hesim: Health Economic Simulation Modeling and Decision Analysis

Devin Incerti
Genentech

Jeroen Jansen
University of California, San Francisco

Abstract

Health economic models simulate the costs and effects of health technologies for use in health technology assessment (HTA) to inform efficient use of scarce resources. Models have historically been developed using spreadsheet software (e.g., Microsoft Excel) and while use of R is growing, general purpose modeling software is still limited. **hesim** helps fill this gap by facilitating parameterization, simulation, and analysis of economic models in an integrated manner. Supported model types include cohort discrete time state transition models (cDTSTMs), individual continuous time state transition models (iCTSTMs), and partitioned survival models (PSMs), encompassing Markov (time-homogeneous and time-inhomogeneous) and semi-Markov processes. A modular design based on **R6** and S3 classes allows users to combine separate submodels for disease progression, costs, and quality-adjusted life years (QALYs) in a flexible way. Probabilistic sensitivity analysis (PSA) is used to propagate uncertainty in model parameters to model outputs. Simulation code is written in C++ so complex simulations such as those combining PSA and individual simulation can be run much more quickly than previously possible. Decision analysis within a cost-effectiveness framework is performed using simulated costs and QALYs from a PSA.

Keywords: health economic evaluation, cost-effectiveness analysis, simulation, multi-state models, R.

1. Introduction

Health technology assessment (HTA) is a systematic approach for comparing competing health technologies to inform the efficient use of health care resources. Publicly funded health systems such as those in Australia, Canada, and the United Kingdom, among others, use HTA to help maximize health gains for the population given a fixed budget. HTA is also commonly used by private payers to guide coverage decisions and the adoption of new technologies (Trosman, Van Bebber, and Phillips 2011). Such assessments usually rely heavily on cost-effectiveness analysis (CEA) so that decisions can be made on the basis of a formal evaluation of costs and effects (Dakin, Devlin, Feng, Rice, O'Neill, and Parkin 2015). CEA is made feasible by development of health economic models that simulate costs and effects over relevant time horizons using the totality of available evidence.

The most commonly used economic models are Markov state transition models (STMs) that simulate transitions between mutually exclusive health states. When the Markov assumption holds so that transition probabilities are either constant over time or depend only on model

time, a cohort-level model can be used (Briggs and Sculpher 1998). If the Markov assumption is relaxed—e.g., with a semi-Markov model so that transition probabilities depend on time in an intermediate health state—then individual-level models are required in most cases (Brennan, Chick, and Davies 2006; Fiocco, Putter, and van Houwelingen 2008). Individual-level models afford considerably more flexibility and allow patient history to be tracked over time.

Decisions informed by economic models and CEA are subject to uncertainty. One source of decision uncertainty stems from uncertainty in the underlying model parameters. Parameter uncertainty is typically quantified using probabilistic sensitivity analysis (PSA), which involves randomly sampling the model parameters from suitable probability distribution and simulating the model for each sampled parameter set (Claxton, Sculpher, McCabe, Briggs, Akehurst, Buxton, Brazier, and O'Hagan 2005). When combined with individual-level simulation, PSA can take an appreciable amount of time to run (O'Hagan, Stevenson, and Madan 2007).

Estimation of model parameters should ideally be performed using statistical models that are aligned with the structure of the economic model. For instance, when patient-level data is available, a multi-state model can be used to parameterize all possible transitions in a STM while accounting for censoring and competing risks (Williams, Lewsey, Briggs, and Mackay 2017). Similarly, in oncology, partitioned survival models (PSMs) can be parameterized from estimates of progression-free survival (PFS) and overall survival (OS). In other cases, parameters might be combined from disparate sources, such as within a single Bayesian model (Baio 2012). When the clinical evidence base is not limited to a single study, a formal evidence synthesis, such as a network-meta analysis (NMA), might even be performed (Dias, Ades, Welton, Jansen, and Sutton 2018).

Despite their computational demands and foundations in statistics, health economic models have historically been developed with specialized commercial software (e.g., TreeAge) or more commonly with a spreadsheet (almost always Microsoft Excel). The limitations of such software relative to programming languages like R have been increasingly emphasized in the literature (Baio and Heath 2017; Incerti, Thom, Baio, and Jansen 2019; Jalal, Pechlivanoglou, Krijkamp, Alarid-Escudero, Enns, and Hunink 2017). It is therefore no surprise that a number of related R packages have recently been developed, such as **BCEA** (Baio, Berardi, and Heath 2017), **SAVI** (Strong, Oakley, and Brennan 2014), **survHE** (Baio 2020), and **heemod** (Filipović-Pierucci, Zarca, and Durand-Zaleski 2017). Still, of the available packages, only **heemod** provides a general purpose framework for developing simulation models and it is limited to cohort Markov models.

hesim is an R package that advances the functionality and performance of the existing software. Multiple model types are supported including cohort discrete time state transition models (DTSTMs), N-state PSMs, and individual-level continuous time state transition models (CTSTMs), encompassing both Markov (time-homogeneous and time-inhomogeneous) and semi-Markov processes. To maximize flexibility and facilitate integration of the statistical methods and economic model, parameters can be estimated either by fitting a model in R or by inputting parameters obtained from external sources. So that individual-level simulation and PSA can be run quickly, **Rcpp** and **data.table** are heavily utilized. After simulating costs and quality-adjusted life-years (QALYs) from a PSA, decision analysis can be performed within a cost-effectiveness framework.

The remainder of this article is organized as follows. Section 2 describes the economic models

supported by **hesim**. An overview of the coding framework is provided in Section 3. The next two sections contain example analyses. Section 4 shows how both partitioned survival and multi-state models can be used in an ideal case where rich patient level data is available. Section 5 then moves to the common scenario where only aggregate level data is available shows how both cohort and individual models can be employed. The cost-effectiveness framework is described in Section 6 along with worked examples. Section 7 makes comparisons to other software and discusses possible extensions. Finally, Section 8 concludes.

