_trtB = rlnorm(n_sim, meanlog = log(0.6), sdlog = 0.02), # hazard ratio of becoming Sicker w

    # State rewards
    # Costs
    c_H      = rgamma(n_sim, shape = 100, scale = 20),     # cost of remaining one cycle in Healthy
    c_S1     = rgamma(n_sim, shape = 177.8, scale = 22.5), # cost of remaining one cycle in Sick
    c_S2     = rgamma(n_sim, shape = 225, scale = 66.7),   # cost of remaining one cycle in Sicker
    c_D      = 0,                                          # cost of being dead (per cycle)
    c_trtA   = rgamma(n_sim, shape = 73.5, scale = 163.3), # cost of treatment A
    c_trtB   = rgamma(n_sim, shape = 86.2, scale = 150.8), # cost of treatment B

    # Utilities
    u_H      = rbeta(n_sim, shape1 = 200, shape2 = 3),     # utility when Healthy
    u_S1     = rbeta(n_sim, shape1 = 130, shape2 = 45),    # utility when Sick
    u_S2     = rbeta(n_sim, shape1 = 230, shape2 = 230),   # utility when Sicker
    u_D      = 0,                                          # utility when Dead
    u_trtA   = rbeta(n_sim, shape1 = 300, shape2 = 15)     # utility when being treated with A
  )
  return(df_psa)
}

# Try it
generate_psa_params(10)
```

```
##           r_HD      p_HS1     p_S1H     p_S1S2    hr_S1     hr_S2 hr_S1S2_trtB
## 1  0.002028152 0.09824029 0.5127637 0.09753778 3.055410 10.148768    0.5937824
## 2  0.002000921 0.14028588 0.5472914 0.10372958 3.005252  9.928044    0.5950295
## 3  0.001982103 0.11912372 0.5563895 0.10145687 2.982392 10.021998    0.6131617
## 4  0.002004616 0.16814566 0.4513392 0.13328700 2.981918 10.167486    0.6066259
## 5  0.001973180 0.12386074 0.5451874 0.11709463 3.024975 10.006505    0.5839462
## 6  0.001994982 0.20236810 0.4959245 0.09820372 2.971345 10.012715    0.5861758
## 7  0.002029022 0.12808209 0.4467345 0.10922783 2.989181 10.001860    0.5838734
## 8  0.001995376 0.17956555 0.5889240 0.11869605 2.984080  9.919332    0.6081428
## 9  0.001986669 0.14451930 0.4441287 0.09900264 2.988974 10.142331    0.5784071
## 10 0.002004075 0.19389383 0.5076377 0.07977953 2.968692  9.749869    0.6141500
##         c_H     c_S1     c_S2 c_D    c_trtA   c_trtB       u_H      u_S1
## 1  1738.307 3448.146 15401.50   0 11760.17 12545.42 0.9936986 0.7921002
## 2  1934.791 4656.666 15198.97   0 10216.98 16021.51 0.9890348 0.8069535
## 3  2113.222 3879.536 14070.22   0 13948.46 11876.24 0.9943986 0.7796555
## 4  2123.062 3911.558 16006.75   0 12209.42 13905.64 0.9903866 0.7634364
## 5  2387.026 4497.693 15402.85   0 13348.28 13951.11 0.9922080 0.6960808
## 6  1600.231 4198.111 16322.33   0 10982.80 14379.26 0.9923096 0.7201886
## 7  1916.467 4087.810 15848.52   0 11359.07 13869.89 0.9951053 0.7368862
## 8  1977.534 4088.065 16378.74   0 15146.19 14538.77 0.9838664 0.7871166
## 9  1785.850 4110.476 15114.98   0 12174.76 14097.89 0.9789327 0.7801267
## 10 1852.581 4181.534 15081.51   0 10827.73 11962.58 0.9888362 0.6948902
##         u_S2 u_D    u_trtA
```

```
## 1  0.5120295   0 0.9618487
## 2  0.5596476   0 0.9588147
## 3  0.5227040   0 0.9500272
## 4  0.4972854   0 0.9503212
## 5  0.4890811   0 0.9527867
## 6  0.5479399   0 0.9474899
## 7  0.4880988   0 0.9272372
## 8  0.5286497   0 0.9504067
## 9  0.5192718   0 0.9562053
## 10 0.4905318   0 0.9443718
```

```r
# Number of simulations
n_sim <- 1000

# Generate PSA input dataset
df_psa_input <- generate_psa_params(n_sim = n_sim)
# First six observations
head(df_psa_input)
```

```
##          r_HD     p_HS1     p_S1H     p_S1S2    hr_S1     hr_S2 hr_S1S2_trtB
## 1 0.002028152 0.1602643 0.4968849 0.09399301 3.021339 10.423829    0.6174156
## 2 0.002000921 0.1185122 0.5553071 0.09355730 2.954005 10.122297    0.5953304
## 3 0.001982103 0.1239255 0.5275639 0.11118352 3.040920 10.466481    0.6037374
## 4 0.002004616 0.1823176 0.5340465 0.10680270 3.008736  9.836515    0.6013755
## 5 0.001973180 0.1250214 0.5540891 0.08518676 3.009890 10.315554    0.5977386
## 6 0.001994982 0.1437542 0.4900105 0.08858359 2.983996 10.204962    0.5847355
##        c_H      c_S1     c_S2 c_D   c_trtA   c_trtB       u_H      u_S1
## 1 2024.100 4111.210 15516.92   0 14536.17 12753.80 0.9884166 0.7043466
## 2 2269.089 3825.014 15036.51   0 11084.58 15073.93 0.9782958 0.8001897
## 3 2020.638 4091.684 14959.30   0 12668.41 14888.98 0.9752874 0.7430207
## 4 1945.772 3989.084 13147.33   0 13583.26 11832.31 0.9856703 0.7422257
## 5 2089.839 3663.629 15669.36   0 13687.70 10075.88 0.9811165 0.7274420
## 6 2107.511 3553.373 15294.76   0 12141.25 12343.99 0.9894955 0.7223727
##        u_S2 u_D    u_trtA
## 1 0.5079267   0 0.9745802
## 2 0.5128133   0 0.9452539
## 3 0.4741263   0 0.9521755
## 4 0.4981919   0 0.9490015
## 5 0.4955725   0 0.9571041
## 6 0.5264675   0 0.9504759
```

