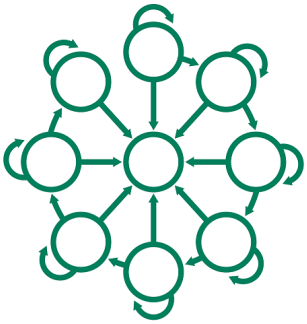


DECISION MODELLING FOR HEALTH ECONOMIC EVALUATION

Advanced Markov Models

Advanced Course Module 1

Nichola Naylor & Jack Williams

May/June 2021 Course

Overview

In this exercise we will build a Markov model in R. Of particular interest is the use of time dependent transition functions to represent probabilities. The exercise will get you to construct two types of time varying transition probabilities: (i) time-varying transitions from a life-table, and (ii) time-varying transitions estimated from a parametric survival model.

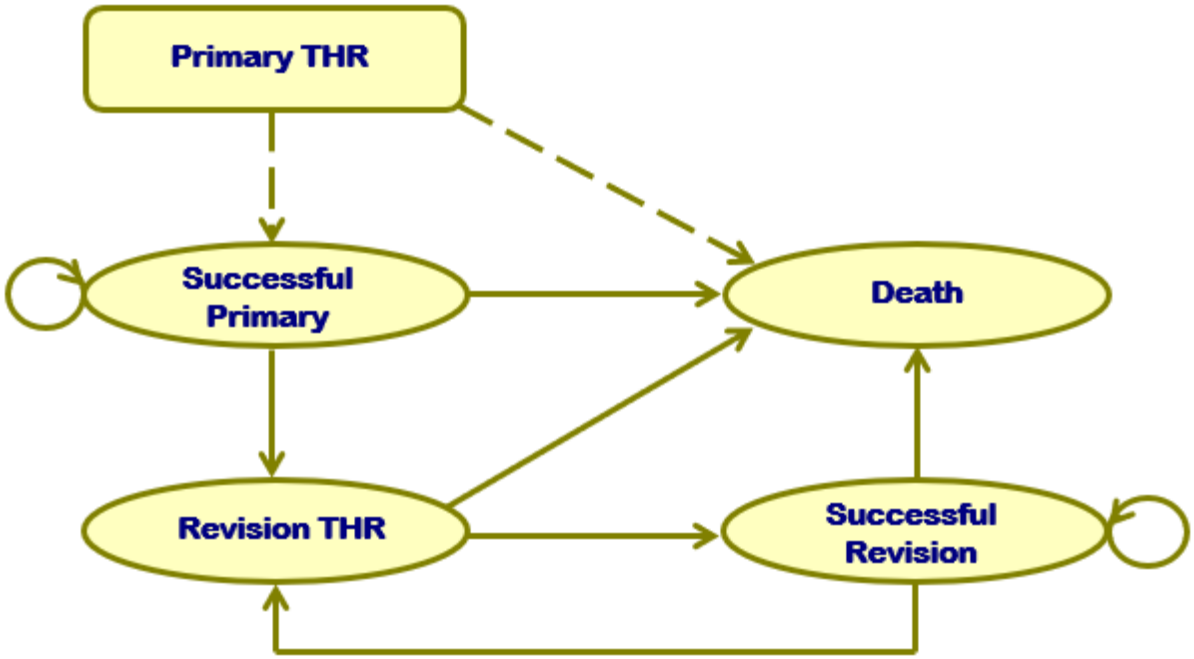
In this exercise we will be focusing on:

1. Preparing parameters
2. Parametric time-dependent transitions from a survival analysis
3. Life-table transitions for background mortality
4. Building a Markov model for the standard prosthesis
5. Adapting the model for a new prosthesis
6. Estimating cost-effectiveness (deterministically)

Before beginning the exercise take a moment to familiarise yourself with the R template for the exercise, 'A1.3.1_Advanced_Markov_Modelling_Template.R' and the figure below. Figure A1.3.1a contains the overall structure of the model you are about to build and it will be worth you spending a little time familiarising yourself with this figure. It contains a state transition diagram of the four-state Markov model you will be building and contains a key to the different parameters of the model, which are seperated into three categories: transition probabilities, state costs and state utilities.

The model itself is constructed across a number of sections with separate sections for the parameters of the model, for the survival analysis used to estimate prosthesis failure rates, for the life tables used for mortality rates, for each of the Markov model arms and the final analysis. Of course, there are many ways to structure models – much being down to personal preference. However, we hope that you will find that our chosen way of presenting the model does not deviate too much from your own preferences!

