A Tutorial on Time-Dependent Cohort State-Transition Models in R using a Cost-Effectiveness Analysis Example

Fernando Alarid-Escudero, PhD[*] Eline Krijkamp, MSc[†] Eva A. Enns, PhD[‡]
Alan Yang, MSc[§] Myriam G.M. Hunink, PhD[†¶] Petros Pechlivanoglou, PhD[‖]
Hawre Jalal, MD, PhD[**]

2022-05-31

Abstract

In an introductory tutorial, we illustrated building cohort state-transition models (cSTMs) in R, where the state transitions probabilities were constant over time. However, in practice, many cSTMs require transitions, rewards, or both to vary over time (time-dependent). This tutorial illustrates adding two types of time-dependency using a previously published cost-effectiveness analysis of multiple strategies as an example. The first is simulation-time dependence, which allows for the transition probabilities to vary as a function of time as measured since the start of the simulation (e.g., varying probability of death as the cohort ages). The second is state-residence time dependence, allowing for history by tracking the time spent in any particular health state using tunnel states. We use these time-dependent cSTMs to conduct cost-effectiveness and probabilistic sensitivity analyses. We also obtain various epidemiological outcomes of interest from the outputs generated from the cSTM, such as survival probability and disease prevalence, often used for model calibration and validation. We present the mathematical notation first, followed by the R code to execute the calculations. The full R code is provided in a public code repository for broader implementation.

[*]Division of Public Administration, Center for Research and Teaching in Economics (CIDE), Aguascalientes, AGS, Mexico
[†]Department of Epidemiology, Erasmus University Medical Center, Rotterdam, The Netherlands
[‡]Division of Health Policy and Management, University of Minnesota School of Public Health, Minneapolis, MN, USA
[§]The Hospital for Sick Children, Toronto
[¶]Center for Health Decision Sciences, Harvard T.H. Chan School of Public Health, Boston, USA
[‖]The Hospital for Sick Children, Toronto and University of Toronto, Toronto, Ontario, Canada
[**]University of Ottawa, Ottawa, ON, Canada

1 Introduction

Cohort state-transition models (cSTMs), commonly known as Markov models, are decision models that simulate disease dynamics over time. cSTMs are widely used in health decision sciences to evaluate various health policies and clinical strategies, such as screening and surveillance programs,[1,2] diagnostic procedures,[3] disease management programs,[4] and interventions.[5,6]. The simplest cSTMs are time-independent (time-homogeneous), meaning that the transition probabilities and other model parameters remain fixed over the simulated time horizon. The implementation of time-independent cSTMs is described in an introductory tutorial.[7] However, a time-independent cSTM is limited in many applications, because key parameters may vary over time. For example, background mortality changes as the cohort ages, the risk of cancer recurrence might change as a function of time since diagnosis, or the incidence of vector-borne diseases may have temporal or seasonal trends. The costs and utility of residing in a particular health state might also vary over time. For example, cancer-related health care costs and utilities might depend on whether a patient is in their first year of remission or their fifth. Similarly, a person's last year of life is often the most expensive in health care spending. Therefore, capturing time-varying dynamics is often essential in realistic models.

This tutorial expands the cSTM framework described in the introductory time-independent cSTM tutorial by allowing time dependence on transition probabilities, costs, and utilities. We distinguish between two types of time-dependency that require different approaches: (1) simulation-time dependence, which represents dependence on the time since the start of the simulation, and (2) state-residence time dependence, representing dependence on the time spent in a health state. A common example of simulation-time dependence is age-specific background mortality. Since all members of the cohort in the simulation age together, we can implement this dependency by changing the transition to death as the simulation progresses.[8] Similarly, in a model simulating a cohort starting from disease diagnosis, any dependence on time since diagnosis can be implemented as dependent on the time since the simulation starts. Simulation-time dependence can also be used to incorporate seasonal or temporal variation in disease incidence, where the risk of developing a disease

can vary based on the current season or year in the simulation.

Unlike simulation-time dependence, state-residence time dependence captures time dependence on events that members of the cohort could experience at different times. For example, members of a cohort of healthy individuals may experience disease onset at different times. Thus, parameters that depend on the time since disease onset cannot be implemented based on simulation time; instead, we need to track cohort members *from their disease onset time*. We implement state-residence time dependence by expanding the model state space to capture the history of the disease since disease onset. For example, instead of a single "Sick" state, individuals would transition first to the "Sick - cycle 1" state at disease onset, then "Sick - cycle 2" at the next cycle, and so on. Each intermediate "Sick" state can have different transition probabilities (e.g., mortality, risk of complications, etc.), costs, or utilities. We can also use this state expansion approach to model simulation-time dependence. However, as we see below, substantial state expansion can be cumbersome and potentially computationally expensive. For simplicity, modelers should always consider simulation-time dependence first. Besides describing the two forms of time dependence, we illustrate the concept and implementation of transition rewards, which capture one-time event-driven outcomes (e.g., costs or utilities) that individuals experience when transitioning between two states.[9] For example, a higher cost immediately at disease onset due to diagnostic tests or the one-time cost of transition to the dead state incurred from end-of-life interventions or management.

This tutorial describes how to incorporate simulation-time and state-residence time dependence in a cSTM in R[10] using a cost-effectiveness analysis (CEA) example. We illustrate the calculation of various epidemiological outcomes beyond the simple cohort trace, including survival, life expectancy, and prevalence. Such epidemiological outcomes can be informative, but they are also used to calibrate and validate the model against observed real-world data.

Readers can find the R code for this tutorial's analysis and graphs in the accompanying GitHub repository (https://github.com/DARTH-git/cohort-modeling-tutorial-timedep). We encourage readers first to review the basics of decision modeling[11] and the introductory tutorial on time-independent cSTMs in R.[7]

2 Simulation-time dependence

Simulation-time dependence represents the time since the model starts and can be represented by defining the transition probability matrix as a function of time, P_t,

$$P_t = \begin{bmatrix} p_{[1,1,t]} & p_{[1,2,t]} & \cdots & p_{[1,n_S,t]} \\ p_{[2,1,t]} & p_{[2,2,t]} & \cdots & p_{[2,n_S,t]} \\ \vdots & \vdots & \ddots & \vdots \\ p_{[n_S,1,t]} & p_{[n_S,2,t]} & \cdots & p_{[n_S,n_S,t]} \end{bmatrix},$$

where $p_{[i,j,t]}$ is the transition probability of moving from state i to state j in time t, $\{i, j\} = 1, \ldots, n_S$, $t = 0, \ldots, n_T$, n_S is the number of health states of the model, and n_T is the number of cycles that represent total simulation time. We implement this dependence in R with a three-dimensional transition probability array, $\mathbf{P}$, coded in R as `a_P`, where the third dimension represents the time since the start of the simulation. Note that in each cycle t all rows of the transition probability matrix must sum to one, $\sum_{j=1}^{n_S} p_{[i,j,t]} = 1$ for all $i = 1, \ldots, n_S$ and $t = 0, \ldots, n_T$.

Next, we specify the initial distribution of the cohort at $t = 0$. We then define $\mathbf{m}_0$ as the row vector that captures the distribution of the cohort among the states at $t = 0$ (i.e., the initial state vector). As illustrated in the introductory tutorial, we iteratively compute the cohort distribution across the health states in each cycle using the transition probability matrix and the cohort's distribution at the prior cycle. The row state vector at the next cycle $t + 1$ ($\mathbf{m}_{t+1}$) is then calculated as the matrix product of the row state vector at cycle t, $\mathbf{m}_t$, and the transition probability matrix that the cohort faces in cycle t, P_t,

$$\mathbf{m}_{t+1} = \mathbf{m}_t P_t \qquad \text{for} \qquad t = 0, \ldots, (n_T - 1). \tag{1}$$

Equation (1) is iteratively evaluated until $t = n_T$.

The cohort trace matrix, M, is a matrix of dimensions $(n_T + 1) \times n_S$ where each row is a state vector $(-\mathbf{m}_t-)$, such that

$$M = \begin{bmatrix} -\mathbf{m}_0- \\ -\mathbf{m}_1- \\ \vdots \\ -\mathbf{m}_{n_T}- \end{bmatrix}.$$

M stores cohort's distribution across states over time. Thus, M can be used to compute various epidemiological outcomes, such as prevalence and survival probability over time, and economic outcomes, such as cumulative resource use and costs.

3 Time dependence on state-residence

The implementation of state-residence time dependence is more involved than simulation-time dependence. To account for state-residence time dependence, we expand the state of interest with as many transient states as the number of cycles required to track the history of the cohort in that state. For example, if a transition rate decreases over the first three cycles spent in a state and then is constant from the fourth cycle onward, four transient states would be needed for that state. These transient states are often called *tunnel* states, where the cohort resides in each transient state for exactly one cycle, after which they either exit the tunnel or transition to the next tunnel state. The total number of states for a cSTM with tunnels is $n_{S_{\text{tunnels}}} = n_S + n_{\text{tunnels}} - 1$, where n_{tunnels} is the total number of times a health state needs to be expanded to capture the relevant residence-time dependence. We subtract one because the original state is replaced by the n_{tunnels} tunnel states. The transition probability matrix also needs to be expanded to incorporate these additional transient states, resulting in a transition probability matrix of dimensions $n_{S_{\text{tunnels}}} \times n_{S_{\text{tunnels}}}$. If the transition probabilities are also dependent on simulation time, as described in the previous section, we need to expand to a 3-dimensional transition array, with dimensions $n_{S_{\text{tunnels}}} \times n_{S_{\text{tunnels}}} \times n_T$.

The algorithm below (Algorithm 1) describes populating the transition probability array for state-residence time-dependent cSTMs.

The transition probability array with tunnels is populated similarly to the simulation-time dependence case. However, we need a different approach for the state expanded with tunnel states and any state(s) with

Algorithm 1 Populate transitions from, to, and within state expanded with tunnels.

Ensure: P_t is the transition probability matrix for cycle t, $P_t[i, j]$ is the entry corresponding to the transition probability from state i to state j at time t, parameterized as $p_{[i,j,t]}$.
Ensure: r is the state expanded as a tunnel with n_{tunnels} tunnel states and r_τ is the tunnel state corresponding to state-residence cycle τ, where $\tau = 1, \ldots, n_{\text{tunnels}}$.
  for For all i and j states different from r **do**
    $P_t[i, j] = p_{[i,j,t]}$ ▷ This is similar to the simulation-time dependence case
  end for
  for States i that can transition to the r state **do**
    $P_t[i, r_1] = p_{[i,r_1,t]}$
  end for
  for States i that can transition from any r_τ state **do**
    for $\tau = 1$ to n_{tunnels} **do**
      $P_t[r_\tau, i] = p_{[r_\tau,i,\tau]}$
      if $\tau = n_{\text{tunnels}}$ **then**
        $P_t[r_\tau, r_\tau] = p_{[r_\tau,r_\tau,\tau]}$
      else
        $P_t[r_\tau, r_{\tau+1}] = p_{[r_\tau,r_{\tau+1},\tau]}$
      end if
    end for
  end for

transitions to or from the tunnel. Filling in the transition probabilities for these states requires iterating through the tunnel states. There is typically some probability that the cohort remains in that state in the last tunnel state, like for other non-transient states. Here we only discuss models with a single tunnel. The same principles can be applied for adding more tunnels as needed.

