###############################################################################

### Cohort state-transition models in R: ##

### From conceptualization to implementation 2019 ##

###############################################################################

# This code forms the basis for the state-transition model of the article:

# 'Cohort state-transition models in R: From conceptualization to implementation'

# Authors:

# - Fernando Alarid-Escudero <fernando.alarid@cide.edu>

# - Eline Krijkamp

# - Eva A. Enns

# - Alan Yang

# - Myriam G.M. Hunink

# - Petros Pechlivanoglou

# - Hawre Jalal

# Please cite the article when using this code

#

# To program this tutorial we made use of

# R version 3.5.0 (2018-04-23)

# Platform: x86\_64-apple-darwin15.6.0 (64-bit)

# Running under: macOS 10.14.5

# RStudio: Version 1.1.453 2009-2018 RStudio, Inc

###############################################################################

################# Code of Appendix ##########################################

###############################################################################

# Implements an age- and history-dependent Sick-Sicker cSTM model

# + include code for a probabilistic sensitivity analysis (PSA)

##################################### Initial setup ###########################

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

library(dplyr) # to manipulate data

library(reshape2) # to transform data

library(ggplot2) # for nice looking plots

library(scales) # for dollar signs and commas

# WE ARE NOT USING TRUNC NORM ANY MORE!

library(truncnorm)

library(pryr)

# devtools::install\_github("DARTH-git/dampack") # to install dampack form GitHub

library(dampack) # for CEA and calculate ICERs

# Define required functions

calc\_logit <- function(p) {

logit <- log(p/(1-p))

return(logit)

}

calc\_invlogit <- function(x) {

invlogit <- exp(x)/(1+exp(x))

return(invlogit)

}

################################## DARTH colors ###############################

# code for the DARTH colors for the figures

DARTHgreen <- '#009999'

DARTHyellow <- '#FDAD1E'

DARTHblue <- '#006699'

DARTHlightgreen <- '#00adad'

DARTHgray <- '#666666'

##################################### Model input ##############################

## General setup

n\_age\_init <- 25 # age at baseline

n\_age\_max <- 110 # maximum age of follow up

n\_t <- n\_age\_max - n\_age\_init # time horizon, number of cycles

## Tunnel inputs

# Number of tunnels

n\_tunnel\_size <- n\_t

# Name for tunnels states of Sick state

v\_Sick\_tunnel <- paste("S1\_", seq(1, n\_tunnel\_size), "Yr", sep = "")

# Create variables for model with tunnels

v\_n\_tunnels <- c("H", v\_Sick\_tunnel, "S2", "D") # health state names

n\_states\_tunnels <- length(v\_n\_tunnels) # number of health states

v\_hcc <- rep(1, n\_t + 1) # vector of half-cycle correction

v\_hcc[1] <- v\_hcc[n\_t + 1] <- 0.5

d\_c <- 0.03 # discount rate for costs

d\_e <- 0.03 # discount rate for QALYs

v\_names\_str <- c("Usual care", "New treatment 1", "New treatment 2", "New treatments 1 & 2") # store the strategy names

n\_str <- length(v\_names\_str) # number of strategies

## Transition probabilities (per cycle) and hazard ratios

p\_HS1 <- 0.15 # probability to become Sick when Healthy

p\_S1H <- 0.5 # probability to become Healthy when Sick

hr\_S1 <- 3 # hazard ratio of death in Sick vs Healthy

hr\_S2 <- 10 # hazard ratio of death in Sicker vs Healthy

## History-dependent transition from S1 to S2

# Weibull parameters

n\_lambda <- 0.08 # scale

n\_gamma <- 1.1 # shape

# Weibull function

p\_S1S2\_tunnels <- n\_lambda \* n\_gamma \* (1:n\_tunnel\_size)^{n\_gamma-1}

# For new treatment 2

or\_S1S2 <- 0.7 # odds ratio of becoming Sicker when Sick under New treatment 2

lor\_S1S2 <- log(or\_S1S2) # log-odd ratio of becoming Sicker when Sick

logitp\_S1S2 <- calc\_logit(p\_S1S2\_tunnels) # log-odds of becoming Sicker when Sick

p\_S1S2\_tunnels\_trt2 <- calc\_invlogit(logitp\_S1S2 + lor\_S1S2) # probability to become Sicker when Sick under New treatment 2