2. Model taxonomy

STMs simulate transitions between mutually exclusive health states. A common assumption is that the STM is a Markov model, meaning that transitions to the next health state can only depend on the present health state. In a time homogeneous Markov model transition probabilities are constant over time, whereas in a time inhomogeneous model they can depend on time since the start of the model. A semi-Markov model relaxes the Markov assumption and allows transitions to depend on time since entering an intermediate state.

Markov and semi-Markov models can be formulated in either continuous or discrete time, at either the cohort or individual level. In health economics, Markov cohort models tend to be formulated in discrete time (i.e., as cDTSTMs), although state probabilities can be computed in continuous time models using the Aalen-Johansen estimator (Aalen and Johansen 1978) or the Kolmogorov forward equation (Cox and Miller 1977). While tunnel states can be used to approximate a semi-Markov model using a cohort approach, they can only be simulated in a general fashion using individual level models. Discrete time individual simulation is possible, but we use a continuous time models (i.e., iCTSTMs) because they do not require specification of model cycles and can be run considerably faster.

PSMs are specialized models that can be parameterized using survival curves and are especially useful in oncology where PFS and OS are commonly reported. They are "area under the curve" models, although they can also be formulate as STMs by using the survival curves to construct transition probabilities.

2.1. Cohort discrete time state transition models

cDTSTMs simulate the probability that a cohort of patients is in each of H health states over time. Time is measured at discrete times with each time point known as a model cycle. A $1 \times H$ state vector that stores the probability of being in each health state at time t is written as $x_t = (x_{1t}, x_{2t}, \dots, x_{Ht})$ where $\sum_i x_{ht} = 1$. A transition probability matrix is denoted by P_t where the (r, s)th element represents a transition from state r to state s between times t and $t + 1$. The state vector at time $t + 1$ for each of T model cycles is given by,

$$x_{t+1}^T = x_t^T P_t, \quad t = 0, \dots, T. \tag{1}$$

Costs and QALYs are computed by assigning (potentially time-varying) values to each health state. Utility, a measure of preference for a health state that normally ranges from 0 (dead) to 1 (perfect health), is used when computing QALYs (Torrance 1986). Assuming model time is in years, state values for costs are estimated by annualizing costs. In a Markov model, state values, like transition probabilities, can depend on time since the start of the model but not

on time since entering an intermediate health state.

Expected values are computed by integrating the "weighted" probability of being in each state, where weights are a function of the discount factor and state values. That is, for a time horizon T, discounted costs and QALYs in health state h are computed as,

$$\int_0^T z_h(t)e^{-rt}P_h(t)dt, \tag{2}$$

where $z_h(t)$ is the predicted cost or utility value at time t, r is the discount rate, and $P_h(t)$ is the probability of being in a given health state.

In a discrete time, the integral is approximated with

$$\sum_{j=1}^{T} f(t_j^*)\Delta t_j \tag{3}$$

where $\Delta t_j = t_j - t_{j-1}$, $t_j^* \in [t_{j-1}, t_j]$, and $f(t_j^*) = z_h(t_j^*)e^{-rt_j^*}P_h(t_j^*)$. Three methods can be used to estimate $f(t_j^*)$. First, a left Riemann sum uses values at the start of each time interval $f(t_j^*) = f(t_{j-1})$. Second, a right Riemann sum uses values at the end of each time interval $f(t_j^*) = f(t_j)$. Finally, the trapezoid rule averages values at the start and end of each interval $f(t_j^*) = \frac{1}{2}\Delta t_j[f(t_{j-1}) + f(t_j)]$.

2.2. Individual continuous time state transition models

iCTSTMs simulate individual trajectories between health states using random number generation. Trajectories are simulated for multiple patients and costs and QALYs are computed by averaging across the simulated patients. A reasonably large number of patients must be simulated to ensure that expected values are stable (O'Hagan *et al.* 2007).

In continuous time, a patient is in state $X(t)$ at time t. State transitions are modeled using a multi-state modeling framework (Putter, Fiocco, and Geskus 2007) where the probability of a transition from state r to state s is governed by the hazard function,

$$h_{rs}(t) = \lim_{\Delta t \to 0} \frac{\mathsf{P}(X(t + \Delta t) = s | X(t) = r)}{\Delta t}. \tag{4}$$

Simulated disease progression is characterized by J distinct jumps between health states $D = \{(t_0, X(t_0)), (t_1, X(t_1)), \dots (t_J, X(t_J))\}$ with a patient remaining in a health state from time t_j until transitioning to the next state at t_{j+1}. Jumps between health states are simulated using parametric and flexible parametric survival models as implemented in the **flexsurv** package (Jackson 2016). Specifically, if a patient enters state r at time t_j, then a probability density function for the time-to-event t^* for the $r \to s$ transition is,

$$f_{rs}(t^*|\theta(z), t_j), \quad t^* \geq 0, \tag{5}$$

where parameters $\theta = (\theta_1, \theta_2, \dots, \theta_p)$ may depend on covariates z_p through the link function $g(\theta_p) = z_p^T\gamma$ and γ is a vector of regression coefficients. In a time inhomogeneous Markov model, time $t^* = t_{j+1}$ is conditional on not experiencing event s until time t_j (i.e., it is

left-truncated at time t_j). In a semi-Markov model, time-to-event is expressed in terms of $t^* = t_{j+1} - t_j$ and a patient enters state s at time $t_j + t^*$.

A survival distribution is specified for each permitted transition in the multi-state model. A trajectory through the model can then be simulated as described in Algorithm 1. While the algorithm repeats until a patient dies, it can also be stopped at a specified time t or when a patient reaches a maximum age. In the latter scenario, death is assume to occur at the maximum age.