Table 1 describes the core components of time-dependent cSTMs and their suggested R code names. The Supplementary Material contains a more detailed description of these components, including their R data structure and dimensions.

Table 1: Core components of time-dependent cSTMs with their R name.

Element	Description	R name
n_S	Number of states	`n_states`
$\mathbf{m}_0$	Initial state vector (row vector)	`v_m_init`
$\mathbf{m}_t$	State vector in cycle t (row vector)	`v_mt`
M	Cohort trace matrix	`m_M`
$\mathbf{P}$	Time-dependent transition probability array	`a_P`
$\mathbf{A}$	Transition-dynamics array	`a_A`
n_{tunnels}	Number of tunnel states	`n_tunnel_size`
$n_{S_{\text{tunnels}}}$	Number of states including tunnel states	`n_states_tunnels`
$\mathbf{m}_{\text{tunnels}_0}$	Initial state vector for the model with tunnel states	`v_m_init_tunnels`

For a more detailed description of these components with their variable types, data structure, R name, and initialization of the input parameters, please see the Supplemental Material.

4 Case study: a cost-effectiveness analysis using a time-dependent Sick-Sicker model

We demonstrate simulation-time and state-residence time-dependence in R by expanding the time-independent

In the introductory tutorial, healthy individuals faced a constant risk of death. In this tutorial, we assume that healthy individuals face an *age-specific* background mortality. Healthy individuals who survive a given cycle face a risk of becoming ill, transitioning to the "Sick" state (denoted by "S1"). To account for the unique quality-of-life impacts and health care needs at the initial onset of the illness, we include a one-time utility decrement of 0.01 (`du_HS1`) and a transition cost of $1,000 (`ic_HS1`) for each person making the transition from H to S1 in a given cycle. When individuals die, they transition to the absorbing "D" state, where they remain. We assume that the transition to "D" incurs a one-time cost of $2,000 (`ic_D`) for the expected acute care received right before dying. These impacts are included to illustrate the implementation of transition rewards.

Individuals in S1 and S2 face an increased hazard of death, compared to healthy individuals, in the form of a hazard ratio (HR) of 3 and 10, respectively, relative to the background age-specific mortality hazard rate. In the state-residence time-dependent model, the risk of progressing from S1 to S2 depends on the time spent in S1. All transitions between non-death states are assumed to be conditional on surviving each cycle.

We use this cSTM to evaluate the cost-effectiveness of different treatment strategies involving up to two different treatments (Treatment A and Treatment B) for the Sick-Sicker disease. We assume that it is not clinically possible to distinguish between individuals in S1 and S2, so any treatment strategy involves treating individuals in both S1 and S2, regardless of whether they experience a benefit. Treatment A increases the QoL of individuals only in S1 from 0.75 (utility without treatment, `u_S1`) to 0.95 (utility with Treatment A, `u_trtA`), costs $12,000 per year (`c_trtA`), and does not impact the transition rates. Treatment B reduces the progression rate from S1 to S2, with a hazard ratio (HR) of 0.6 (`hr_S1S2_trtB`), costs $13,000 per year (`c_trtB`), and does not affect QoL. We consider four treatment strategies: Standard of care with no treatment (SoC Strategy), Treatment A only (Strategy A), Treatment B only (Strategy B), and Treatment A and B used in combination (Strategy AB). When combined, the effects of Treatment A and B are assumed to be independent and treatment costs are additive. Note that for Strategy A, the model has identical transition probabilities to SoC. The differences are the added cost of the treatment for S1 and S2, and QoL increases for S1. After comparing the four strategies in terms of expected QALYs and costs, we calculate the incremental cost per QALY gained between non-dominated strategies as explained below.

We discount both costs and QALYs at an annual rate of 3%. Model parameters and the corresponding R variable names are presented in Table 2. We follow the notation described in the DARTH coding framework.[14]

Table 2: Description of parameters, their R variable name, base-case value and distribution.

Parameter	**R name**	**Base-case**	**Distribution**
Number of cycles (n_T)	`n_cycles`	75 years	-
Names of health states	`v_names_states`	H, S1, S2, D	-
Annual discount rate for costs	`d_c`	3%	-
Annual discount rate for QALYs	`d_e`	3%	-
Number of PSA samples (K)	`n_sim`	1,000	-
Annual transition rates			
- Disease onset (H to S1)	`r_HS1`	0.15	gamma(30, 200)
- Recovery (S1 to H)	`r_S1H`	0.5	gamma(60, 120)

Parameter	R name	Base-case	Distribution
- Time-independent disease progression (S1 to S2)	`r_S1S2`	0.105	gamma(84, 800)
- Time-dependent disease progression (S1 to S2)	`v_r_S1S2_tunnels`		
Weibull parameters			
Scale (λ)	`r_S1S2_scale`	0.08	lognormal(log(0.08), 0.02)
Shape (γ)	`r_S1S2_shape`	1.10	lognormal(log(1.10), 0.05)
Annual mortality			
- Age-dependent background mortality rate (H to D)	`v_r_HDage`	age-specific	-
- Hazard ratio of death in S1 vs H	`hr_S1`	3.0	lognormal(log(3.0), 0.01)
- Hazard ratio of death in S2 vs H	`hr_S2`	10.0	lognormal(log(10.0), 0.02)
Annual costs			
- Healthy individuals	`c_H`	$2,000	gamma(100.0, 20.0)
- Sick individuals in S1	`c_S1`	$4,000	gamma(177.8, 22.5)
- Sick individuals in S2	`c_S2`	$15,000	gamma(225.0, 66.7)
- Dead individuals	`c_D`	$0	-
Utility weights			
- Healthy individuals	`u_H`	1.00	beta(200, 3)
- Sick individuals in S1	`u_S1`	0.75	beta(130, 45)
- Sick individuals in S2	`u_S2`	0.50	beta(230, 230)
- Dead individuals	`u_D`	0.00	-
Treatment A: cost and effectiveness			
- Cost of Treatment A, additional to state-specific health care costs	`c_trtA`	$12,000	gamma(73.5, 163.3)
- Utility for treated individuals in S1	`u_trtA`	0.95	beta(300, 15)
Treatment B: cost and effectiveness			
- Cost of Treatment B, additional to state-specific health care costs	`c_trtB`	$12,000	gamma(86.2, 150.8)
- Reduction in rate of disease progression (S1 to S2) as hazard ratio (HR)	`hr_S1S2_trtB`	log(0.6)	lognormal(log(0.6), 0.02)
Transition rewards			
- Utility decrement of healthy individuals when transitioning to S1	`du_HS1`	0.01	beta(11,1088)
- Cost of healthy individuals when transitioning to S1	`ic_HS1`	$1,000	gamma(25, 40)
- Cost of dying when transitioning to D	`ic_D`	$2,000	gamma(100, 20)

4.1 Incorporating simulation-time dependence

To illustrate simulation-time dependence in the Sick-Sicker cSTM, we model all-cause mortality as a function of age. We obtain all-cause mortality from life tables in the form of age-specific mortality hazard rates, $\mu(a)$, where a refers to age. For this example, we create a vector `v_r_mort_by_age` to represent age-specific background mortality hazard rates for 25 to 100 year-olds obtained from US life tables.[15] The .csv file with the life tables is included in the Supplementary Material and the GitHub repository. We repeat each of the one-year-specific rates in $\mu(a)$ (R variable name `v_r_mort_by_age`) as many times as the number of cycles in a year for these ages. To compute the transition probability from state H to state D, corresponding to the cohort's age at each cycle, we transform the rate $\mu(a)$ to a transition probability assuming a constant exponential hazard rate within each year of age

$$p_{[H,D,t]} = 1 - \exp\left\{-\mu(a_0 + t)\right\},$$

where $a_0 = 25$ is the starting age of the cohort. We transform the resulting R variable, `v_r_HDage`, to a vector of probabilities, `v_p_HDage`, adjusting by the cycle length.

```
## Load life table mortality rates from `.csv` format
lt_usa_2005 <- read.csv("LifeTable_USA_Mx_2015.csv")
# Extract age-specific all-cause mortality for ages in model time horizon
v_r_mort_by_age <- lt_usa_2015 %>%
  dplyr::filter(Age >= n_age_init & Age < n_age_max) %>%
  dplyr::select(Total) %>%
  as.matrix()
# Age-specific mortality rate in the Healthy state (background mortality)
# for all cycles
v_r_HDage  <- rep(v_r_mort_by_age, each = 1/cycle_length)
# Transform to age-specific background mortality risk for all cycles adjusting
# by cycle length
v_p_HDage  <- 1 - exp(-v_r_HDage * cycle_length)
```

Because mortality in S1 and S2 are relative to background mortality which depends on age, mortality in S1 and S2 will also be age-dependent. To generate the age-specific mortality in S1 and S2, we multiply the age-specific background mortality rate, `v_r_HDage`, by the constant hazard ratios `hr_S1` and `hr_S2`, respectively. We then convert the resulting age-specific mortality rates to probabilities.

```
## Age-specific mortality rates in the Sick and Sicker states
v_r_S1Dage <- v_r_HDage * hr_S1 # when Sick
v_r_S2Dage <- v_r_HDage * hr_S2 # when Sicker
## Age-specific probabilities of dying in the Sick and Sicker states
v_p_S1Dage <- 1 - exp(-v_r_S1Dage) # when Sick
v_p_S2Dage <- 1 - exp(-v_r_S2Dage) # when Sicker
```

To incorporate simulation-time dependence into the transition probability matrix, we expand the dimensions of the matrix and create a 3-dimensional transition probability array, $\mathbf{P}$ and use `a_P` in R, of dimensions $n_S \times n_S \times n_T$. The first two dimensions of this array correspond to transitions between states, where the

rows determine departing states and columns destination states. The third dimension refers to the time since the simulation started. The third dimension refers to time since simulation start. The t-th element in the third dimension corresponds to the transition probability matrix at cycle t. A visual representation of `a_P` is shown in Figure 1.

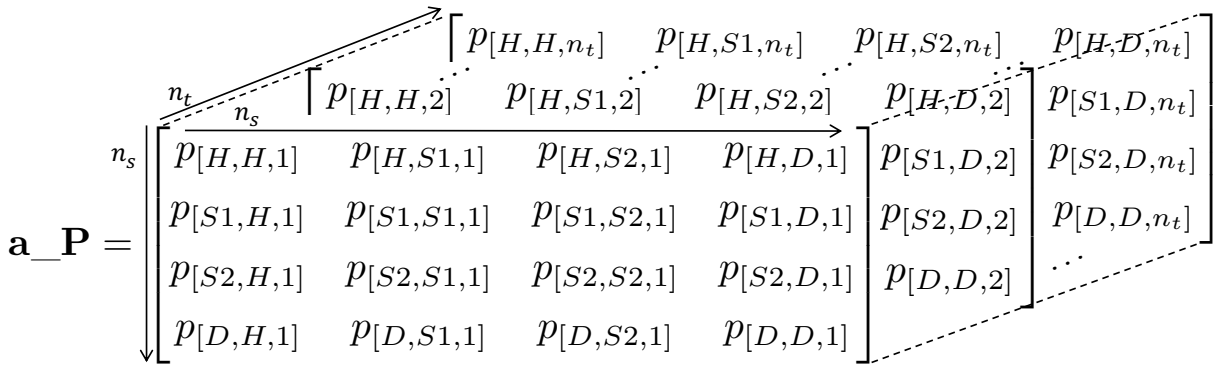


Figure 1: A 3-dimensional representation of the transition probability array of the Sick-Sicker model with simulation-time dependence.

