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, 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. An illustrative example using a three state model of disease progression in oncology is provided in Section 4. Analyses are conducted for each of the three supported model types, illustrating approaches suitable for both patient level and summary level data. The cost-effectiveness framework is described in Section 5 along with worked examples based on the output from the prior section. Section 6 makes comparisons to other software and discusses possible extensions. Finally, Section 7 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 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 formulated 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 proceeds 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.

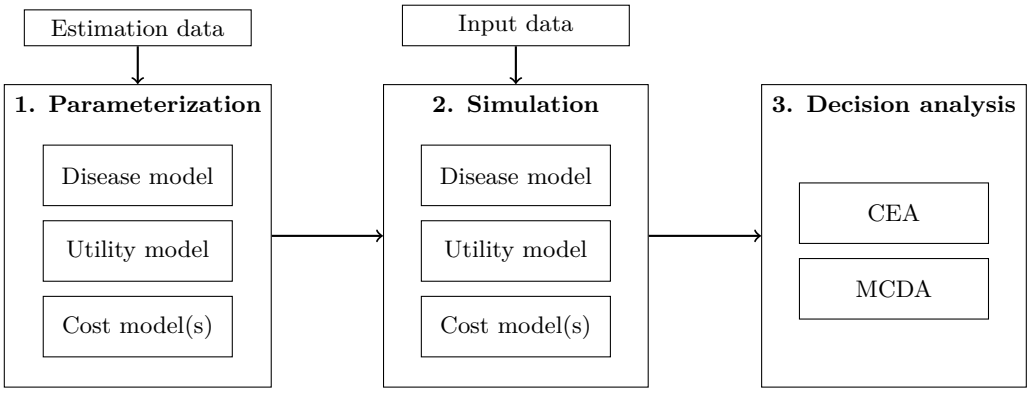


Figure 1: Economic modeling process

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 5.

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 regression coefficients of a logistic regression are example of a parameter objects while the predicted probabilities are examples of a transformed parameter object.

Transformed parameter objects are often useful when parameterizing a model using external sources. Two common examples are predicted probabilities for transition matrices and predicted means for assigning values to health states. The advantage of a transformed parameter object is that an explicit statistical model does not need to be specified which affords more flexibility to the user.

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`	`msm::msm`
	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()`.

`tparams_transprobs` objects contain transition probability matrices that have been predicted for each PSA sample, treatment strategy, and patient from the target population. Transition probabilities may also vary by time interval to allow for time inhomogeneous Markov models. Transition probabilities can be predicted from a fitted multi-state model using the **msm** package or constructed "by hand" in a custom manner.

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.

Table 2: Parameterization of state value models

Statistical model	Parameter object	Model object
Predicted means	`tparams_mean`	`stateval_tbl::lm`
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 (i.e. predicted means) 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).

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. Illustrative example

We consider a three state model commonly used describe disease progression in oncology. As shown in Figure 2, the three health states are stable disease (i.e. not progressed), progressed, and death. Patients can transition to a more severe health state (stable → progressed, stable → death, and progression → death) but cannot recover to a less severe health state. We assume that patients remain on first line (1L) treatment until progression, at which time they switch to a second line (2L) treatment.

Figure 2: Three state model of disease progression in oncology

This model can be simulated using either of the model types described in Section 2. We demonstrate use of all three below, starting with the iCTSTM.

4.1. Individual continuous time state transition model

We setup the model by defining the model structure, target population, and treatment strategies of interest. The transitions are characterized by a matrix where each element denotes a transition from a row to a column. If a transition is not possible it is marked with `NA`; otherwise, an integer denotes the transition number.

```
R> tmat <- rbind(
+    c(NA, 1, 2),
+    c(NA, NA, 3),
+    c(NA, NA, NA)
+  )
R> colnames(tmat) <- rownames(tmat) <- c("Stable", "Progression", "Dead")
R> print(tmat)

            Stable Progression Dead
Stable          NA           1    2
Progression     NA          NA    3
Dead            NA          NA   NA
```

The target population, treatment strategies, and (non-death) health states are stored in a `hesim_data` object. There are 3 treatment strategies: standard of care (SOC) and two new interventions (New 1 and New 2). The target population consists of a heterogeneous population of 1000 patients, which is large enough to ensure that averages across patients in the individual simulation are reasonably stable.

```
R> library("hesim")
R> library("data.table")
```

```
R> n_patients <- 1000
R> patients <- data.table(
+    patient_id = 1:n_patients,
+    age = rnorm(n_patients, mean = 45, sd = 7),
+    female = rbinom(n_patients, size = 1, prob = .51)
+  )
R>
R> states <- data.table(
+    state_id = c(1, 2),
+    state_name = c("Stable", "Progression") # Non-death health states
+  )
R>
R> strategies <- data.frame(
+    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>
R> hesim_dat <- hesim_data(
+    strategies = strategies,
+    patients = patients,
+    states = states
+  )
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      age female
   1:          1 46.26366      1
   2:          2 50.49314      1
   3:          3 35.52785      1
   4:          4 58.88309      0
   5:          5 53.66930      0
  ---
 996:        996 54.60555      0
 997:        997 50.10654      0
 998:        998 45.02035      0
 999:        999 45.60573      1
1000:       1000 39.45426      0
```

```
$states
   state_id  state_name
1:        1      Stable
2:        2 Progression

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

Parameterization

In a STM, the disease model governs transitions between health states. For now, we assume that that patient level data with exact transition times is available so that a multi-state statistical model can be used to parameterize the disease model.

Multi-state data consists of one row for each possible transition from a given health state where only one transition is observed and all others are censored. We use the simulated dataset `onc3` from the **hesim** package.

