An Introductory Tutorial to Cohort State-Transition Models in R

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[**]

2021-06-15

Abstract

Decision models can synthesize evidence from different sources to simulate the long-term consequences of alternative strategies in the presence of uncertainty. Cohort state-transition models (cSTM) are decision models commonly used in medical decision-making to simulate hypothetical cohorts' transitions among various health states over time. cSTMs can be time-independent, where transition probabilities are constant over time, or time-dependent, where transition probabilities change over time. In this tutorial, we implement time-independent cSTM in R, an open-source mathematical and statistical programming language. This is the first part of a two-part tutorial on implementing cSTMs in R, where the focus of the accompanying tutorial is on time-dependent cSTM. This tutorial aims to construct time-independent cSTMs using a previously published decision model, calculate costs and effectiveness outcomes, conduct a cost-effectiveness analysis of multiple strategies, and conduct a probabilistic sensitivity analysis. We provide open-source code in R to facilitate wider adoption.

[*]Division of Public Administration, Center for Research and Teaching in Economics (CIDE), Aguascalientes, AGS, Mexico
[†]Department of Epidemiology and Department of Radiology, 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 Pittsburgh, Pittsburgh, PA, USA

1 Introduction

Policymakers are often tasked with allocating healthcare resources under constrained budgets and uncertainty about future outcomes. Health economic evaluations might inform their final decisions. These economic evaluations often rely on decision models to synthesize evidence from different sources and project long-term outcomes of various alternative strategies. A commonly used decision model is the discrete-time cohort state-transition model (cSTM), often referred to as a Markov model.[1]

In a recent review, we illustrated the increased use of R's statistical programming framework in health decision sciences. We provided a summary of available resources to apply to medical decision making.[2] Many packages have been explicitly developed to estimate and construct cSTMs in R. For example, the `markovchain`[3] and `heemod`[4] packages are designed to build cSTMs using a pre-defined structure. The `markovchain` package focuses on simulating time-independent and time-dependent Markov chains but is not designed to conduct economic evaluations. `heemod` is a well-structured R package for economic evaluations. Still, it requires users to set up the cSTM structure, specify the parameters, and run analyses in a pre-specified approach, limiting the understanding of how cSTMs work and are constructed. However, these packages are necessarily stylized and inflexible, requiring users to follow a specific cSTM structure. If the desired cSTM does not fit within this structure, using these packages can be challenging.

This tutorial demonstrates how to conduct a full cost-effectiveness analysis (CEA) comparing multiple interventions and implementing probabilistic sensitivity analysis (PSA). We first describe each of the components of a time-independent cSTM. Then, we illustrate the implementation of these components with an example. Our general conceptualization should apply to other programming languages (e.g., MATLAB, Python, C++, Julia). The reader can find the most up-to-date R code of the time-independent cSTM and the R code to create the tutorial graphs in the accompanying GitHub repository (https://github.com/DARTH-git/cohort-modeling-tutorial-intro) to replicate and modify the example to fit their needs. We assume that the reader is familiar with the basics of decision modeling and has a basic understanding of programming. Thus, a prior introduction to R and linear algebra for decision modelers is recommended.

This introductory tutorial aims to (1) conceptualize time-independent cSTMs for implementation in a programming language and (2) provide a template for implementing these cSTMs in *base* R. We focus on using R *base* packages, ensuring modelers understand the concept and structure of cSTMs and avoid the limitation of constructing cSTMs in a package-specific structure. We used previously developed R packages for visualizing CEA results and checking cSTMs are correctly specified.

2 Cohort state-transition models (cSTMs)

A cSTM is a dynamic mathematical model in which a hypothetical cohort of individuals transition between different health states over time. In contrast, an individual-based state-transition model (iSTM) is a type of STM where simulated individuals transition between health states over time.[5] We have previously published a tutorial on the implementation of iSTM in R.[6]

A cSTM is most appropriate when the decision problem has a dynamic component (e.g., the disease process can vary over time) and can be described using a reasonable number of health states. cSTMs are often used because of their transparency, efficiency, ease of development, and debugging. cSTMs are usually computationally less demanding than iSTMs, providing the ability to conduct PSA and value-of-information (VOI) analyses that otherwise might not be computationally feasible with iSTMs.[5] cSTMs have been used to evaluate screening and surveillance programs,[7,8] diagnostic procedures,[9] disease management programs,[10], interventions,[11] and policies.[12]

A cSTM consists of a set of n_S mutually exclusive and collectively exhaustive health states. The cohort is assumed to be homogeneous within each health state. Individuals in the cohort residing in a particular health state are assumed to have the same characteristics and are indistinguishable from one another. The cohort transitions between health states with defined probabilities, which are called "transition probabilities". A transition probability represents the chance that individuals in the cohort residing in a state in a given cycle transition to another state or remain in the same state. In cSTMs, a one-cycle transition probability reflects a conditional probability of transitioning during the cycle, given that the person is alive at the beginning of the cycle.[13] Transition probabilities only depend on the current health state in a given cycle. They do not depend on the history before that cycle, often referred to as the "Markovian assumption."[14–16] This means that in a Markovian cSTM, transition probabilities do not depend on the history of past transitions or time spent in a given state.

cSTMs can be classified as either time-independent (time-homogeneous) or time-dependent (time-inhomogeneous). Time-independent cSTMs have constant transition probabilities (i.e., the probability of any state transition is independent of time). In contrast, time-dependent cSTMs have transition probabilities or rewards (e.g., costs or utilities associated with being in a particular health state) that vary over time. A reward refers to a value assigned to individuals for being in a given health state (state reward) or transitioning between health states (transition reward). Time-independent models are more straightforward to implement than time-dependent ones, but most problems in healthcare are best modeled with time-dependent cSTMs. For example, time-dependent cSTMs can capture the increasing age-specific background mortality as the cohort ages (age dependency) and dependency on the amount of time spent in a given state (state residence dependence). However, it is easiest to start implementing the structure of a time-independent model and expand it to account for various types of time dependency. Therefore, this tutorial focuses on time-independent cSTMs. In the second part tutorial, we describe how to construct time-dependent cSTMs.[17]

2.1 Rates versus probabilities

Probabilities and rates are commonly used input parameters in cSTMs. While they are often numerically similar in practice, there is a subtle but important conceptual difference. A rate represents the *instantaneous* force of an event occurrence per unit time, while a probability represents the cumulative risk of an event over a defined time period.

To illustrate this difference further, let us assume that after 10,000 person-years of observation of healthy individuals (e.g., 10,000 individuals observed for an average of 1 year, or 5,000 individuals observed for an average of 2 years, etc.), we observe 500 events of interest (e.g., becoming sick from some disease). The annual event rate of becoming sick, μ_{yearly}, is then equal to $\mu_{yearly} = 500/10,000 = 0.05$.

If we then wanted to know what proportion of an initially healthy cohort becomes sick at the end of the year, we can convert the annual rate of becoming sick into an annual probability of becoming sick using the following equation:

$$p_{yearly} = 1 - \exp(-\mu_{yearly}), \tag{1}$$

resulting in $p_{yearly} = 1 - \exp(-0.05) = 0.0488$. Equation (1) assumes that the rate of becoming sick is constant over the year, implying that the time until a healthy person becomes sick is exponentially distributed. The parameter p_{yearly} is the transition probability from healthy to sick in a cSTM when using an annual cycle length.

If we were concerned that an annual cycle length was too long to capture disease dynamics accurately, we could use a monthly cycle length. This would require us to parameterize the model with monthly transition probabilities. To convert the annual probability of becoming sick to a monthly probability, we first calculate the monthly rate of becoming sick:

$$\mu_{yearly} = -\ln(1 - p_{yearly}). \tag{2}$$

Because rates are instantaneous, the monthly rate is then just the annual rate divided by 12:

$$\mu_{monthly} = \mu_{yearly}/12. \tag{3}$$

We divide by 12 because this is the number of months (desired cycle length) in a year (cycle length of the given data). If the original or desired cycle length were different, we would divide by a different factor (e.g., annual to weekly: 52; annual to daily: 365.25, etc.).

The monthly probability can then be calculated from the monthly rate using equation (1):

$$p_{monthly} = 1 - \exp(-\mu_{monthly}). \tag{4}$$

Sometimes, transition probabilities might be available from published literature in one time period (e.g., annual) and might differ from the model's cycle length time scale (e.g., monthly). This discrepancy will require the analyst to convert the transition probability to the desired cycle length using the equations described above. Furthermore, the conversion may differ for different parameters if they come from various sources and cycle lengths.

