SUPPLEMENTARY MATERIAL OF "A MULTIDIMENSIONAL ARRAY REPRESENTATION OF STATE-TRANSITION MODEL DYNAMICS"

Eline M. Krijkamp[*],
Fernando Alarid-Escudero[*,†],
Eva A. Enns,
Petros Pechlivanoglou,
M.G. Myriam Hunink,
Alan Yang,
Hawre J. Jalal

October 16, 2019

1 Dynamics-array approach of the stylistic 3-state model

Model description
We follow a cohort of healthy 70-year-old individuals over their remaining lifetime, using 30 annual cycles. The healthy individuals can transition to the sick health state, they can die or remain healthy. Sick individuals can fully recover, transitioning back to healthy, remain sick or die. Remaining in each of these health states is associated with some utilities and costs (the state rewards). In addition to these state rewards, transition dis-utilities and costs apply. Getting sick is associated with a sudden decrease of quality of life of 0.1. In addition, transitioning to dead incurs a one-time cost of $4,000. Both the state and transition rewards are constant over time. The R code to use the dynamics-array approach for this case example is shown below. All parameters of this model are fictitious, not based on a specific disease. Figure 1 shows the health states and possible transitions. We describe this simple 3-state example in more detail on GitHub - (`https://github.com/DARTH-git/state-transition-model-dynamics`).

[*]Contributed equally to this work.
[†]Corresponding author

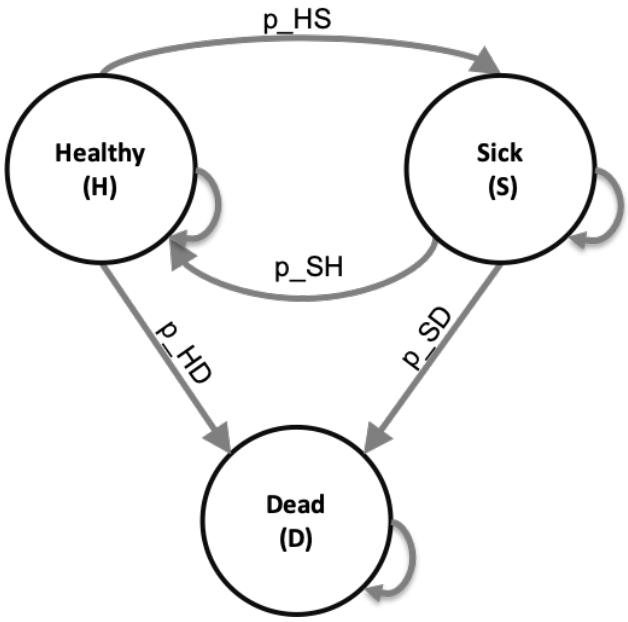

Figure 1: State-transition diagram of the 3-state model.