```
R> onc3[patient_id %in% c(1, 2)]
```

```
          from          to strategy_name female      age patient_id
1:      Stable Progression         New 2      0 59.85813          1
2:      Stable       Death         New 2      0 59.85813          1
3: Progression       Death         New 2      0 59.85813          1
4:      Stable Progression         New 2      0 62.57282          2
5:      Stable       Death         New 2      0 62.57282          2
   time_start time_stop status transition_id strategy_id      time
1:   0.000000  2.420226      1             1           3  2.420226
2:   0.000000  2.420226      0             2           3  2.420226
3:   2.420226 14.620258      1             3           3 12.200032
4:   0.000000  7.497464      0             1           3  7.497464
5:   0.000000  7.497464      0             2           3  7.497464
```

Multi-state models can be fit in one of two ways. First, a joint survival model with interaction terms for different transition can be estimated, which is useful if there are constraints in the parameters across transitions, such as coefficients that are assumed equal across transitions. Here, we will take a second approach that is computationally more efficient and fit a separate model for each transition. We illustrate with parametric Weibull models but multiple distributions should be compared in a real application (Williams *et al.* 2017). The treatment effect is assumed to only influence transitions from stable disease since patients are assumed to switch to a 2L treatment after progression.

```
R> library("flexsurv")
R> n_trans <- max(tmat, na.rm = TRUE)
R> wei_fits <- vector(length = n_trans, mode = "list")
R> f <- as.formula(Surv(time, status) ~ factor(strategy_name) + female + age)
R> for (i in 1:length(wei_fits)){
```

```
+    if (i == 3) f <- update.formula(f, .~.-factor(strategy_name))
+    wei_fits[[i]] <- flexsurvreg(f, data = onc3,
+                                 subset = (transition_id == i),
+                                 dist = "weibull")
+  }
R> wei_fits <- flexsurvreg_list(wei_fits)
```

Utility and cost values are stored in `stateval_tbl` objects. Utility is assumed to equal 0.8 with stable disease (standard error (SE) = 0.02) and 0.6 (SE = 0.05) with progressed disease. Utility values are assumed to follow a beta distribution, which can be parameterized either using its shape parameters or via the mean and standard error (in which case the method of moments is used to derive the shape parameters).

```
R> utility_tbl <- stateval_tbl(
+    data.table(state_id = states$state_id,
+               mean = c(.8, .6),
+               se = c(0.02, .05)
+    ),
+    dist = "beta",
+    hesim_data = hesim_dat)
R> print(utility_tbl)

   state_id mean   se
1:        1  0.8 0.02
2:        2  0.6 0.05
```

Both medical and drug costs are considered. Medical costs are assumed to follow a gamma distribution which is often appropriate as they tend to be right skewed. Like utility, the distribution is characterized by the mean ($2,000 with stable disease, $9,500 with progressed disease) and standard error (assumed equal to the mean) and the method of moments is used to derive the underlying shape and scale parameters.

```
R> medcost_tbl <- stateval_tbl(
+    data.table(state_id = states$state_id,
+               mean = c(2000, 9500),
+               se = c(2000, 9500)
+    ),
+    dist = "gamma",
+    hesim_data = hesim_dat)
R> print(medcost_tbl)

   state_id mean   se
1:        1 2000 2000
2:        2 9500 9500
```

Drug costs are assumed to be fixed (i.e., measured without uncertainty). Costs are $2,000 with SOC, $12,000 with New 1, and $15,000 with New 2 when on 1L treatment. All patients

are assumed to switch to the same treatment at 2L after progression, which costs $1,500 for the first 3 months and $1,200 thereafter.

```
R> drugcost_tbl <- stateval_tbl(
+    drugcost_dt,
+    dist = "fixed",
+    hesim_data = hesim_dat
+  )
R> print(drugcost_tbl)
```

```
    strategy_id state_id time_id time_start time_stop   est
 1:           1        1       1       0.00      0.25  2000
 2:           1        1       2       0.25       Inf  2000
 3:           1        2       1       0.00      0.25  1500
 4:           1        2       2       0.25       Inf  1200
 5:           2        1       1       0.00      0.25 12000
 6:           2        1       2       0.25       Inf 12000
 7:           2        2       1       0.00      0.25  1500
 8:           2        2       2       0.25       Inf  1200
 9:           3        1       1       0.00      0.25 15000
10:           3        1       2       0.25       Inf 15000
11:           3        2       1       0.00      0.25  1500
12:           3        2       2       0.25       Inf  1200
```

Simulation

Before constructing the economic model, we specify the number of parameters samples that will be drawn for each submodel for the PSA.