Algorithm 1 Simulation of individual continuous time state transition model

1. Let r be the state entered at time t_j. The number of permitted transitions from state r is given by n_r. If $j = 0$, then $t_j = 0$.
2. Simulate times $\mathcal{T} = \{t_{1,j+1}, t_{2,j+1}, \dots t_{n_r,j+1}\}$ to each of the n_r permitted transitions.
3. Set the time of the transition t_{j+1} equal to the minimum simulated time in $\mathcal{T}$ and the next state s to the state with the minimum simulated time.
4. Set $r = s$ and $t_j = t_{j+1}$. If the patient is still alive, repeat the previous steps until death.

Costs and QALYs are computed using the continuous time present value given a flow of state values, which change as patients transition between health states or as costs vary as a function of time. The state values can be partitioned into M time intervals indexed by $m = 1, \dots, M$ where interval m contains times t such that $t_m \leq t \leq t_{m+1}$ and values for state h are equal to z_{hm} during interval m. z_{hm} will equal zero during time intervals in which a patient is not in state h. Discounted costs and QALYs for health state h are then given by,

$$\sum_{m=1}^{M} \int_{t_m}^{t_m+1} z_{hm} e^{-rt} dt = \sum_{m=1}^{M} z_{hm} \left(\frac{e^{-rt_m} - e^{-rt_{m+1}}}{r} \right), \tag{6}$$

where $r > 0$ is again the discount rate. If $r = 0$, then the present value simplifies to $\sum_{m=1}^{M} z_{hm}(t_{m+1} - t_m)$.

Note that while state values in cohort models can depend on time since the start of the model, state values in individual-level models can depend on either time since the start of the model or time since entering the most recent health state. Individual-level models consequently not only afford more flexibility than cohort models when simulating disease progression, but when simulating costs and/or QALYs as well.

2.3. Partitioned survival models

PSMs are conceptually similar to STMs in that they are characterized by mutually exclusive health states. They differ, however, in that state probabilities are not computed via matrix multiplication or individual simulation, but from a set of non-mutually exclusive survival curves (Glasziou, Simes, and Gelber 1990; Woods, Sideris, Palmer, Latimer, and Soares 2018). Each survival curve represents time to transitioning to that state or to a more severe health state.

In N-state model, $N-1$ non-mutually exclusive survival curves are required. The cumulative survival function, $S_n(t)$, represents the probability that a patient survives to health state n or to a lower indexed state beyond time t. The probability that a patient is in health state 1 is $S_1(t)$. State membership in health states $2, \dots, N-1$ is computed as $S_n(t) - S_{n-1}(t)$. Finally, the probability of being in the final health state n (i.e., the death state) is $1 - S_{N-1}(t)$, or one minus overall survival function.

Survival functions are estimated by fitting parametric or flexible parametric survival models as described in Section 2.2. The nth fitted survival model with density $f_n(t)$ has cumulative density function $F_n(t)$, survivor function $1 - F_n(T)$, cumulative hazard $H_n(t) = -\log S_n(t)$, and hazard $h_n(t) = f_n(t)/S_n(t)$. State probabilities for each health states can be computed for an arbitrarily fine grid of time points to produce the health state vector x_t for each time t in the grid. Costs and QALYs are then computed in the same manner as in the cohort model described in Section 2.1.

3. Framework

Economic models consist of a disease model, a utility model, and a set of cost models for each cost category. Model development proceed in 4 steps. The first step is to setup the model by defining the model structure, target population, and treatment strategies of interest, and storing the relevant information in a `hesim_data` object.

The analysis is performed in the next three steps as shown in Figure 1. First, the disease progression, costs, and utility submodels are parameterized using "estimation" datasets. Next, the submodels are combined to construct an economic model. The economic model is then used to simulate, disease progression, QALYs, and costs as a function of "input data". The input data always contains variables describing the target population and treatment strategies of interest, but can also contain variables related to time to facilitate use of time-varying covariates or parameters. Finally, the simulated outcomes are used to perform decision analysis within a cost-effectiveness framework. While other approaches such as multi-criteria decision analysis (MCDA) could, in principle, be used as well, only CEA is currently supported. All models are simulated using PSA so that decision uncertainty can be represented.

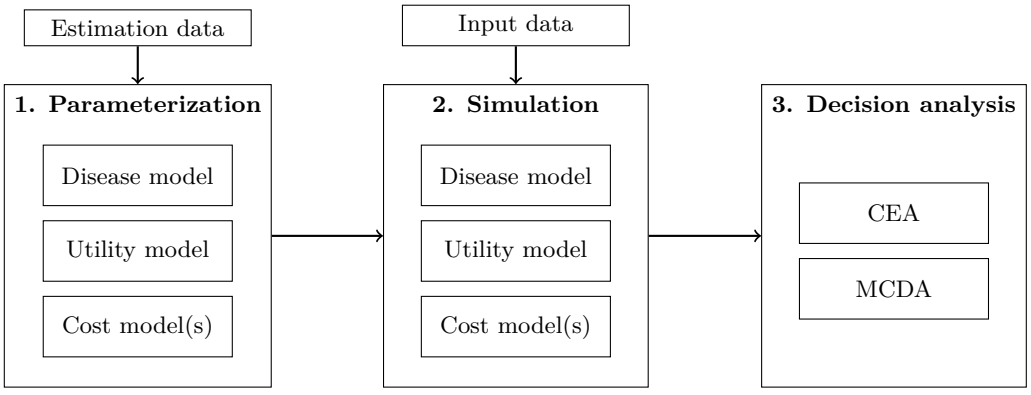

Figure 1: Economic modeling process

The three economic models described in Section 2 are implemented as the **R6** classes `CohortDtstm`, `IndivCtstm`, and `Psm`. Each class contains methods for simulating disease progression, QALYs,

and costs. These methods are made possible by separate **R6** classes for the disease, utility, and cost submodels. The remainder of this section describes the framework for parameterization and simulation. The cost-effectiveness framework is described in Section 6.