R code for the dynamics-array approach
We **highly recommend downloading this code from GitHub** (see link above) to avoid errors due to copying from the manuscript and to obtain the latest version of the code. We use the coding convention as recommended in our coding Framework.[1]

```r
# Load the packages
library(reshape2) # to transform data
library(ggplot2)  # for nice looking plots

# initial set up
age          <- 70  # age of starting cohort
n_t          <- 30  # number of cycles
v_age_names <- age:(age + n_t - 1) # vector with age names
v_n <- c("H", "S", "D") # vector with the 3 health states of the model:
# Healthy (H), Sick (S), Dead (D)
n_states <- length(v_n) # number of health states

#### Generate initial set of base-case external parameters ####
# Costs
c_H    <- 1000    # cost of remaining one cycle healthy
c_S    <- 3000    # cost of remaining one cycle sick
c_D    <- 0       # cost of being dead (per cycle)
# State utilities
u_H    <- 1       # utility when healthy
u_S    <- 0.60    # utility when sick
u_D    <- 0       # utility when healthy
# Transition probabilities (per cycle)
p_HS   <- 0.30    # probability to become sick when healthy
p_HD   <- 0.05    # probability to die when healthy
p_SH   <- 0.15    # probability to become healthy when sick
p_SD   <- 0.20    # probability to die when sick
# Transition rewards
du_HS <- 0.10  # one-time utility decrement when becoming sick
ic_D   <- 4000    # one-time cost of dying

#### Transition probability matrix ####
# matrix m_P at the first cycle
m_P <- matrix(NA,
              nrow = n_states,
              ncol = n_states,
```

```r
                dimnames = list(v_n, v_n))

# Fill in matrix
# From Healthy
m_P["H", "H"]  <- 1 - (p_HS + p_HD)
m_P["H", "S"]  <- p_HS
m_P["H", "D"]  <- p_HD
# From Sick
m_P["S", "H"]  <- p_SH
m_P["S", "S"]  <- 1 - (p_SH + p_SD)
m_P["S", "D"]  <- p_SD
# From Death
m_P["D", "H"]  <- 0
m_P["D", "S"]  <- 0
m_P["D", "D"]  <- 1

#### Cohort trace matrix ####
## Initial state vector
v_m0 <- c(H = 1, S = 0, D = 0) # all the cohort starts in the Healthy state

## Create the Markov cohort trace matrix m_M that captures the proportion of
## the cohort in each state at each cycle
m_M <- matrix(0,
              nrow = (n_t + 1),
              ncol = n_states,
              dimnames = list(0:n_t, v_n)) # initialize cohort trace matrix
m_M[1, ] <- v_m0 # store the initial state vector in the first row of the cohort trace

#### Multidimensional array ####
## Create the multidimensional array a_A that captures the proportion of the
## cohort that transitioned between health states at each cycle
a_A <- array(0,
             dim = c(n_states, n_states, n_t + 1),
             dimnames = list(v_n, v_n, 0:n_t)) # initialize multidimensional array

diag(a_A[, , 1]) <- v_m0 # store the initial state vector in the diagonal of the first
    slice of A

#### State and transition rewards ####
## Create matrices to store rewards
m_R_costs <- m_R_effects <- matrix(NA,
                                   nrow = n_states,
                                   ncol = n_states,
                                   dimnames = list(v_n, v_n))

# Fill in matrix for costs
# To Healthy
m_R_costs["H", "H"]  <- c_H
m_R_costs["S", "H"]  <- c_H
m_R_costs["D", "H"]  <- c_H
# To Sick
m_R_costs["H", "S"]  <- c_S
m_R_costs["S", "S"]  <- c_S
m_R_costs["D", "S"]  <- c_S
# To Death
m_R_costs["H", "D"]  <- c_D + ic_D
m_R_costs["S", "D"]  <- c_D + ic_D
m_R_costs["D", "D"]  <- c_D

# Fill in matrix for effects
# To Healthy
m_R_effects["H", "H"]  <- u_H
m_R_effects["S", "H"]  <- u_H
m_R_effects["D", "H"]  <- u_H
# To Sick
m_R_effects["H", "S"]  <- u_S - du_HS
m_R_effects["S", "S"]  <- u_S
```

```r
m_R_effects["D", "S"]  <- u_S
# To Death
m_R_effects["H", "D"]  <- u_D
m_R_effects["S", "D"]  <- u_D
m_R_effects["D", "D"]  <- u_D

#### Expected QALYs and Costs per cycle for each strategy ####
## Create multidimensional arrays to store expected outcomes
a_Y_costs <- a_Y_effects <- array(0,
                                  dim = c(n_states, n_states, n_t + 1),
                                  dimnames = list(v_n, v_n, 0:n_t))

# Initialize arrays
a_Y_costs[, , 1]   <- a_A[, , 1] * m_R_costs
a_Y_effects[, , 1] <- a_A[, , 1] * m_R_effects

#### Run the cSTM ####
for(t in 1:n_t){  # loop through the number of cycles
  # estimate the state vector for the next cycle (t + 1)
  m_M[t + 1, ] <- m_M[t, ] %*% m_P
  a_A[, , t + 1] <- diag(m_M[t, ]) %*% m_P  # estimate the transition dynamics at t + 1

  # element-wise-multiplication of array A with the rewards matrices
  a_Y_costs[, , t + 1]   <- a_A[, , t + 1] * m_R_costs
  a_Y_effects[, , t + 1] <- a_A[, , t + 1] * m_R_effects
}

#### Aggregate outcomes ####
v_costs <- rowSums(t(colSums(a_Y_costs)))     # calculate the expected costs per cycle
v_QALYs <- rowSums(t(colSums(a_Y_effects)))   # calculate the expected QALYs per cycle
TC <- sum(v_costs)                            # calculate the total expected costs
TE <- sum(v_QALYs)                            # calculate the total expected QALYS
v_results <- c(TC, TE)                        # combine the total expected costs and QALYs
names(v_results) <- c("Costs", "Effect")      # name the vector
v_results                                     # print the results


################################################################################
### Ratio of those that transitioned from sick to dead at each cycle to those that transitioned to dead from both healthy and sick.
v_e <- numeric(n_t + 1)   # create the vector v_e
v_e[1] <- 0               # initiate the vector

### calculate the ratio across all cycles starting in cycle 2
v_e[-1] <-  a_A["S", "D", -1] / (a_A["H", "D", -1] +  a_A["S", "D", -1])
```