```
R> n_samples <- 100
```

The input data for the transition model consists of one row for each treatment strategy and patient and can be easily generated from a `hesim_data` object with the `expand` function.

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

```
   strategy_id patient_id strategy_name soc new1 new2      age female
1:           1          1           SOC   1    0    0 46.26366      1
2:           1          2           SOC   1    0    0 50.49314      1
3:           1          3           SOC   1    0    0 35.52785      1
4:           1          4           SOC   1    0    0 58.88309      0
5:           1          5           SOC   1    0    0 53.66930      0
6:           1          6           SOC   1    0    0 53.40432      1
```

The transition model is then constructed as a function of the parameters (from the fitted Weibull models) and the input data. We must also specify the allowed transitions, the desired number of PSA samples (drawn from the multivariate normal distribution of the maximum likelihood estimate of the regression coefficients), the "clock" (recall that we used a clock reset approach when fitting the Weibull models), and the starting age of the patients. The starting age of the patients is useful because it allows us to specify a maximum age for patients during the simulation, ensuring that the simulated patients do not live unrealistically long lives.

```
R> transmod <- create_IndivCtstmTrans(wei_fits, transmod_data,
+                                     trans_mat = tmat, n = n_samples,
+                                     clock = "reset",
+                                     start_age = patients$age)
```

The utility and cost models can be easily constructed from the `stateval_tbl` objects with the `create_StateVals()` function, in which the (transformed) parameters are `tparams_mean` objects. The `hesim_data` object provides the information required to ensures that state values are constant within groups not specified within a particular `stateval_tbl`; for instance, since utility only varies by health state, it is assumed constant across patients, treatment strategies, and time. The models for each cost category are combined in a list.

```
R> # Utility
R> utilitymod <- create_StateVals(utility_tbl, n = n_samples)
R>
R> # Costs
R> drugcostmod <- create_StateVals(drugcost_tbl, n = n_samples,
+                                  time_reset = TRUE)
R> medcostmod <- create_StateVals(medcost_tbl, n = n_samples)
R> costmods <- list(Drug = drugcostmod,
+                   Medical = medcostmod)
```

The complete economic model is an constructed by combining the transition, utility, and cost submodels.

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

Once the economic model has been constructed, it is straightforward to simulate outcomes. A trajectory through the multi-state model is simulated with the `$sim_disease()` method with patients assumed to live to a maximum age of 100.

```
R> ictstm$sim_disease(max_age = 100)
R> head(ictstm$disprog_)

   sample strategy_id patient_id grp_id from to final time_start
1:      1           1          1      1    1  3     1   0.000000
```

```
2:      1           1          2      1    1  2     0   0.000000
3:      1           1          2      1    2  3     1   7.122132
4:      1           1          3      1    1  2     0   0.000000
5:      1           1          3      1    2  3     1   3.815690
6:      1           1          4      1    1  2     0   0.000000
   time_stop
1:  2.964441
2:  7.122132
3: 12.852926
4:  3.815690
5:  9.809671
6:  3.374204
```

State probabilities (averaged across patients in a given subgroup) can be constructed from the disease progression simulation output with the `$sim_stateprobs()` method. As shown in Figure 3, patients using the newer treatments remain in stable disease longer and have longer expected survival.

```
R> ictstm$sim_stateprobs(t = seq(0, 30 , 1/12))
R> head(ictstm$stateprobs_)
```

```
   sample strategy_id grp_id state_id          t  prob
1:      1           1      1        1 0.00000000 1.000
2:      1           1      1        1 0.08333333 1.000
3:      1           1      1        1 0.16666667 1.000
4:      1           1      1        1 0.25000000 1.000
5:      1           1      1        1 0.33333333 0.999
6:      1           1      1        1 0.41666667 0.998
```

QALYs and costs are simulated from the disease progression simulation output using the utility and cost submodels with the `$sim_qalys()` and `$sim_costs()` methods. Outcomes are simulated for each discount rate specified with the `dr` argument. By default, `$sim_qalys()` will return life-years (i.e., a utility value of 1 for each year of life) in addition to QALYs.

```
R> ictstm$sim_qalys(dr = c(0,.03))
R> head(ictstm$qalys_)
```

```
   sample strategy_id grp_id state_id dr    qalys      lys
1:      1           1      1        1  0 4.806160 6.165409
2:      1           1      1        2  0 2.865952 4.924144
3:      1           2      1        1  0 5.703347 7.316331
4:      1           2      1        2  0 2.930233 5.034588
5:      1           3      1        1  0 6.545572 8.396750
6:      1           3      1        2  0 2.882091 4.951873
```