## Age-dependent mortality rates

lt\_usa\_2005 <- read.csv("data/LifeTable\_USA\_Mx\_2015.csv")

v\_r\_mort\_by\_age <- lt\_usa\_2005 %>%

# filter(Age >= age & Age <= n\_age\_max) %>%

select(Total) %>%

as.matrix()

## State rewards

# Costs

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\_D <- 0 # cost of being dead (per cycle)

c\_trt1 <- 12000 # cost of New treatment 1 (per cycle)

c\_trt2 <- 13000 # cost of New treatment 2 (per cycle)

# 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 Healthy

u\_trt1 <- 0.95 # utility when being treated

## Transition rewards

du\_HS1 <- 0.01 # disutility when transitioning from Healthy to Sick

ic\_HS1 <- 1000 # increase in cost when transitioning from Healthy to Sick

ic\_D <- 2000 # increase in cost when dying

# Discount weight (equal discounting is assumed for costs and effects)

v\_dwc <- 1 / ((1 + d\_e) ^ (0:(n\_t)))

v\_dwe <- 1 / ((1 + d\_c) ^ (0:(n\_t)))

## Age-specific transition probabilities

# Age-specific probability of dying when Healthy (all-cause mortality)

v\_p\_HDage <- 1 - exp(-v\_r\_mort\_by\_age[(n\_age\_init + 1) + 0:(n\_t - 1)])

# Age-specific mortality risk in the Sick state

v\_p\_S1Dage <- 1 - exp(-v\_r\_mort\_by\_age[(n\_age\_init + 1) + 0:(n\_t - 1)] \* hr\_S1)

# Age-specific mortality risk in the Sicker state

v\_p\_S2Dage <- 1 - exp(-v\_r\_mort\_by\_age[(n\_age\_init + 1) + 0:(n\_t - 1)] \* hr\_S2)

######################## Construct state-transition models ####################

#### Create transition matrix ####

# Initialize 3-D array

a\_P\_tunnels <- array(0, dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:(n\_t - 1)))

### Fill in array

## From H

a\_P\_tunnels["H", "H", ] <- 1 - (p\_HS1 + v\_p\_HDage)

a\_P\_tunnels["H", v\_Sick\_tunnel[1], ] <- p\_HS1

a\_P\_tunnels["H", "D", ] <- v\_p\_HDage

## From S1

for(i in 1:(n\_tunnel\_size - 1)){

a\_P\_tunnels[v\_Sick\_tunnel[i], "H", ] <- p\_S1H

a\_P\_tunnels[v\_Sick\_tunnel[i],

v\_Sick\_tunnel[i + 1], ] <- 1 - (p\_S1H + p\_S1S2\_tunnels[i] + v\_p\_S1Dage)

a\_P\_tunnels[v\_Sick\_tunnel[i], "S2", ] <- p\_S1S2\_tunnels[i]

a\_P\_tunnels[v\_Sick\_tunnel[i], "D", ] <- v\_p\_S1Dage

}

# repeat code for the last cycle to force the cohort stay in the last tunnel state of Sick

a\_P\_tunnels[v\_Sick\_tunnel[n\_tunnel\_size], "H", ] <- p\_S1H

a\_P\_tunnels[v\_Sick\_tunnel[n\_tunnel\_size],

v\_Sick\_tunnel[n\_tunnel\_size], ] <- 1 - (p\_S1H +

p\_S1S2\_tunnels[n\_tunnel\_size] +

v\_p\_S1Dage)

a\_P\_tunnels[v\_Sick\_tunnel[n\_tunnel\_size], "S2", ] <- p\_S1S2\_tunnels[n\_tunnel\_size]

a\_P\_tunnels[v\_Sick\_tunnel[n\_tunnel\_size], "D", ] <- v\_p\_S1Dage

## From S2

a\_P\_tunnels["S2", "S2", ] <- 1 - v\_p\_S2Dage

a\_P\_tunnels["S2", "D", ] <- v\_p\_S2Dage

## From D

a\_P\_tunnels["D", "D", ] <- 1

# For New treatment 2

# Only need to update the probabilities involving p\_S1S2

a\_P\_tunnels\_trt2 <- a\_P\_tunnels

for(i in 1:(n\_tunnel\_size - 1)){

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[i], "H", ] <- p\_S1H

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[i],

v\_Sick\_tunnel[i + 1], ] <- 1 - (p\_S1H + p\_S1S2\_tunnels\_trt2[i] + v\_p\_S1Dage)