2 Traditional cohort trace approach of the stylistic 3-state model

Model structure

In the traditional cohort trace approach, the Markov cohort trace is calculated at every cycle t. The cohort trace only shows how the cohort is distributed among the different health states over time, but does not store information about the transitions among health states. In our stylistic 3-state model, getting sick is associated with a sudden decrease of quality of life of 0.1 and in addition transitioning to dead incurs a one-time cost of $4,000. To incorporate these transition rewards, we created two extra temporary health states at which individuals can only stay for one cycle. One temporary state for those that transitioned from healthy to sick, S_{temp}, and one temporary state for those that die, D_{temp}, coming from either healthy or sick. The S_{temp} health state allows to incorporate the sudden decrease of quality of life when getting sick and the D_{temp} state is used to incorporate the one-time cost dying. In total, the stylistic three-state cSTM with traditional cohort approach now has five health states: Healthy, Sick temporary (S_{temp}), Sick, Dead temporary (D_{temp}) and Dead. Figure 2 shows the state-transition diagram with health states and possible transitions needed when using the traditional cohort trace approach. As shown in the figure, healthy individuals can transition to S_{temp}, D_{temp} or stay healthy. Individuals in the S_{temp} (i.e., those that just turned sick) can fully recover, transitioning back to healthy, transition to the D_{temp} or remain sick, which means that they transition to Sick state. Sick individuals, can also fully

recover, transitioning back to healthy, can transition to D_{temp} or stay sick, which means they remain in the Sick state. All individuals in D_{temp} transition to the Dead state. The Dead state is the absorbing state.

This examples shows that when using the traditional cohort trace approach for the original 3-state model with only two transition rewards, the size of the number of states almost doubles. This means that incorporating transition rewards in more realistic models, that already start out with more health states, using the traditional approach results in state explosion and consequently, it is more likely to make errors while coding these models.