```
R> ictstm$sim_costs(dr = .03)
R> head(ictstm$costs_)
```

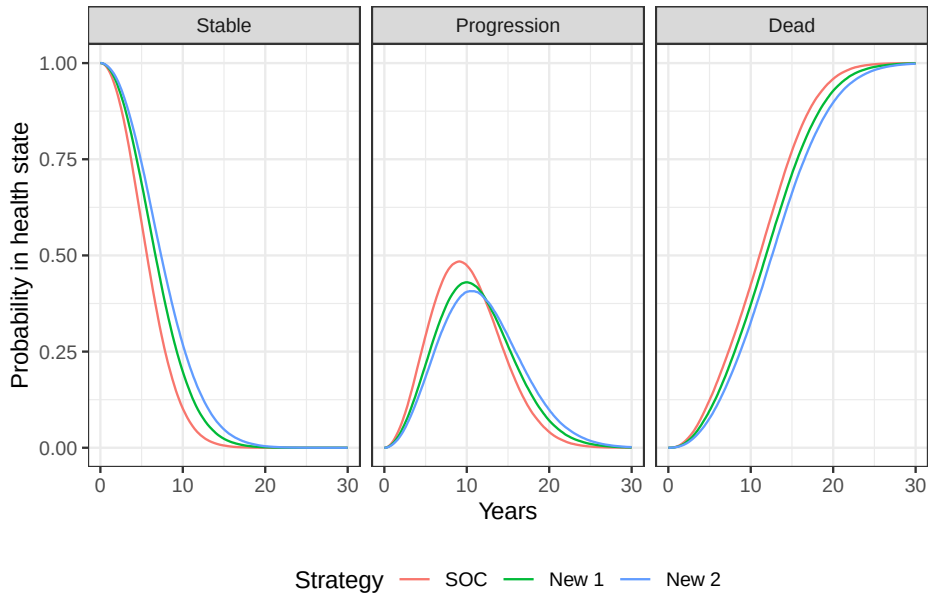


Figure 3: Mean simulated state probabilities from the probabilistic sensitivity analysis

```
   sample strategy_id grp_id state_id   dr category      costs
1:      1           1      1        1 0.03     Drug  11012.687
2:      1           1      1        2 0.03     Drug   4432.177
3:      1           2      1        1 0.03     Drug  76896.149
4:      1           2      1        2 0.03     Drug   4373.490
5:      1           3      1        1 0.03     Drug 108224.004
6:      1           3      1        2 0.03     Drug   4165.827
```

For a given subgroup, QALYs and costs are simulated for each PSA sample, treatment strategy, and health state. They are summarized (i.e., averaged) across PSA samples and treatment strategies (and subgroups) with the `$summarize()` method.

```
> ce_sim_ictstm <- ictstm$summarize()

Error: Cannot summarize costs without first simulating 'costs_' with '$sim_costs()'.

> head(ce_sim_ictstm$costs)

Error in head(ce_sim_ictstm$costs): object 'ce_sim_ictstm' not found
```

The output is a generic cost-effectiveness object (of class `ce`) that can be used to perform CEA. We demonstrate a formal CEA in Section 5. Here, we make a simple pairwise comparisons between each of the new interventions (New 2 and New 1) and the comparator (SOC), and compute the incremental cost-effectiveness ratio (ICER).

```
R> ce_sim_ictstm <- ictstm$summarize()
R> cea_pw_ictstm <- cea_pw(ce_sim_ictstm, comparator = 1,
```

```
+                          dr_qalys = .03, dr_costs = .03,
+                          k = seq(0, 25000, 500))
R> icer_tbl(cea_pw_ictstm, colnames = strategies$strategy_name,
+           k = 100000)

                 SOC New 1
Incremental QALYs "-" "0.53 (0.24, 0.83)"
Incremental costs "-" "62,718 (52,601, 74,091)"
Incremental NMB   "-" "-9,968 (-36,778, 16,020)"
ICER              "-" "118,898"
Conclusion        "-" "Not cost-effective"
                 New 2
Incremental QALYs "0.95 (0.51, 1.35)"
Incremental costs "90,778 (78,247, 105,749)"
Incremental NMB   "4,246 (-28,891, 35,486)"
ICER              "95,532"
Conclusion        "Cost-effective"
```

4.2. Partitioned survival model

```
R> hesim_dat$patients <- data.table(
+    patient_id = 1:4,
+    patient_wt = rep(1/4, 4),
+    age = c(55, 65, 55, 65),
+    female = c(1, 1, 0, 0)
+  )
```

Parameterization

```
R> surv_est_data <- as_pfs_os(onc3, patient_vars = c("patient_id", "female", "age",
+                                                    "strategy_name"))
R> surv_est_data[patient_id %in% c(1, 2)]

   patient_id female      age strategy_name pfs_time pfs_status
1:          1      0 59.85813         New 2 2.420226          1
2:          2      0 62.57282         New 2 7.497464          0
   os_status   os_time
1:         1 14.620258
2:         0  7.497464

R> fit_pfs_wei <- flexsurv::flexsurvreg(
+    Surv(pfs_time, pfs_status) ~ strategy_name + female,
+    data = surv_est_data,
+    dist = "weibull")
```

```
R>
R> fit_os_wei <- flexsurvreg(
+    Surv(os_time, os_status) ~ strategy_name + female,
+    data = surv_est_data,
+    dist = "weibull")
R>
R> psmfit_wei <- flexsurvreg_list(fit_pfs_wei, fit_os_wei)

R> attr(utility_tbl, "patients") <- hesim_dat$patients
R> attr(medcost_tbl, "patients") <- hesim_dat$patients

R> drugcost_tbl <- stateval_tbl(
+    data.table(strategy_id = strategies$strategy_id,
+               est = c(2000, 12000, 15000)),
+    dist = "fixed",
+    hesim_data = hesim_dat)
R> print(drugcost_tbl)

   strategy_id   est
1:           1  2000
2:           2 12000
3:           3 15000
```

Simulation