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[i], "S2", ] <- p\_S1S2\_tunnels\_trt2[i]

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[i], "D", ] <- v\_p\_S1Dage

}

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[n\_tunnel\_size],

v\_Sick\_tunnel[n\_tunnel\_size], ] <- 1 - (p\_S1H +

p\_S1S2\_tunnels\_trt2[n\_tunnel\_size] +

v\_p\_S1Dage)

a\_P\_tunnels\_trt2[v\_Sick\_tunnel[n\_tunnel\_size], "S2", ] <- p\_S1S2\_tunnels\_trt2[n\_tunnel\_size]

#### Run Markov model ####

## Initial state vector

# All starting healthy

v\_s\_init\_tunnels <- c(1, rep(0, n\_tunnel\_size), 0, 0)

## Initialize cohort trace for history-dependent cSTM

m\_M\_tunnels <- matrix(0,

nrow = (n\_t + 1), ncol = n\_states\_tunnels,

dimnames = list(0:n\_t, v\_n\_tunnels))

# Store the initial state vector in the first row of the cohort trace

m\_M\_tunnels[1, ] <- v\_s\_init\_tunnels

# For New treatment 2

m\_M\_tunnels\_trt2 <- m\_M\_tunnels

## Initialize transition array

a\_A\_tunnels <- array(0,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t))

# Set first slice of A with the initial state vector in its diagonal

diag(a\_A\_tunnels[, , 1]) <- v\_s\_init\_tunnels

# For New treatment 2

a\_A\_tunnels\_trt2 <- a\_A\_tunnels

## Iterative solution of age-dependent cSTM

for(t in 1:n\_t){

# Fill in cohort trace

m\_M\_tunnels[t + 1, ] <- m\_M\_tunnels[t, ] %\*% a\_P\_tunnels[, , t]

m\_M\_tunnels\_trt2[t + 1,] <- m\_M\_tunnels\_trt2[t, ] %\*% a\_P\_tunnels\_trt2[, , t]

# Fill in transition dynamics array

a\_A\_tunnels[, , t + 1] <- m\_M\_tunnels[t, ] \* a\_P\_tunnels[, , t]

a\_A\_tunnels\_trt2[, , t + 1] <- m\_M\_tunnels\_trt2[t, ] \* a\_P\_tunnels\_trt2[, , t]

}

# Create aggregated trace

m\_M\_tunnels\_sum <- cbind(H = m\_M\_tunnels[, "H"],

S1 = rowSums(m\_M\_tunnels[, 2:(n\_tunnel\_size +1)]),

S2 = m\_M\_tunnels[, "S2"],

D = m\_M\_tunnels[, "D"])

m\_M\_tunnels\_sum\_trt2 <- cbind(H = m\_M\_tunnels\_trt2[, "H"],

S1 = rowSums(m\_M\_tunnels\_trt2[, 2:(n\_tunnel\_size +1)]),

S2 = m\_M\_tunnels\_trt2[, "S2"],

D = m\_M\_tunnels\_trt2[, "D"])

#### Plot Outputs ####

### Cohort trace

## Define colors and line types

cols <- c("H" = DARTHgreen, "S1" = DARTHblue,

"S2" = DARTHyellow, "D" = DARTHgray)

lty <- c("H" = 1, "S1" = 2, "S2" = 4, "D" = 3)

## Plot the cohort trace

ggplot(melt(m\_M\_tunnels\_sum), aes(x = Var1, y = value,

color = Var2, linetype = Var2)) +

geom\_line(size = 1) +

scale\_colour\_manual(name = "Health state",

values = cols) +

scale\_linetype\_manual(name = "Health state",

values = lty) +

xlab("Cycle") +

ylab("Proportion of the cohort") +

theme\_bw(base\_size = 14) +

theme(legend.position = "bottom",

legend.background = element\_rect(fill = NA))

# For New treatment 2

ggplot(melt(m\_M\_tunnels\_sum\_trt2), aes(x = Var1, y = value,

color = Var2, linetype = Var2)) +

geom\_line(size = 1) +

scale\_colour\_manual(name = "Health state",

values = cols) +

scale\_linetype\_manual(name = "Health state",

values = lty) +

xlab("Cycle") +

ylab("Proportion of the cohort") +

theme\_bw(base\_size = 14) +

theme(legend.position = "bottom",

legend.background = element\_rect(fill = NA))