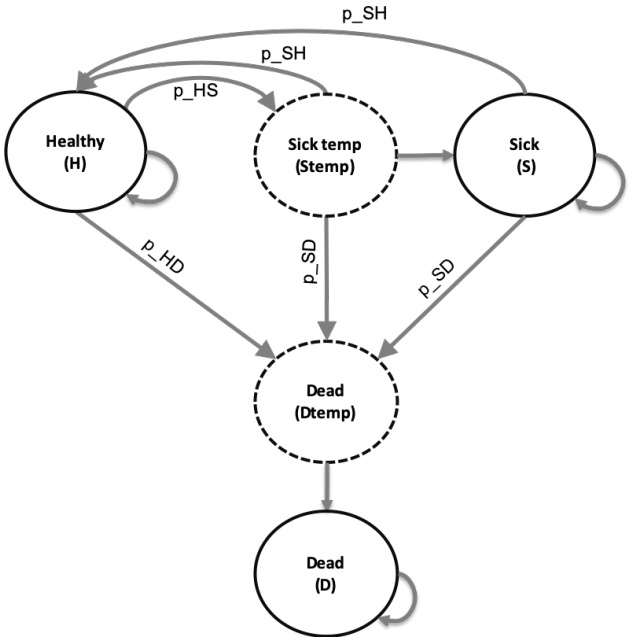

Figure 2: State-transition diagram of the healthy-sick-dead model when using the traditional cohort approach to incorporate transition rewards.

R code for the traditional cohort trace approach

We recommend downloading this code from GitHub (`https://github.com/DARTH-git/state-transition-model-dynamics`) to avoid errors due to copying from the manuscript and to obtain the latest version of the code. We use the coding convention as recommended in our coding Framework.[1] GitHub also includes the code showing that both approaches give identical model results.

```r
# Load the packages
library(reshape2) # to transform data
library(ggplot2)  # for nice looking plots

# initial set up
age          <- 70  # age of starting cohort
n_t          <- 30  # time horizon, number of cycles
v_age_names <- age:(age + n_t - 1) # vector with age names
v_n <- c("H", "Stemp", "S", "Dtemp", "D") # vector with the 3 health states of the
    model:
# Healthy (H), Sick (S), Dead (D) and two temporary health states one for Sick for the
     first time (Stemp) and one for dying (Dtemp)
n_states <- length(v_n) # number of health states


#### Generate initial set of base-case external parameters ####
# Costs
c_H    <- 1000    # cost of remaining one cycle healthy
c_S    <- 3000    # cost of remaining one cycle sick
c_D    <- 0       # cost of being dead (per cycle)
# State utilities
u_H    <- 1       # utility when healthy
u_S    <- 0.60    # utility when sick
```

```r
u_D   <- 0       # utility when healthy
# Transition probabilities (per cycle)
p_HS  <- 0.30    # probability to become sick when healthy
p_HD  <- 0.05    # probability to die when healthy
p_SH  <- 0.15    # probability to become healthy when sick
p_SD  <- 0.20    # probability to die when sick
# Transition rewards
du_HS <- 0.10    # one-time utility decrement when becoming sick
ic_D  <- 4000    # one-time cost of dying

#### Transition probability matrix ####
# matrix m_P at the first cycle
m_P <- matrix(NA,
              nrow = n_states,
              ncol = n_states,
              dimnames = list(v_n, v_n))

# Fill in matrix
# From Healthy
m_P["H", "H"]       <- 1 - (p_HS + p_HD)
m_P["H", "Stemp"]   <- p_HS
m_P["H", "S"]       <- 0
m_P["H", "Dtemp"]   <- p_HD
m_P["H", "D"]       <- 0

# From Sick temporary (first cycle being sick)
m_P["Stemp", "H"]     <- p_SH
m_P["Stemp", "Stemp"] <- 0