```
R> survmods_data <- expand(hesim_dat, by = c("strategies", "patients"))
R> head(survmods_data)

   strategy_id patient_id strategy_name soc new1 new2 patient_wt age
1:           1          1           SOC   1    0    0       0.25  55
2:           1          2           SOC   1    0    0       0.25  65
3:           1          3           SOC   1    0    0       0.25  55
4:           1          4           SOC   1    0    0       0.25  65
5:           2          1         New 1   0    1    0       0.25  55
6:           2          2         New 1   0    1    0       0.25  65
   female
1:      1
2:      1
3:      0
4:      0
5:      1
6:      1

R> survmods <- create_PsmCurves(psmfit_wei,
+                               input_data = survmods_data,
+                               n = n_samples,
+                               bootstrap = FALSE, est_data = surv_est_data)
```

```
R> utilitymod <- create_StateVals(utility_tbl, n = n_samples)
R> drugcostmod <- create_StateVals(drugcost_tbl, n = n_samples)
R> medcostmod <- create_StateVals(medcost_tbl, n = n_samples)
R> costmods <- list(Drug = drugcostmod, Medical = medcostmod)

R> psm <- Psm$new(survival_models = survmods,
+                 utility_model = utilitymod,
+                 cost_models = costmods)

R> times <- seq(0, 20, by = .1)
R> psm$sim_survival(t = times)
R> head(psm$survival_)

   sample strategy_id patient_id grp_id patient_wt curve   t
1:      1           1          1      1       0.25     1 0.0
2:      1           1          1      1       0.25     1 0.1
3:      1           1          1      1       0.25     1 0.2
4:      1           1          1      1       0.25     1 0.3
5:      1           1          1      1       0.25     1 0.4
6:      1           1          1      1       0.25     1 0.5
    survival
1: 1.0000000
2: 0.9997999
3: 0.9991202
4: 0.9979090
5: 0.9961371
6: 0.9937847


R> psm$sim_stateprobs()
R> psm$stateprobs_[sample == 1 & patient_id == 1 & state_id == 2 & t == 12]

   sample strategy_id patient_id grp_id patient_wt state_id  t
1:      1           1          1      1       0.25        2 12
2:      1           2          1      1       0.25        2 12
3:      1           3          1      1       0.25        2 12
       prob
1: 0.3253905
2: 0.4097814
3: 0.4114750

R> psm$sim_costs(dr = .03)
R> psm$sim_qalys(dr = .03)

R> ce_sim_psm <- psm$summarize()
R> cea_pw_psm <- cea_pw(ce_sim_psm, comparator = 1,
```

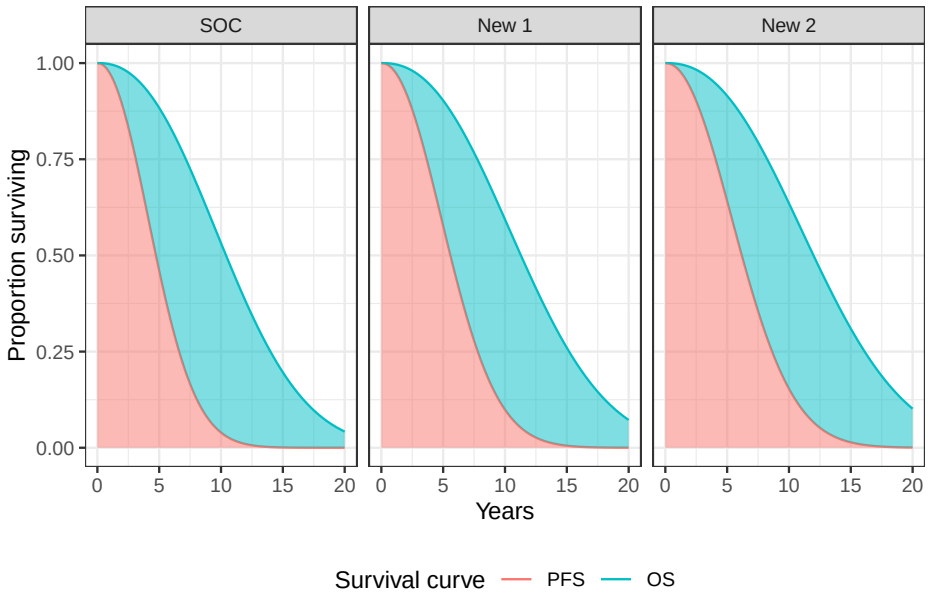


Figure 4: Mean progression free survival and overall survival, weighted by weight of each patient in the target population. Solid and dotted lines represent means and 95% confidence intervals from the probabilistic sensitivity analysis, respectively. weighted

```
+                       dr_qalys = .03, dr_costs = .03,
+                       k = seq(0, 25000, 500))
R> icer_tbl(cea_pw_psm, colnames = strategies$strategy_name,
+           k = 100000)

                  SOC New 1
Incremental QALYs "-" "0.48 (0.26, 0.68)"
Incremental costs "-" "96,085 (85,520, 102,554)"
Incremental NMB   "-" "-47,654 (-67,645, -31,208)"
ICER              "-" "198,395"
Conclusion        "-" "Not cost-effective"
                  New 2
Incremental QALYs "0.83 (0.60, 1.07)"
Incremental costs "130,960 (114,269, 140,997)"
Incremental NMB   "-47,850 (-66,228, -29,855)"
ICER              "157,575"
Conclusion        "Not cost-effective"
```

4.3. Cohort disctete time state transition model

Parameterization