#### State and Transition Rewards ####

### State rewards

## Vector of utilities for S1 under Usual care

v\_u\_S1\_UC <- rep(u\_S1, n\_tunnel\_size)

names(v\_u\_S1\_UC) <- v\_Sick\_tunnel

## Vector of state utilities under Usual care

v\_u\_UC <- c(H = u\_H, v\_u\_S1\_UC, S2 = u\_S2, D = u\_D)

## Vector of costs per cycle for S1 under Usual care

v\_c\_S1\_UC <- rep(c\_S1, n\_tunnel\_size)

names(v\_c\_S1\_UC) <- v\_Sick\_tunnel

## Vector of state costs per cycle under Usual care

v\_c\_UC <- c(H = c\_H, v\_c\_S1\_UC, S2 = c\_S2, D = c\_D)

## Vector of utilities for S1 under New treatment 1

v\_u\_S1\_trt1 <- rep(u\_trt1, n\_tunnel\_size)

names(v\_u\_S1\_trt1) <- v\_Sick\_tunnel

## Vector of state utilities under New treatment 1

v\_u\_trt1 <- c(H = u\_H, v\_u\_S1\_trt1, S2 = u\_S2, D = u\_D)

## Vector of costs per cycle for S1 under New treatment 1

v\_c\_S1\_trt1 <- rep(c\_S1 + c\_trt1, n\_tunnel\_size)

names(v\_c\_S1\_trt1) <- v\_Sick\_tunnel

## Vector of state costs per cycle under New treatment 1

v\_c\_trt1 <- c(H = c\_H, v\_c\_S1\_trt1, S2 = c\_S2 + c\_trt1, D = c\_D)

## Vector of utilities for S1 under New treatment 2

v\_u\_S1\_trt2 <- rep(u\_S1, n\_tunnel\_size)

names(v\_u\_S1\_trt2) <- v\_Sick\_tunnel

## Vector of state utilities under New treatment 2

v\_u\_trt2 <- c(H = u\_H, v\_u\_S1\_trt2, S2 = u\_S2, D = u\_D)

## Vector of costs per cycle for S1 under New treatment 2

v\_c\_S1\_trt2 <- rep(c\_S1 + c\_trt2, n\_tunnel\_size)

names(v\_c\_S1\_trt2) <- v\_Sick\_tunnel

## Vector of state costs per cycle under New treatment 2

v\_c\_trt2 <- c(H = c\_H, v\_c\_S1\_trt2, S2 = c\_S2 + c\_trt2, D = c\_D)

## Vector of utilities for S1 under New treatments 1 & 2

v\_u\_S1\_trt1\_2 <- rep(u\_trt1, n\_tunnel\_size)

names(v\_u\_S1\_trt1\_2) <- v\_Sick\_tunnel

## Vector of state utilities under New treatment 2

v\_u\_trt1\_2 <- c(H = u\_H, v\_u\_S1\_trt1\_2, S2 = u\_S2, D = u\_D)

## Vector of costs per cycle for S1 under New treatment 2

v\_c\_S1\_trt1\_2 <- rep(c\_S1 + (c\_trt1 + c\_trt2), n\_tunnel\_size)

names(v\_c\_S1\_trt1\_2) <- v\_Sick\_tunnel

## Vector of state costs per cycle under New treatment 2

v\_c\_trt1\_2 <- c(H = c\_H, v\_c\_S1\_trt1\_2, S2 = c\_S2 + (c\_trt1 + c\_trt2), D = c\_D)

### Arrays of rewards

## Array of state and transition utilities under Usual care

a\_R\_u\_UC <- aperm(array(v\_u\_UC,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of state and transition costs under Usual care

a\_R\_c\_UC <- aperm(array(v\_c\_UC,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of utilities under New treatment 1

a\_R\_u\_trt1 <- aperm(array(v\_u\_trt1,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of costs per cycle under New treatment 1

a\_R\_c\_trt1 <- aperm(array(v\_c\_trt1,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of utilities under New treatment 2

a\_R\_u\_trt2 <- aperm(array(v\_u\_trt2,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of costs per cycle under New treatment 2

a\_R\_c\_trt2 <- aperm(array(v\_c\_trt2,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of utilities under New treatment 1 & 2

a\_R\_u\_trt1\_2 <- aperm(array(v\_u\_trt1\_2,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