First, we initialize the transition probability array for SoC, `a_P_SoC`, with a default value of zero for all transition probabilities.

```
# Initialize the transition probability array
a_P_SoC <- array(0, dim = c(n_states, n_states, n_cycles),
                 dimnames = list(v_names_states, v_names_states, 0:(n_cycles - 1)))
```

The code below illustrates how to assign age-dependent transition probabilities in the third dimension of the array `a_P_SoC`.

```
### Fill in array
## From H
a_P_SoC["H", "H", ]   <- (1 - v_p_HDage) * (1 - p_HS1)
a_P_SoC["H", "S1", ]  <- (1 - v_p_HDage) * p_HS1
a_P_SoC["H", "D", ]   <- v_p_HDage
## From S1
a_P_SoC["S1", "H", ]  <- (1 - v_p_S1Dage) * p_S1H
a_P_SoC["S1", "S1", ] <- (1 - v_p_S1Dage) * (1 - (p_S1H + p_S1S2))
a_P_SoC["S1", "S2", ] <- (1 - v_p_S1Dage) * p_S1S2
a_P_SoC["S1", "D", ]  <- v_p_S1Dage
## From S2
a_P_SoC["S2", "S2", ] <- 1 - v_p_S2Dage
a_P_SoC["S2", "D", ]  <- v_p_S2Dage
## From D
a_P_SoC["D", "D", ]   <- 1
```

```r
## Initialize transition probability matrix for Strategy A as a copy of SoC's
a_P_strA <- a_P_SoC
```

Most of this code is equivalent to populating the matrix P for time-independent cSTM. The only difference is the empty index added to the third dimension of `a_P_SoC` to represent the vector of transitions over time. For time-varying transitions, we provide a vector of transition probabilities. We only need to provide one value for the transition probability for constant transitions over time, and R replicates the value of such transitions as many times as the number of cycles ($n_T + 1$ times in our example). To ensure that the sum of each row of each cycle-specific matrix equals one, we define the probability of staying in each state by subtracting the sum of the exiting probability vectors from one after conditioning on surviving each cycle. Finally, we initialize the transition probability array for Strategy A as a copy of SoC's because Treatment A does not alter the cohort's transition probabilities.

Each slice along the third dimension of `a_P_SoC` corresponds to a transition probability matrix from t to $t + 1$. For example, the transition matrix for 25-year-olds in the Sick-Sicker model under the SoC Strategy can be retrieved by indexing the first slice of the array using:

```r
a_P_SoC[, , 1]
```

```
##            H        S1         S2           D
## H  0.8598357 0.1391509 0.00000000 0.001013486
## S1 0.3922742 0.5053157 0.09937273 0.003037378
## S2 0.0000000 0.0000000 0.98991124 0.010088764
## D  0.0000000 0.0000000 0.00000000 1.000000000
```

Similarly, the code below defines `a_P_strB` for Strategy B.

```r
## Initialize transition probability array for Strategy B
a_P_strB <- a_P_SoC
## Update only transition probabilities from S1 involving p_S1S2
a_P_strB["S1", "S1", ] <- (1 - v_p_S1Dage) * (1 - (p_S1H + p_S1S2_trtB))
a_P_strB["S1", "S2", ] <- (1 - v_p_S1Dage) * p_S1S2_trtB

## Initialize transition probability matrix for Strategy AB as a copy of B's
a_P_strAB <- a_P_strB
```

We instantiate `a_P_strB` as a copy of `a_P_SoC` and update the probability of remaining in S1, and the transition probability from S1 to S2 (i.e., `p_S1S2` is replaced with `p_S1S2_trtB`). Next, we create the transition probability array for Strategy AB, `a_P_strAB`, as a copy of `a_P_strB` since the cSTMs for Strategies B and AB have identical transition probabilities but are only different in cost, as illustrated below.

We used the functions `check_sum_of_transition_array` and `check_transition_probability` in the `darthtools` package (https://github.com/DARTH-git/darthtools) to check if the transition probability arrays are valid (i.e., ensuring transition probabilities are between 0 and 1, and transition probabilities from each state sum to 1). The Supplementary Material includes the code for these functions.

In the Sick-Sicker model, the entire cohort starts in the Healthy state. Therefore, we create the $1 \times n_S$ initial

state vector `v_m_init` with all of the cohort assigned to the H state:

```
v_m_init <- c(H = 1, S1 = 0, S2 = 0, D = 0) # initial state vector
v_m_init
#  H S1 S2  D
#  1  0  0  0
```

We use the variable `v_m_init` to initialize the cohort trace, M, represented by `m_M` for the cohort under the SoC Strategy. We also create a trace for each of the other treatment-based strategies. Note that the initial state vector, `v_m_init`, can be modified to account for the distribution of the cohort across the states at the start of the simulation and might vary by strategy. To simulate the cohort over the n_T cycles for the simulation-time-dependent cSTM under SoC, we initialize the cohort trace matrix.

```
## Initialize cohort trace for age-dependent cSTM under SoC
m_M_SoC <- 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_SoC[1, ] <- v_m_init
```

The code below iteratively computes the cohort's distribution across health states over time.

The state vector of the cohort's distribution at cycle $t + 1$ is obtained by applying Equation (1), the matrix product between the row vector `m_M_SoC[t, ]` and the transition probability matrix `a_P_SoC[, , t]`. This equation is comparable to the one described for the time-independent model in the introductory tutorial.[7] The only modification required is to index the third dimension of the transition probability arrays by t to obtain the SoC's cycle-specific transition probability matrix. The vectors across all cycles are obtained by iteratively applying Equation (1) to each transition probability matrix of `a_P_SoC` across the third dimension using a `for` loop. Figure 2 shows the cohort's trace for all cycles of the age-dependent cSTM under SoC. The complete code for all the strategies is included in the Supplemental Material.

```
# Iterative solution of age-dependent cSTM under SoC
for(t in 1:n_cycles){
  m_M_SoC[t + 1, ] <- m_M_SoC[t, ] %*% a_P_SoC[, , t]
}
```

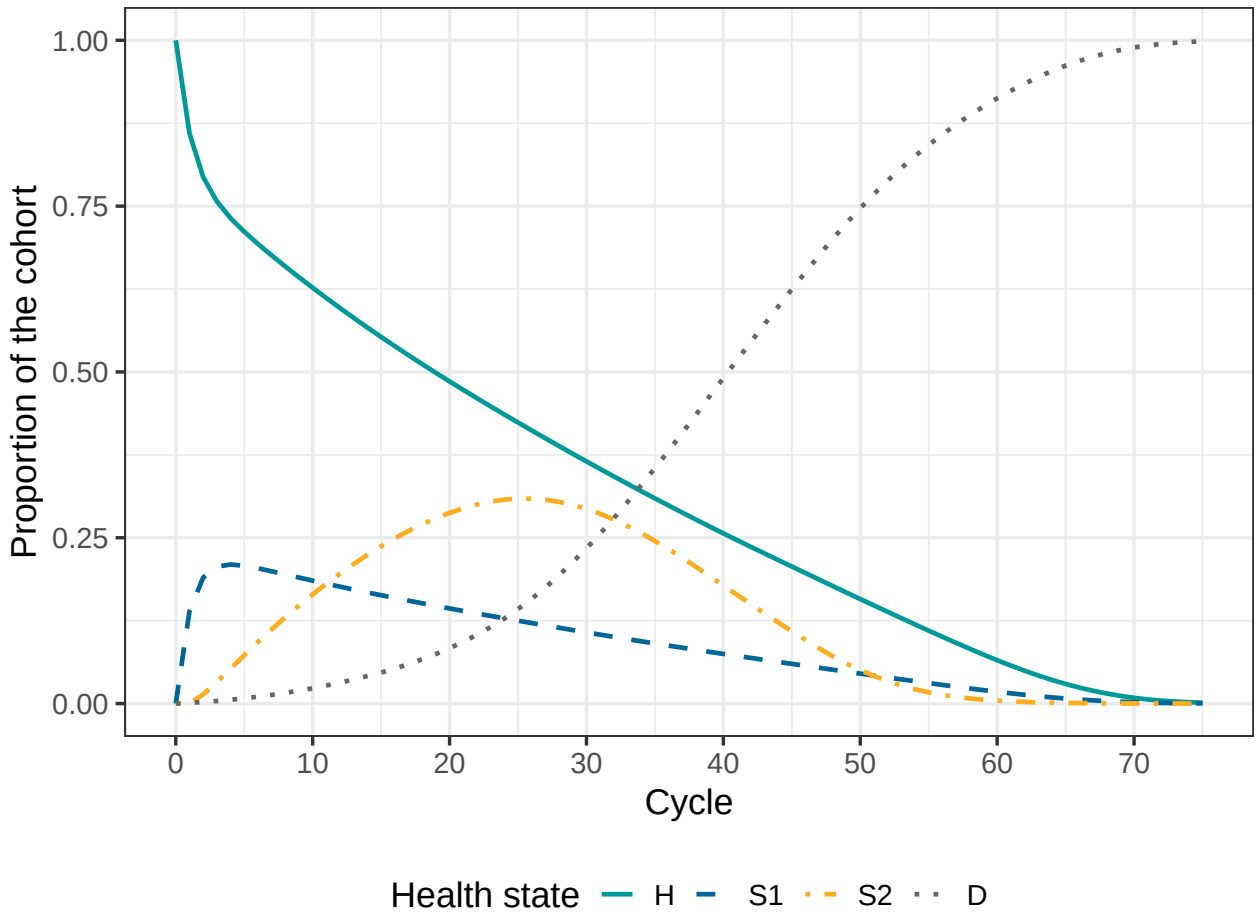


Figure 2: Cohort trace of the age-dependent cSTM under SoC.

4.2 Incorporating time dependence on state residence (tunnel states)

Here, we add the dependence on state-residence to the simulation-time-dependent Sick-Sicker model defined above. We assume the risk of progression from S1 to S2 increases as a function of the time $\tau = 1, \ldots, n_{\text{tunnels}}$ the cohort remains in the S1 state. This increase follows a Weibull hazard function, $h(\tau)$, defined as

$$h(\tau) = \gamma\lambda(\lambda\tau)^{\gamma-1},$$

with a corresponding cumulative hazard, $H(\tau)$,

$$H(\tau) = (\lambda\tau)^{\gamma}, \tag{2}$$

where λ and γ are the scale and shape parameters of the Weibull hazard function, respectively.