```
R> library("msm")
```

```
R> qinit <- matrix(0, nrow = 3, ncol = 3)
R> qinit[1, 2] <-  0.28
R> qinit[1, 3] <-  0.013
R> qinit[2, 3] <-  0.10
R> msm_fit <- msm(state_id ~ time, subject = patient_id,
+                 data = onc3p[strategy_name == "SOC"],
+                 exacttimes = FALSE,
+                 covariates = ~ age + female,
+                 qmatrix = qinit, gen.inits = FALSE)

R> # msm with exact transition times
R> tmat2 <- tmat; tmat2[is.na(tmat2)] <- 0
R> msm_efit <- msm(state_id ~ time, subject = patient_id,
+                  data = onc3p[strategy_name == "SOC"],
+                  qmatrix = tmat2, exacttimes = TRUE)
R>
R> # flexsurv with exponential model
R> library("flexsurv")
R> exp_fits <- vector(length = n_trans, mode = "list")
R> for (i in 1:length(wei_fits)){
+    exp_fits[[i]] <- flexsurvreg(Surv(time, status) ~ 1, data = onc3,
+                                 subset = (transition_id == i &
+                                           strategy_name == "SOC"),
+                                 dist = "exponential")
+  }
R>
R> # Compare
R> msm_efit
R> exp_fits[[1]]$res
R> exp_fits[[2]]$res
R> exp_fits[[3]]$res

R> params2 <- list(
+    lrr_12_est = c(soc = log(1), new1 = log(.80), new2 = log(.71)),
+    lrr_12_se = c(soc = 0, new1 = .03, new2 = .04),
+    lrr_13_est = c(soc = log(1), new1 = log(.90), new2 = log(.85)),
+    lrr_13_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)
+  )
R>
R> params2_def <- define_rng({
+    list(
+      rr_12 = lognormal_rng(lrr_12_est, lrr_12_se),
+      rr_13 = lognormal_rng(lrr_13_est, lrr_13_se),
+      u = beta_rng(mean = u_mean, sd = u_se)
+    )}, n = n_samples)
```

```
R>
R> params2_rng <- eval_rng(params2_def, params2)

R> transmod_data <- survmods_data
R> qmat_soc <- qmatrix(msm_fit, newdata = transmod_data[strategy_name == "SOC"],
+                      uncertainty = "normal", n = n_samples)
R> qmat_soc[,, 1]

           [,1]       [,2]      [,3]
[1,] -0.3991107  0.3553603 0.0437504
[2,]  0.0000000 -0.1059250 0.1059250
[3,]  0.0000000  0.0000000 0.0000000

R> cycle_len <- 1/4
R> pmat_soc <- expmat(qmat_soc, t = cycle_len)
R> pmat_soc[,, 1]

          [,1]       [,2]       [,3]
[1,] 0.9050386 0.08342368 0.01153771
[2,] 0.0000000 0.97386631 0.02613369
[3,] 0.0000000 0.00000000 1.00000000

R> tpmat_id <- tpmatrix_id(transmod_data, n_samples)
R> head(tpmat_id)

   sample strategy_id patient_id patient_wt
1:      1           1          1       0.25
2:      1           1          2       0.25
3:      1           1          3       0.25
4:      1           1          4       0.25
5:      1           2          1       0.25
6:      1           2          2       0.25

R> xbeta <- function(x, beta) c(x %*% t(beta))

R> x_rr <- as.matrix(transmod_data[, .(soc, new1, new2)])
R> rr <- cbind(xbeta(x_rr, params2_rng$rr_12),
+              xbeta(x_rr, params2_rng$rr_13))

R> pmat <- apply_rr(pmat_soc, rr = rr,
+                   index = list(c(1, 2), c(1, 3)))