3.1. Parameterization

Each submodel contains fields for the model parameters and the input data. Parameters can either be created from a statistical model fit using R or input directly by the user. There are two types of parameter objects, standard parameter objects prefixed by `"params"` and transformed parameter objects prefixed by `"tparams"`. The former contain the underlying parameters of a statistical model while the latter contain parameters more immediate to prediction that have been transformed as function of the input data. The coefficients from a Weibull regression model are an example of a parameter object and the predicted shape and scale parameters of the Weibull distribution are an example of a transformed parameter object.

Although economic models are ideally parameterized using explicit statistical models, it is not always feasible. Parameters are often taken from a variety of sources or from summary level data. In these cases, it can be convenient to build models with mathematical expressions usin non-standard evaluation. This can be done when building cDTSTMs using `defime_model()`.

Disease progression

The statistical model used to parameterize the disease model depends on the type of economic model. For example, multinomial logistic regressions can be used to parameterize a cDTSTM, a set of N-1 independent survival models are used to parameterize an N-state partitioned survival model, and multi-state models can be used to parameterize an iCTSTM (Table 2).

Table 1: Parameterization of disease models

Economic model	Statistical model	Parameter object	Model object
`CohortDtstm`	Custom	`tparams_transprobs`	`define_model()`
	Multinomial logistic regressions	`params_mlogit`	`multinom_list`
`IndivCtstm`	Multi-state model (joint likelihood)	`params_surv`	`flexsurv::flexsurvreg`
	Multi-state model (transition-specific)	`params_surv_list`	`flexsurvreg_list`
`Psm`	Independent survival models	`param_surv_list`	`flexsurvreg_list`

The parameters of a survival model are stored in a `params_surv` object and a `params_surv_list` can be used to store the parameters of multiple survival models. The latter is useful for storing the parameters of a multi-state model or the independent survival models required for a PSM. The parameters of a multinomial logistic regression are stored in a `params_mlogit` and can be created by fitting a model with `nnet::multinom()`.

Although disease models are ideally parameterized by fitting a single statistical model, this is often infeasible. Parameters may be taken from multiple sources, frequently using summary level data. In these cases it can be more convenient to define transition probability matrices using mathematical expressions relating parameters and input data to transformed parameters. Transition probabilities for cDTSTMs can be constructed in this manner using `define_model()`.

Costs and utility

State values (i.e., costs and utilities) do not depend on the choice of disease model. They can currently either be modeled using a linear model or using predicted means. The latter is an example of a transformed parameter object since the predicted means are parameters that are presumably a function of the underlying parameters of a statistical model and input data.

Table 2: Parameterization of state value models

Statistical model	Parameter object	Model object
Predicted means	`tparams_mean`	`stateval_tbl`
	`tparams_mean`	`define_model()`
Linear model	`params_lm`	`stats::lm`

The most straightforward way to construct state values is with a `stateval_tbl`, which is a special object used to assign values to health states that can vary across PSA samples, treatment strategies, patients, and/or time intervals. State values can be specified either as moments (i.e., mean and standard error) or parameters (e.g., shape and scale of gamma distribution) of a probability distribution, or by pre-simulating values from a suitable probability distribution (e.g., from a Bayesian model). Like transition probabilities, state value can also be constructed with `define_model`.

3.2. Simulation

The utility and cost models are always **R6** objects of class `StateVals`, whereas the disease models vary by economic model (Table 3). The disease model is used to simulate survival curves in a PSM and health state transitions in a cDTSTM and iCTSTM.

Table 3: Submodels comprising an economic model

Economic model	Disease model	Utility model	Cost model(s)
`CohortDtstm`	`CohortDtstmTrans`	`StateVals`	`StateVals`
`Psm`	`PsmCurves`	`StateVals`	`StateVals`
`IndivCtstm`	`IndivCtstmTrans`	`StateVals`	`StateVals`

The disease and state value models are most easily instantiated using `S3` generic functions prefixed by "`create`" from parameter, transformed parameter, or statistical model objects. An `IndivCtstmTrans` can, for instance, be created from a `flexsurvreg_list` or `params_surv_list` object with `create_IndivCtstmTrans()`. Similarly, a `StateVals` object can, for example, be created from a `stateval_tbl` object with `create_StateVals()`. The complete economic model is instantiated by combining the submodels using the **R6** constructor method `$new()`.

Each economic model contains methods for simulating disease progression, QALYs, and costs (Table 4). The cost and utility models always simulate costs and QALYs from the simulated progression of disease with the methods `$sim_qalys()` and `$sim_costs()`, respectively. Methods for simulating disease progression differ slightly since the simulation approaches differ as described in Section 2. In an N-state PSM $n - 1$ survival curves are generated and in a iCTSTM a trajectory through state transitions is simulated for multiple patients via individual simulation. All models can simulate state probabilities over time.

Table 4: Methods for simulating outcomes from economic models

Economic model	Disease progression	QALYs	Costs
`CohortDtstm`	`sim_stateprobs()`	`sim_qalys()`	`sim_costs()`
`Psm`	`sim_survival()`, `sim_stateprobs()`	`sim_qalys()`	`sim_costs()`
`IndivCtstm`	`sim_disease()`, `sim_stateprobs()`	`sim_qalys()`	`sim_costs()`

4. Economic modeling with patient level data

4.1. Partitioned survival analysis

4.2. Multi-state model

5. Economic modeling with aggregate level data

Health economic models are commonly developed using aggregate level data taken from multiple sources. In these cases it is helpful to be able input model parameters rather than fitting a statistical model. We illustrate with the 4 state disease progression model displayed in Figure 2. Patients advance between successively more severe disease states. It is possible to move from State 2 back to State 1 but recovery is not possible from State 3. Death can occur from any state.