To derive a transition probability from S1 to S2 as a function of the time the cohort spends in S1, $p_{[S1_\tau,S2,\tau]}$, we assume constant rates within each cycle interval (i.e., piecewise exponential transition times), where the cycle-specific probability of a transition is

$$p_{[S1_\tau,S2,\tau]} = 1 - \exp\left(-\mu_{[S1_\tau,S2,\tau]}\right), \tag{3}$$

where $\mu_{[S1_\tau,S2,\tau]}$ is the rate of transition from S1 to S2 in cycle τ defined as the difference in cumulative hazards between consecutive cycles[16]

$$\mu_{[S1_\tau,S2,\tau]} = H(\tau) - H(\tau - 1). \tag{4}$$

Substituting the Weibull cumulative hazard from Equation (2) into Equation (4) gives

$$\mu_{[S1_\tau,S2,\tau]} = (\lambda\tau)^\gamma - (\lambda(\tau - 1))^\gamma, \tag{5}$$

and the transition probability

$$p_{[S1_\tau,S2,\tau]} = 1 - \exp\left(-\left((\lambda\tau)^\gamma - (\lambda(\tau - 1))^\gamma\right)\right). \tag{6}$$

We assume that state-residence dependence affects the cohort in the S1 state throughout the whole simulation (i.e., $n_{\text{tunnels}} = n_T$) and create a new variable called `n_tunnel_size` with the length of the tunnel equal to `n_cycles`. Thus, there will be 75 S1 tunnel states plus 3 more states (H, S2, D) resulting in a total of $n_{S_{\text{tunnels}}} = 78$.

Figure 3 shows the state-transition diagram of the Sick-Sicker model with state-residence dependency with n_{tunnels} tunnel states for S1.

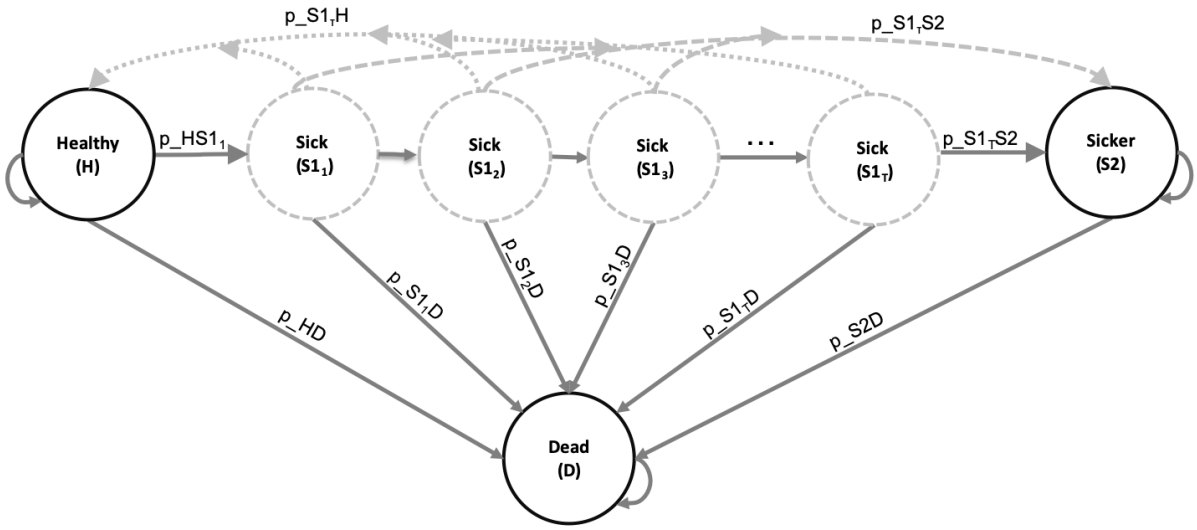


Figure 3: State-transition diagram of the Sick-Sicker model with tunnel states expanding the Sick state $(S1_1, S1_2, ..., S1_{n_{\text{tunnels}}})$. The circles represent the health states, and the arrows represent the possible transition probabilities. The labels next to the arrows represent the variable names for these transitions.

To implement state-residence dependence in the Sick-Sicker cSTM, we create the vector variables `v_Sick_tunnel` and `v_names_states_tunnels` with the names of the Sick tunnel states' and all the states of the cSTM, including tunnels, respectively, and use the parameters listed in Table 1.

```
## Number of tunnels
n_tunnel_size <- n_cycles
## Vector with cycles for tunnels
v_cycles_tunnel <- 1:n_tunnel_size
## Vector with the names of the Sick tunnel state
v_Sick_tunnel <- paste("S1_", seq(1, n_tunnel_size), "Yr", sep = "")
## Create variables for model with tunnels
v_names_states_tunnels <- c("H", v_Sick_tunnel, "S2", "D") # state names
n_states_tunnels <- length(v_names_states_tunnels)         # number of states
## Initialize first cycle of Markov trace accounting for the tunnels
v_m_init_tunnels <- c(1, rep(0, n_tunnel_size), 0, 0)
```

Then, the transition rate and probability dependent on state residence from Sick to Sicker, `v_r_S1S2_tunnels` and `v_p_S1S2_tunnels`, respectively, based on a Weibull hazard function are:

```
# Weibull parameters
r_S1S2_scale <- 0.08 # scale
r_S1S2_shape <- 1.10 # shape
# Weibull function
v_r_S1S2_tunnels <- (v_cycles_tunnel*r_S1S2_scale)^r_S1S2_shape -
                    ((v_cycles_tunnel - 1)*r_S1S2_scale)^r_S1S2_shape

v_p_S1S2_tunnels <- 1 - exp(-v_r_S1S2_tunnels*cycle_length)
```

To adapt the 3-dimensional transition probability array to incorporate both age and state-residence dependence in the Sick-Sicker model under SoC, we first create an expanded 3-dimensional array accounting for tunnels, `a_P_tunnels_SoC`. The dimensions of this array are $n_{S_{\text{tunnels}}} \times n_{S_{\text{tunnels}}} \times n_T$. A visual representation of `a_P_tunnels_SoC` of the Sick-Sicker model with tunnel states expanding the Sick state is shown in Figure 4.

```
# Initialize array
a_P_tunnels_SoC <- array(0, dim = c(n_states_tunnels, n_states_tunnels, n_cycles),
                         dimnames = list(v_names_states_tunnels,
                                         v_names_states_tunnels,
                                         0:(n_cycles - 1)))
```

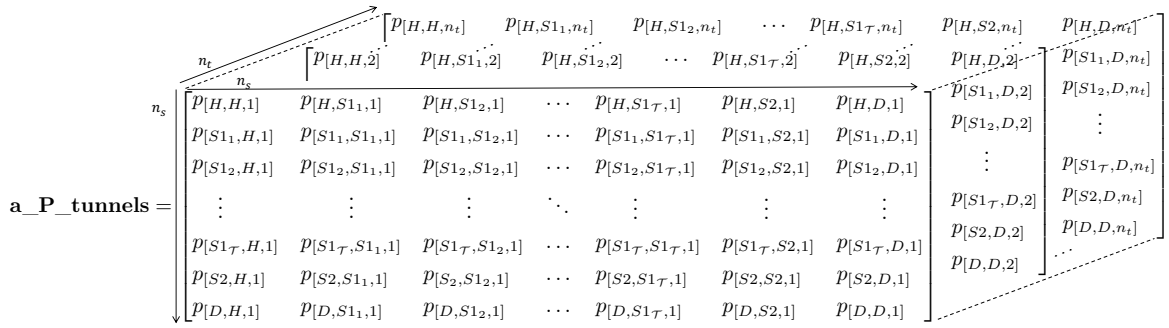


Figure 4: The 3-dimensional transition probability array of the Sick-Sicker model expanded to account for simulation-time and state-residence time dependence using τ tunnel states for S1.

Filling `a_P_tunnels_SoC` with the corresponding transition probabilities works similarly to `a_P_SoC` above. The difference is that we now fill the transition probabilities to/from tunnel states by iterating over the tunnel and assigning the appropriate disease progression transition probabilities for each state, as described in Section 3 and Algorithm 1.

```
### Fill in array
## From H
a_P_tunnels_SoC["H", "H", ]               <- (1 - v_p_HDage) * (1 - p_HS1)
a_P_tunnels_SoC["H", v_Sick_tunnel[1], ] <- (1 - v_p_HDage) * p_HS1
a_P_tunnels_SoC["H", "D", ]               <- v_p_HDage
## From S1
for(i in 1:n_tunnel_size){
  a_P_tunnels_SoC[v_Sick_tunnel[i], "H", ]  <- (1 - v_p_S1Dage) * p_S1H
  a_P_tunnels_SoC[v_Sick_tunnel[i], "S2", ] <- (1 - v_p_S1Dage) * v_p_S1S2_tunnels[i]
  a_P_tunnels_SoC[v_Sick_tunnel[i], "D", ]  <- v_p_S1Dage
  if(i == n_tunnel_size){
    # If reaching last tunnel state, ensure that the cohort that does not
    # transition out of the Sick state, remain in the last Sick tunnel state
    a_P_tunnels_SoC[v_Sick_tunnel[i],
                    v_Sick_tunnel[i], ] <- (1 - v_p_S1Dage) *
      (1 - (p_S1H + v_p_S1S2_tunnels[i]))
  } else{
    a_P_tunnels_SoC[v_Sick_tunnel[i],
                    v_Sick_tunnel[i + 1], ]   <- (1 - v_p_S1Dage) *
      (1 - (p_S1H + v_p_S1S2_tunnels[i]))
  }
}
### From S2
a_P_tunnels_SoC["S2", "S2", ] <- 1 - v_p_S2Dage
a_P_tunnels_SoC["S2", "D", ]  <- v_p_S2Dage
# From D
```

```
a_P_tunnels_SoC["D", "D", ] <- 1
```

Again, we check the three-dimensional transition probability array with tunnels is valid using the `darthtools` package.

To simulate the cohort and store its state occupation over the n_T cycles for the cSTM accounting for state-residence dependency, we initialize a cohort trace matrix for SoC Strategy, `m_M_tunnels_SoC`. The dimensions of the matrix are $(n_T + 1) \times n_{S_{\text{tunnels}}}$.

```
# Initialize cohort for state-residence cSTM under SoC
m_M_tunnels_SoC <- matrix(0,
                      nrow = (n_cycles + 1), ncol = n_states_tunnels,
                      dimnames = list(0:n_cycles, v_names_states_tunnels))
# Store the initial state vector in the first row of the cohort trace
m_M_tunnels_SoC[1, ] <- v_m_init_tunnels
```

We then use the matrix product, similar to the simulation-time-dependent cSTM, to generate the full cohort trace.

```
# Iterative solution of state-residence-dependent cSTM under SoC
for(t in 1:n_cycles){
  m_M_tunnels_SoC[t + 1, ] <- m_M_tunnels_SoC[t, ] %*% a_P_tunnels_SoC[, , t]
}
```

To compute a summarized cohort trace to capture occupancy in the H, S1, S2, D states under SoC, we aggregate over the S1 tunnel states in each cycle. The complete code for all the strategies is included in the Supplemental Material.

```
# Create aggregated trace
m_M_tunnels_SoC_sum <- cbind(H = m_M_tunnels_SoC[, "H"],
                             S1 = rowSums(m_M_tunnels_SoC[, which(v_names_states=="S1"):
                                                            (n_tunnel_size +1)]),
                             S2 = m_M_tunnels_SoC[, "S2"],
                             D = m_M_tunnels_SoC[, "D"])
```

5 Economic outcomes

In a CEA, the main outcomes are typically the total expected discounted QALYs and costs accrued by the cohort over the predefined time horizon. Below, we calculate economic outcomes from state and transition rewards. A "state reward" refers to a value (e.g., cost, utility) assigned to individuals for remaining in a given health state for one cycle. A "transition reward" refers to the increase or decrease in either costs or utilities of transitioning from one health state to another, which may be associated with a one-time cost or utility impact. The introductory tutorial describes how to incorporate state rewards in CEA in detail.[7] Here, we describe and illustrate how to implement state and transition rewards together using a transition dynamics array.