## Array of costs per cycle under New treatment 1 & 2

a\_R\_c\_trt1\_2 <- aperm(array(v\_c\_trt1\_2,

dim = c(n\_states\_tunnels, n\_states\_tunnels, n\_t + 1),

dimnames = list(v\_n\_tunnels, v\_n\_tunnels, 0:n\_t)),

perm = c(2, 1, 3))

### Transition rewards

## For Usual care

# Add disutility due to transition from H to S1

a\_R\_u\_UC["H", "S1\_1Yr", ] <- a\_R\_u\_UC["H", "S1\_1Yr", ] - du\_HS1

# Add transition cost per cycle due to transition from H to S1

a\_R\_c\_UC["H", "S1\_1Yr", ] <- a\_R\_c\_UC["H", "S1\_1Yr", ] + ic\_HS1

# Add transition cost per cycle of dying from all non-dead states

a\_R\_c\_UC[-n\_states\_tunnels, "D", ] <- a\_R\_c\_UC[-n\_states\_tunnels, "D", ] + ic\_D

## For New treatment 1

# Add disutility due to transition from Healthy to Sick

a\_R\_u\_trt1["H", "S1\_1Yr", ] <- a\_R\_u\_trt1["H", "S1\_1Yr", ] - du\_HS1

# Add transition cost per cycle due to transition from Healthy to Sick

a\_R\_c\_trt1["H", "S1\_1Yr", ] <- a\_R\_c\_trt1["H", "S1\_1Yr", ] + ic\_HS1

# Add transition cost per cycle of dying from all non-dead states

a\_R\_c\_trt1[-n\_states\_tunnels, "D", ] <- a\_R\_c\_trt1[-n\_states\_tunnels, "D", ] + ic\_D

## For New treatment 2

# Add disutility due to transition from Healthy to Sick

a\_R\_u\_trt2["H", "S1\_1Yr", ] <- a\_R\_u\_trt2["H", "S1\_1Yr", ] - du\_HS1

# Add transition cost per cycle due to transition from Healthy to Sick

a\_R\_c\_trt2["H", "S1\_1Yr", ] <- a\_R\_c\_trt2["H", "S1\_1Yr", ] + ic\_HS1

# Add transition cost per cycle of dying from all non-dead states

a\_R\_c\_trt2[-n\_states\_tunnels, "D", ] <- a\_R\_c\_trt2[-n\_states\_tunnels, "D", ] + ic\_D

## For New treatment 1 & 2

# Add disutility due to transition from Healthy to Sick

a\_R\_u\_trt1\_2["H", "S1\_1Yr", ] <- a\_R\_u\_trt1\_2["H", "S1\_1Yr", ] - du\_HS1

# Add transition cost per cycle due to transition from Healthy to Sick

a\_R\_c\_trt1\_2["H", "S1\_1Yr", ] <- a\_R\_c\_trt1\_2["H", "S1\_1Yr", ] + ic\_HS1

# Add transition cost per cycle of dying from all non-dead states

a\_R\_c\_trt1\_2[-n\_states\_tunnels, "D", ] <- a\_R\_c\_trt1\_2[-n\_states\_tunnels, "D", ] + ic\_D

#### Expected QALYs and Costs for all transitions per cycle ####

# QALYs = life years x QoL

# NOTE: all parameters are annual

### For Usual Care

a\_Y\_c\_UC <- a\_A\_tunnels \* a\_R\_c\_UC

a\_Y\_u\_UC <- a\_A\_tunnels \* a\_R\_u\_UC

### For New treatment 1

a\_Y\_c\_trt1 <- a\_A\_tunnels \* a\_R\_c\_trt1

a\_Y\_u\_trt1 <- a\_A\_tunnels \* a\_R\_u\_trt1

### For New treatment 2

a\_Y\_c\_trt2 <- a\_A\_tunnels\_trt2 \* a\_R\_c\_trt2

a\_Y\_u\_trt2 <- a\_A\_tunnels\_trt2 \* a\_R\_u\_trt2

### For New treatment 1 & 2

a\_Y\_c\_trt1\_2 <- a\_A\_tunnels\_trt2 \* a\_R\_c\_trt1\_2

a\_Y\_u\_trt1\_2 <- a\_A\_tunnels\_trt2 \* a\_R\_u\_trt1\_2

#### Expected QALYs and Costs per cycle ####

## Vector of qalys under Usual Care

v\_qaly\_UC <- rowSums(t(colSums(a\_Y\_u\_UC)))