m_P["Stemp", "S"]     <- 1 - (p_SH + p_SD)
m_P["Stemp", "Dtemp"] <- p_SD
m_P["Stemp", "D"]     <- 0

# From Sick
m_P["S", "H"]         <- p_SH
m_P["S", "Stemp"]     <- 0
m_P["S", "S"]         <- 1 - (p_SH + p_SD)
m_P["S", "Dtemp"]     <- p_SD
m_P["S", "D"]         <- 0

# From Death temporary
m_P["Dtemp", "H"]     <- 0
m_P["Dtemp", "Stemp"] <- 0
m_P["Dtemp", "S"]     <- 0
m_P["Dtemp", "Dtemp"] <- 0
m_P["Dtemp", "D"]     <- 1

# From Death
m_P["D", "H"]         <- 0
m_P["D", "Stemp"]     <- 0
m_P["D", "S"]         <- 0
m_P["D", "Dtemp"]     <- 0
m_P["D", "D"]         <- 1

#### Cohort trace matrix ####
## Initial state vector
v_m0 <- c(H = 1, Stemp = 0, S = 0, D = 0, Dtemp = 0) # the cohort starts healthy

## Create the Markov cohort trace matrix m_M that captures the proportion of
## the cohort in each state at each cycle
m_M <- matrix(0,
              nrow = (n_t + 1),
              ncol = n_states,
              dimnames = list(0:n_t, v_n)) # initialize cohort trace matrix
m_M[1, ] <- v_m0 # store the initial state vector in the first row of the cohort trace


#### Run the cSTM ####
```

```r
for(t in 1:n_t){  # loop through the number of cycles
  # estimate the state vector for the next cycle (t + 1)
  m_M[t + 1, ] <- m_M[t, ] %*% m_P
}


#### Expected QALYs and Costs per cycle for each strategy ####

#### State and transition rewards ####
## Create a vector to store rewards
v_R_costs   <- c(c_H, c_S, c_S, c_D + ic_D, c_D)
v_R_effects <- c(u_H, u_S - du_HS, u_S, u_D, u_D)
names(v_R_costs) <- names(v_R_effects) <- v_n

#### Aggregate outcomes ####
v_costs <- m_M %*% v_R_costs          # calculate the expected costs per cycle
v_QALYs <- m_M %*% v_R_effects        # calculate the expected QALYs per cycle

TC <- sum(v_costs)                    # calculate the total expected costs
TE <- sum(v_QALYs)                    # calculate the total expected QALYS
v_results <- c(TC, TE)                # combine the total expected costs and QALYs
names(v_results) <- c("Costs", "Effect")  # name the vector
v_results                             # print the results

##############################################################################
## Calculation of the ratio of those that transitioned from sick to dead at each cycle
     to those that transitioned to dead from both healthy and sick would require an
    additional health state to distinguish those that died from healthy from those that
     died from being sick.
```