```r
# Histogram of parameters
ggplot(melt(df_psa_input, variable.name = "Parameter"), aes(x = value)) +
  facet_wrap(~Parameter, scales = "free") +
  geom_histogram(aes(y = ..density..)) +
  scale_x_continuous(breaks = scales::pretty_breaks(n = 3)) +
  theme_bw(base_size = 16) +
  theme(axis.text = element_text(size=6))
```

09.1 Conduct probabilistic sensitivity analysis

```r
# Initialize data.frames with PSA output
# data.frame of costs
df_c <- as.data.frame(matrix(0,
                             nrow = n_sim,
```

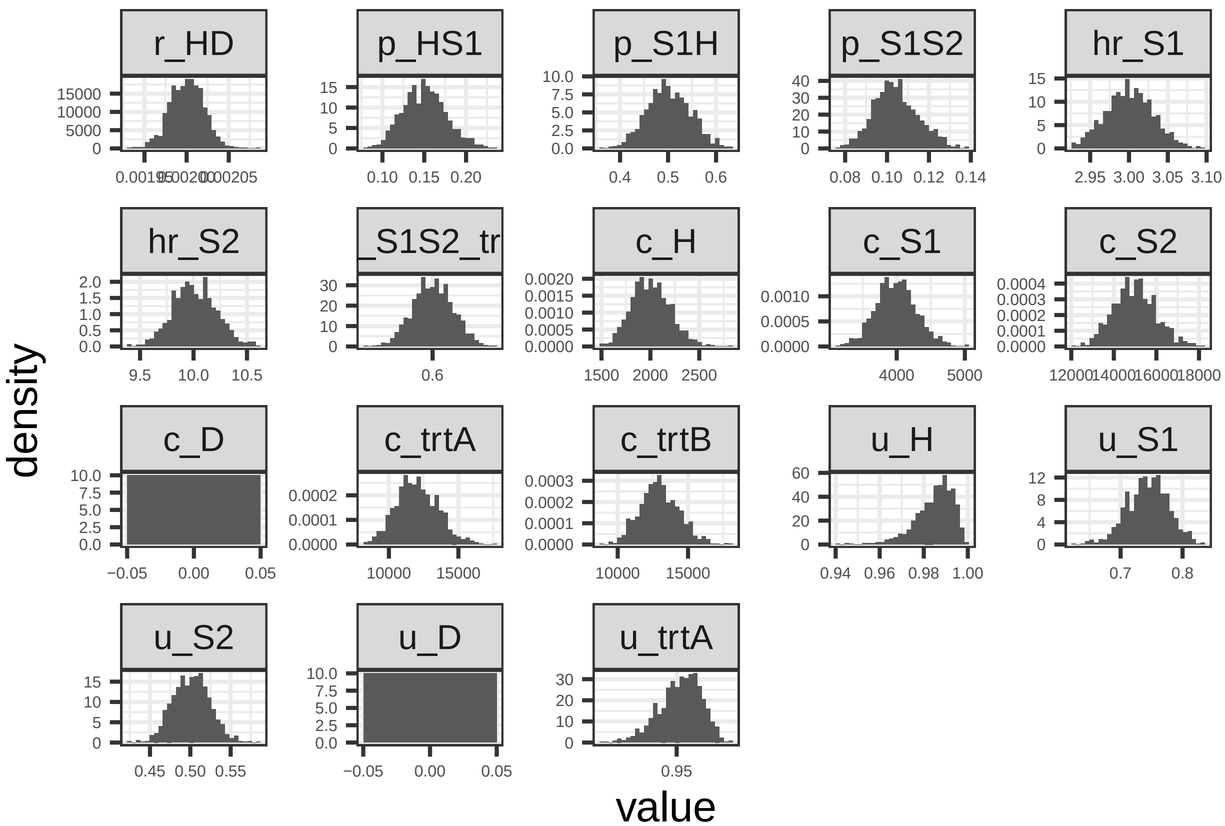

Figure 11: Histogram of parameters distributions for the time-independent Sick-Sicker model.

```
                              ncol = n_str))
colnames(df_c) <- v_names_str
# data.frame of effectiveness
df_e <- as.data.frame(matrix(0,
                              nrow = n_sim,
                              ncol = n_str))
colnames(df_e) <- v_names_str
# Run Markov model on each parameter set of PSA input dataset
n_time_init_psa_series <- Sys.time()
for(i in 1:n_sim){
  l_out_temp <- calculate_ce_out(df_psa_input[i, ])
  df_c[i, ] <- l_out_temp$Cost
  df_e[i, ] <- l_out_temp$Effect
  # Display simulation progress
  if(i/(n_sim/10) == round(i/(n_sim/10), 0)) { # display progress every 10%
    cat('\r', paste(i/n_sim * 100, "% done", sep = " "))
  }
}
```

```
##  10 % done 20 % done 30 % done 40 % done 50 % done 60 % done 70 % done 80 % done 90 % done 100 % done
```