5.1 State rewards

To add state rewards to the Sick-Sicker model, we first create a vector of utilities and costs for each of the four strategies considered. The vectors of utilities and costs, `v_u_SoC` and `v_c_SoC`, respectively, contain the utilities and costs corresponding to being in each of the four health states under SoC, shown in Table 2.

```
# Vector of state utilities under SoC
v_u_SoC <- c(H = u_H, S1 = u_S1, S2 = u_S2, D = u_D)
# Vector of state costs under SoC
v_c_SoC <- c(H = c_H, S1 = c_S1, S2 = c_S2, D = c_D)
```

We account for the benefits and costs of both treatments individually and their combination to create the state-reward vectors under treatments A and B (strategies A and B, respectively) and when applied jointly (Strategy AB). Only Treatment A affects QoL, so we create a vector of utilities for Strategy A, `v_u_strA`, substituting the utility of being in S1 under SoC, `u_S1`, with that under Treatment A, `u_trtA`. Treatment B does not affect QoL, so the vector of utilities for Strategy B, `v_u_strB`, is the same as SoC's vector. However, when both treatments A and B are applied jointly (Strategy AB), the resulting vector of utilities `v_u_strAB` equals that of Strategy A.

```
# Vector of state utilities for Strategy A
v_u_strA <- c(H = u_H, S1 = u_trtA, S2 = u_S2, D = u_D)
# Vector of state utilities for Strategy B
v_u_strB <- v_u_SoC
# Vector of state utilities for Strategy AB
v_u_strAB <- v_u_strA
```

Both treatments A and B incur a cost. To create the vector of state costs for Strategy A, `v_c_strA`, we add the cost of Treatment A, `c_trtA`, to S1 and S2 state costs. Similarly, when constructing the vector of state costs for Strategy B, `v_c_strB`, we add the cost of Treatment B, `c_trtB`, to S1 and S2 state costs. Finally, for the vector of state costs for Strategy AB, `v_c_strAB`, we add both treatment costs to the state costs of S1 and S2.

```
# Vector of state costs for 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 costs for 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 costs for 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)
```

5.1.1 Transition rewards

As previously mentioned, dying (i.e., transitioning to the Dead state) incurs a one-time cost of $2,000 that reflects the acute care that might be received immediately preceding death. We also assume a disutility and a cost increment on the transition from H to S1. Incorporating transition rewards requires keeping track of the proportion of the cohort that transitions between health states in each cycle while capturing the origin and destination states for each transition. The cohort trace, M, does not capture this information. However, obtaining this information is relatively straightforward in a cSTM and is described in detail by Krijkamp et al. (2020).[9] Briefly, this approach involves changing the core computation in a traditional cSTM, from $m_t P_t$ to $\text{diag}(m_t)P_t$. This simple change allows us to compute the proportion of the cohort that transitions between any two states in a cycle t. The result is no longer a cohort trace matrix but rather a three-dimensional array that we refer to as a transition-dynamics array ($\mathbf{A}$) with dimensions $n_S \times n_S \times [n_T + 1]$. The $t-$th slice of $\mathbf{A}$, A_t, is the matrix that stores the proportion of the population that transitioned between any two states from cycles $t-1$ to t. Similarly, we define the transition rewards by the states of origin and destination.

To account for state and transition rewards, we create a *matrix* of rewards R_t of dimensions $n_S \times n_S$. The off-diagonal entries of R_t store the transition rewards, and the diagonal of R_t stores the state rewards for cycle t and assumes that rewards occur at the beginning of the cycle.[9] Finally, we multiply this matrix by A_t, the t-th slice of A, apply discounting, within-cycle correction, and compute the overall reward for each strategy outcome. Below, we illustrate these computations in R.

To compute $\mathbf{A}$ for the simulation-time-dependent Sick-Sicker model under SoC, we initialize a three-dimensional array `a_A_SoC` of dimensions $n_S \times n_S \times [n_T + 1]$ and set the diagonal of the first slice to the initial state vector `v_m_init`.

```
# Initialize transition-dynamics array under SoC
a_A_SoC <- array(0,
                 dim = c(n_states, n_states, (n_cycles + 1)),
                 dimnames = list(v_names_states, v_names_states, 0:n_cycles))
# Set first slice to the initial state vector in its diagonal
diag(a_A_SoC[, , 1]) <- v_m_init
```

We then compute a matrix multiplication between a diagonal matrix of each of the t-th rows of the cohort trace matrix under SoC, denoted as `diag(m_M_SoC[t, ])`, by the t-th matrix of the array of the transition matrix, `a_P_SoC[, , t]`, over all n_T cycles.

```
# Iterative solution to produce the transition-dynamics array under SoC
for (t in 1:n_cycles){
  a_A_SoC[, , t + 1] <- diag(m_M_SoC[t, ]) %*% a_P_SoC[, , t]
}
```

To create the arrays of rewards for costs and utilities for the simulation-time-dependent Sick-Sicker cSTM under SoC, we initialize three-dimensional arrays of rewards and fill each row across the third dimension with the vector of state rewards.

```r
# Arrays of state and transition rewards under SoC
# Utilities
a_R_u_SoC <- array(matrix(v_u_SoC, nrow = n_states, ncol = n_states, byrow = T),
                  dim = c(n_states, n_states, n_cycles + 1),
                  dimnames = list(v_names_states, v_names_states, 0:n_cycles))
# Costs
a_R_c_SoC <- array(matrix(v_c_SoC, nrow = n_states, ncol = n_states, byrow = T),
                  dim = c(n_states, n_states, n_cycles + 1),
                  dimnames = list(v_names_states, v_names_states, 0:n_cycles))
```

To account for the transition rewards, we either add or subtract them in the corresponding location of the reward matrix representing the transitions of interest. Thus, for example, to account for the disutility of transitioning from H to S1 under SoC, we subtract the disutility to the entry of the array of rewards corresponding to the transition from H to S1 across all cycles.

```r
# Add disutility due to transition from H to S1
a_R_u_SoC["H", "S1", ] <- a_R_u_SoC["H", "S1", ] - du_HS1
```

In a similar approach, we add the costs of transitioning from H to S1 and the cost of dying.

```r
# Add transition cost due to transition from H to S1
a_R_c_SoC["H", "S1", ] <- a_R_c_SoC["H", "S1", ] + ic_HS1
# Add transition cost of dying from all non-dead states
a_R_c_SoC[-n_states, "D", ] <- a_R_c_SoC[-n_states, "D", ] + ic_D
a_R_c_SoC[, , 1]
```

```
##       H   S1    S2    D
## H  2000 5000 15000 2000
## S1 2000 4000 15000 2000
## S2 2000 4000 15000 2000
## D  2000 4000 15000    0
```

The state and transition rewards are applied to the model dynamics by element-wise multiplication between $\mathbf{A}$ and $\mathbf{R}$, indicated by the $\odot$ sign, which produces the array of economic outcomes for all n_T cycles, $\mathbf{Y}$. Formally,

$$\mathbf{Y} = \mathbf{A} \odot \mathbf{R} \tag{7}$$

To obtain $\mathbf{Y}$ for QALYs and costs for all four strategies, we apply Equation (7) by the element-wise multiplication of the transition array `a_A_SoC` by the corresponding array of rewards.

```r
# For SoC
a_Y_c_SoC <- a_A_SoC * a_R_c_SoC
a_Y_u_SoC <- a_A_SoC * a_R_u_SoC
```

The total rewards for each health state at cycle t, $\mathbf{y}_t$, is obtained by summing the rewards across all

$j = 1, \ldots, n_S$ health states for all n_T cycles.

$$\mathbf{y}_t = \mathbf{1}^T Y_t = \left[\sum_{i=1}^{n_S} Y_{[i,1,t]}, \sum_{i=1}^{n_S} Y_{[i,2,t]}, \ldots, \sum_{i=1}^{n_S} Y_{[i,n_S,t]} \right]. \tag{8}$$

To obtain the expected costs and QALYs per cycle under SoC, $\mathbf{y}$, we apply Equation (8) again across all the matrices of the third dimension of $\mathbf{Y}$ for all the outcomes. The complete code for all the strategies is presented in the Supplementary Material.

```
# Vectors of rewards under SoC
# QALYs
v_qaly_SoC <- rowSums(t(colSums(a_Y_u_SoC)))
# Costs
v_cost_SoC <- rowSums(t(colSums(a_Y_c_SoC)))
```

5.1.2 Within-cycle correction and discounting future rewards

We use Simpson's 1/3rd rule for within-cycle correction (WCC),[18] and use exponential discounting for costs and QALYs. Please see the Supplemental material and introductory cSTM tutorial for a detailed description of implementing discounting and WCC in R.[7]

6 Incremental cost-effectiveness ratios (ICERs)

We followed the coding approach described in the introductory cSTM tutorial to conduct the cost-effectiveness analysis.[7] We use the R package `dampack` (https://cran.r-project.org/web/packages/dampack/)[19] to calculate the incremental costs and effectiveness and the incremental cost-effectiveness ratio (ICER) between non-dominated strategies. `dampack` organizes and formats the results as a data frame, `df_cea`, that can be printed as a formatted table.

The SoC Strategy is the least costly and effective strategy, followed by Strategy B, producing an expected benefit of 1.32 QALYs per individual for an additional expected cost of $86,162 with an ICER of $65,288/QALY followed by Strategy AB with an ICER $104,461/QALY. Strategy A is a dominated strategy. The results of the CEA of the simulation-time-dependent Sick-Sicker model are presented in Table 3. The non-dominated strategies, SoC, B, and AB, form the cost-effectiveness efficient frontier of the CEA based on the simulation-time-dependent Sick-Sicker model (Figure 5).

Table 3: Cost-effectiveness analysis results for the simulation-time-dependent Sick-Sicker model. ND: Non-dominated strategy; D: Dominated strategy.