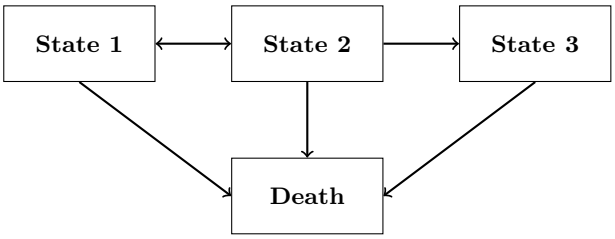

Figure 2: Four state disease progression model

The four health states and five possible transitions are characterized below.

```
        State 1 State 2 State 3 Death
State 1      NA       1      NA     2
State 2       3      NA       4    NA
State 3      NA      NA      NA     5
Death        NA      NA      NA    NA
```

There are 3 treatment strategies: standard of care (SOC) and two new interventions (New 1 and New 2). For illustration, we will consider a target population approximated by two representative patients of age 60 and 70. The 60-year old patient represents 1/3 of all patients with the disease while the 70-year old patient represents 2/3. The `hesim_data` class is again used to store this data as well as the number of non-death health states.

```
R> library("hesim")
R> strategies <- data.table(
```

```
+    strategy_id = 1:3,
+    strategy_name = c("SOC", "New 1", "New 2"),
+    soc = c(1, 0, 0),
+    new1 = c(0, 1, 0),
+    new2 = c(0, 0, 1)
+  )
R> patients <- data.table(
+    patient_id = 1:2,
+    grp_id = 1:2,
+    age = c(60, 70),
+    patient_wt = c(1/3, 2/3)
+  )
R> hesim_dat <- hesim_data(
+    strategies = strategies,
+    patients = patients,
+    states = states[state_name != "Death"]
+  )
R> print(hesim_dat)

$strategies
   strategy_id strategy_name soc new1 new2
1:           1           SOC   1    0    0
2:           2         New 1   0    1    0
3:           3         New 2   0    0    1

$patients
   patient_id grp_id age patient_wt
1:          1      1  60  0.3333333
2:          2      2  70  0.6666667

$states
   state_id state_name
1:        1    State 1
2:        2    State 2
3:        3    State 3

attr(,"class")
[1] "hesim_data"
```

5.1. Cohort model

Parameterization

The core of a cohort DTSTM is the disease model which is determined by the transition matrices P_t. In this example, model parameters related to the transition probabilities for SOC are obtained from a hypothetical analysis of a historical cohort. If data is collected at

unevenly spaced intervals, then transitions can be described with transition intensity matrices, Q_t, by fitting a continuous time multi-state model designed for panel data (e.g., with the R package **msm**) (Jackson *et al.* 2011). We assume that regression coefficients and a covariance matrix are available from such an analysis and that a piecewise exponential model was fit so that intensities change after year 2.

```
R> q_coef_soc
```

```
    trans         var    name        loghr
 1:  1->2   intercept  int_12 -0.105360516
 2:  1->4   intercept  int_14 -1.609437912
 3:  2->1   intercept  int_21 -0.693147181
 4:  2->3   intercept  int_23  0.095310180
 5:  3->4   intercept  int_34  0.405465108
 6:  1->2 time[2,Inf) time_12  0.182321557
 7:  1->4 time[2,Inf) time_14  0.182321557
 8:  2->1 time[2,Inf) time_21  0.182321557
 9:  2->3 time[2,Inf) time_23  0.182321557
10:  3->4 time[2,Inf) time_34  0.182321557
11:  1->2         age  age_12  0.002995509
12:  1->4         age  age_14  0.002995509
13:  2->1         age  age_21  0.002995509
14:  2->3         age  age_23  0.002995509
15:  3->4         age  age_34  0.002995509
```

```
R> q_vcov_soc[1:4, 1:4]
```

```
              [,1]          [,2]          [,3]          [,4]
[1,]  9.743459e-04 -8.692414e-05 -3.037960e-04  0.0008443623
[2,] -8.692414e-05  1.401786e-03 -2.759369e-05 -0.0002954624
[3,] -3.037960e-04 -2.759369e-05  1.249238e-03 -0.0004715665
[4,]  8.443623e-04 -2.954624e-04 -4.715665e-04  0.0012019299
```

If many studies are available, then relative treatment effects can be estimated by means of a network meta-analysis (NMA). We assume that a NMA was performed to estimate the relative risk of transitions from State 1 to both State 2 and Death. The SOC was the reference arm. Estimation is performed in terms of the log relative risk since it is approximately normally distributed.

The utility associated with each state is assumed to follow a beta distribution, which is bounded between 0 and 1. The beta distribution is defined in terms of 2 shape parameters, but can be derived from the mean and standard error using the method of moments. Point estimates for utility in State 1, State 2, and State 3 are 0.9, 0.75, and 0.5, respectively. Standard errors in the same three states are 0.02, 0.03, and 0.05. Utility (and costs) are always assumed to be 0 in the death state.

We assume that two categories of costs are measured: treatment and medical. The former are the costs of the three treatment strategies and are assumed fixed. The SOC is the cheapest

option at $2,000 per annum while New 1 and New 2 cost $10,000 and $12,000 per year, respectively.

While there is no uniform way to define medical costs, they often consist of costs accrued from hospitalizations and inpatient visits. They are assumed to follow a gamma distribution given that they are usually heavily right skewed. Like the beta distribution, the parameters of the gamma distribution are derived from the mean and standard error using the method of moments. Mean medical costs are $2,000, $4,000, and $15,000 in State 1, State 2, and State 3, respectively. We assume the variance is unavailable and set the standard error in each state equal to the mean.

```
R> params <- list(
+    q_coef_soc = q_coef_soc,
+    q_vcov_soc = q_vcov_soc,
+    lrr_12_est = c(soc = log(1), new1 = log(.80), new2 = log(.71)),
+    lrr_12_se = c(soc = 0, new1 = .03, new2 = .04),
+    lrr_14_est = c(soc = log(1), new1 = log(.90), new2 = log(.85)),
+    lrr_14_se = c(soc = 0, new1 = .02, new2 = .03),
+    u_mean = c(s1 = .9, s2 = .75, s3 = .05),
+    u_se = c(s1 = .02, s2 = .03, s3 = .05),
+    c_medical = c(s1 = 2000, s2 = 4000, s3 = 15000),
+    c_tx = c(soc = 2000, new1 = 10000, new2 = 12000)
+  )
```