```
n_time_end_psa_series <- Sys.time()
n_time_total_psa_series <- n_time_end_psa_series - n_time_init_psa_series
print(paste0("PSA with ", comma(n_sim), " simulations run in series in ",
             round(n_time_total_psa_series, 2), " ",
             units(n_time_total_psa_series)))
```

```
## [1] "PSA with 1,000 simulations run in series in 1.68 secs"
```

09.2 Create PSA object for dampack

```
l_psa <- make_psa_obj(cost          = df_c,
                      effectiveness = df_e,
                      parameters    = df_psa_input,
                      strategies    = v_names_str)
l_psa$strategies <- v_names_str
colnames(l_psa$effectiveness)<- v_names_str
colnames(l_psa$cost)<- v_names_str
```

09.2.1 Save PSA objects

```
save(df_psa_input, df_c, df_e, v_names_str, n_str, l_psa,
     file = "markov_sick-sicker_intro_tutorial_PSA_dataset.RData")
```

09.3 Create probabilistic analysis graphs

```
load(file = "markov_sick-sicker_intro_tutorial_PSA_dataset.RData")
```

Vector with willingness-to-pay (WTP) thresholds.

```
v_wtp <- seq(0, 250000, by = 5000)
```

09.3.1 Cost-Effectiveness scatter plot

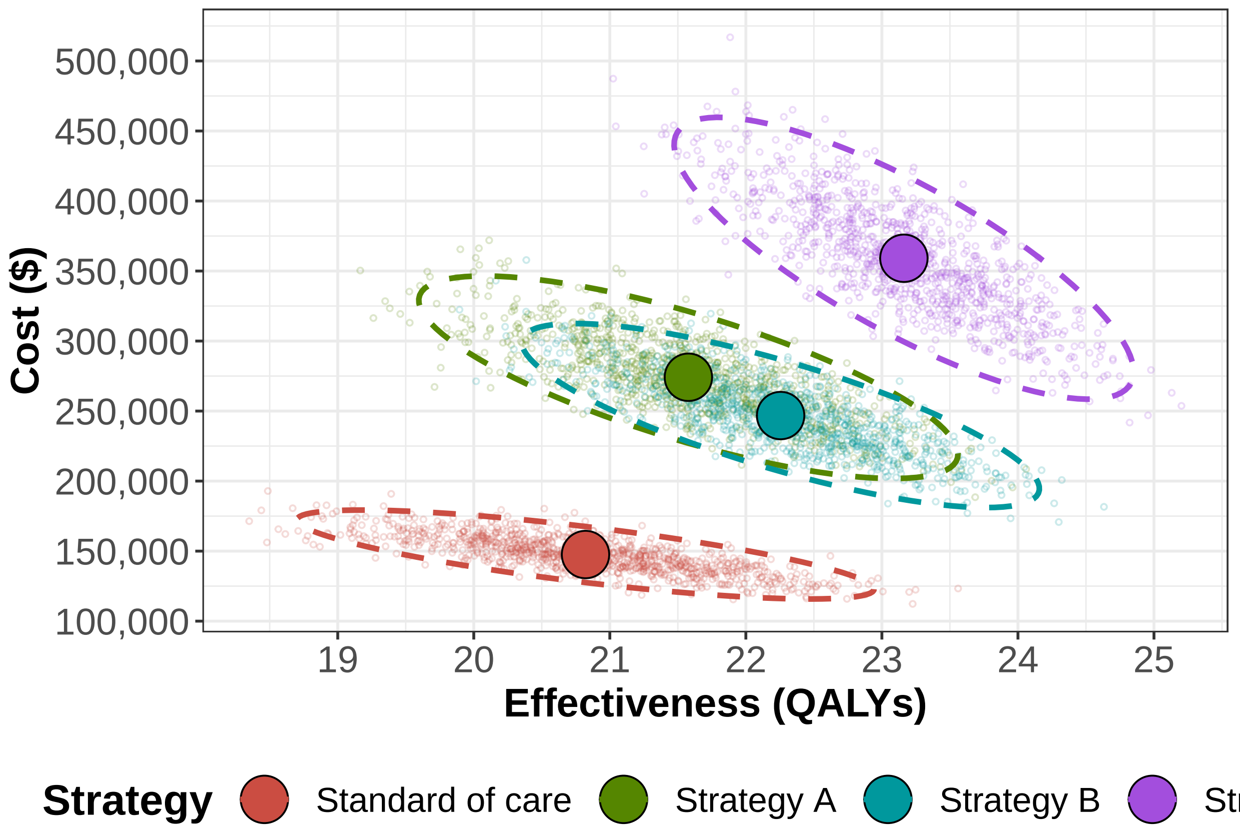

Figure 12: Cost-effectiveness scatter plot.

09.4 Conduct CEA with probabilistic output

```
# Compute expected costs and effects for each strategy from the PSA
df_out_ce_psa <- summary(l_psa)

# Calculate incremental cost-effectiveness ratios (ICERs)
df_cea_psa <- calculate_icers(cost       = df_out_ce_psa$meanCost,
                              effect     = df_out_ce_psa$meanEffect,
                              strategies = df_out_ce_psa$Strategy)
df_cea_psa
```

```
##           Strategy     Cost   Effect  Inc_Cost Inc_Effect      ICER Status
## 1 Standard of care 147556.9 20.82141        NA         NA        NA     ND
## 2       Strategy B 246796.5 22.25558  99239.59  1.4341754  69196.27     ND
## 3      Strategy AB 359123.0 23.16154 112326.53  0.9059587 123986.38     ND
## 4       Strategy A 274149.6 21.57760        NA         NA        NA      D
```