Strategy	Costs ($)	QALYs	Incremental Costs ($)	Incremental QALYs	ICER ($/QALY)	Status
Standard of care	116,374	18.879	NA	NA	NA	ND
Strategy B	202,536	20.199	86,162	1.320	65,288	ND
Strategy AB	296,300	21.097	93,764	0.898	104,461	ND
Strategy A	218,789	19.636	NA	NA	NA	D

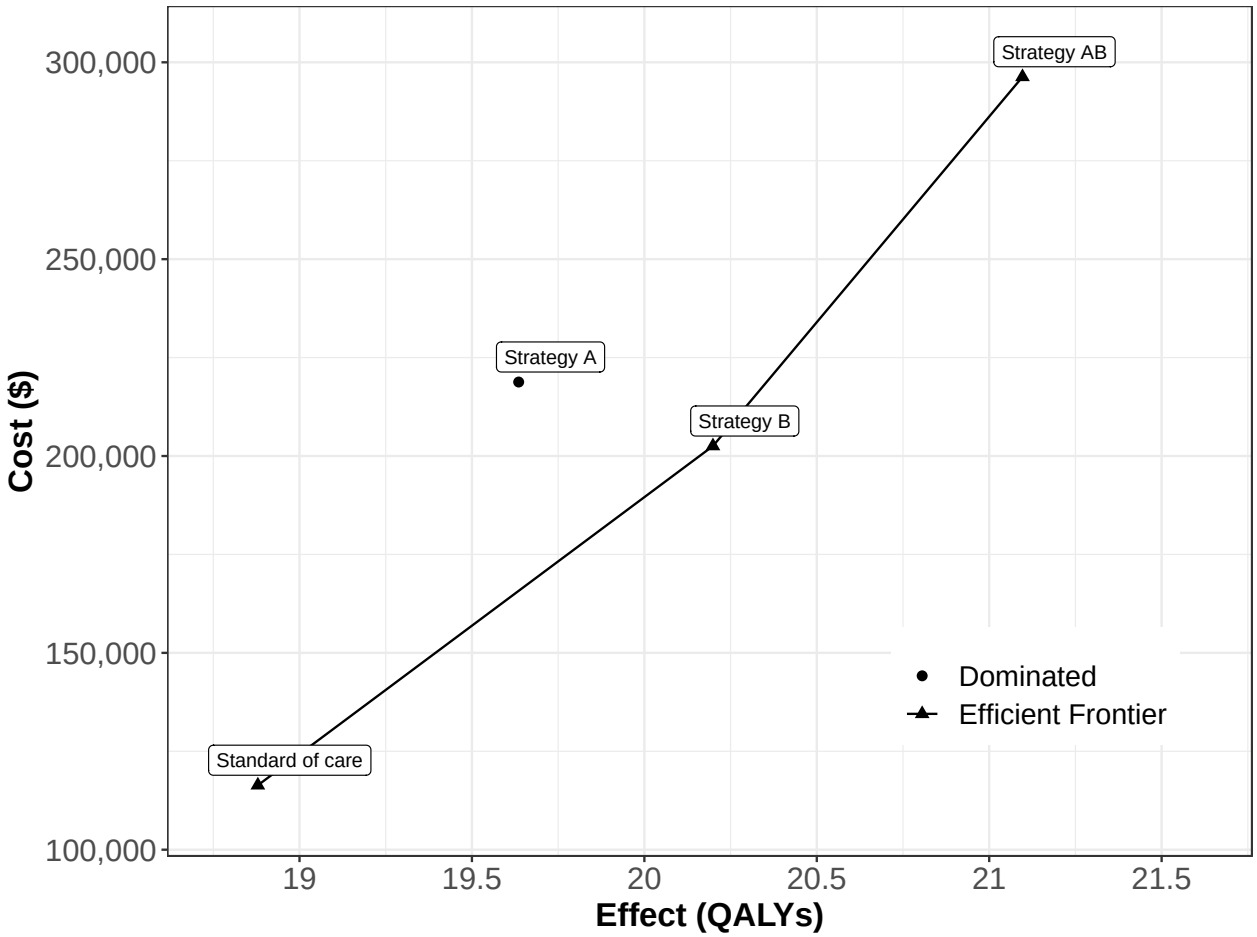


Figure 5: Cost-effectiveness efficient frontier of all four strategies for the simulation-time-dependent Sick-Sicker model.

7 Probabilistic sensitivity analysis

We conducted a probabilistic sensitivity analysis (PSA) to quantify the effect of model parameter uncertainty on cost-effectiveness outcomes.[20] In a PSA, we randomly draw parameter sets from distributions that reflect the current uncertainty in model parameter estimates. The parameters' distributions and their values are described in Table 2 and in more detail in the Supplementary Material. We compute model outcomes for each sampled set of parameter values (e.g., total discounted cost and QALYs) for each strategy. We follow the steps to conduct a PSA from a previously published article describing the Decision Analysis in R for technologies in Health (DARTH) coding framework.[14] The R code to generate the PSA is included in the Supplementary Material. The description of the functions and steps and a brief discussion on choosing distributions are described in more detail in the introductory tutorial.[7]

To conduct the PSA of the CEA using the simulation time-dependent Sick-Sicker cSTM, we sampled 1,000 parameter sets. We computed the total discounted costs and QALYs of each simulated strategy for each parameter set. We generated the cost-effectiveness scatter plot,[21] where each of the 4,000 simulations (i.e., 1,000 combinations of total discounted expected costs and QALYs for each of the four strategies) are plotted as a point in the graph (Figure 6A)). We also added the 95% confidence ellipse, and the expected values of

the total discounted costs and QALYs for each strategy. The CE scatter plot shows that Strategy AB has the highest expected costs and QALYs. Standard of care has the lowest expected cost and QALYs. Strategy B is more effective and least costly than Strategy A. Therefore, Strategy A is strongly dominated by Strategy B.

In Figure 6B, we present the cost-effectiveness acceptability curves (CEACs), showing the probability that each strategy is cost-effective, and the cost-effectiveness frontier (CEAF), displaying the probability of the optimal strategy being cost-effective over a range of willingness-to-pay (WTP) thresholds. At WTP thresholds less than $70,000 per QALY, SoC is the strategy with the highest probability of being cost-effective and the highest expected NMB. Strategy B has the highest probability of being cost-effective and the highest expected NMB for WTP thresholds greater than $70,000 and lower than $105,000 per QALY. Strategy AB has the highest expected NMB for WTP thresholds greater than or equal to $105,000 and is the strategy with the highest probability of being cost-effective.

We quantify the expected loss from each strategy over a range of WTP thresholds with the expected loss curves (ELCs) to complement the PSA results (Figure 6C). The expected loss considers both the probability of making the wrong decision and the magnitude of the loss due to this decision, representing the foregone benefits of choosing a suboptimal strategy. The SoC Strategy has the lowest expected loss for WTP thresholds less than $65,000 per QALY, Strategy B has the lowest expected loss for WTP thresholds greater than or equal to $70,000 and less than $105,000. Strategy AB has the lowest expected loss for WTP thresholds greater than or equal $105,000 per QALY. At a WTP threshold of $100,000 per QALY, the EVPI is highest at $7,836 (Figures 6C-D). For a more detailed description of these outputs and the R code to generate them, we refer the reader to a previous publication by our group.[22]

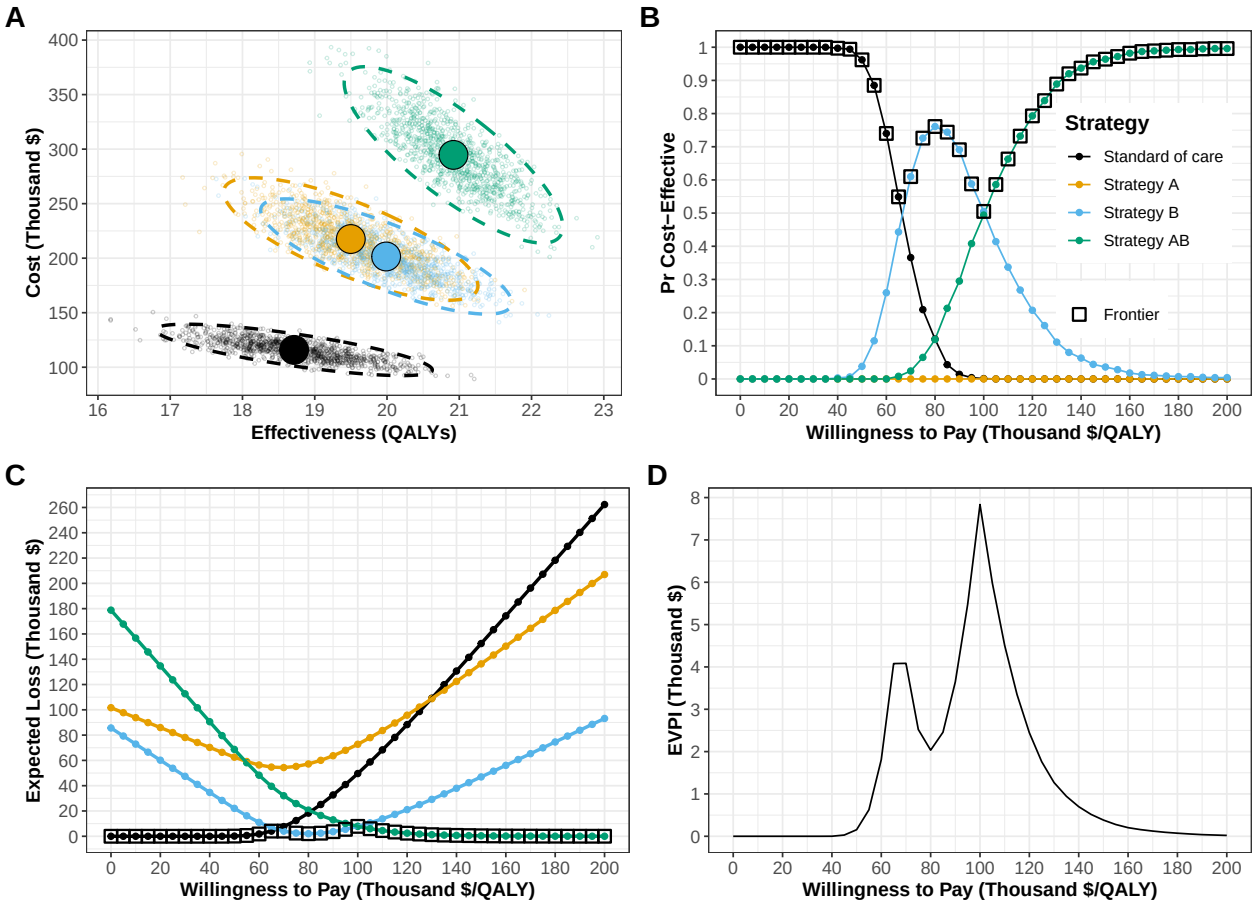


Figure 6: Figures generated from the probabilistic sensitivity analysis (PSA) output. A) Cost-effectiveness scatter plot. B) Cost-effectiveness acceptability curves (CEACs) and frontier (CEAF). C) Expected loss curves (ELCs). D) Expected value of perfect information (EVPI).

8 Epidemiological outcomes

cSTMs can be used to generate epidemiological outcomes that can be of direct interest to the decision maker or be helpful for model calibration and validation. Some common epidemiological outcomes include survival, prevalence, incidence, the average number of events, and lifetime risk of events.[23] We provide the epidemiological definitions of some of these outcomes and how they can be generated from the trace and transition probability objects of a cSTM using the simulation time-dependent Sick-Sicker cSTM under SoC. Similar to the PSA of the economic outcomes, we generated the epidemiological outcomes for each of the parameter sets drawn from their distributions and applied the equations described below. We summarized the outcomes with the 95% posterior predicted interval (PI), defined as the estimated range between the 2.5th and 97.5th percentiles of the model-predicted posterior outputs. The Supplemental Material provides the code to generate these outcomes from the state-residence dependent cSTM.