Figure A1.3.1a



To work through this exercise, you will need to have the following csv files available within the same working-directory folder as the template and solution files:

- "hazardfunction.csv"
- "life-tables.csv"

Step by step guide

Open the corresponding 'A1.3.2_Advanced_Markov_Modelling_Template.R' file and start at the 'Parameters' section of the script.

(1) Preparing parameters

The aim of this section of the exercise is to prepare the 'Parameters' section to facilitate the use of parameter names throughout the rest of the modelling exercise.

On opening the section you will see that some of the work has been done for you, including entry of patient characteristics (age and sex), the inclusion of stated discounting rates for the exercise, and (further down the script) the cost of the two prostheses that we are seeking to evaluate. This section will guide you through the completion of the parameter information for the constant transition probabilities (taken from the model figure), the state costs and the state utilities.

i) We will start with the three transition probabilities that will be assumed to be constant in the model: the operative mortality rates and the re-revision rate parameters, which are named as "`tp.`" variables. We will assume that the operative mortality is 2% for both primary and revision procedures and that the re-revision rate is 4% per annum. Enter these values now (working on a 0 to 1 scale to represent percentage values).

ii) Now we deal with the costs in the model. Note that the costs of the standard and new prostheses (named `c.SP0` and `c.NP1`) have already been entered for you. That leaves the costs for the states of the model. The state cost for the primary hospital procedure is assumed to be the same irrespective of the implant – we therefore leave this out (since it is received by both arms and nets out of the analysis, but would be important if you were interested in total costs for each arm). We also assume for the purposes of this model that there is no ongoing cost to the health service if people have a successful implant in place. As such, the health state cost can be assigned zero in the model. Therefore the only cost to consider at this point is the cost of the revision procedure itself. This has been estimated as £5,294 and this should be entered into `c.revision`. Afterwards, create a vector (named `state.costs`) that combines all of these costs, and remember that the order is important and should match the order of states that you will use in the state transition matrix (and should match the order of states listed in the `state.names` vector above).

iii) Create two effect vectors; one for the life year effects of each health state and one for quality-adjusted life years (hint: these vectors also need to be the same length as the numer of health states - 5 - and have the same order of corresponding health states as `state.names`). For the life years vector, the values will consist of 0 or 1, since the cycle length is one year and therefore one cycle alive equals one life year). The quality of life utilities of a year spent in each of the model health states has been estimated to be 0.85, 0.3 and 0.75 for the Successful primary, Revision, and Successful revision health states respectively. The utility value for those in the actual 'Primary' surgery state are not counted since these individuals do not remain in this state after the first cycle. You can therefore assign a value of 0 quality of life to this state. Enter the utility values into the respective `u.` variables. Then pull this information together by creating a vector of these utility values, and remember this should match the same state order for costs as previously defined.

You should now have the following vectors defined as follows:

```
print(seed)
```

```
## [1] 1 0 0 0 0
```

```
print(state.costs)
```

```
## [1]    0    0 5294    0    0
```

```
print(state.lys)
```

```
## [1] 1 1 1 1 0
```

```
print(state.utilities)
```

```
## [1] 0.00 0.85 0.30 0.75 0.00
```

(2) **Parametric time-dependent transitions from a survival analysis**

If you go to the 'Hazard function & Associated Parameters' section you will see the importing of an output of a regression analysis on prosthesis failure. After importing the csv file `hazardfunction.csv` type in `View(hazards)` into the console to see the hazard ratios. A parametric Weibull model was fitted to patient level survival time data to estimate this function. The regression model shows that prosthesis survival time is significantly related to age, sex, and type of prosthesis (new versus standard). In addition, the significance of the gamma parameter indicates that there is an important time dependency to the risk of failure – which increases over time.

Note that the estimation of this regression model was undertaken on the log hazard scale. We therefore have to exponentiate the results to get the actual hazard rate. The exponents of the individual coefficients are interpreted as hazard ratios (column "coefficient" in hazards data.frame). For example, the new prosthesis (`NP1`) has a hazard ratio of 0.26 indicating that it has a much lower hazard compared to the standard prosthesis.