```
# Save CEA table with ICERs
# As .RData
save(df_cea_psa,
     file = "markov_sick-sicker_intro_tutorial_probabilistic_CEA_results.RData")
# As .csv
write.csv(df_cea_psa,
          file = "markov_sick-sicker_intro_tutorial_probabilistic_CEA_results.csv")
```

09.4.1 Plot cost-effectiveness frontier

```
plot(df_cea_psa)
```

09.4.2 Cost-effectiveness acceptability curves (CEACs) and frontier (CEAF)

```
##   range_min range_max   cost_eff_strat
## 1         0     70000 Standard of care
## 2         0    125000       Strategy B
## 3     70000    250000       Strategy B
## 4    125000    250000      Strategy AB
## 5    125000        NA      Strategy AB
```

09.4.3 Expected Loss Curves (ELCs)

The expected loss is the the quantification of the foregone benefits when choosing a suboptimal strategy given current evidence.

```
##        WTP         Strategy  Expected_Loss On_Frontier
## 1        0 Standard of care      0.0000000        TRUE
## 2        0       Strategy A 126592.7446448       FALSE
## 3        0       Strategy B  99239.5859762       FALSE
## 4        0      Strategy AB 211566.1182551       FALSE
## 5     5000 Standard of care      0.0000000        TRUE
## 6     5000       Strategy A 122811.8022610       FALSE
## 7     5000       Strategy B  92068.7090661       FALSE
## 8     5000      Strategy AB 199865.4479683       FALSE
## 9    10000 Standard of care      0.0000000        TRUE
## 10   10000       Strategy A 119030.8598772       FALSE
## 11   10000       Strategy B  84897.8321559       FALSE
```

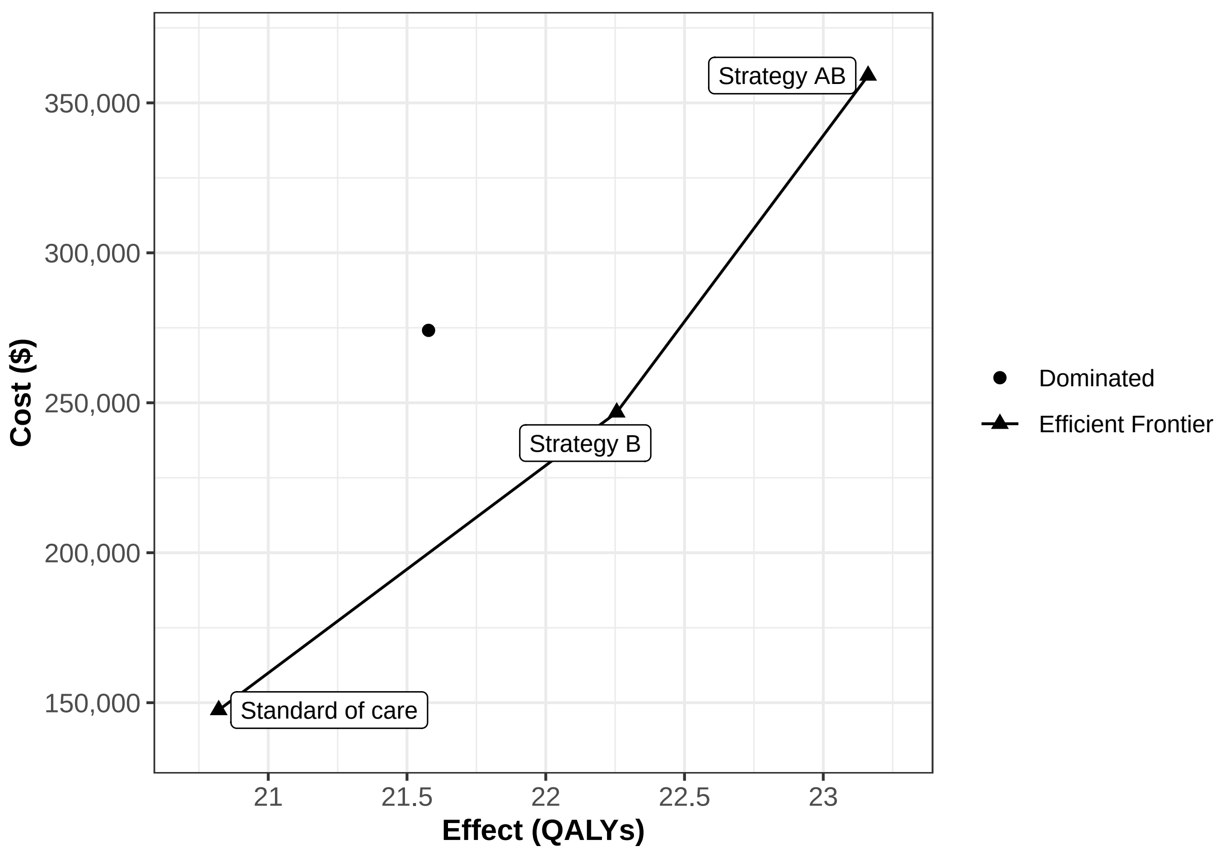

Figure 13: Cost-effectiveness efficient frontier from probabilistic outputs for the time-independent Sick-Sicker model.

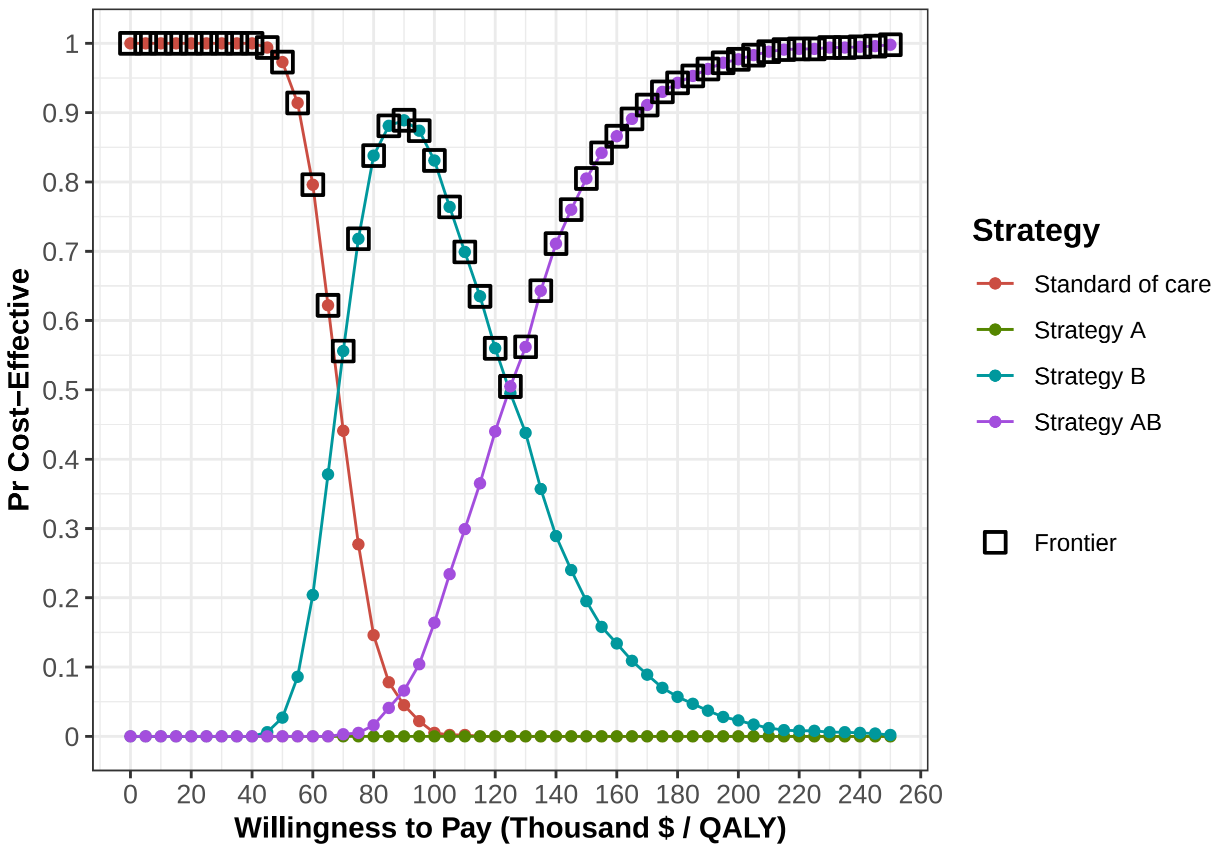

Figure 14: Cost-effectiveness acceptability curves (CEACs) and frontier (CEAF).