3 Comparison of approaches using a simulation study

We conducted a simulation study on computation efficiency of the two approaches: (1) dynamics-array approach and (2) traditional cohort trace approach. We defined computation efficiency as computation time in seconds and memory storage in megabytes (MB). We conducted a full factorial design with the number of states, n_{states}, and the number of cycles, n_t, in a cSTM as the factors of the simulation study. We varied the number of states from 2 to 62, incremented by 5, while the number of cycles varied from 12 to 1,320, incremented by 12. In total, we evaluated $110 * 13 = 1,430$ different scenarios. The reasoning for these numbers is as follows. The simplest cohort model is a 2-state model, therefore the minimum number of health states is 2. The maximum number of health states is set to 62, which represents the number of states of complex realistic cSTM. For the number of cycles, we assume that the maximum time horizon for a model is 110 years (modelling an individual's lifetime). When modelling this time horizon in monthly cycles, we get a total of 1320 cycles. We ran this full factorial experiment 10 times and took the average of the required time and memory to smooth out the variations in the computation time of R. Correcting for variation in R computation time is important, because the total required time is small (<50 seconds) and this would meant that even a small variation in time (e.g. 3-5 seconds) could affect our results.

To capture and calculate transition and state rewards using the cohort trace approach we created temporary health states, because a transition reward for a state is only obtained when it is first transitioned to. Without loss of generality, we assume that there are no absorbing states and every state can be visited from any other state. The transition probability matrices for both approaches were randomly sampled such that each entry is between 0 and 1 and each row sums to 1. The rewards vectors and matrices for both approaches were also randomly sampled from appropriate distributions. We set the seed of R's random number generator to assure reproducibility of these results.