3 Time-independent cSTM dynamics

A cSTM consists of three core components: (1) a state vector, $\mathbf{m}_t$, that stores the distribution of the cohort across all health states in cycle t where $t = 0, \dots, n_T$; (2) the cohort trace matrix, M, that stacks $\mathbf{m}_t$ for all t and represents the distribution of the cohort in the various states over time; and (3) a transition probability matrix, P.[18] If the cSTM is comprised of n_S discrete health states, $\mathbf{m}_t$ is a $1 \times n_S$ vector and P is a $n_S \times n_S$

matrix. The i-th element of $\mathbf{m}_t$, where $i = 1, \ldots, n_S$, represents the proportion of the cohort in the i-th health state in cycle t, referred to as $m_{[t,i]}$. Thus, $\mathbf{m}_t$ is written as:

$$\mathbf{m}_t = \begin{bmatrix} m_{[t,1]} & m_{[t,2]} & \cdots & m_{[t,n_S]} \end{bmatrix}.$$

The elements of P are the transition probabilities of moving from state i to state j, $p_{[i,j]}$, where $\{i, j\} = 1, \ldots, n_S$ and all should have values between 0 and 1.

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

For P to be a correctly specified transition probability matrix, all rows of the transition probability matrix must sum to one, $\sum_{j=1}^{n_S} p_{[i,j]} = 1$ for all $i = 1, \ldots, n_S$.

The state vector at cycle $t + 1$ ($\mathbf{m}_{t+1}$) is then calculated as the matrix product of the state vector at cycle t, $\mathbf{m}_t$, and the transition probability matrix, P, such that

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

where $\mathbf{m}_1$ is computed from $\mathbf{m}_0$, the initial state vector with the distribution of the cohort across all health states at the start of the simulation (cycle 0). Then, we iteratively apply this equation through $t = (n_T - 1)$.

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

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

Note that the initial cycle (i.e., cycle 0) corresponds to $t = 0$, which is on the first row of M. Thus, M stores the output of the cSTM, which could be used to compute various epidemiological, and economic outcomes, such as life expectancy, prevalence, cumulative resource use, and costs, etc. Table 1 describes the elements related to the core components of cSTM and their suggested R code names. For a more detailed description of the variable types, data structure, R name for all cSTM elements, please see the Supplementary Material.

Table 1: Components of a cSTM with their R name.

Element	Description	R name
n_S	Number of states	n_states
$\mathbf{m}_0$	Initial state vector	v_m_init
$\mathbf{m}_t$	State vector in cycle t	v_mt
M	Cohort trace matrix	m_M

Element	Description	R name
P	Time-independent transition probability matrix	m_P

4 Case study: Sick-Sicker model

In this tutorial, we will use the previously published 4-state "Sick-Sicker" model for conducting a CEA of multiple strategies to illustrate the various aspects of cSTM implementation in R.[6,19] Figure 1 represents the state-transition diagram of the Sick-Sicker model.
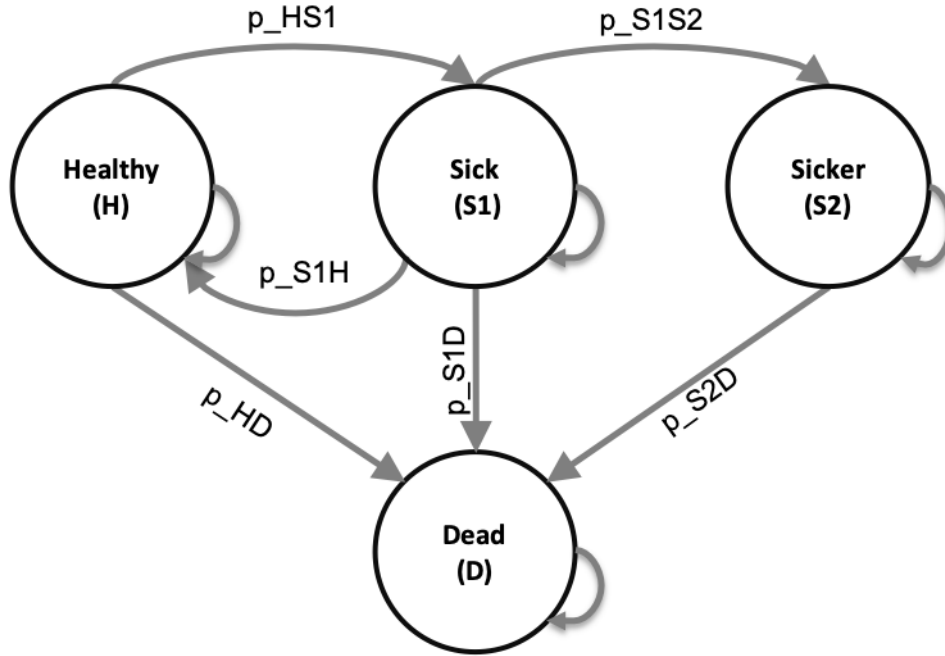

Figure 1: State-transition diagram of the time-independent Sick-Sicker cohort state-transition model, showing all possible states (labeled with state names) and transitions (labeled with transition probability variable names).

The model simulates a cohort at risk of a hypothetical disease with two stages, "Sick" and "Sicker", to compute the expected costs and quality-adjusted life years (QALYs) of the cohort over time. All the parameters of the Sick-Sicker model and the corresponding R variable names are presented in Table 2. The naming of these parameters and variables follows the notation described in the DARTH coding framework.[20] Briefly, we define variables by <x>_<y>_<var_name>, where x is the prefix that indicates the data type (e.g., scalar (no prefix), v for vector, m for matrix, a for array, df for data frame, etc.), y is the prefix indicating variable type (e.g., p for probability, r for rate, hr for hazard ratio, lor for log-odds ratio, c for cost c, u for utility, etc.), and var_name is some description of the variable presented separated by underscores. For example, v_p_HD denotes the vector of transition probabilities from health state "H" to health state "D". In later sections we will define and name all the other parameters.

In the Sick-Sicker model, we simulate a hypothetical cohort of 25-year-old individuals over their lifetime

(until to a maximum age of 100 years) who all start in the "Healthy" state (denoted "H"). We will simulate the cohort dynamics in annual cycle lengths, requiring a total of 75 one-year cycles. The total number of cycles is denoted as n_T and defined in R as `n_cycles`. Healthy individuals are at risk of developing the disease when they transition to the "Sick" state (denoted by "S1"). Sick individuals are at risk of further progressing to a more severe disease stage, the "Sicker" health state (denoted by "S2"). Individuals in S1 can recover and return to H. However, once individuals reach S2, they cannot recover; that is, the probability of transitioning to S1 or H from S2 is zero. Individuals in H face constant background mortality. 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 mortality rate. Individuals in S1 and S2 also experience increased health care costs, and reduced quality of life (QoL) compared to individuals in H. When individuals die, they transition to the absorbing "Dead" state (denoted by "D"), where they remain. All transitions between non-death states are assumed to be conditional on surviving each cycle. We discount both costs and QALYs at an annual rate of 3%.

We are interested in evaluating the cost-effectiveness of four strategies: the standard of care (strategy SoC), strategy A, strategy B, and a combination of strategies A and B (strategy AB). Strategy A involves administering treatment A that increases the QoL of individuals in S1 from 0.75 (utility without treatment, `u_S1`) to 0.95 (utility with treatment A, `u_trtA`) and costs $12,000 per year (`c_trtA`).[6] This strategy does not impact the QoL of individuals in S2, nor does it change the risk of becoming sick or progressing through the sick states. Strategy B uses treatment B to reduce only the rate of Sick individuals progressing to the Sicker state with a hazard ratio (HR) of 0.6 (`hr_S1S2_trtB`), costs $13,000 per year (`c_trtB`), and does not affect QoL. Strategy AB involves administering both treatments A and B.

We assume that it is not possible to distinguish between Sick and Sicker patients; therefore, for treatment strategies, individuals in both disease states receive the treatment. After comparing the four strategies in terms of expected QALYs and costs, we calculate the incremental cost per QALY gained between non-dominated strategies.