Take a moment to understand this survival model. Remember that these values are on the log hazard scale. If you are having trouble with the appropriate interpretation then do post any questions in the discussion board - we are about to implement this model and you may struggle if you are not clear on the interpretation from the start.R

i) To start with, we need to generate a link from the "Parameters" section to the corresponding results of the survival analysis from the "`hazards`" data.frame for the constant, age and male coefficients. Fill in the remaining "`r.`" variable definitions within the template file.

ii) We now want to calculate the lambda value of Weibull distribution (this, together with the gamma parameter - which we've defined for you - enables us to specify the baseline hazard). In estimating a standard survival analysis, the linear sum of the coefficients multiplied by the explanatory variables (e.g. coefficient(age)x65yearsold + coefficient(male_dummy)x1) is the log of the lambda. Therefore to get the value of log lambda, multiply the coefficients of age and sex by the age and sex characteristics and add together, not forgetting to also add the constant term. Using this information, define `lambda`.

iii) Exponentiate the NP1 coefficient to get the relative risk (RR) of revision for new prosthesis 1 compared to standard.

Your gamma value should be equal to **1.4536782** , your lambda value should be equal to **0.000455983** and your relative risk of revision for new prosthesis 1 compared to standard **0.2606768**. *Note: values here are rounded.*

(3) Life-table transitions for background mortality

We can still employ time dependent transitions in other model states providing the time dependency relates to time in the model rather than time in the state itself. This is the case for a background mortality rate, for example, which depends on the age of a subject rather than which state of the model they are in. In this section we will illustrate the use of a time dependent transition to model background mortality in the model from a life-table.

Go to the 'Life tables' section and familiarise yourself with the contents. View the imported lifetable data.frame, containing figures on age-sex specific mortality rates taken from a standard life-table, published as deaths per thousand per year. These are converted to the corresponding annual probabilities. Notice the`Index` column – the need for this will become apparent.

```
##           Age Index   Males  Female
## 1       35-44    35 0.00151 0.00099
## 2       45-54    45 0.00393 0.00260
## 3       55-64    55 0.01090 0.00670
## 4       65-74    65 0.03160 0.01930
## 5       75-84    75 0.08010 0.05350
## 6 85 and over    85 0.18790 0.15480
```

i) Create a vector ('`cycle.v`') that gives the number of model cycles from 1 to 60. Then create another vector ('`current.age`') which combines the initial age (`age`) of the cohort with the number of annual cycles, to provide the age of the cohort in each cycle of the model.

ii) Next, you can run the `findInterval()` function, which performs a lookup. The first argument is the current age values in each model cycle, and the second argument is the position of this age in the Index column of the 'life.table' data frame. The output should match the age in each model cycle to the appropriate position in `life.table$Index`, i.e. the row in the life.table data.frame:

Note, for those of you struggling to understand how this function works, you may find it easier to run the function using two simple vectors to see how it finds the position of the first vector in the second, for example:

```
findInterval(c(1:10), c(1,2,3,6,10))
```

iii) Once you have the interval values, you can use the positions within the Index column of the life.table, to select the corrent risk of death from the same table. For males (column 3), we can do this with th following code `life.table[interval,3]`, and can do the same for females by selecting column 4. We have provided the code for you, but we would recommend that you check the code to ensure that you understand how this works.

Note: There are many ways in which this type of matching or lookup can be performed in R. For those who are interested in learning alternatives, another method to perform this is using the 'data.table' package, and using `setkey()` and `roll()` functions, which gives the same output. However we have focused on using the base R functions wherever possible for this course.

(4) Building a Markov model for the standard prosthesis

In preparation for building the Markov model we are going to specify the time dependent transition probability for prosthesis failure after initial surgery. Recall that the cumulative hazard rate for the Weibull is given by:

$$H(t) = \lambda t^{y}$$

and that the time dependent transition probability (for a yearly cycle length and time measured in years) is given by:

$$tp(t) = 1 - exp\{H(t-1) - H(t)\}$$

$$tp(t) = 1 - exp\{\lambda(t-1)^{\gamma} - \lambda t^{\gamma}\}$$

$$tp(t) = 1 - exp\{\lambda[(t-1)^{\gamma} - t^{\gamma}]\}$$

Where tp(t) = the transition probability at time t, λ = lambda and γ = gamma.

i) Go to the 'Standard' section of the template file. Use the above formula to calculate the time-dependent transition probability for the standard prosthesis (referred to in the script as `revision.risk`) using the cycle number for the time vector (`cycle.v`) and the lambda and gamma parameters already defined in this exercise. Do the same for the NP1 treatment arm. *Hint: cycle.v can be used as t directly within the formula. Additionaly, for NP1, remember that the relative risk reduction is affecting the lambda value.*

ii) Now combine age and sex dependent risk of death (from `death.risk`), and age dependent risk of revision (which we have just defined for SP0 and NP1), into one data.frame (`tdtps`). Based on the column numbers we set a column key which switches between pulling risk of death from the "Men" or "Women" values.