```
## 12   10000      Strategy AB 188164.7776815       FALSE
## 13   15000 Standard of care      0.0000000        TRUE
## 14   15000       Strategy A 115249.9174934       FALSE
## 15   15000       Strategy B  77726.9552457       FALSE
## 16   15000      Strategy AB 176464.1073947       FALSE
## 17   20000 Standard of care      0.0000000        TRUE
## 18   20000       Strategy A 111468.9751096       FALSE
## 19   20000       Strategy B  70556.0783355       FALSE
## 20   20000      Strategy AB 164763.4371079       FALSE
## 21   25000 Standard of care      0.0000000        TRUE
## 22   25000       Strategy A 107688.0327258       FALSE
## 23   25000       Strategy B  63385.2014254       FALSE
## 24   25000      Strategy AB 153062.7668211       FALSE
## 25   30000 Standard of care      0.0000000        TRUE
## 26   30000       Strategy A 103907.0903420       FALSE
## 27   30000       Strategy B  56214.3245152       FALSE
## 28   30000      Strategy AB 141362.0965343       FALSE
## 29   35000 Standard of care      0.0000000        TRUE
## 30   35000       Strategy A 100126.1479582       FALSE
## 31   35000       Strategy B  49043.4476050       FALSE
## 32   35000      Strategy AB 129661.4262475       FALSE
## 33   40000 Standard of care      0.0000000        TRUE
## 34   40000       Strategy A  96345.2055744       FALSE
## 35   40000       Strategy B  41872.5706949       FALSE
## 36   40000      Strategy AB 117960.7559607       FALSE
## 37   45000 Standard of care     20.4105025        TRUE
## 38   45000       Strategy A  92584.6736931       FALSE
## 39   45000       Strategy B  34722.1042872       FALSE
## 40   45000      Strategy AB 106280.4961764       FALSE
## 41   50000 Standard of care    127.3151594        TRUE
## 42   50000       Strategy A  88910.6359661       FALSE
## 43   50000       Strategy B  27658.1320339       FALSE
## 44   50000      Strategy AB  94686.7305464       FALSE
## 45   55000 Standard of care    484.6710630        TRUE
## 46   55000       Strategy A  85487.0494860       FALSE
## 47   55000       Strategy B  20844.6110274       FALSE
## 48   55000      Strategy AB  83343.4161633       FALSE
## 49   60000 Standard of care   1443.8027334        TRUE
## 50   60000       Strategy A  82665.2387725       FALSE
## 51   60000       Strategy B  14632.8657876       FALSE
## 52   60000      Strategy AB  72601.8775469       FALSE
## 53   65000 Standard of care   3565.3369993        TRUE
## 54   65000       Strategy A  81005.8306546       FALSE
## 55   65000       Strategy B   9583.5231433       FALSE
## 56   65000      Strategy AB  63022.7415260       FALSE
## 57   70000 Standard of care   6910.0632690       FALSE
## 58   70000       Strategy A  80569.6145406       FALSE
## 59   70000       Strategy B   5757.3725029        TRUE
## 60   70000      Strategy AB  54666.7975089       FALSE
## 61   75000 Standard of care  11505.0686181       FALSE
## 62   75000       Strategy A  81383.6775058       FALSE
## 63   75000       Strategy B   3181.5009418        TRUE
## 64   75000      Strategy AB  47561.1325712       FALSE
## 65   80000 Standard of care  17238.3093138       FALSE
```