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

Parameter	R name	Base-case	Distribution
Number of cycles (n_cycles)	`n_cycles`	75 years	-
Names of health states (n)	`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 probabilities conditional on surviving			
- Disease onset (H to S1)	`p_HS1`	0.150	beta(30, 170)
- Recovery (S1 to H)	`p_S1H`	0.500	beta(60, 60)
- Disease progression (S1 to S2)	`p_S1S2`	0.105	beta(84, 716)
Annual mortality			
- Background mortality rate (H to D)	`r_HD`	0.002	-

Parameter	R name	Base-case	Distribution
- 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(576.0, 20.8)
- 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(676.0, 19.2)
- Reduction in rate of disease progression (S1 to S2) as hazard ratio (HR)	`hr_S1S2_trtB`	log(0.6)	lognormal(log(0.6), 0.1)

The following sections include R code snippets. All R code is stored as a GitHub repository and can be accessed at https://github.com/DARTH-git/cohort-modeling-tutorial-intro. In the R code below, we initialize the input parameters by setting the variables to their base-case values. We do this process as the first coding step and all in one place, so when a parameter value changes, the updated value will carry through the rest of the code.

```
## General setup
cycle_length <- 1 # cycle length equal one year
n_age_init <- 25  # age at baseline
n_age_max  <- 100 # maximum age of follow up
n_cycles <- n_age_max - n_age_init # number of cycles
v_names_states <- c("H", "S1", "S2", "D") # the 4 health states of the model:
                            # Healthy (H), Sick (S1), Sicker (S2), Dead (D)
n_states <- length(v_names_states) # number of health states
d_e <- 0.03 # discount rate for QALYs of 3% per cycle
d_c <- 0.03 # discount rate for costs of 3% per cycle
v_names_str <- c("Standard of care", # store the strategy names
              "Strategy A",
```

```
                    "Strategy B",
                    "Strategy AB")

## Transition probabilities (per cycle), hazard ratios and odds ratio (OR)
r_HD    <- 0.002 # constant rate of dying when Healthy (all-cause mortality rate)
p_HS1   <- 0.15  # probability of becoming Sick when Healthy
p_S1H   <- 0.5   # probability of becoming Healthy when Sick
p_S1S2  <- 0.105 # probability of becoming Sicker when Sick
hr_S1   <- 3     # hazard ratio of death in Sick vs Healthy
hr_S2   <- 10    # hazard ratio of death in Sicker vs Healthy

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

## State rewards
## Costs
c_H    <- 2000  # cost of being Healthy for one cycle
c_S1   <- 4000  # cost of being Sick for one cycle
c_S2   <- 15000 # cost of being Sicker for one cycle
c_D    <- 0     # cost of being dead for one cycle
c_trtA <- 12000 # cost of receiving treatment A for one cycle
c_trtB <- 13000 # cost of receiving treatment B for one cycle
# Utilities
u_H    <- 1     # utility of being Healthy for one cycle
u_S1   <- 0.75  # utility of being Sick for one cycle
u_S2   <- 0.5   # utility of being Sicker for one cycle
u_D    <- 0     # utility of being dead for one cycle
u_trtA <- 0.95  # utility when receiving treatment A for one cycle
```

To compute the background mortality risk, `p_HD`, from the background mortality rate for the same cycle length (i.e., `cycle_length=1`), we apply Equation (1) to `r_HD`. To compute the mortality risks of the cohort in S1 and S2, we multiply the background mortality rate `r_HD` by the hazard ratios `hr_S1` and `hr_S2`, respectively, and then convert back to probabilities using Equation (1). These calculations are required because hazard ratios only apply to rates and not to probabilities. The code below performs the computation in R. In the `darthtools` package (https://github.com/DARTH-git/darthtools), we provide R functions that compute transformations between rates and probabilities since these transformations are frequently used.

```
## Mortality rates
r_S1D <- r_HD * hr_S1 # rate of dying when Sick
r_S2D <- r_HD * hr_S2 # rate of dying when Sicker
## Probabilities of dying
cycle_length <- 1
p_HD  <- 1 - exp(-r_HD*cycle_length)  # background mortality risk (i.e., probability)
p_S1D <- 1 - exp(-r_S1D*cycle_length) # probability of dying when Sick
```

```
p_S2D <- 1 - exp(-r_S2D*cycle_length) # probability of dying when Sicker
```

To compute the risk of progression from S1 to S2 under treatment B, we first transform `p_S1S2` to a rate, `r_S1S2`, using Equation (2). Then, we multiply the hazard ratio of treatment B by the rate of progressing from S1 to S2 and transform it back to probabilities by applying Equation (1).

```
## Transition probability of becoming Sicker when Sick for treatment B
# transform probability to rate
r_S1S2 <- -log(1 - p_S1S2)/cycle_length
# apply hazard ratio to rate to obtain transition rate of becoming Sicker when Sick
# for treatment B
r_S1S2_trtB <- r_S1S2 * hr_S1S2_trtB
# transform rate to probability
p_S1S2_trtB <- 1 - exp(-r_S1S2_trtB*cycle_length) # probability to become Sicker when Sick
                                                  # under treatment B conditional on surviving
```

For the Sick-Sicker model, the entire cohort starts in the H 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
```

The variable `v_m_init` is used to initialize M represented by `m_M` for the cohort under strategy SoC. We also create a trace for each of the other treatment-based strategies.

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

Note that the initial state vector, `v_m_init`, can be modified to account for the cohort's distribution across the states at the start of the simulation and could, but does not always, vary by strategy.

Since the Sick-Sicker model consists of 4 states, we create a 4×4 transition probability matrix for strategy SoC, `m_P`. We initialize the matrix with default values of zero for all transition probabilities and then populate it with the corresponding transition probabilities. To access an element of `m_P`, we specify first the row name (or number) and then the column name (or number) separated by a comma. For example, we could access the transition probability from state Healthy (H) to state Sick (S1) using the corresponding row or column state-names as characters `m_P["H", "S1"]`. We assume that all transitions to non-death states are conditional on not dying in a cycle. Thus, we first condition on surviving by multiplying the transition probabilities times `1 - p_HD`, the probability of not dying in a cycle. For example, to obtain the probability

of transitioning from H to S1, we multiply the transition probability from H to S1 conditional on being alive, `p_HS1` by `1 - p_HD`. We create the transition probability matrix for strategy A as a copy of SoC's because treatment A does not alter the cohort's transition probabilities.

```r
## Initialize transition probability matrix for strategy SoC
m_P <- matrix(0,
              nrow = n_states, ncol = n_states,
              dimnames = list(v_names_states, v_names_states)) # row and column names
## Fill in matrix
# From H
m_P["H", "H"]   <- (1 - p_HD) * (1 - p_HS1)
m_P["H", "S1"]  <- (1 - p_HD) * p_HS1
m_P["H", "D"]   <- p_HD
# From S1
m_P["S1", "H"]  <- (1 - p_S1D) * p_S1H
m_P["S1", "S1"] <- (1 - p_S1D) * (1 - (p_S1H + p_S1S2))
m_P["S1", "S2"] <- (1 - p_S1D) * p_S1S2
m_P["S1", "D"]  <- p_S1D
# From S2
m_P["S2", "S2"] <- 1 - p_S2D
m_P["S2", "D"]  <- p_S2D
# From D
m_P["D", "D"]   <- 1


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

Because treatment B alters progression from S1 to S2, we created a different transition probability matrix to model this treatment, `m_P_strB`. We initialize `m_P_strB` as a copy of `m_P` and update only the transition probabilities from S1 to S2 (i.e., `p_S1S2` is replaced with `p_S1S2_trtB`). Strategy AB also alters progression from S1 to S2 because it uses treatment B, so we create this strategy's transition probability matrix as a copy of the transition probability matrix of strategy B.

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


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

Once all transition matrices are created, we verify they are valid by checking that all of their rows sum to one and that each of the transition probabilities of both matrices is between 0 and 1 using the functions `check_sum_of_transition_array` and `check_transition_probability`, respectively, which have been used

11

previously elsewhere[20] and are provided in the `darthtools` package.

```
### Check if transition probability matrices are valid
## Check that transition probabilities are [0, 1]
check_transition_probability(m_P)
check_transition_probability(m_P_strA)
check_transition_probability(m_P_strB)
check_transition_probability(m_P_strAB)
## Check that all rows sum to 1
check_sum_of_transition_array(m_P,      n_states = n_states, n_cycles = n_cycles)
check_sum_of_transition_array(m_P_strB, n_states = n_states, n_cycles = n_cycles)
check_sum_of_transition_array(m_P_strA, n_states = n_states, n_cycles = n_cycles)
check_sum_of_transition_array(m_P_strAB, n_states = n_states, n_cycles = n_cycles)
```

Next, we obtain the cohort distribution across the 4 states over 75 cycles using a time-independent cSTM under all four strategies. To achieve this, we iteratively compute the matrix product between each of the rows of `m_M` and `m_P`, and between `m_M_strB` and `m_P_strB`, respectively, using the `%*%` symbol in R at each cycle using a `for` loop

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

Table 3 shows the cohort trace matrix M of the Sick-Sicker model under strategies SoC and A for the first six cycles. The whole cohort starts in the H state and transitions to the rest of the states over time. Given that the D state is absorbing, the proportion in this state increases over time. A graphical representation of the cohort trace for all the cycles is shown in Figure 2.

Table 3: The distribution of the cohort under strategies SoC and A for the first six cycles of the time-independent Sick-Sicker model. The first row, labeled with cycle 0, contains the distribution of the cohort at time zero.