For example, setting the column key to 3 means we are selecting column from the data.table (first 6 rows displayed below), i.e. selecting the Male values for risk of death.

```
head(tdtps)
```

```
##   age  males females revision.risk.sp0 revision.risk.np1
## 1  61 0.0109  0.0067      0.0004558791      0.0001188571
## 2  62 0.0109  0.0067      0.0007926651      0.0002066900
## 3  63 0.0109  0.0067      0.0010023327      0.0002613817
## 4  64 0.0109  0.0067      0.0011684956      0.0003047313
## 5  65 0.0316  0.0193      0.0013099613      0.0003416420
## 6  66 0.0316  0.0193      0.0014349705      0.0003742621
```

What you have just implemented is a relatively sophisticated use of survival analysis to model a time dependent transition within a Markov model. Note that it only works because we will be using this time dependent transition from the initial state of the model – i.e. we know exactly how long subjects have spent in the initial model state. Take a moment to make sure you understand what you have just implemented and the limitations on its use in a Markov framework.

You have now implemented all three key types of transition probability: constant, time dependent (function) and time dependent (tabular).

Having specified the time dependent transition parameters we can now proceed to construct the Markov model. To begin with we need to create an array of time dependent state transition matrices. Remember the course reader has an overview of data structures, including arrays, in R.

i) Start by creating an empty array of an empty array of dimensions (number of states, number of states, number of cycles). *[Hint: putting 0s instead of NAs allows for 0 values to be pre-filled for where transitions can't occur].* It is recommended to provide dimension names (where the transition matrix column and row names match the state names). See below to give an idea of the structure of what this should look like (with only cycle 1 and 2 results displayed below, which we will now attempt to fill!).

```
tm.SP0[,,1:2]
```

```
## , , 1
##
##               P-THR successP-THR        R-THR successR-THR  Death
## P-THR             0    0.9800000 0.0000000000       0.0000 0.0200
## successP-THR      0    0.9928441 0.0004558791       0.0000 0.0067
## R-THR             0    0.0000000 0.0000000000       0.9733 0.0267
## successR-THR      0    0.0000000 0.0400000000       0.9533 0.0067
## Death             0    0.0000000 0.0000000000       0.0000 1.0000
##
## , , 2
##
##               P-THR successP-THR        R-THR successR-THR  Death
## P-THR             0    0.9800000 0.0000000000       0.0000 0.0200
## successP-THR      0    0.9925073 0.0007926651       0.0000 0.0067
## R-THR             0    0.0000000 0.0000000000       0.9733 0.0267
## successR-THR      0    0.0000000 0.0400000000       0.9533 0.0067
## Death             0    0.0000000 0.0000000000       0.0000 1.0000
```

The structure of a for loop has been created for you. You want this loop to create the correct transition matrix for each corresponding cycle, from cycle 1 to cycle 60 (hence why there are 60 loops). The start of the loop has been completed for you, with the correct transitions from the 'P-THR' and 'successP-THR' health states completed. Take note of how the appropriate time dependent transitions are selected. For background mortality, this is taken from the `death.risk` data frame, selecting the appropriate value using the cycle number - indicated by 'i', and the column key defined above (to select male or female risk value). Similarly for the revision risk, the 'i' element is selected from the `revision.risk.sp0` vector, to assign the correct risk to the appropriate transition matrix for that cycle, accounting for patient age for that cycle). *Remember we are still only interested in the SP0 arm at the moment.*

ii) Refamiliarise yourself with the diagram of the model, and then complete the remaining transitions for each cycle, from the 'R-THR' and 'successR-THR' health states. *Note: a revision after R-THR is a re-revision, therefore we need to use the associated transition probability of re-revision, which you defined earlier in this exercise.*

Once you have completed the array with all transition matrices, we then are concerned with generating the Markov trace: that is showing the numbers of patients that are in any one state at any one time. We've set up the initial trace matrix (`trace.SP0`) for you.