For the comparison between approaches on the running time and storage memory as a function of number of health states, we fixed the number of cycles at 1,320, the maximum value tested. The top left panel of Figure 3 shows that as the number of health states increases, the run time of the traditional cohort trace approach increases almost exponentially (from 0.004 seconds with 2 health states to 46 seconds with 62 health states) while that of the dynamics-array approach varies very little (from 0.009 seconds with 2 health states to 0.344 seconds with 62 health states). In the top right panel Figure 3 we see that the time benefit of the dynamics-array approach increase as the number of health states increases. And at the point where we simulate a model with 62 health states for 1320 cycles the dynamics-array approach is 140 times faster compared to the traditional cohort trace approach.

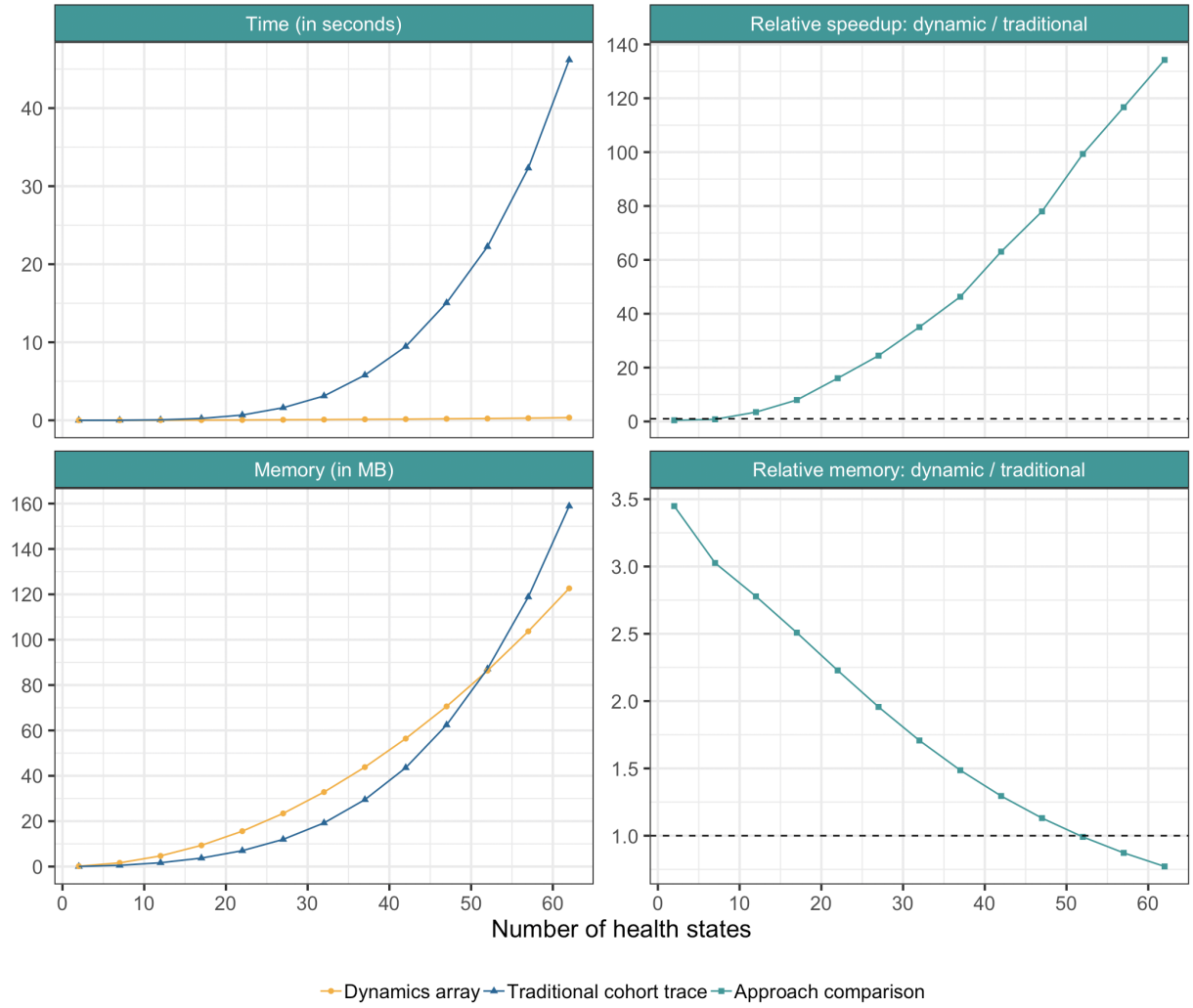

Figure 3: Computation time and memory storage of the two approaches as a function of the number of states when running the model for 1,320 cycles. The **top left** panel shows the absolute computation time in seconds of both approaches. The **top right** panel shows the relative speedup of the dynamics-array approach compared to the traditional cohort trace approach. The horizontal line at y-axis equals 0 indicates when the two approached are equally fast. The **bottom left** panel shows the absolute memory storage in megabytes (MB) of the two approached, while the **bottom right** panel shows the relative required memory of the dynamics-array approach compared to the traditional cohort trace approach. The horizontal line at y-axis equals 1 indicate when both approached required the same memory storage. Above the line the traditional cohort trace requires less memory, while below the line the dynamics-array approach requires less memory. All results are based on the average of 10 simulations. This was done to smooth out the variations caused by the computation time of R.

The bottom panel of Figure 3, shows that as the number of health states increases, the traditional cohort trace approach takes up less storage than the dynamics-array approach when the number of health states is less than 52. However, when the number of health states is greater than 52, the dynamics-array approach uses less storage memory than the traditional cohort approach.

Figure 4 illustrates how computation time of the two approaches vary as the number of health states and the number of cycles increase simultaneously. The figure shows that the run time of the cohort trace approach increases when either the number of health states or the number of cycles increases. On the contrary, the run time of the dynamics-array approach is significantly less and is invariant to increases in either the number of health states nor the number of cycles.