Cycle	H	S1	S2	D
0	1.000	0.000	0.000	0.000
1	0.848	0.150	0.000	0.002
2	0.794	0.186	0.016	0.005
3	0.766	0.192	0.035	0.008
4	0.745	0.190	0.054	0.011
5	0.726	0.186	0.073	0.015
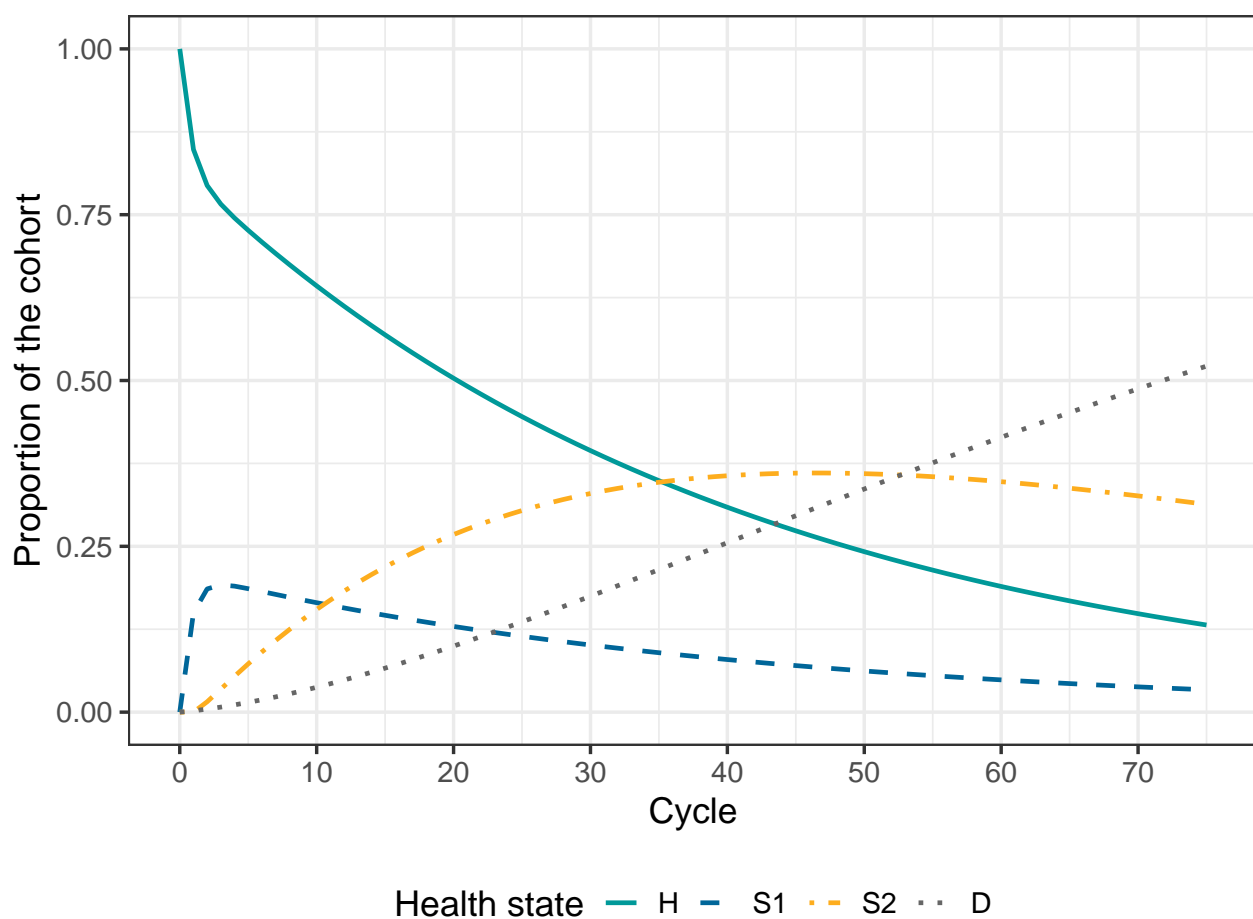
Figure 2: Cohort trace of the time-independent cSTM under strategies SoC and A.

5 Cost and effectiveness outcomes

We are interested in computing the total QALYs and total costs accrued by the cohort over a predefined time horizon for a CEA. In an accompanying tutorial,[17] we describe how to compute epidemiological outcomes from cSTMs, such as survival, prevalence and life expectancy.[5] Epidemiological outcomes are often used to

produce other measures of interest for model calibration and validation.

5.1 State rewards

A state reward refers to a value assigned to individuals for being in a given health state. In a CEA context, these could be either utilities or costs associated with remaining in a specific health state for one cycle. The column vector **y** of size $n_T + 1$ can represent the total expected reward of an outcome of interest for the entire cohort at each cycle. To calculate **y**, we compute the matrix product of the cohort trace matrix times a *vector* of state rewards **r** of the same dimension as the number of states (n_S), such that

$$\mathbf{y} = M\mathbf{r}. \tag{5}$$

For the Sick-Sicker model, we create a vector of utilities and costs for each of the four strategies considered. The vectors of utilities and costs in R, `v_u_SoC` and `v_c_SoC`, respectively, represent the utilities and costs corresponding with being in each of the four health states under SoC, which are 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 specific to strategies involving treatment A (strategies A and AB), `v_u_strA` and `v_u_strAB`. These vectors will have the same utility weights as for strategy SoC except for being in S1, which we assign the utility associated with the benefit of treatment A in being in that state, `u_trtA`. Treatment B does not affect QoL, so the vector of utilities for strategy involving treatment B, `v_u_strB`, is the same as for SoC.

```
# 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 <- c(H = u_H, S1 = u_S1, S2 = u_S2, D = u_D)
# Vector of state utilities for strategy AB
v_u_strAB <- c(H = u_H, S1 = u_trtA, S2 = u_S2, D = u_D)
```

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 the state costs of S1 and S2. Similarly, when constructing the vector of state costs for strategy B, `v_c_strB`, we add the cost of treatment B, `c_trtB`, to the state costs of S1 and S2. 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)
```

To compute the expected QALYs and costs for the Sick-Sicker model under SoC and strategy A, we apply Equation (5) by multiplying the cohort trace matrix, `m_M`, times the corresponding strategy-specific state vectors of rewards. Similarly, to compute the expected rewards for strategies B and AB, we multiply the cohort trace matrix accounting for the effectiveness of treatment B, `m_M_strB`, times their corresponding state vectors of rewards.

```
# Vector of QALYs under SoC
v_qaly_SoC <- m_M %*% v_u_SoC
# Vector of costs under SoC
v_cost_SoC <- m_M %*% v_c_SoC
# Vector of QALYs for strategy A
v_qaly_strA <- m_M_strA %*% v_u_strA
# Vector of costs for strategy A
v_cost_strA <- m_M_strA %*% v_c_strA
# Vector of QALYs for strategy B
v_qaly_strB <- m_M_strB %*% v_u_strB
# Vector of costs for strategy B
v_cost_strB <- m_M_strB %*% v_c_strB
# Vector of QALYs for strategy AB
v_qaly_strAB <- m_M_strAB %*% v_u_strAB
# Vector of costs for strategy AB
v_cost_strAB <- m_M_strAB %*% v_c_strAB
```

5.2   Within-cycle correction

A discrete-time cSTM is an approximation of dynamics that, in reality, occur in continuous time. The discretization of the occurrence of events into discrete-time cycles in a discrete-time cSTM might introduce biases when estimating outcomes based on state occupancy.[21] One approach to reducing these biases is to reduce the cycle length, requiring to simulate the model for a larger number of cycles, which can be computationally burdensome. Another approach is to use within-cycle corrections (WCC).[22] In this tutorial, we use Simpson's 1/3rd rule as the WCC method by multiplying the rewards (e.g., costs and effectiveness) by 1/3 in the first and last cycles, by 4/3 if cycle number is odd, and by 2/3 if the cycle number is even.[24] We implement the WCC by generating a column vector **wcc** of size $n_T + 1$ with values corresponding to the first,

15

$t = 0$, and last cycle, $t = n_T$, equal to 1/3, and the entries corresponding to the even and odd cycles with 2/3 and 4/3, respectively.

$$\mathbf{wcc} = \left[\frac{1}{3}, \frac{2}{3}, \frac{4}{3}, \cdots, \frac{1}{3} \right]$$

The within-cycle correction vector is the same for both costs and QALYs; thus, only one vector, `v_wcc`, is required.