8.1 Survival probability

The survival probability, $S(t)$, captures the proportion of the cohort remaining alive by cycle t. To estimate $S(t)$ from the simulated cohort of the simulation-time-dependent Sick-Sicker model, shown in Figure 8.3A, we sum the proportions of the non-death states for all n_T cycles in `m_M_SoC`.

```
v_S_SoC <- rowSums(m_M_SoC[, -which(v_names_states == "D")]) # vector with survival curve
```

8.2 Life expectancy

Life expectancy (LE) refers to the expected number of time units remaining to be alive.[24] In continuous-time, LE is the area under the entire survival curve.[25]

$$LE = \int_{t=0}^{\infty} S(t)dt.$$

In discrete-time using cSTMs, we often calculate restricted LE over a fixed time horizon (e.g., n_T) at which most of the cohort has transitioned to the Dead state and is defined as

$$LE = \sum_{t=0}^{n_T} S(t).$$

In the simulation-time dependent Sick-Sicker model, where we simulated a cohort over n_T= 75 cycles, restricted life expectancy `le_SoC` is 40.8 (95% credible interval:[38.4, 43.6]) cycles (Figure 8.3B), which is calculated as

```
le_SoC <- sum(v_S_SoC) # life expectancy
```

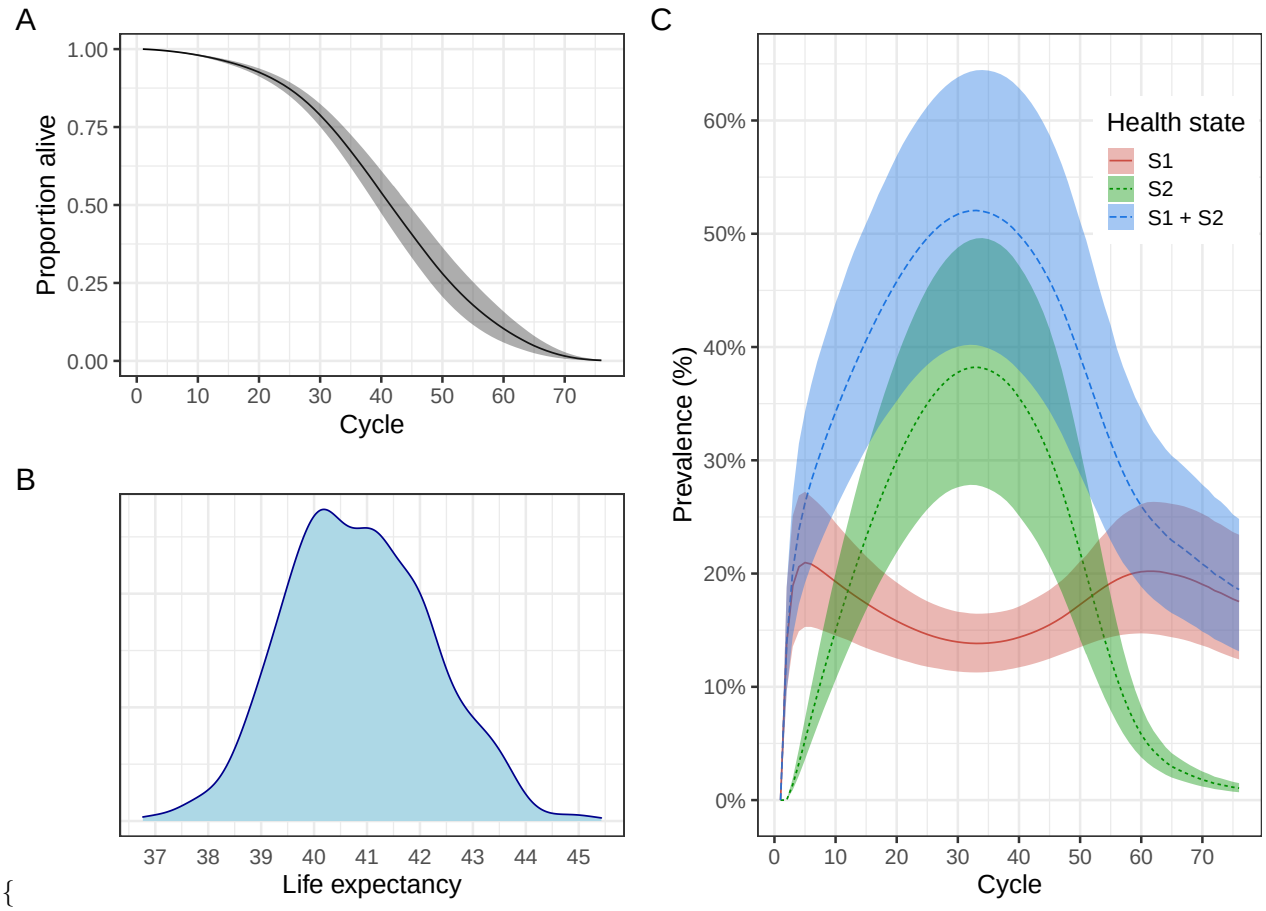
Note that this equation expresses LE in the units of t. We use an annual cycle length; thus, the resulting LE will be in years. Analysts can also use other cycle lengths (e.g., monthly or daily), but the LE must be correctly converted to the desired unit if different than the cycle length units.

8.3 Prevalence

Prevalence is defined as the proportion of the population or cohort with a specific condition (or being in a particular health state) among those alive.[26] To calculate the prevalence of S1 at cycle t, prev$(t)_i$, we compute the ratio between the proportion of the cohort in S1 and the proportion alive at that cycle.[27] The proportion of the cohort alive is given by the survival probability $S(t)$ defined above. The individual prevalence of the S1 and S2 health states and the overall prevalence of sick individuals (i.e., S1 + S2) of the age-dependent Sick-Sicker cSTM at each cycle t is computed as follows and are shown in Figure 8.3C.

```
v_prev_S1_SoC   <- m_M_SoC[, "S1"] / v_S_SoC          # vector with prevalence of Sick
v_prev_S2_SoC   <- m_M_SoC[, "S2"] / v_S_SoC          # vector with prevalence of Sicker
v_prev_S1S2_SoC <- rowSums(m_M_SoC[, c("S1", "S2")])/v_S_SoC # prevalence of Sick and Sicker
```

\begin{figure}[H]



{

}

\caption{Epidemiological outcomes generated from the probabilistic sensitivity analysis (PSA) output for the simulation-time dependent cSTM. A) Survival curve. B) Density of life expectancy. C) Prevalence of sick states over time. The shaded area shows the 95% posterior model-predictive interval of the outcomes and colored lines shows the posterior model-predicted mean based on 1,000 simulations} \end{figure}

9 Discussion

In this tutorial, we conceptualize time-dependent cSTMs with their mathematical description and a walk-through of their implementation for a CEA in R using the Sick-Sicker example. We described two types of time dependence: on the time since the start of the simulation (simulation-time dependence) or on the time spent in a health state (state-residence time dependence). We illustrated how to generate various epidemiological outcomes from the model, incorporate transition rewards in CEAs, and conduct a PSA.

We implemented simulation-time dependence by expanding the two-dimensional transition probability matrix into a three-dimensional transition probability array, where the third dimension captures time since the start of the simulation. However, there are alternative implementations of simulation-time dependence in cSTMs. For example, the model could be coded such that the time-varying elements of the transition probability matrix P_t are updated at each time point t as the simulation is run. This would eliminate the need for the transition probability array `a_P`, reducing computer memory requirements. But this comes at the expense of

increasing the number of operations at every cycle, potentially slowing down the simulation.

We incorporated state-residence time dependence using tunnel states by expanding the corresponding health states on the first and second dimensions of the 3-dimensional array to account for time spent in the current state in addition to simulation-time dependence. Another approach to account for state-residence time dependence is to use a three-dimensional transition probability array with dimensions for the current state, future state, and time in the current state.[28] However, combining simulation-time and state-residence time dependencies could necessitate a four-dimensional array, which may prove challenging.

Any time-varying feature in a discrete-time model can most generally be implemented as tunnel states with, at the extreme, every state having a different tunnel state for each time step. The cohort would then progressively move through these tunnel states to capture their progression through time and the model features (e.g., transition probabilities, costs, or utilities) that change over time. Using time-varying transition probabilities is a shortcut when the cohort experiences these time-varying processes simultaneously as a function of the time from the simulation start. Even if the time-varying process has a different periodicity than the cycle length, either tunnel states or time-varying transition probabilities can be used to capture these effects. However, this time-varying process must be represented or approximated over an integer number of cycle lengths.

As described in the introductory tutorial, cSTMs are recommended when the number of states is considered "not too large."[23] Incorporating time dependence in cSTMs using the provided approaches requires expanding the number of states by creating a multidimensional array for simulation-time dependence and/or creating tunnel states for state-residence time dependence, increasing the amount of computer memory (RAM) required. For example, as state-residence time dependence extends in more health states, the total number of health states will increase, causing a "state explosion". It is possible to build reasonably complex time-dependent cSTMs in R as long as there is sufficient RAM to store the transition probability array and outputs of interest. For example, a typical PC with 8GB of RAM can handle a transition probability array of about 1,000 states and 600 time-cycle slices. However, if a high degree of granularity is desired, the dimensions of these data objects can grow quickly; if the required number of states gets too large and difficult to code, it may be preferable to use a stochastic (Monte Carlo) version of the state-transition model – often called individual-based state-transition models (iSTM) or microsimulation models – rather than a cohort simulation model.[23] In an iSTM, the risks and rewards of simulated individuals need not depend only on a current health state; they may also depend on an individual's characteristics and attributes. In addition, modelers can store health state history and other events over time for each individual to determine the risk of new events and corresponding costs and effects. Thus, we recommend carefully considering the required model structure before implementing it. An iSTM will require additional functions to describe the dependence of transition probabilities and rewards on individuals' histories. A previous tutorial showed how to construct these functions for an iSTM using the Sick-Sicker example.[29]

We described the concept and implementation of transition rewards. While these event-driven impacts could instead be captured by expanding the model state-space to include an initial disease state or a pre-death, end-of-life state, we can avoid state-space expansion by calculating incurred rewards based on the proportion of the cohort making the relevant transition. For implementation, this requires storing not just the cohort trace, which reflects how many individuals are in each state at a given cycle, but also a cohort state-transition array, which records how many individuals are making each possible transition at a given cycle.[9] We model the effectiveness of Treatment B as a hazard ratio. However, other effect measures can be

used, such as odds ratio, risk ratio, or risk differences depending on the methods used to estimate them.[30–32] We assume that all parameters are known for the case study. However, in a real-world CEA, parameters are estimated from electronic health records,[33] clinical trials, and observational studies, and when there is no individual-level data to inform them, through model calibration.[34]

We use `base` R to implement time-dependent cSTMs as the most transparent way to illustrate how to translate the mathematical specification of cSTMs into code. Using a `base` R implementation maximizes flexibility and customizability to model any disease that can be specified as a cSTM. However, there are R packages for building and running cSTMs and conducting CEAs, such as `heemod`[35] and `hesim`,[36] that are made such that they can support modeling arbitrary patient populations or many treatment strategies. Since much of the programming burden has already been done, these packages can reduce programming time. Users don't need to write functions themselves but instead need to call a series of prewritten functions. And in the case of `hesim`, all the memory pre-allocation and loops are made at the `C++` level, making it more efficient than looping over `base` R but might require more RAM. Ultimately, it is up to the analyst to decide the most appropriate approach to implementing their cSTM-based CEAs.

This tutorial extends our conceptualization of time-independent cSTMs to allow for time dependence. It provides a step-by-step guide to implementing time-dependent cSTMs in R to generate epidemiological and economic outcomes, accounts for transition rewards, and conducts a CEA and a corresponding PSA. We hope that health decision scientists and health economists find this tutorial helpful in developing their cSTMs in a more flexible, efficient, and open-source manner. Ultimately, our goal is to facilitate R in health economic evaluations with the overall aim to increase model transparency and reproducibility.