To perform a PSA, it is necessary to randomly draw the parameters from suitable probability distributions. These distributions can be defined using `define_rng()`. While any random number generation function can be used, a number of commonly used distributions are specifically designed for use within a `define_rng()` block (see `?rng_distributions`). The built in distributions do not require specification of the number of PSA iterations (since it is defined within the block) and automatically perform certain operations (e.g., sampling from a gamma distribution from means and standard errors by usin methods of moments to derive the underlying shape and scale parameters). We include utility for illustrative purposes but omit the cost parameters since they can be sampled more directly with `stateva_tbl()`.

```
R> rng_def <- define_rng({
+    list( # Parameters to return
+      q_coef_soc = multi_normal_rng(q_coef_soc$loghr, q_vcov_soc,
+                                    names = q_coef_soc$name),
+      rr_12 = lognormal_rng(lrr_12_est, lrr_12_se),
+      rr_14 = lognormal_rng(lrr_14_est, lrr_14_se),
+      u = beta_rng(mean = u_mean, sd = u_se)
+    )
+  }, n = 1000)
```

The random number generation expressions can are evaluated with `eval_rng()`.

```
R> params_rng <- eval_rng(rng_def, params = params)
```

The transition intensity matrices and relative risks can be combined to derive the relevant transition probability matrices, stored as a `tparams_transprobs` object. A `tparams_transprobs` is simply a 3-dimensional array of transition probability matrices with a unique predicted matrix for each PSA sample, treatment strategy, representative patient, and (optionally) time interval. As in the prior example, the underlying input data can be constructed with `expand`.

```
R> transmod_data <- expand(hesim_dat, by = c("strategies", "patients"),
+                          times = 2)
R> head(transmod_data)
```

```
   strategy_id patient_id time_id strategy_name soc new1 new2 grp_id
1:           1          1       1           SOC   1    0    0      1
2:           1          1       2           SOC   1    0    0      1
3:           1          2       1           SOC   1    0    0      2
4:           1          2       2           SOC   1    0    0      2
5:           2          1       1         New 1   0    1    0      1
6:           2          1       2         New 1   0    1    0      1
   age patient_wt time_start time_stop
1:  60  0.3333333          0         2
2:  60  0.3333333          2       Inf
3:  70  0.6666667          0         2
4:  70  0.6666667          2       Inf
5:  60  0.3333333          0         2
6:  60  0.3333333          2       Inf
```

A complete dataset of ID variables that also incorporate the PSA iterations is constructed with `tpmatrix_id()`. The objective is to construct a transition probability for each row in this dataset. The user has complete flexibility to do this.

```
R> tpmat_id <- tpmatrix_id(transmod_data, rng_def$n)
R> head(tpmat_id)
```

```
   sample strategy_id patient_id grp_id patient_wt time_id time_start
1:      1           1          1      1  0.3333333       1          0
2:      1           1          1      1  0.3333333       2          2
3:      1           1          2      2  0.6666667       1          0
4:      1           1          2      2  0.6666667       2          2
5:      1           2          1      1  0.3333333       1          0
6:      1           2          1      1  0.3333333       2          2
   time_stop
1:         2
2:       Inf
3:         2
4:       Inf
5:         2
6:       Inf
```

In this example we do this by first using the coefficients and input data to to predict transition intensity matrices for SOC (by age group and time interval). This is done by separately predicting intensities for all feasible transitions. There are $n = 4$ observations for SOC and $B = 1000$ random draws of the coefficients for the PSA. A transition from state r to state s is modeled as $log(q_{rs}) = X\beta^T$ where X is an $n \times 3$ design matrix and β is a $B \times 3$ sample of the coefficients. The resulting $n \times B$ matrix can be conveniently flattened into a vector of length nB with the `c()` function so that predicted values are stacked by parameter sample as in `tpmatrix_id()`.

```
R> predict_q_soc <- function(data, beta){
+    data_soc <- data[strategy_name == "SOC"]
+    x <- model.matrix(~factor(time_start) + age, data = data_soc)
+    n_trans <- 5
+    qmat_soc <- matrix(NA, nrow = nrow(beta) * nrow(x), ncol = n_trans)
+    colnames(qmat_soc) <- c("q_12", "q_14", "q_21", "q_23", "q34")
+    for (i in 1:n_trans) {
+       beta_i <- as.matrix(beta)[, seq(i, ncol(beta), n_trans)]
+       qmat_soc[, i] <- exp(c(x %*% t(beta_i)))
+    }
+    return(qmat_soc)
+  }
R> q_soc <- predict_q_soc(transmod_data, params_rng$q_coef_soc)
R> head(q_soc)
```

```
         q_12        q_14       q_21       q_23        q34
[1,] 14.82301 0.011725932 1.86899144 0.08533259  0.8018565
[2,] 18.13121 0.013273955 2.28667643 0.10488687  0.9435241
[3,] 23.60066 0.007292289 2.32903289 0.05555563  0.7307565
[4,] 28.86785 0.008254996 2.84952861 0.06828641  0.8598626
[5,] 63.16646 0.004109621 0.09778717 0.03292046 25.5371554
[6,] 76.96592 0.004753191 0.11946125 0.04040065 31.1453112
```