## Vector of costs under Usual Care

v\_cost\_UC <- rowSums(t(colSums(a\_Y\_c\_UC)))

## Vector of qalys under New Treatment 1

v\_qaly\_trt1 <- rowSums(t(colSums(a\_Y\_u\_trt1)))

## Vector of costs under New Treatment 1

v\_cost\_trt1 <- rowSums(t(colSums(a\_Y\_c\_trt1)))

## Vector of qalys under New Treatment 2

v\_qaly\_trt2 <- rowSums(t(colSums(a\_Y\_u\_trt2)))

## Vector of costs under New Treatment 2

v\_cost\_trt2 <- rowSums(t(colSums(a\_Y\_c\_trt2)))

## Vector of qalys under New Treatment 1 & 2

v\_qaly\_trt1\_2 <- rowSums(t(colSums(a\_Y\_u\_trt1\_2)))

## Vector of costs under New Treatment 1 & 2

v\_cost\_trt1\_2 <- rowSums(t(colSums(a\_Y\_c\_trt1\_2)))

#### Discounted total expected QALYs and Costs per strategy ####

### For Usual Care

## QALYs

n\_totqaly\_UC <- t(v\_qaly\_UC) %\*% (v\_dwe \* v\_hcc)

## Costs

n\_totcost\_UC <- t(v\_cost\_UC) %\*% (v\_dwc \* v\_hcc)

### For New treatment 1

## QALYs

n\_totqaly\_trt1 <- t(v\_qaly\_trt1) %\*% (v\_dwe \* v\_hcc)

## Costs

n\_totcost\_trt1 <- t(v\_cost\_trt1) %\*% (v\_dwc \* v\_hcc)

### For New treatment 2

## QALYs

n\_totqaly\_trt2 <- t(v\_qaly\_trt2) %\*% (v\_dwe \* v\_hcc)

## Costs

n\_totcost\_trt2 <- t(v\_cost\_trt2) %\*% (v\_dwc \* v\_hcc)

### For New treatment 1 & 2

## QALYs

n\_totqaly\_trt1\_2 <- t(v\_qaly\_trt1\_2) %\*% (v\_dwe \* v\_hcc)

## Costs

n\_totcost\_trt1\_2 <- t(v\_cost\_trt1\_2) %\*% (v\_dwc \* v\_hcc)

########################### Cost-effectiveness analysis #######################

### Vector of total costs for all strategies

v\_ted\_cost <- c(n\_totcost\_UC, n\_totcost\_trt1, n\_totcost\_trt2, n\_totcost\_trt1\_2)

### Vector of effectiveness for all strategies

v\_ted\_qaly <- c(n\_totqaly\_UC, n\_totqaly\_trt1, n\_totqaly\_trt2, n\_totqaly\_trt1\_2)

### Calculate incremental cost-effectiveness ratios (ICERs)

df\_cea <- calculate\_icers(cost = v\_ted\_cost,

effect = v\_ted\_qaly,

strategies = v\_names\_str)

df\_cea

### Create CEA table

table\_cea <- df\_cea

## Format column names

colnames(table\_cea)[2:6] <- c("Costs ($)", "QALYs",

"Incremental Costs ($)", "Incremental QALYs",

"ICER ($/QALY)") # name the columns

## Format rows

table\_cea$`Costs ($)` <- comma(round(table\_cea$`Costs ($)`, 0))

table\_cea$`Incremental Costs ($)` <- comma(round(table\_cea$`Incremental Costs ($)`, 0))

table\_cea$QALYs <- round(table\_cea$QALYs, 2)

table\_cea$`Incremental QALYs` <- round(table\_cea$`Incremental QALYs`, 2)

table\_cea$`ICER ($/QALY)` <- comma(round(table\_cea$`ICER ($/QALY)`, 0))

table\_cea

### CEA frontier

plot(df\_cea, label="all") +

expand\_limits(x = max(table\_cea$QALYs + 0.5)) # change this

########################### Probabalistic Sensitivty Analysis #######################