10 Acknowledgements

Dr. Alarid-Escudero was supported by grants U01-CA199335 and U01-CA253913 from the National Cancer Institute (NCI) as part of the Cancer Intervention and Surveillance Modeling Network (CISNET), and a grant by the Gordon and Betty Moore Foundation. Miss Krijkamp was supported by the Society for Medical Decision Making (SMDM) fellowship through a grant by the Gordon and Betty Moore Foundation (GBMF7853). Dr. Enns was supported by a grant from the National Institute of Allergy and Infectious Diseases of the National Institutes of Health under award no. K25AI118476. Dr. Hunink received research funding from the American Diabetes Association, the Netherlands Organization for Health Research and Development, the German Innovation Fund, Netherlands Educational Grant ("Studie Voorschot Middelen"), and the Gordon and Betty Moore Foundation. Dr. Jalal was supported by a grant from the National Institute on Drug Abuse of the National Institute of Health under award no. K01DA048985. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. The funding agencies had no role in the design of the study, interpretation of results, or writing of the manuscript. The funding agreement ensured the authors' independence in designing the study, interpreting the data, writing, and publishing the report. We also want to thank the anonymous reviewers of *Medical Decision Making* for their valuable suggestions and the students who took our classes where we refined these materials.

References

1.
Suijkerbuijk AWM, Van Hoek AJ, Koopsen J, De Man RA, Mangen MJJ, De Melker HE, et al. Cost-effectiveness of screening for chronic hepatitis B and C among migrant populations in a low endemic country. PLoS ONE. 2018;13(11):1–6.

2.
Sathianathen NJ, Konety BR, Alarid-Escudero F, Lawrentschuk N, Bolton DM, Kuntz KM. Cost-effectiveness Analysis of Active Surveillance Strategies for Men with Low-risk Prostate Cancer. European Urology [Internet]. 2019;75(6):910–7. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0302283818308534

3.
Lu S, Yu Y, Fu S, Ren H. Cost-effectiveness of ALK testing and first-line crizotinib therapy for non-small-cell lung cancer in China. PLoS ONE. 2018;13(10):1–2.

4.
Djatche LM, Varga S, Lieberthal RD. Cost-Effectiveness of Aspirin Adherence for Secondary Prevention of Cardiovascular Events. PharmacoEconomics - Open [Internet]. 2018;2(4):371–80. Available from: https://doi.org/10.1007/s41669-018-0075-2

5.
Pershing S, Enns EA, Matesic B, Owens DK, Goldhaber-Fiebert JD. Cost-Effectiveness of Treatment of Diabetic Macular Edema. Annals of Internal Medicine. 2014;160(1):18–29.

6.
Smith-Spangler CM, Juusola JL, Enns EA, Owens DK, Garber AM. Population Strategies to Decrease Sodium Intake and the Burden of Cardiovascular Disease: A Cost-Effectiveness Analysis. Annals of Internal Medicine [Internet]. 2010;152(8):481–7. Available from: http://annals.org/article.aspx?articleid=745729

7.
Alarid-Escudero F, Krijkamp EM, Enns EA, Yang A, Hunink MGM, Pechlivanoglou P, et al. An Introductory Tutorial on Cohort State-Transition Models in R Using a Cost-Effectiveness Analysis Example. Medical Decision Making. 2022;In Press.

8.
Snowsill T. A New Method for Model-Based Health Economic Evaluation Utilizing and Extending Moment-Generating Functions. Medical Decision Making [Internet]. 2019;39(5):523–39. Available from: http://journals.sagepub.com/doi/10.1177/0272989X19860119

9.
Krijkamp EM, Alarid-Escudero F, Enns E, Pechlivanoglou P, Hunink MM, Jalal H. A Multidimensional Array Representation of State-Transition Model Dynamics. Medical Decision Making. 2019;In Press.

10.

Jalal H, Pechlivanoglou P, Krijkamp E, Alarid-Escudero F, Enns EA, Hunink MGM. An Overview of R in Health Decision Sciences. Medical Decision Making [Internet]. 2017;37(7):735–46. Available from: http://journals.sagepub.com/doi/10.1177/0272989X16686559

11.

Hunink MGGM, Weinstein MC, Wittenberg E, Drummond MF, Pliskin JS, Wong JB, et al. Decision Making in Health and Medicine [Internet]. 2nd ed. Cambridge: Cambridge University Press; 2014. Available from: http://ebooks.cambridge.org/ref/id/CBO9781139506779

12.

Enns EA, Cipriano LE, Simons CT, Kong CY. Identifying Best-Fitting Inputs in Health-Economic Model Calibration: A Pareto Frontier Approach. Medical Decision Making [Internet]. 2015;35(2):170–82. Available from: http://www.ncbi.nlm.nih.gov/pubmed/24799456

13.

O'Mahony JF, Newall AT, Rosmalen J van. Dealing with Time in Health Economic Evaluation: Methodological Issues and Recommendations for Practice. PharmacoEconomics [Internet]. 2015;33(12):1265–8. Available from: http://www.ncbi.nlm.nih.gov/pubmed/26105525

14.

Alarid-Escudero F, Krijkamp E, Pechlivanoglou P, Jalal H, Kao S-YZ, Yang A, et al. A Need for Change! A Coding Framework for Improving Transparency in Decision Modeling. PharmacoEconomics [Internet]. 2019;37(11):1329–39. Available from: https://doi.org/10.1007/s40273-019-00837-x

15.

Arias E, Heron M, Xu J. United States Life Tables, 2014. National Vital Statistics Reports [Internet]. 2017;66(4):63. Available from: https://www.cdc.gov/nchs/data/nvsr/nvsr66/nvsr66%7B/_%7D04.pdf

16.

Diaby V, Adunlin G, Montero AJ. Survival modeling for the estimation of transition probabilities in model-based economic evaluations in the absence of individual patient data: A tutorial. PharmacoEconomics. 2014 Feb;32(2):101–8.

17.

Elbasha EH, Chhatwal J. Theoretical foundations and practical applications of within-cycle correction methods. Medical Decision Making. 2016;36(1):115–31.

18.

Elbasha EH, Chhatwal J. Myths and misconceptions of within-cycle correction: a guide for modelers and decision makers. PharmacoEconomics. 2016;34(1):13–22.

19.

Alarid-Escudero F, Knowlton G, Easterly CA, Enns EA. Decision analytic modeling package (dampack) [Internet]. 2021. Available from: https://cran.r-project.org/web/packages/dampack/%20https://github.com/DARTH-git/dampack

20.

Briggs AH, Weinstein MC, Fenwick EAL, Karnon J, Sculpher MJ, Paltiel AD. Model Parameter Estimation and Uncertainty Analysis: A Report of the ISPOR-SMDM Modeling Good Research Practices Task Force Working Group-6. Medical Decision Making. 2012 Sep;32(5):722–32.

21.

Briggs AH, Goeree R, Blackhouse G, O'Brien BJ. Probabilistic Analysis of Cost-Effectiveness Models: Choosing between Treatment Strategies for Gastroesophageal Reflux Disease. Medical Decision Making [Internet]. 2002 Jul;22(4):290–308. Available from: http://mdm.sagepub.com/cgi/doi/10.1177/027298902400448867

22.

Alarid-Escudero F, Enns EA, Kuntz KM, Michaud TL, Jalal H. "Time Traveling Is Just Too Dangerous" But Some Methods Are Worth Revisiting: The Advantages of Expected Loss Curves Over Cost-Effectiveness Acceptability Curves and Frontier. Value in Health. 2019;22(5):611–8.

23.

Siebert U, Alagoz O, Bayoumi AM, Jahn B, Owens DK, Cohen DJ, et al. State-Transition Modeling: A Report of the ISPOR-SMDM Modeling Good Research Practices Task Force-3. Medical Decision Making [Internet]. 2012;32(5):690–700. Available from: http://mdm.sagepub.com/cgi/doi/10.1177/0272989X12455463

24.

Lee ET, Wang JW. Statistical methods for Survival Data Analysis. 3rd ed. Hoboken, NJ: Wiley; 2003.

25.

Klein JP, Moeschberger ML. Survival Analysis: Techniques for Censored and Truncated Data [Internet]. 2nd ed. Springer-Verlag; 2003. Available from: http://www.springer.com/statistics/life+sciences,+medicine+%7B/&%7D+health/book/978-0-387-95399-1

26.

Rothman KJ, Greenland S, Lash TL. Modern Epidemiology. 3rd ed. Rothman KJ, Greenland S, Lash TL, editors. Lippincott Williams & Wilkins; 2008.

27.

Keiding N. Age-Specific Incidence and Prevalence: A Statistical Perspective. Journal of the Royal Statistical Society Series A (Statistics in Society). 1991;154(3):371–412.

28.

Hawkins N, Sculpher M, Epstein D. Cost-effectiveness analysis of treatments for chronic disease: Using R to incorporate time dependency of treatment response. Medical Decision Making [Internet]. 2005;25(5):511–9. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16160207

29.

Krijkamp EM, Alarid-Escudero F, Enns EA, Jalal HJ, Hunink MGM, Pechlivanoglou P. Microsimulation Modeling for Health Decision Sciences Using R: A Tutorial. Medical Decision Making [Internet]. 2018 Apr;38(3):400–22. Available from: http://journals.sagepub.com/doi/10.1177/0272989X18754513

30.

Kuntz KM, Weinstein MC. Modelling in economic evaluation. In: Drummond MF, McGuire A, editors. Economic evaluation in health care: Merging theory with practice. 2nd ed. New York, NY: Oxford University Press; 2001. p. 141–71.

31.

Ades AE, Lu G, Claxton K. Expected value of sample information calculations in medical decision modeling. Medical Decision Making [Internet]. 2004;24(2):207–27. Available from: http://www.ncbi.nlm.nih.gov/pubmed/15090106

32.

Gidwani R, Russell LB. Estimating Transition Probabilities from Published Evidence: A Tutorial for Decision Modelers. PharmacoEconomics [Internet]. 2020;38(11):1153–64. Available from: https://doi.org/10.1007/s40273-020-00937-z

33.

Rodriguez PJ, Ward ZJ, Long MW, Austin SB, Wright DR. Applied Methods for Estimating Transition Probabilities from Electronic Health Record Data. Medical Decision Making [Internet]. 2021;41(2):143–52. Available from: https://journals.sagepub.com/doi/abs/10.1177/0272989X20985752?casa_token=SIhw-ApH8ogAAAAA:lxalsZ3tBihW8nVpK1WhSjkuVNR3Z7kKUe44gC5UJk4Rs8KfYoAQ6Y9efV8oVvR1GeLeTDwWdRXKtWk

34.

Welton NJ, Ades AE. Estimation of markov chain transition probabilities and rates from fully and partially observed data: uncertainty propagation, evidence synthesis, and model calibration. Medical Decision Making [Internet]. 2005;25(6):633–45. Available from: http://www.ncbi.nlm.nih.gov/pubmed/16282214

35.

Filipović-Pierucci A, Zarca K, Durand-Zaleski I. Markov Models for Health Economic Evaluation: The R Package heemod. arXiv:170203252v1 [Internet]. 2017;April:30. Available from: http://arxiv.org/abs/1702.03252

36.

Incerti D, Jansen JP. hesim: Health Economic Simulation Modeling and Decision Analysis. arXiv:210209437v2 [Internet]. 2021 Feb;March:38. Available from: http://arxiv.org/abs/2102.09437