The predicted transition intensities are, in turn, use to form the complete transition intensity matrix. Specifically, `qmatrix()` forms an array of transition intensity matrices where each row sums to 0 so that $q_{rr} = \sum_{s \neq r} -q_{rs}$.

```
R> qmat_soc <- qmatrix(q_soc, tmat)
R> qmat_soc[,, 1]
```

```
           [,1]      [,2]        [,3]       [,4]
[1,] -14.834738 14.823012  0.00000000 0.01172593
[2,]   1.868991 -1.954324  0.08533259 0.00000000
[3,]   0.000000  0.000000 -0.80185645 0.80185645
[4,]   0.000000  0.000000  0.00000000 0.00000000
```

Transition probability matrices are constructed from the transition intensity matrices using the matrix exponential (Jackson *et al.* 2011). In continuous time, $P(t) = \exp(tQ)$ so a

transition probability matrix with model cycles of length t years is given by $\exp(Q_t)$. In this example we will assume model cycles are 1/2 of a year.

```
R> pmat_soc <- expmat(qmat_soc, t = 1/2)
R> pmat_soc[,, 1]
```

```
          [,1]      [,2]       [,3]        [,4]
[1,] 0.1087938 0.8571878 0.02748697 0.006531414
[2,] 0.1080804 0.8536447 0.03095643 0.007318499
[3,] 0.0000000 0.0000000 0.66969813 0.330301874
[4,] 0.0000000 0.0000000 0.00000000 1.000000000
```

Transition probability matrices can be derived for treatment strategies New 1 and New 2 using the relative risks. Relative risks are predicted for each row in the `tmat_id` dataset above.

```
R> x_rr <- as.matrix(transmod_data[, .(soc, new1, new2)])
R> rr_12 <- c(x_rr %*% t(params_rng$rr_12))
R> rr_14 <- c(x_rr %*% t(params_rng$rr_14))
R> rr <- cbind(rr_12, rr_14)
```

Given reference transition probability matrices (i.e., those for SOC), we can apply the relative risks for the other treatment strategies using the `apply_rr()` function. An array of transition probability matrices is returned with one matrix for each row in `rr`.

```
R> pmat <- apply_rr(pmat_soc, rr = rr,
+                   index = list(c(1, 2), c(1, 4)))
```

Finally, the array of transition probability matrices is combined with the relevant ID variables indexing each array slice to create the `tparams_transprobs` object.

```
R> tprobs <- tparams_transprobs(pmat, tpmat_id)
```

The utility and cost parameters are much more straightforward to store. As in the previous example we consider two cost categories and store parameter values in a `stateval_tbl`.

```
R> txcost_tbl <- stateval_tbl(
+    data.table(
+      strategy_id = 1:3,
+      est = c(soc = 2000, new1 = 10000, new2 = 12000)
+    ),
+    dist = "fixed",
+    hesim_data = hesim_dat
+  )
R> head(txcost_tbl)
```

```
   strategy_id   est
1:           1  2000
2:           2 10000
3:           3 12000
```

```
R> medcost_tbl <- stateval_tbl(
+    data.table(
+      state_id = 1:3,
+      mean = c(s1 = 2000, s2 = 4000, s3 = 15000),
+      se = c(s1 = 2000, s2 = 4000, s3 = 15000)
+    ),
+    dist = "gamma",
+    hesim_data = hesim_dat
+  )
R> head(medcost_tbl)
```

```
   state_id  mean    se
1:        1  2000  2000
2:        2  4000  4000
3:        3 15000 15000
```

Although we could do the same for utility, we illustrate an alternative approach in which the distribution of utility values has been previously drawn from a suitable probability distribution. While not necessary in this case, this feature might be useful in settings where samples have already been drawn from the posterior distribution of a Bayesian model.

```
R> utility_tbl <- stateval_tbl(
+    data.table(
+      state_id = rep(c(1, 2, 3), each = rng_def$n),
+      sample = rep(1:rng_def$n, nrow(hesim_dat$states)),
+      value = c(as.matrix(params_rng$u))
+    ),
+    dist = "custom",
+    hesim_data = hesim_dat
+  )
R> head(utility_tbl)
```

```
   sample state_id     value
1:      1        1 0.8949624
2:      2        1 0.8635926
3:      3        1 0.8628418
4:      4        1 0.8561286
5:      5        1 0.8999304
6:      6        1 0.8833792
```

Simulation

The simulation proceeds the same way as in the prior examples. We first create the transition,

cost, and utility models, here specifying that model cycles are 6 months long since the model is in discrete time.

```
R> transmod <- CohortDtstmTrans$new(params = tprobs, cycle_length = 1/2)

R> # Utility
R> utilitymod <- create_StateVals(utility_tbl)
R> # Costs
R> txcostmod <- create_StateVals(txcost_tbl, n = rng_def$n)
R> medcostmod <- create_StateVals(medcost_tbl, n = rng_def$n)
R> costmods <- list(Treatment = txcostmod,
+                   Medical = medcostmod)

R> econmod <- CohortDtstm$new(trans_model = transmod,
+                             utility_model = utilitymod,
+                             cost_models = costmods)
```

State probabilities are simulated for 45 years, or 90 model cycles.

```
R> econmod$sim_stateprobs(n_cycles = 45 * 2)
```

Costs and QALYs are again simulated with 3 percent discount rates.

```
R> econmod$sim_costs(dr = .03)
R> econmod$sim_qalys(dr = .03)
```

Costs and QALYs are first summarized by subgroup.

```
R> ce_sim_grp <- econmod$summarize(by_grp = TRUE)
R> cea_pw_out <- cea_pw(ce_sim_grp, comparator = 1, dr_costs = .03, dr_qalys = .03)
R> icer_tbl(cea_pw_out, output = "data.table")

   strategy_id grp_id               iqalys                      icosts
1:           2      1 0.15 (-16.68, 16.80) 40,504 (-108,707, 260,009)
2:           3      1 0.24 (-16.39, 17.00)  51,255 (-98,706, 319,900)
3:           2      2 0.15 (-18.28, 18.58) 45,798 (-122,815, 292,707)
4:           3      2 0.23 (-18.06, 18.78) 57,694 (-128,181, 342,653)
                          inmb    icer         conclusion
1: -32,873 (-818,670, 626,401) 265,403 Not cost-effective
2: -39,246 (-801,777, 601,635) 213,407 Not cost-effective
3: -38,432 (-879,511, 693,499) 310,889 Not cost-effective
4: -46,065 (-885,921, 669,918) 248,063 Not cost-effective
```