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

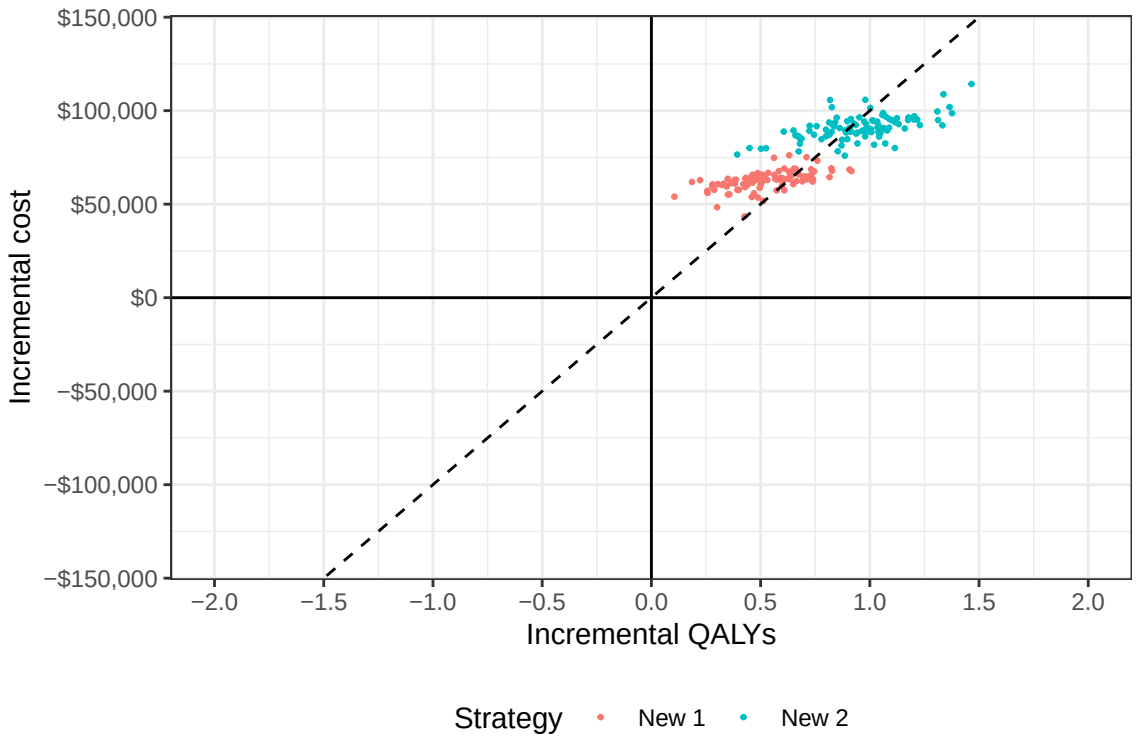
Simulation

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

R> cdtstm <- CohortDtstm$new(trans_model = transmod,
+                            utility_model = psm$utility_model,
+                            cost_models = psm$cost_models)

R> cdtstm$sim_stateprobs(n_cycles = 30/cycle_len)

R> cdtstm$sim_qalys(dr = .03)
R> cdtstm$sim_costs(dr = .03)

R> ce_sim_cdtstm <- cdtstm$summarize()
R> cea_pw_cdtstm <- cea_pw(ce_sim_cdtstm, comparator = 1,
+                          dr_costs = .03, dr_qalys = .03)
R> icer_tbl(cea_pw_cdtstm, output = "data.table")

   strategy_id grp_id             iqalys                     icosts
1:           2      1 0.36 (-0.38, 0.85)  88,958 (70,153, 101,615)
2:           3      1 0.57 (-0.35, 1.11) 118,415 (88,231, 133,621)
                          inmb    icer         conclusion
1:  -71,131 (-93,219, -51,890) 249,502 Not cost-effective
2: -90,057 (-110,127, -70,101) 208,787 Not cost-effective
```

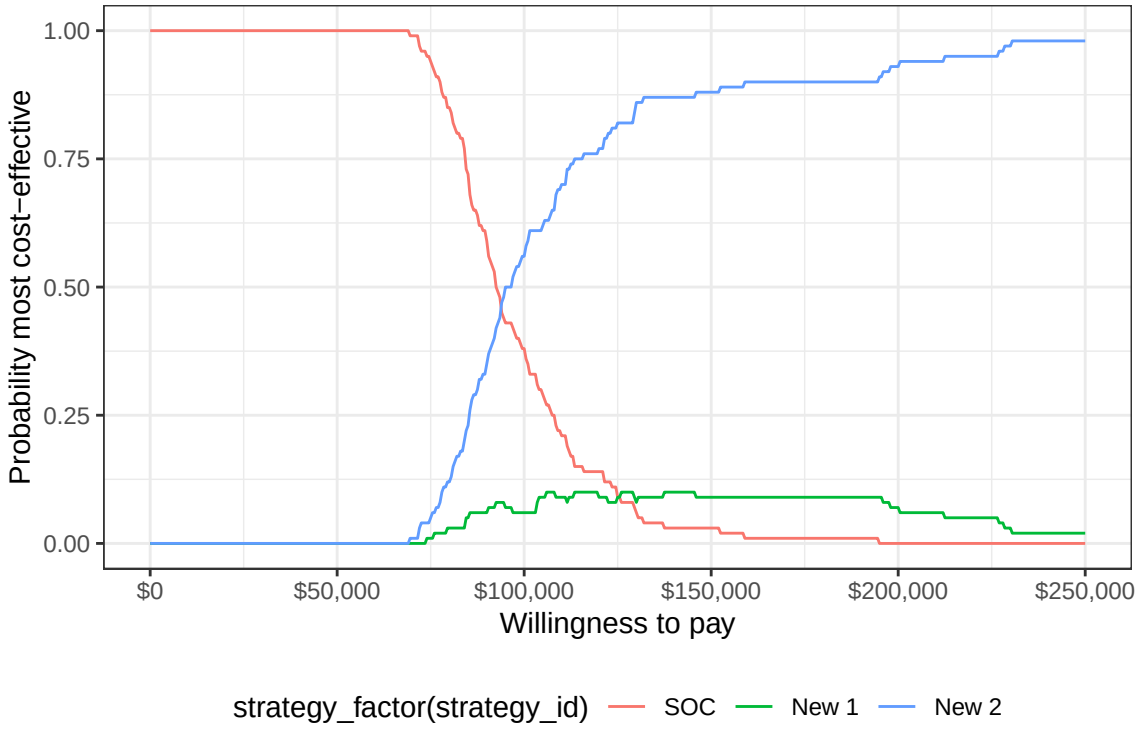
5. Cost-effectiveness analysis