```
## 66   80000       Strategy A  83335.9758178       FALSE
## 67   80000       Strategy B   1743.8647274        TRUE
## 68   80000      Strategy AB  41593.7029801       FALSE
## 69   85000 Standard of care  23755.3452939       FALSE
## 70   85000       Strategy A  86072.0694140       FALSE
## 71   85000       Strategy B   1090.0237973        TRUE
## 72   85000      Strategy AB  36410.0686734       FALSE
## 73   90000 Standard of care  30806.4109083       FALSE
## 74   90000       Strategy A  89342.1926446       FALSE
## 75   90000       Strategy B    970.2125015        TRUE
## 76   90000      Strategy AB  31760.4640010       FALSE
## 77   95000 Standard of care  38230.5409662       FALSE
## 78   95000       Strategy A  92985.3803187       FALSE
## 79   95000       Strategy B   1223.4656493        TRUE
## 80   95000      Strategy AB  27483.9237721       FALSE
## 81  100000 Standard of care  46063.3396948       FALSE
## 82  100000       Strategy A  97037.2366635       FALSE
## 83  100000       Strategy B   1885.3874676        TRUE
## 84  100000      Strategy AB  23616.0522139       FALSE
## 85  105000 Standard of care  54281.8445962       FALSE
## 86  105000       Strategy A 101474.7991811       FALSE
## 87  105000       Strategy B   2933.0154589        TRUE
## 88  105000      Strategy AB  20133.8868285       FALSE
## 89  110000 Standard of care  62817.0836292       FALSE
## 90  110000       Strategy A 106229.0958303       FALSE
## 91  110000       Strategy B   4297.3775817        TRUE
## 92  110000      Strategy AB  16968.4555747       FALSE
## 93  115000 Standard of care  71721.1628151       FALSE
## 94  115000       Strategy A 111352.2326324       FALSE
## 95  115000       Strategy B   6030.5798575        TRUE
## 96  115000      Strategy AB  14171.8644738       FALSE
## 97  120000 Standard of care  80956.0704381       FALSE
## 98  120000       Strategy A 116806.1978715       FALSE
## 99  120000       Strategy B   8094.6105703        TRUE
## 100 120000      Strategy AB  11706.1018100       FALSE
## 101 125000 Standard of care  90527.9620256       FALSE
## 102 125000       Strategy A 122597.1470752       FALSE
## 103 125000       Strategy B  10495.6252476       FALSE
## 104 125000      Strategy AB   9577.3231107        TRUE
## 105 130000 Standard of care 100377.6364126       FALSE
## 106 130000       Strategy A 128665.8790785       FALSE
## 107 130000       Strategy B  13174.4227245       FALSE
## 108 130000      Strategy AB   7726.3272109        TRUE
## 109 135000 Standard of care 110517.5028876       FALSE
## 110 135000       Strategy A 135024.8031696       FALSE
## 111 135000       Strategy B  16143.4122893       FALSE
## 112 135000      Strategy AB   6165.5233991        TRUE
## 113 140000 Standard of care 120962.5035865       FALSE
## 114 140000       Strategy A 141688.8614847       FALSE
## 115 140000       Strategy B  19417.5360780       FALSE
## 116 140000      Strategy AB   4909.8538112        TRUE
## 117 145000 Standard of care 131681.1198459       FALSE
## 118 145000       Strategy A 148626.5353603       FALSE
## 119 145000       Strategy B  22965.2754272       FALSE
```