i) First create the first cycle trace by multiplying the seed population (cycle0, i.e. `trace.SP0[1,]`) by the transition matrix for cycle1 using `%*%`. *Hint: If you get a non-comformable error, make sure you're only selecting the relevant matrix from `tm.SP0`. Refer back to the reader for some more information on matrix multiplication in R*

ii) Create a for loop for cycle 2 to cycle 60 that multiplies populations within cycles with the correct transition matrix, we've outlined the structure for you. *[Hint: you will have to "look back" if starting from cycle2].*

iii) Take a look at the trace that you've constructed, check the sum across rows must always equal the size of the original cohort, which in our case is 1.

(5) **Calculating costs and effects**

Now that we have the Markov trace we can calculate the cost and effects for each cycle of the model.

i) Calculate the total life years (LYs) for the standard treatment arm using the trace outputs and the life year vector. By doing this without quality adjustment or discounting, this column can be used to calculate life-expectancy (which is often useful – but not particularly in this example). We have provided the code for you here for calculating the LYs for each cycle, so that you can see how the life year vector is multiplied by the trace - using `%*%` for matrix multiplication. Now calculate the total number of LYs over the whole modelled period. *(Hint: we essentially want the sum of the 1st (and only) column of lys.SP0)*

ii) Now calculate quality adjusted life years (QALYs), defining `QALYs.SP0` and `undiscounted.QALYs.SP0` using the same methods as above, but this time with the `state.utilities` vector.

However, this is undiscounted. Therefore, we need to apply the standard discount formula, you can do this by creating a vector (`discount.factor.o`) which specifies the discount factor that needs to be applied for each cycle. The discount formula is given below, where r is the discount rate, and t is time (in years):

$$1/(1+r)^t$$

Note that for the above, if you provide t as a vector, then the formula will return a vector *(Hint: use the 'cycle.v' vector, since each cycle is one year).* Once you have completed this, the vector should look like this:

```
##  [1] 0.9852217 0.9706617 0.9563170 0.9421842 0.9282603 0.9145422 0.9010268
##  [8] 0.8877111 0.8745922 0.8616672 0.8489332 0.8363874 0.8240270 0.8118493
## [15] 0.7998515 0.7880310 0.7763853 0.7649116 0.7536075 0.7424704 0.7314979
## [22] 0.7206876 0.7100371 0.6995439 0.6892058 0.6790205 0.6689857 0.6590992
## [29] 0.6493589 0.6397624 0.6303078 0.6209929 0.6118157 0.6027741 0.5938661
## [36] 0.5850897 0.5764431 0.5679242 0.5595313 0.5512623 0.5431156 0.5350892
## [43] 0.5271815 0.5193907 0.5117149 0.5041527 0.4967021 0.4893617 0.4821297
## [50] 0.4750047 0.4679849 0.4610689 0.4542550 0.4475419 0.4409280 0.4344118
## [57] 0.4279919 0.4216669 0.4154354 0.4092960
```

Then multiply `discount.factor.o` with `QALYs.SP0` to get the discounted QALYs for the standard treatment arm.

iii) Calculate the costs in a similar manner (but this time with the state cost vector). Don't forget to incorporate the cost discount rate (which is different to the outcome discount rate).

iv) We also now want to include the cost of the original prosthesis, there are many ways to do this, but since it is a one off cost occurring in `cycle0`, here we can add on at the end when defining `disc.cost.SP0` (and `undisc.cost.SP0` for completeness).

Now that you have the results for the standard prosthesis, perform the same processes for the NP1 treatment arm within the 'NP1' section. More sections have been made left blank for you to fill out (e.g. generating the transition matrices), but you can use the code above to help you. *Hint: run the chuncks of code as you go to spot major errors early on.*

vi) Finally, in the 'Analysis' section create an output matrix storing the differences in treatment costs and effects, and incremental cost-effectiveness ratio, filling in the formulae needed to calculate these results in `output`. *Hint: use the discounted values for these results.*

You should have an incremental cost-effectiveness ratio of **£2198** per QALY gained.

Congratulations! Your Markov model is now complete.

Try re-running the model for different sex and age groups (between 40 and 90 years), and/or different discount rates, and see differences in the results produced. *Hint: to do this you can go back to the top of your model R script and adapt the* `age` *and* `male` *assignments and rerun.*