# 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)

p\_HS1 = rbeta(n\_sim, 30, 170), # probability to become sick when healthy

p\_S1H = rbeta(n\_sim, 60, 60) , # probability to become healthy when sick

hr\_S1 = rlnorm(n\_sim, log(3), 0.01), # rate ratio of death in S1 vs healthy

hr\_S2 = rlnorm(n\_sim, log(10), 0.02), # rate ratio of death in S2 vs healthy

n\_lambda = rlnorm(n\_sim, log(0.08), 0.01), # transition from S1 to S2 - Weibull scale parameter

n\_gamma = rlnorm(n\_sim, log(1.1), 0.02), # transition from S1 to S2 - Weibull shape parameter

#I WOULD GENERATE OR INSTEAD OF LOR USING RLNORM

lor\_S1S2 = rnorm(n\_sim, log(0.7), 0.1), # log-odd ratio of becoming Sicker when Sick

# State rewards

# Costs

c\_H = rgamma(n\_sim, shape = 100, scale = 20), # cost of remaining one cycle in state H

c\_S1 = rgamma(n\_sim, shape = 177.8, scale = 22.5), # cost of remaining one cycle in state S1

c\_S2 = rgamma(n\_sim, shape = 225, scale = 66.7), # cost of remaining one cycle in state S2

c\_trt = rgamma(n\_sim, shape = 73.5, scale = 163.3), # cost of treatment (per cycle)

c\_D = 0, # cost of being in the death state

#SEE NOTE ABOVE WE RE NOT USING TRUNCNORM ANYMORE

# Utilities

u\_H = rtruncnorm(n\_sim, mean = 1, sd = 0.01, b = 1), # utility when healthy

u\_S1 = rtruncnorm(n\_sim, mean = 0.75, sd = 0.02, b = 1), # utility when sick

u\_S2 = rtruncnorm(n\_sim, mean = 0.50, sd = 0.03, b = 1), # utility when sicker

u\_D = 0, # utility when dead

u\_trt = rtruncnorm(n\_sim, mean = 0.95, sd = 0.02, b = 1) # utility when being treated

)

return(df\_psa)

}

# 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)

# Histogram of parameters

ggplot(melt(df\_psa\_input, variable.name = "Parameter"), aes(x = value)) +

facet\_wrap(~Parameter, scales = "free") +

geom\_histogram(aes(y = ..density..)) +

theme\_bw(base\_size = 16)

# Initialize matrices with PSA output

# Dataframe of costs

df\_c <- as.data.frame(matrix(0,

nrow = n\_sim,

ncol = n\_str))

colnames(df\_c) <- v\_names\_str

# Dataframe of effectiveness

df\_e <- as.data.frame(matrix(0,

nrow = n\_sim,

ncol = n\_str))

colnames(df\_e) <- v\_names\_str

## Conduct probabilistic sensitivity analysis

#WHY CEA OUTPUT?

#PROVIDE MORE DETAIL ON WHAT FUNCTIONS EXIST IN THE R FILE

# Source functions that contain the model and CEA output

source('functions/Functions STM\_03.R')

# Run Markov model on each parameter set of PSA input dataset

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 = " "))

}

}

# Create PSA object for dampack

l\_psa <- make\_psa\_obj(cost = df\_c,

effectiveness = df\_e,

parameters = df\_psa\_input,

strategies = v\_names\_str)

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

v\_wtp <- seq(0, 200000, by = 10000)

# Cost-Effectiveness Scatter plot

plot(l\_psa)

## 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

## Plot cost-effectiveness frontier

plot(df\_cea\_psa, label="all") +

expand\_limits(x = 20.8)

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

ceac\_obj <- ceac(wtp = v\_wtp, psa = l\_psa)

# Regions of highest probability of cost-effectiveness for each strategy

summary(ceac\_obj)

# CEAC & CEAF plot

plot(ceac\_obj)

## Expected Loss Curves (ELCs)

elc\_obj <- calc\_exp\_loss(wtp = v\_wtp, psa = l\_psa)

elc\_obj

# ELC plot

plot(elc\_obj, log\_y = FALSE)

## Expected value of perfect information (EVPI)

evpi <- calc\_evpi(wtp = v\_wtp, psa = l\_psa)

# EVPI plot

plot(evpi, effect\_units = "QALY")