```
# First, we define two functions to identify if a number is even or odd
is_even <- function(x) x %% 2 == 0
is_odd  <- function(x) x %% 2 != 0
## Vector with cycles
v_cycles <- seq(1, n_cycles + 1)
## Generate 2/3 and 4/3 multipliers for even and odd entries, respectively
v_wcc <- is_even(v_cycles)*(2/3) + is_odd(v_cycles)*(4/3)
## Substitute 1/3 in first and last entries
v_wcc[1] <- v_wcc[n_cycles + 1] <- 1/3
```

5.3 Discounting future rewards

We often discount future costs and benefits by a specific rate to calculate the net present value of these rewards. We then use this rate to generate a column vector with cycle-specific discount weights $\mathbf{d}$ of size $n_T + 1$ where its t-th entry represents the discounting for cycle t

$$\mathbf{d}' = \left[1, \frac{1}{(1+d)^1}, \frac{1}{(1+d)^2}, \cdots, \frac{1}{(1+d)^{n_T}} \right],$$

where d is the cycle-length discount rate. At the end of the simulation loop, we multiply the vector of expected rewards, $\mathbf{y}$, by a discounting column vector. The total expected discounted outcome summed over the n_T cycles, y, is obtained by the inner product between $\mathbf{y}$ transposed, $\mathbf{y}'$, and $\mathbf{d}$,

$$y = \mathbf{y}'\mathbf{d}. \tag{6}$$

The discount vectors for costs and QALYs for the Sick-Sicker model with annual cycles, `v_dwc` and `v_dwe`, respectively, are

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

To compute the total expected discounted QALYs and costs under all four strategies accounting for both discounting and WCC, we incorporate $\mathbf{wcc}$ in equation (6) using an element-wise multiplication with $\mathbf{d}$, indicated by the $\odot$ sign.

$$y = \mathbf{y}' \left(\mathbf{d} \odot \mathbf{wcc} \right). \tag{7}$$

16

To compute the total expected discounted and WCC-corrected QALYs under all four strategies in R, we apply Equation (7) to the reward vectors of each strategy.

```
## Expected discounted QALYs under SoC
n_tot_qaly_SoC <- t(v_qaly_SoC) %*% (v_dwe * v_wcc)
## Expected discounted costs under SoC
n_tot_cost_SoC <- t(v_cost_SoC) %*% (v_dwc * v_wcc)
## Expected discounted QALYs for strategy A
n_tot_qaly_strA <- t(v_qaly_strA) %*% (v_dwe * v_wcc)
## Expected discounted costs for strategy A
n_tot_cost_strA <- t(v_cost_strA) %*% (v_dwc * v_wcc)
## Expected discounted QALYs for strategy B
n_tot_qaly_strB <- t(v_qaly_strB) %*% (v_dwe * v_wcc)
## Expected discounted costs for strategy B
n_tot_cost_strB <- t(v_cost_strB) %*% (v_dwc * v_wcc)
## Expected discounted QALYs for strategy AB
n_tot_qaly_strAB <- t(v_qaly_strAB) %*% (v_dwe * v_wcc)
## Expected discounted costs for strategy AB
n_tot_cost_strAB <- t(v_cost_strAB) %*% (v_dwc * v_wcc)
```

Table 4: Total expected discounted QALYs and costs per average individual in the cohort of the Sick-Sicker model by strategy accounting for within-cycle correction.

	Costs	QALYs
Standard of care	$148,657	20.990
Strategy A	$275,937	21.717
Strategy B	$248,571	22.482
Strategy AB	$361,341	23.354

The total expected discounted QALYs and costs for the Sick-Sicker model under the four strategies accounting for within-cycle correction are shown in Table 4.

6   Incremental cost-effectiveness ratios (ICERs)

We combine the total expected discounted costs and QALYs for all four strategies into outcome-specific vectors, `v_cost_str` for costs and `v_qaly_str` for QALYs. We use the R package `dampack` (https://cran.r-project.org/web/packages/dampack/)[25] to calculate the incremental costs and effectiveness and the incremental cost-effectiveness ratio (ICER) between non-dominated strategies and create the data frame `df_cea` with this information.

```
### Vector of costs
v_cost_str <- c(n_tot_cost_SoC, n_tot_cost_strA, n_tot_cost_strB, n_tot_cost_strAB)
### Vector of effectiveness
v_qaly_str <- c(n_tot_qaly_SoC, n_tot_qaly_strA, n_tot_qaly_strB, n_tot_qaly_strAB)
```