```
## 120 145000      Strategy AB   3927.7997838        TRUE
## 121 150000 Standard of care 142590.1349892       FALSE
## 122 150000       Strategy A 155754.6081198       FALSE
## 123 150000       Strategy B  26703.4136603       FALSE
## 124 150000      Strategy AB   3136.1446403        TRUE
## 125 155000 Standard of care 153664.6205903       FALSE
## 126 155000       Strategy A 163048.1513371       FALSE
## 127 155000       Strategy B  30607.0223513       FALSE
## 128 155000      Strategy AB   2509.9599546        TRUE
## 129 160000 Standard of care 164850.4494507       FALSE
## 130 160000       Strategy A 170453.0378137       FALSE
## 131 160000       Strategy B  34621.9743015       FALSE
## 132 160000      Strategy AB   1995.1185282        TRUE
## 133 165000 Standard of care 176129.4072113       FALSE
## 134 165000       Strategy A 177951.0531905       FALSE
## 135 165000       Strategy B  38730.0551519       FALSE
## 136 165000      Strategy AB   1573.4060020        TRUE
## 137 170000 Standard of care 187487.7778505       FALSE
## 138 170000       Strategy A 185528.4814459       FALSE
## 139 170000       Strategy B  42917.5488810       FALSE
## 140 170000      Strategy AB   1231.1063544        TRUE
## 141 175000 Standard of care 198912.2934561       FALSE
## 142 175000       Strategy A 193172.0546677       FALSE
## 143 175000       Strategy B  47171.1875764       FALSE
## 144 175000      Strategy AB    954.9516732        TRUE
## 145 180000 Standard of care 210399.8906218       FALSE
## 146 180000       Strategy A 200878.7094495       FALSE
## 147 180000       Strategy B  51487.9078319       FALSE
## 148 180000      Strategy AB    741.8785521        TRUE
## 149 185000 Standard of care 221925.4154186       FALSE
## 150 185000       Strategy A 208623.2918626       FALSE
## 151 185000       Strategy B  55842.5557186       FALSE
## 152 185000      Strategy AB    566.7330621        TRUE
## 153 190000 Standard of care 233493.9174456       FALSE
## 154 190000       Strategy A 216410.8515057       FALSE
## 155 190000       Strategy B  60240.1808354       FALSE
## 156 190000      Strategy AB    434.5648023        TRUE
## 157 195000 Standard of care 245091.7033528       FALSE
## 158 195000       Strategy A 224227.6950291       FALSE
## 159 195000       Strategy B  64667.0898324       FALSE
## 160 195000      Strategy AB    331.6804227        TRUE
## 161 200000 Standard of care 256711.3313377       FALSE
## 162 200000       Strategy A 232066.3806303       FALSE
## 163 200000       Strategy B  69115.8409072       FALSE
## 164 200000      Strategy AB    250.6381209        TRUE
## 165 205000 Standard of care 268349.6561853       FALSE
## 166 205000       Strategy A 239923.7630940       FALSE
## 167 205000       Strategy B  73583.2888446       FALSE
## 168 205000      Strategy AB    188.2926816        TRUE
## 169 210000 Standard of care 280011.0751160       FALSE
## 170 210000       Strategy A 247804.2396409       FALSE
## 171 210000       Strategy B  78073.8308652       FALSE
## 172 210000      Strategy AB    149.0413256        TRUE
## 173 215000 Standard of care 291683.9400780       FALSE
```