They can also be aggregated across groups using the weighted specified in the `hesim_data` object above.

```
R> ce_sim <- econmod$summarize()
R> cea_pw_out <- cea_pw(ce_sim, comparator = 1, dr_costs = .03, dr_qalys = .03)
R> icer_tbl(cea_pw_out, output = "data.table")

   strategy_id grp_id               iqalys                      icosts
1:           2      1 0.15 (-17.92, 18.09) 44,033 (-118,179, 278,464)
2:           3      1 0.24 (-17.38, 18.27) 55,548 (-118,429, 333,469)
                          inmb    icer         conclusion
1: -36,579 (-855,737, 669,234) 295,368 Not cost-effective
2: -43,792 (-858,110, 645,776) 236,262 Not cost-effective
```

5.2. Individual model

6. Cost-effectiveness analysis

7. Discussion

8. Conclusion

References

Aalen OO, Johansen S (1978). "An empirical transition matrix for non-homogeneous Markov chains based on censored observations." *Scandinavian Journal of Statistics*, pp. 141–150.

Baio G (2012). *Bayesian methods in health economics*. CRC Press.

Baio G (2020). "survHE: Survival Analysis for Health Economic Evaluation and Cost-Effectiveness Modeling." *Journal of Statistical Software*, **95**(1), 1–47.

Baio G, Berardi A, Heath A (2017). *Bayesian cost-effectiveness analysis with the R package BCEA*. Springer.

Baio G, Heath A (2017). "When simple becomes complicated: why excel should lose its place at the top table."

Brennan A, Chick SE, Davies R (2006). "A taxonomy of model structures for economic evaluation of health technologies." *Health economics*, **15**(12), 1295–1310.

Briggs A, Sculpher M (1998). "An introduction to Markov modelling for economic evaluation." *Pharmacoeconomics*, **13**(4), 397–409.

Claxton K, Sculpher M, McCabe C, Briggs A, Akehurst R, Buxton M, Brazier J, O'Hagan T (2005). "Probabilistic sensitivity analysis for NICE technology assessment: not an optional extra." *Health economics*, **14**(4), 339–347.

Cox DR, Miller HD (1977). *The theory of stochastic processes*, volume 134. CRC press.

Dakin H, Devlin N, Feng Y, Rice N, O'Neill P, Parkin D (2015). "The influence of cost-effectiveness and other factors on nice decisions." *Health economics*, **24**(10), 1256–1271.

Dias S, Ades AE, Welton NJ, Jansen JP, Sutton AJ (2018). *Network meta-analysis for decision-making*. John Wiley & Sons.

Filipović-Pierucci A, Zarca K, Durand-Zaleski I (2017). "Markov Models for Health Economic Evaluations: The R Package heemod." *arXiv preprint arXiv:1702.03252*.

Fiocco M, Putter H, van Houwelingen HC (2008). "Reduced-rank proportional hazards regression and simulation-based prediction for multi-state models." *Statistics in Medicine*, **27**(21), 4340–4358.

Glasziou P, Simes R, Gelber R (1990). "Quality adjusted survival analysis." *Statistics in medicine*, **9**(11), 1259–1276.

Incerti D, Thom H, Baio G, Jansen JP (2019). "R you still using excel? The advantages of modern software tools for health technology assessment." *Value in Health*, **22**(5), 575–579.

Jackson CH (2016). "flexsurv: a platform for parametric survival modeling in R." *Journal of statistical software*, **70**.

Jackson CH, *et al.* (2011). "Multi-state models for panel data: the msm package for R." *Journal of statistical software*, **38**(8), 1–29.

Jalal H, Pechlivanoglou P, Krijkamp E, Alarid-Escudero F, Enns E, Hunink MM (2017). "An overview of R in health decision sciences." *Medical decision making*, **37**(7), 735–746.

O'Hagan A, Stevenson M, Madan J (2007). "Monte Carlo probabilistic sensitivity analysis for patient level simulation models: efficient estimation of mean and variance using ANOVA." *Health economics*, **16**(10), 1009–1023.

Putter H, Fiocco M, Geskus RB (2007). "Tutorial in biostatistics: competing risks and multi-state models." *Statistics in medicine*, **26**(11), 2389–2430.

Strong M, Oakley JE, Brennan A (2014). "Estimating multiparameter partial expected value of perfect information from a probabilistic sensitivity analysis sample: a nonparametric regression approach." *Medical Decision Making*, **34**(3), 311–326.

Torrance GW (1986). "Measurement of health state utilities for economic appraisal: a review." *Journal of health economics*, **5**(1), 1–30.

Trosman JR, Van Bebber SL, Phillips KA (2011). "Health technology assessment and private payers' coverage of personalized medicine." *Journal of oncology practice*, **7**(3S), 18s–24s.

Williams C, Lewsey JD, Briggs AH, Mackay DF (2017). "Cost-effectiveness analysis in R using a multi-state modeling survival analysis framework: a tutorial." *Medical Decision Making*, **37**(4), 340–352.

Woods B, Sideris E, Palmer S, Latimer N, Soares M (2018). "NICE DSU technical support document 19. Partitioned survival analysis for decision modelling in health care: a critical review. 2017." *Available from: www.. nicedsu. org. uk/wp-content/uploads/2017/06/Partitioned-Survival-Analysisfinal-report. pdf*.

Affiliation:

Devin Incerti
Genentech
South San Francisco
E-mail: incerti.devin@gene.com

Jeroen Jansen
University of California, San Francisco
San Francisco
E-mail: jeroen.jansen@ucsf.edu