```
### Calculate incremental cost-effectiveness ratios (ICERs)
df_cea <- dampack::calculate_icers(cost = v_cost_str,
                                   effect = v_qaly_str,
                                   strategies = v_names_str)
```

The CEA results of the Sick-Sicker model are presented in Table 5. SoC is the least costly and effective strategy, followed by Strategy B producing an expected incremental benefit of 1.492 QALYs per individual for an additional expected cost of $99,913 with an ICER of $66,960/QALY followed by Strategy AB with an ICER $129,354/QALY. Strategy A is a dominated strategy.

Table 5: Cost-effectiveness analysis results for the Sick-Sicker model. ND: Non-dominated strategy; D: Dominated strategy.

Strategy	Costs ($)	QALYs	Incremental Costs ($)	Incremental QALYs	ICER ($/QALY)	Status
Standard of care	148,657	20.990	NA	NA	NA	ND
Strategy B	248,571	22.482	99,913	1.492	66,960	ND
Strategy AB	361,341	23.354	112,771	0.872	129,354	ND
Strategy A	275,937	21.717	NA	NA	NA	D

Figure 3 shows the cost-effectiveness efficient frontier of all four strategies for the time-independent Sick-Sicker model.
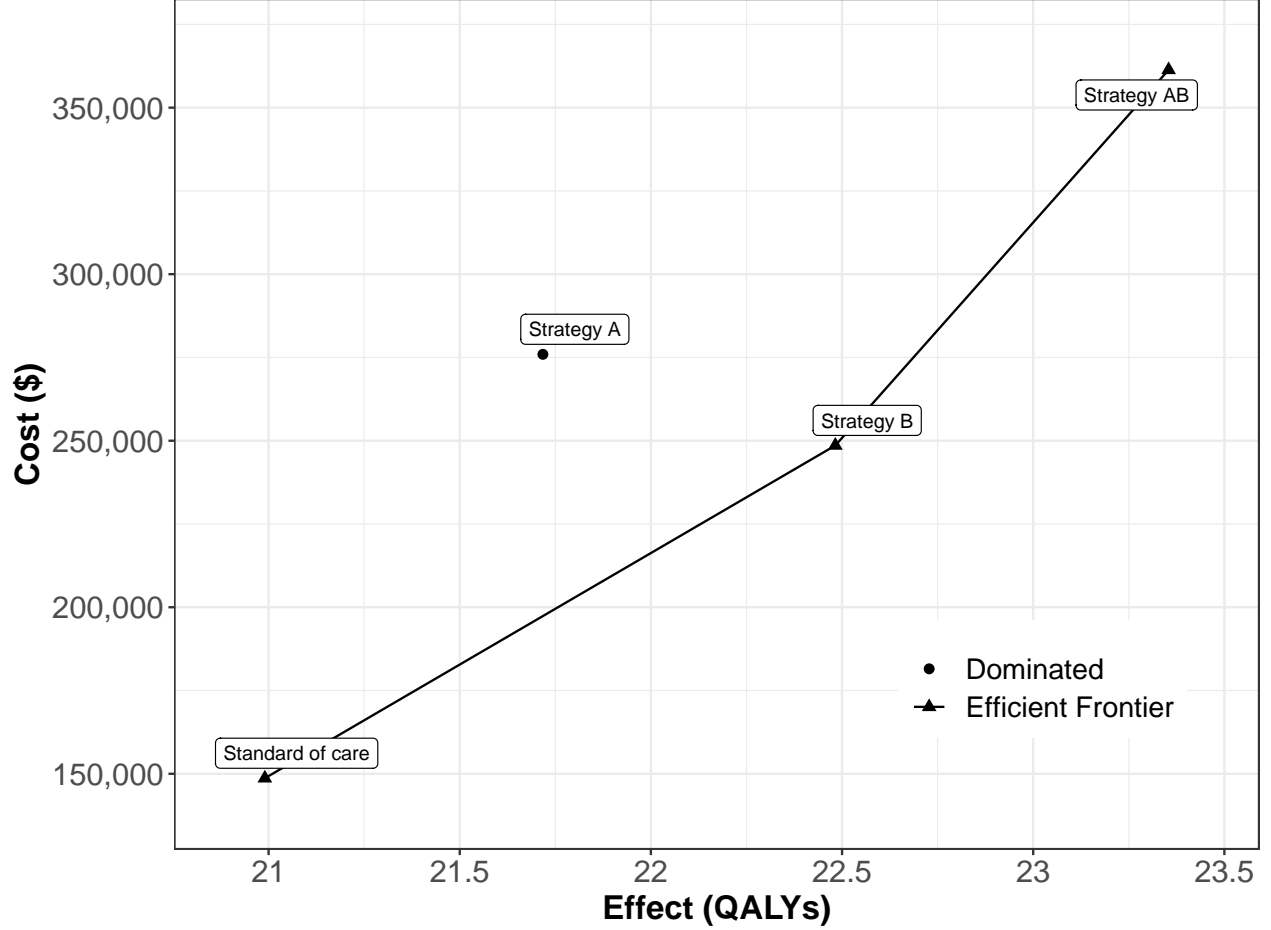
Figure 3: Cost-effectiveness efficient frontier of all four strategies for the time-independent Sick-Sicker model.

7   Probabilistic sensitivity analysis

To quantify the effect of model parameter uncertainty on cost-effectiveness outcomes, we conducted a probabilistic sensitivity analysis (PSA).[26] In a PSA, we randomly draw parameter sets from distributions that reflect the current uncertainty in model parameter estimates. The distribution for all the parameters and their values are described in Table 2 and more detail in the Supplementary Material. For each sampled set of parameter values, we compute model outcomes (e.g., total discounted cost and QALYs) for each strategy. In a previously published manuscript, we describe the implementation of these steps in R.[20] Briefly, to conduct the PSA, we create three R functions:

1. `generate_psa_params(n_sim, seed)`: a function that generates a sample of size `n_sim` for the model parameters, `df_psa_input`, from their distributions defined in Table 2. The function input `seed` sets the seed of the pseudo-random number generator used in sampling parameter values, which ensures reproducibility of the PSA results.

2. `decision_model(l_params_all, verbose = FALSE)`: a function that wraps the R code of the time-independent cSTM described in section 3. This function requires inputting a list of all model parameter values, `l_params_all` and whether the user wants print messages on whether transition probability

matrices are valid via the `verbose` parameter.

3. `calculate_ce_out(l_params_all, n_wtp = 100000)`: a function that calculates total discounted costs and QALYs based on the `decision_model` function output. This function also computes the net monetary benefit (NMB) for a given willingness-to-pay threshold, specified by the argument `n_wtp`. These functions are provided in the accompanying GitHub repository of this manuscript.

To conduct the PSA of the CEA using the time-independent Sick-Sicker cSTM, we sampled 1,000 parameter sets from their distributions. For each sampled parameter set, we simulated each strategy. Results from a PSA can be represented in various ways. For example, the joint distribution, 95% confidence ellipse, and the expected values of the total discounted costs and QALYs for each strategy can be plotted in a cost-effectiveness scatter plot (Figure 4),[27] where each of the 1,000 simulations are plotted as a point in the graph. The CE scatter plot for CEA using the time-independent model 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. Strategy A is a strongly dominated strategy.
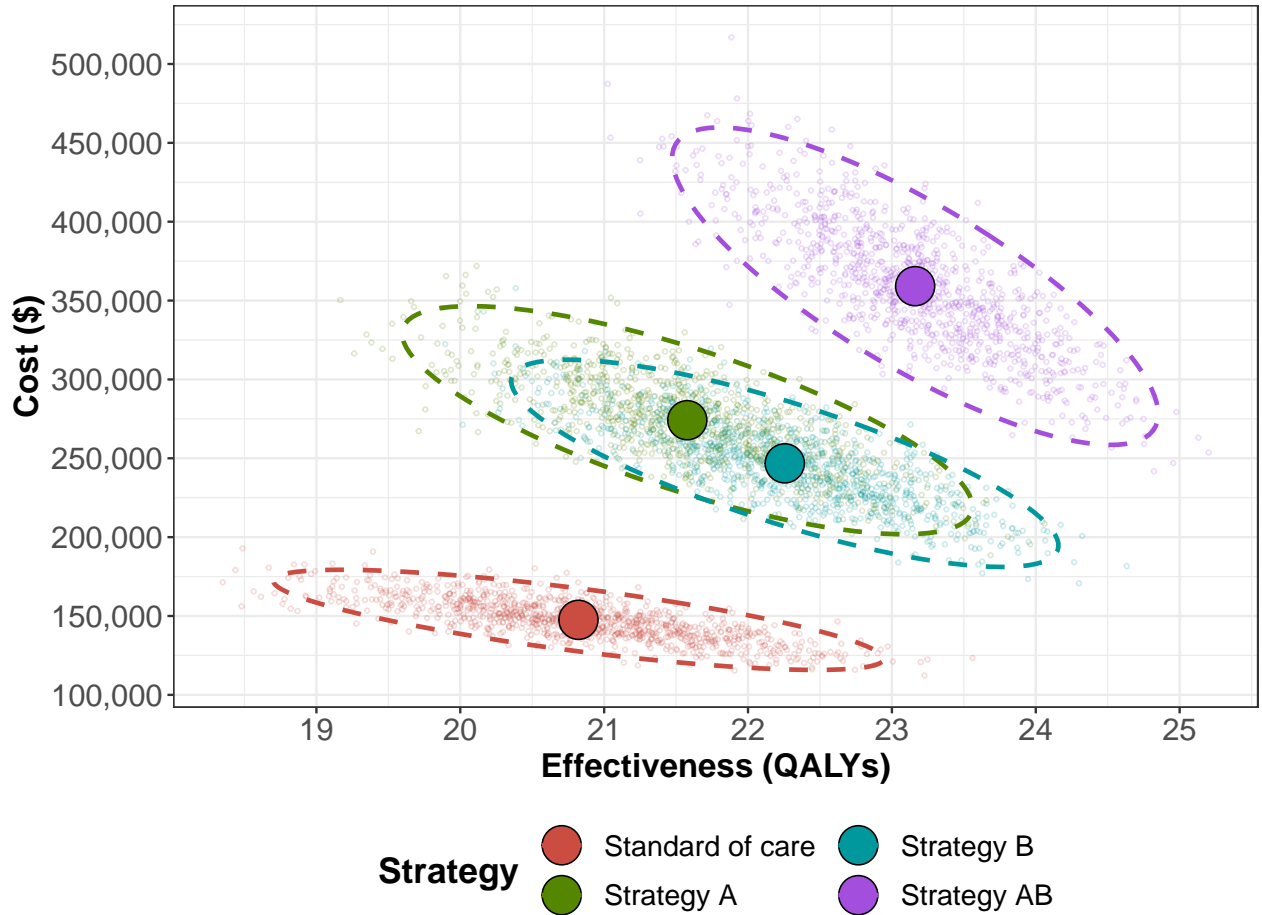
Figure 4: Cost-effectiveness scatter plot.

Figure 5 presents the cost-effectiveness acceptability curves (CEACs), which show the probability that each strategy is cost-effective, and the cost-effectiveness frontier (CEAF), which shows the strategy with the highest expected monetary benefit, over a range of willingness-to-pay (WTP) thresholds. Each strategy's

net monetary benefit (NMB) is computed using $\text{NMB} = \text{QALY} \times \text{WTP} - \text{Cost}$[28] for each PSA sample. At WTP thresholds less than \$70,000 per QALY gained, strategy SoC has both the highest probability of being cost-effective and the highest expected NMB. This switches to strategy B for WTP thresholds between \$70,000 and \$125,000 per QALY gained and to strategy AB for WTP thresholds greater than or equal to \$125,000 per QALY gained.
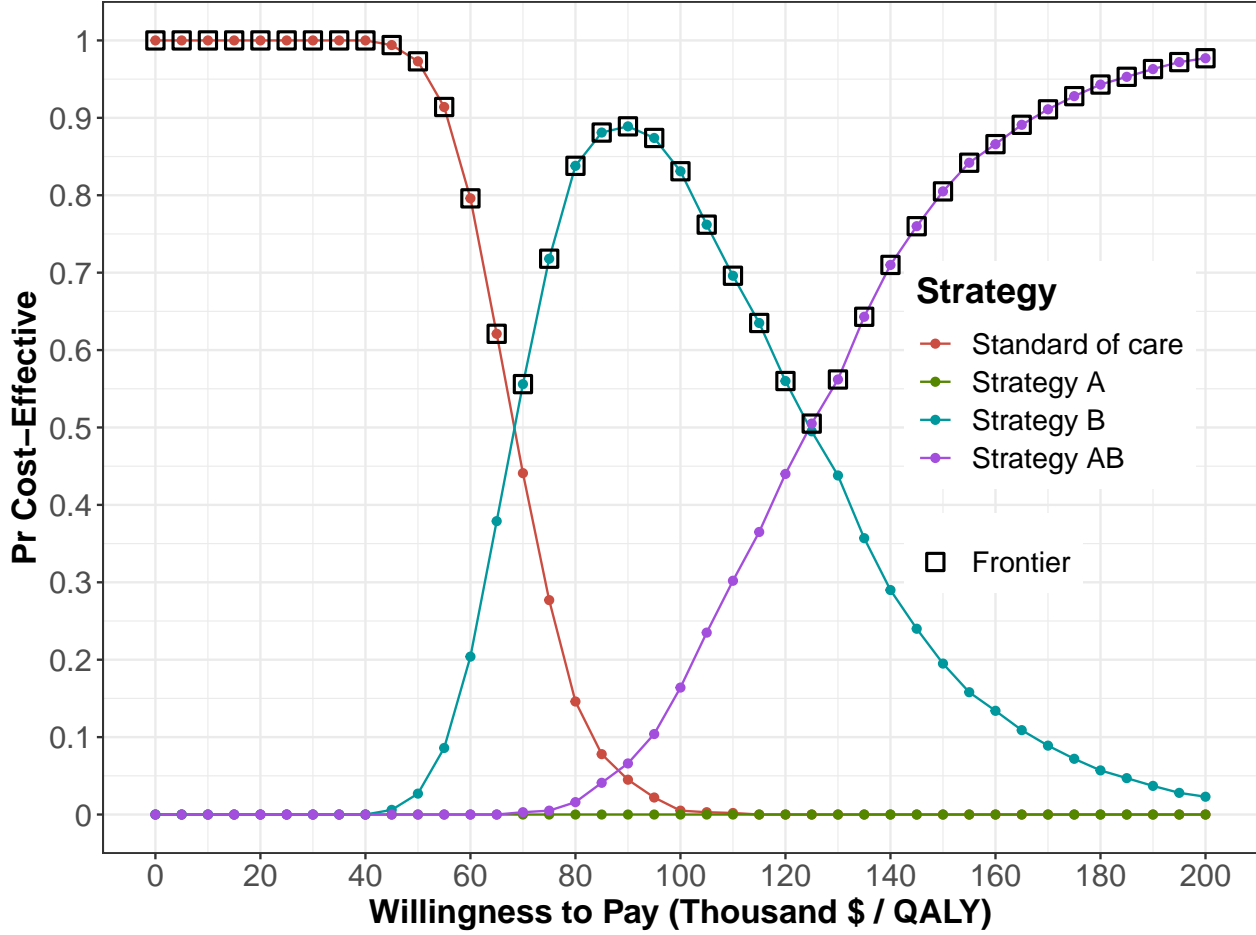

Figure 5: Cost-effectiveness acceptability curves (CEACs) and frontier (CEAF).

The code provided in the GitHub repository also produces expected loss curves (ELCs). These curves quantify the expected loss from each strategy over a range of WTP thresholds (Figure 6). 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 expected loss of the optimal strategy represents the lowest envelope of the ELCs because, given current information, the loss cannot be minimized further. The lower envelope also represents the expected value of perfect information (EVPI), which quantifies the value of eliminating parameter uncertainty. At a WTP threshold of \$125,000 per QALY, the EVPI is highest at \$9,577. For a more detailed description of CEAC, CEAF, ELC and EVPI interpretations and the R code to generate them, we refer the reader to previously published literature.[29]