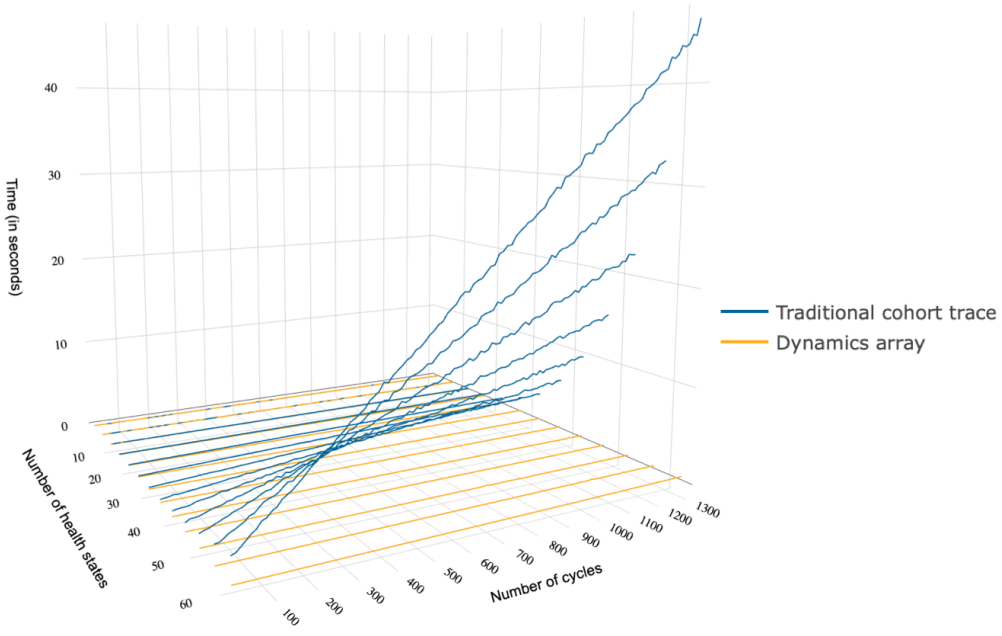

Figure 4: An three-dimensional illustration of how computation time (y-axis) of the two approaches vary as the number of states, n_{states} (x-axis), and the number of cycles, n_t (z-axis), increase simultaneously.

Figure 5 illustrates how computation storage of the two approaches vary as the number of health states and the number of cycles increase simultaneously. We see that the storage increases as when either the number of health states or the number of cycles increases. Both approaches take up approximately the same amount of storage before 52 health states. After 52 health states, the traditional cohort trace approach takes up more memory compared to the dynamics-array approach.

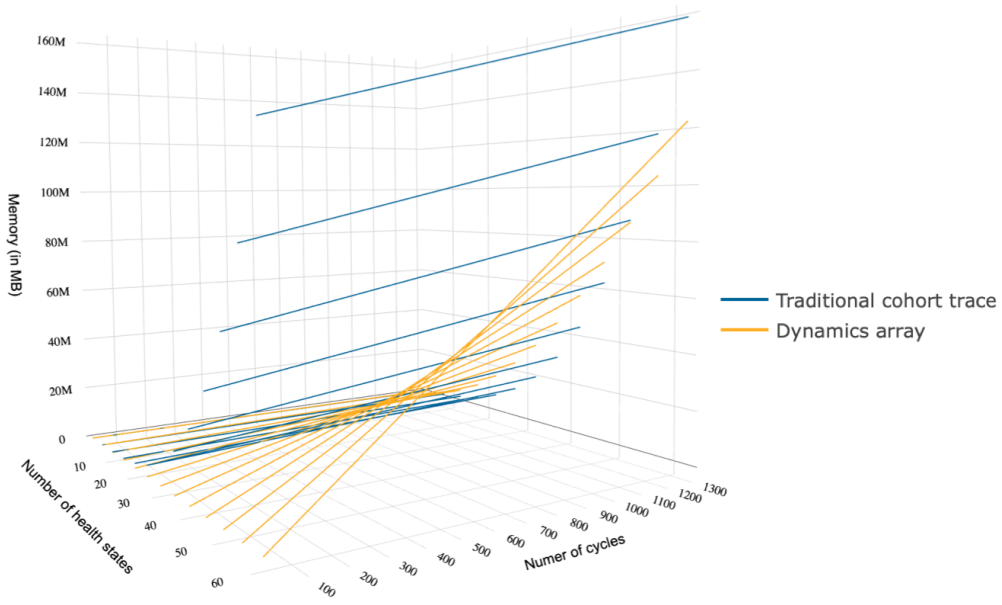

Figure 5: An three-dimensional illustration of how computation memory in bytes (y-axis) of the two approaches vary as the number of states, n_{states} (x-axis), and the number of cycles, n_t (z-axis), increase simultaneously.

Based on the results of our simulation study, we found that the dynamics-array approach is computationally superior to the traditional cohort trace approach. It is substantially faster and the increase in time is minimal when the number of cycles or the number of health states increases. In addition, the required memory of the two approaches is not that different for models with a small number of health states while the dynamics-array approach takes up less storage than the traditional cohort trace approach as the number of states increases. The code of the simulation study can be found on GitHub - (`https://github.com/DARTH-git/state-transition-model-dynamics`).

References

[1] Alarid-Escudero F, Krijkamp EM, Pechlivanoglou P, Jalal H, Kao SYZ, Yang A, Enns EA. A Need for Change! A Coding Framework for Improving Transparency in Decision Modeling. PharmacoEconomics, Sept; 2019.