```
R> wtp <- seq(0, 250000, 500) # Willingness to pay per QALY
R> cea_ictstm <- cea(ce_sim_ictstm, dr_costs = .03, dr_qalys = .03, k = wtp)

R> library("scales")
R> strategy_factor <- function (x) {
+   factor(x, levels = 1:3, labels = c("SOC", "New 1", "New 2"))
+ }
R>
R> xlim <- ceiling(max(cea_pw_ictstm$delta[, ie]) * 1.2)
R> ylim <- max(cea_pw_ictstm$delta[, ic]) * 1.2
R> ggplot(cea_pw_ictstm$delta,
+         aes(x = ie, y = ic, col = strategy_factor(strategy_id))) +
+   geom_jitter(size = .5)  +
+   xlab("Incremental QALYs") +
+   ylab("Incremental cost") +
+   scale_y_continuous(limits = c(-ylim, ylim),
+                      labels = scales::dollar) +
+   scale_x_continuous(limits = c(-xlim, xlim), breaks = seq(-2, 2, .5)) +
+   theme(legend.position = "bottom") +
```

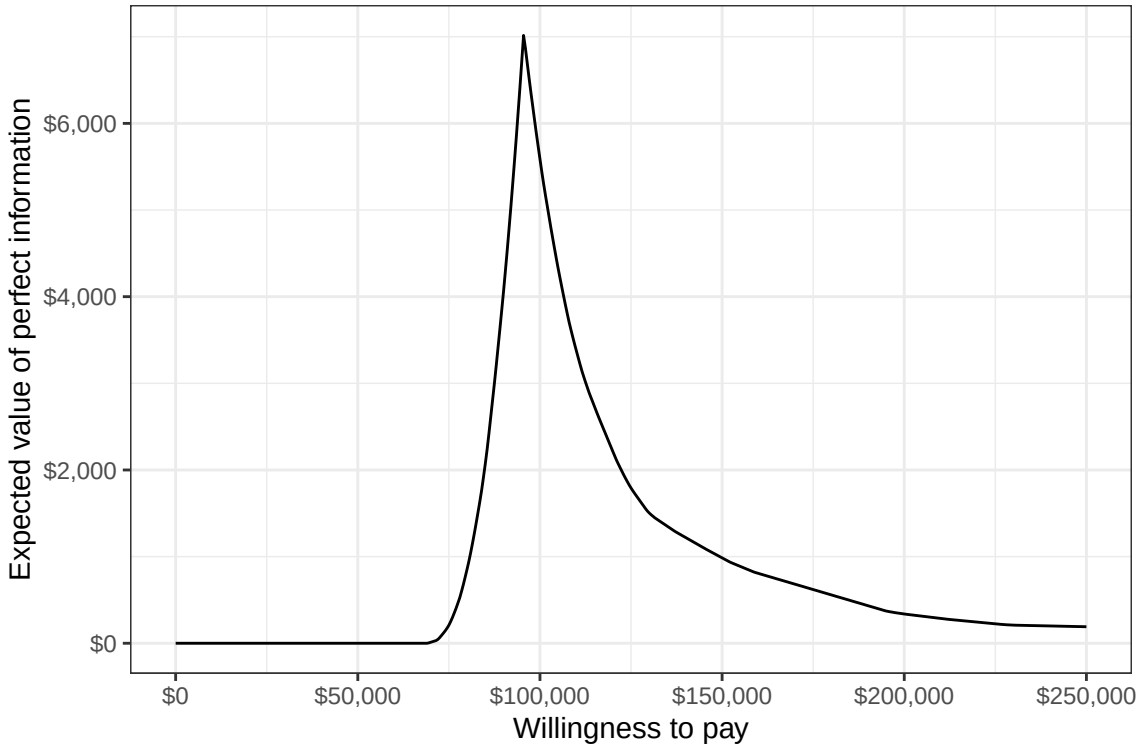
```
+     scale_colour_discrete(name = "Strategy") +
+     geom_abline(slope = 100000, linetype = "dashed") +
+     geom_hline(yintercept = 0) +
+     geom_vline(xintercept = 0)
```




```
R> ggplot(cea_ictstm$mce,
+         aes(x = k, y = prob, col = strategy_factor(strategy_id))) +
+     geom_line() +
+     xlab("Willingness to pay") +
+     ylab("Probability most cost-effective") +
+     scale_x_continuous(breaks = seq(0, max(wtp), length.out = 6),
+                        label = scales::dollar) +
+     theme(legend.position = "bottom")
```




```
R> ggplot(cea_ictstm$evpi, aes(x = k, y = evpi)) +
+    geom_line()  +
+    xlab("Willingness to pay") +
+    ylab("Expected value of perfect information") +
+    scale_x_continuous(breaks = seq(0, max(wtp), length.out = 6),
+                       label = scales::dollar) +
+    scale_y_continuous(label = scales::dollar) +
+    theme(legend.position = "bottom")
```



6. Discussion

7. 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**.

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