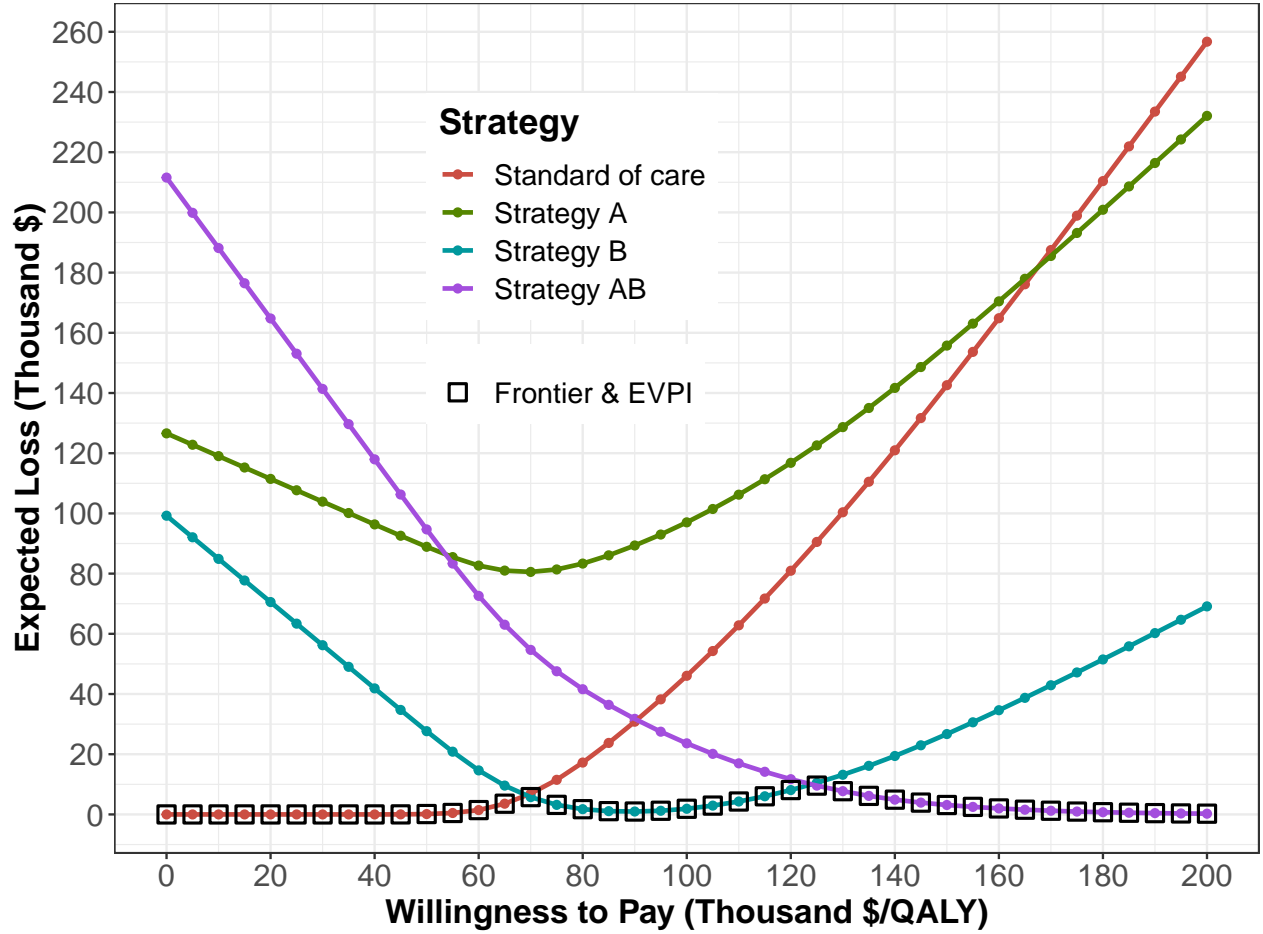Figure 6: Expected loss curves (ELCs) and expected value of perfect information (EVPI).

8   Discussion

In this tutorial, we provided a mathematical conceptualization of time-independent cSTMs and a walk-through of their implementation in R using a hypothetical disease example with accompanying code throughout the tutorial. We used R as the programming language of choice because of its widespread use in the health decision sciences literature and because it is freely available and easily accessible.

The parameterization of our example model assumes all parameters are known, or at least, the characterization of their uncertainty is known (i.e., we know their distributions). However, to construct a real-world cSTM, modelers must conduct a thorough synthesis of current evidence to determine these appropriate structures and inform all parameters based on the current evidence. For example, literature must be carefully considered when determining whether transitions between non-death health states are estimated conditional on being alive or are estimated as competing risks along with mortality risks.[26] Similarly, our PSA analysis simplifies reality where all model parameters are assumed to be independent of each other. However, parameters could be correlated with each other or have a rank order, and appropriate statistical methods that simulate these correlations or rank order might be needed.[30] We encourage modelers to use appropriate statistical methods to synthesize and quantify model parameters uncertainty accurately. In addition, modelers should

appropriately specify all model parameters for the cycle length of the model.[22]

In general, cSTMs are recommended when the number of states is considered "not too large".[5] This recommendation arises because as the number of states increases, it becomes more challenging to keep track of their construction. It is possible to build reasonably complex cSTMs in R as long as the RAM of the computer running the analysis can store the size of the transition probability matrix and outputs of interest. For time-independent cSTMs, in general, this should not be a problem with the capacity of current RAM in personal computers. With increasing model complexity and interdependency of functions to conduct various analyses like PSA, it is essential to ensure all code and functions work as expected and all elements of the cSTM are valid. We can achieve this by creating functions that help with model debugging, validation, and thorough unit testing. In the accompanying GitHub repository, we provide functions to check that transition probability matrices and their elements are valid. However, unit testing is beyond the scope of this tutorial. We refer the reader to a previously published manuscript that describes unit testing in more detail and provides accompanying code.[20]

In this tutorial, we implemented a cSTM using a (discrete-time) transition matrix, but cSTM can also be implemented via a set of (discrete-time) difference equations or (continuous-time) differential equations in R.[31,32] We refer readers interested in learning more on continuous-time cSTMs to previously published manuscripts[21,33–35] and a tutorial using R.[36] Finally, the variable names used in this paper reflect our coding style. While we provide standardized variable names, adopting these conventions is ultimately a personal preference.

In summary, this tutorial provides a conceptualization of time-independent cSTMs and a step-by-step guide to implement them in R. We aim to add to the current body of literature and material on building this type of decision model so that health decision scientists and health economists can develop cSTMs in a more flexible, efficient, open-source manner and to encourage increased transparency and reproducibility. In the accompanying tutorial, we explore generalizing this framework to time-dependent cSTM, generating epidemiological outcomes, and incorporating transition rewards.

9 Acknowledgements

References

1. Kuntz KM, Russell LB, Owens DK, et al. Decision Models in Cost-Effectiveness Analysis. In: Neumann PJ, Sanders GD, Russell LB, et al. (eds) *Cost-effectiveness in health and medicine*. New York, NY: Oxford University Press, 2017, pp. 105–136.

2. Jalal H, Pechlivanoglou P, Krijkamp E, et al. An Overview of R in Health Decision Sciences. *Medical Decision Making*; 37: 735–746, http://journals.sagepub.com/doi/10.1177/0272989X16686559 (2017).

3. Spedicato GA. Discrete Time Markov Chains with R. *The R Journal*; 9: 84–104, https://journal.r-project.org/archive/2017/RJ-2017-036/RJ-2017-036.pdf (2017).

4. Filipović-Pierucci A, Zarca K, Durand-Zaleski I. Markov Models for Health Economic Evaluation: The R Package heemod. *arXiv:170203252v1*; April: 30, http://arxiv.org/abs/1702.03252 (2017).

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

6. Krijkamp EM, Alarid-Escudero F, Enns EA, et al. Microsimulation Modeling for Health Decision Sciences Using R: A Tutorial. *Medical Decision Making*; 38: 400–422, http://journals.sagepub.com/doi/10.1177/0272989X18754513 (2018).

7. Suijkerbuijk AWM, Van Hoek AJ, Koopsen J, et al. Cost-effectiveness of screening for chronic hepatitis B and C among migrant populations in a low endemic country. *PLoS ONE* 2018; 13: 1–16.

8. Sathianathen NJ, Konety BR, Alarid-Escudero F, et al. Cost-effectiveness Analysis of Active Surveillance Strategies for Men with Low-risk Prostate Cancer. *European Urology*; 75: 910–917, https://linkinghub.elsevier.com/retrieve/pii/S0302283818308534 (2019).