```
## 174 215000       Strategy A 255696.1622191       FALSE
## 175 215000       Strategy B  82575.8189170       FALSE
## 176 215000      Strategy AB    121.2360008        TRUE
## 177 220000 Standard of care 303360.5755906       FALSE
## 178 220000       Strategy A 263591.8553479       FALSE
## 179 220000       Strategy B  87081.5775194       FALSE
## 180 220000      Strategy AB     97.2012265        TRUE
## 181 225000 Standard of care 315038.0048799       FALSE
## 182 225000       Strategy A 271488.3422534       FALSE
## 183 225000       Strategy B  91588.1298985       FALSE
## 184 225000      Strategy AB     73.9602290        TRUE
## 185 230000 Standard of care 326717.9270273       FALSE
## 186 230000       Strategy A 279387.3220169       FALSE
## 187 230000       Strategy B  96097.1751357       FALSE
## 188 230000      Strategy AB     53.2120896        TRUE
## 189 235000 Standard of care 338400.8501989       FALSE
## 190 235000       Strategy A 287289.3028048       FALSE
## 191 235000       Strategy B 100609.2213972       FALSE
## 192 235000      Strategy AB     35.4649745        TRUE
## 193 240000 Standard of care 350084.9926938       FALSE
## 194 240000       Strategy A 295192.5029159       FALSE
## 195 240000       Strategy B 105122.4869819       FALSE
## 196 240000      Strategy AB     18.9371825        TRUE
## 197 245000 Standard of care 361773.7469258       FALSE
## 198 245000       Strategy A 303100.3147641       FALSE
## 199 245000       Strategy B 109640.3643038       FALSE
## 200 245000      Strategy AB      7.0211278        TRUE
## 201 250000 Standard of care 373467.7937629       FALSE
## 202 250000       Strategy A 311013.4192174       FALSE
## 203 250000       Strategy B 114163.5342307       FALSE
## 204 250000      Strategy AB      0.3976781        TRUE
```

09.4.4 Expected value of perfect information (EVPI)

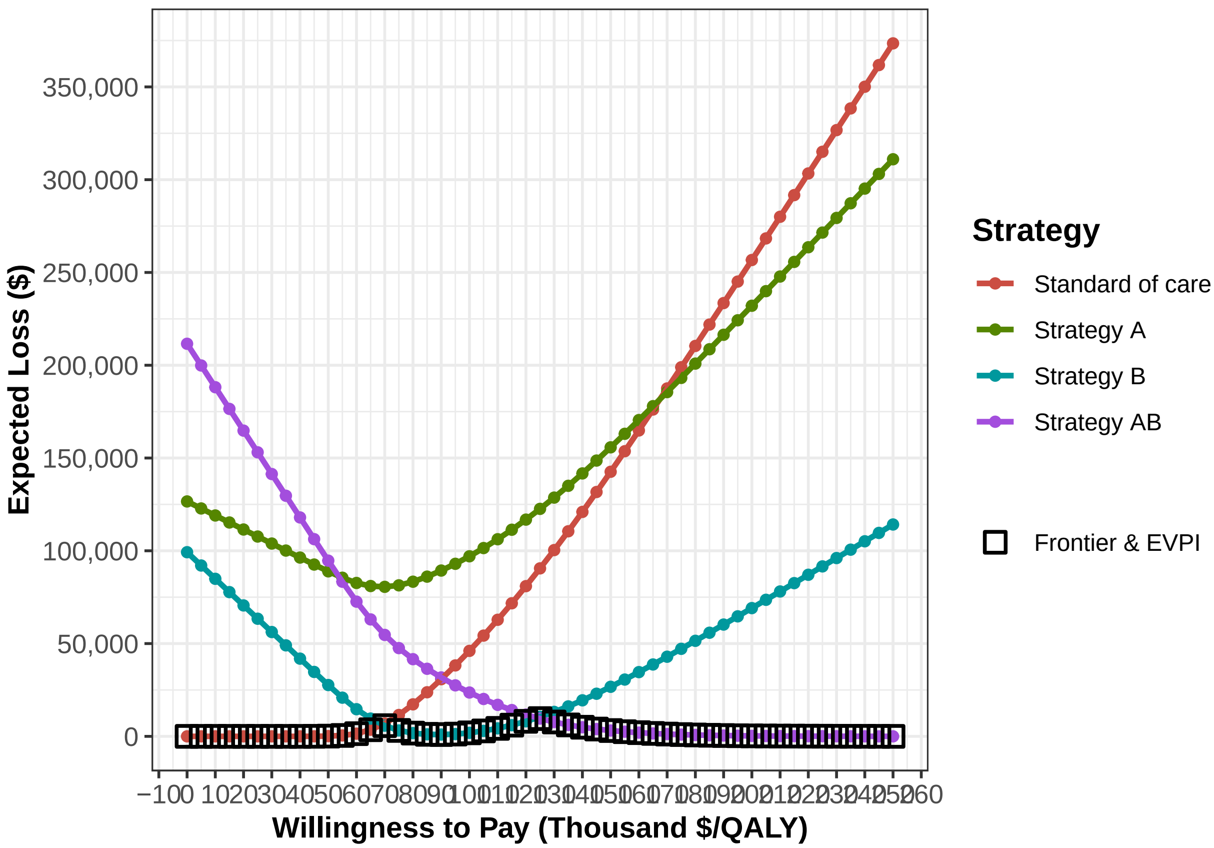

Figure 15: Expected loss curves (ELCs).

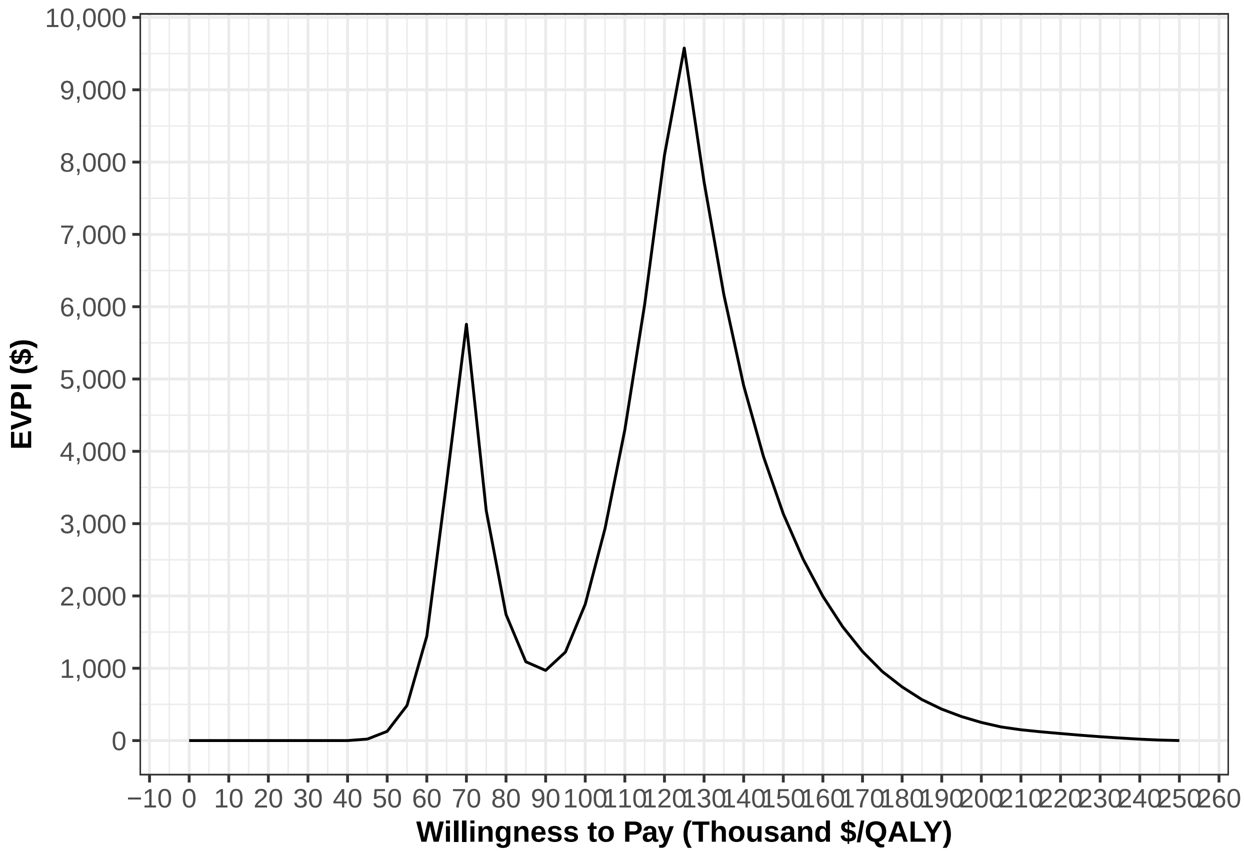

Figure 16: Expected value of perfect information (EVPI).