9. Lu S, Yu Y, Fu S, et al. Cost-effectiveness of ALK testing and first-line crizotinib therapy for non-small-cell lung cancer in China. *PLoS ONE* 2018; 13: 1–12.

10. Djatche LM, Varga S, Lieberthal RD. Cost-Effectiveness of Aspirin Adherence for Secondary Prevention of Cardiovascular Events. *PharmacoEconomics - Open*; 2: 371–380, https://doi.org/10.1007/s41669-018-0075-2 (2018).

11. Smith-Spangler CM, Juusola JL, Enns EA, et al. Population Strategies to Decrease Sodium Intake and the Burden of Cardiovascular Disease: A Cost-Effectiveness Analysis. *Annals of Internal Medicine*; 152: 481–487, http://annals.org/article.aspx?articleid=745729 (2010).

12. Pershing S, Enns EA, Matesic B, et al. Cost-Effectiveness of Treatment of Diabetic Macular Edema. *Annals of Internal Medicine* 2014; 160: 18–29.

13. Miller DK, Homan SM. Determining Transition Probabilities: Confusion and Suggestions. *Medical Decision Making*; 14: 52–58, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X9401400107 (1994).

14. Kuntz KM, Weinstein MC. Modelling in economic evaluation. In: Drummond MF, McGuire A (eds) *Economic evaluation in health care: Merging theory with practice*. New York, NY: Oxford University Press, 2001, pp. 141–171.

15. Sonnenberg FA, Beck JR. Markov models in medical decision making: A practical guide. *Medical Decision Making*; 13: 322–338, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X9301300409 (1993).

16. Beck JR, Pauker SG. The Markov process in medical prognosis. *Medical Decision Making*; 3: 419–458, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X8300300403 (1983).

17. Alarid-Escudero F, Krijkamp E, Enns EA, et al. A Tutorial on Time-Dependent Cohort State-Transition Models in R. 2021.

18. Iskandar R. A theoretical foundation of state-transition cohort models in health decision analysis. *PLOS ONE*; 13: e0205543, https://www.biorxiv.org/content/early/2018/09/28/430173 (2018).

19. Enns EA, Cipriano LE, Simons CT, et al. Identifying Best-Fitting Inputs in Health-Economic Model Calibration: A Pareto Frontier Approach. *Medical Decision Making*; 35: 170–182, http://www.ncbi.nlm.nih.gov/pubmed/24799456 (2015).

20. Alarid-Escudero F, Krijkamp E, Pechlivanoglou P, et al. A Need for Change! A Coding Framework for Improving Transparency in Decision Modeling. *PharmacoEconomics*; 37: 1329–1339, https://doi.org/10.1007/s40273-019-00837-x (2019).

21. Rosmalen J van, Toy M, O'Mahony JF. A Mathematical Approach for Evaluating Markov Models in Continuous Time without Discrete-Event Simulation. *Medical Decision Making*; 33: 767–779, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X13487947 (2013).

22. Hunink MGGM, Weinstein MC, Wittenberg E, et al. *Decision Making in Health and Medicine*. 2nd ed. Cambridge: Cambridge University Press, http://ebooks.cambridge.org/ref/id/CBO9781139506779 (2014).

23. Elbasha EH, Chhatwal J. Theoretical foundations and practical applications of within-cycle correction methods. *Medical Decision Making* 2016; 36: 115–131.

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

25. Alarid-Escudero F, Knowlton G, Easterly CA, et al. Decision analytic modeling package (dampack), https://cran.r-project.org/web/packages/dampack/%20https://github.com/DARTH-git/dampack (2021).

26. Briggs AH, Weinstein MC, Fenwick EAL, et al. 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; 32: 722–732.

27. Briggs AH, Goeree R, Blackhouse G, et al. Probabilistic Analysis of Cost-Effectiveness Models: Choosing between Treatment Strategies for Gastroesophageal Reflux Disease. *Medical Decision Making*; 22: 290–308, http://mdm.sagepub.com/cgi/doi/10.1177/027298902400448867 (2002).

28. Stinnett AA, Mullahy J. Net Health Benefits: A New Framework for the Analysis of Uncertainty in Cost-Effectiveness Analysis. *Medical Decision Making*; 18: S68–S80, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X9801800209 (1998).

29. Alarid-Escudero F, Enns EA, Kuntz KM, et al. "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: 611–618.

30. Goldhaber-Fiebert JD, Jalal HJ. Some Health States Are Better Than Others: Using Health State Rank Order to Improve Probabilistic Analyses. *Medical Decision Making*; 36: 927–940, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X15605091 (2015).

31. Grimmett G, Welsh D. Markov Chains. In: *Probability: An introduction.* Oxford University Press, pp. 203–, www.statslab.cam.ac.uk/{~}grg/teaching/chapter12.pdf (2014).

32. Axler S, Gehring FW, Ribet KA. Difference Equations. New York, NY: Springer, http://link.springer.com/10.1007/0-387-27645-9 (2005).

33. Cao Q, Buskens E, Feenstra T, et al. Continuous-Time Semi-Markov Models in Health Economic Decision Making: An Illustrative Example in Heart Failure Disease Management. *Medical Decision Making*; 36: 59–71, http://mdm.sagepub.com/cgi/doi/10.1177/0272989X15593080 (2016).

34. Begun A, Icks A, Waldeyer R, et al. Identification of a multistate continuous-time nonhomogeneous Markov chain model for patients with decreased renal function. *Medical Decision Making*; 33: 298–306, http://www.ncbi.nlm.nih.gov/pubmed/23275452 (2013).

35. Soares MO, Canto E Castro L. Continuous time simulation and discretized models for cost-effectiveness analysis. *PharmacoEconomics*; 30: 1101–1117, http://www.ncbi.nlm.nih.gov/pubmed/23116289 (2012).

36. Frederix GWJ, Hasselt JGC van, Severens JL, et al. Development of a framework for cohort simulation in cost-effectiveness analyses using a multistep ordinary differential equation solver algorithm in R. *Medical Decision Making*; 33: 780–92, http://www.ncbi.nlm.nih.gov/pubmed/23515213 (2013).