Practical Health State Transition Model 2A - Solutions

X. Pouwels

2021-09-02

```
rm(list  = ls()) # clear environment
options(scipen = 999) # Disable scientific notations
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.5     v dplyr   1.0.3
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(knitr)
```

Questions and answers

1. As you can see both strategies (prescribing aspirin or not) have similar diagrams. However, there will be differences in the values of the parameters applied in these strategies. First, we will focus on the course of patients not using aspirin. Have a look at the parameters which are already defined in the `Assignment_start.R` script.

```
# Setting parameters
n_cycles <- 10 # number of cycles
r_d_effects <- 0.015 # annual discount rate, health effects
r_d_costs <- 0.04 # annual discount rate, costs
v_names_hs <- c("Well", "Post-minor_stroke", "Post-major_stroke", "Post-MI",
                "Death_stroke", "Death_MI", "Death_other") # vector of names of health states
n_hs <- length(v_names_hs) # number of health states
n_ind <- 100000 # number of individuals to simulate
v_start_hs <- c(n_ind, 0, 0, 0, 0, 0, 0) # vector of starting position in the model

# Input parameters

## Rates & probabilities
r_fatal_mi  <- 0.25 # rate fatal MI
```

```
r_fatal_stroke  <- 0.3 # rate fatal stroke

## Treatment effectiveness
eff_mi  <- 0.6 # Treatment effectiveness of Aspirin on the probability of experiencing a MI
eff_stroke  <- 1.2 # Treatment effectiveness of Aspirin on the probability of experiencing a stroke

## Utility values
u_healthy   <- 1 # utility value health state: Well
u_post_mi   <- 0.85 # utility value health state: Post-MI
u_post_minor_stroke <- 0.75 # utility value health state: Post-minor stroke
u_post_major_stroke <- 0.5 # utility value health state: Post-major stroke
u_aspirin_use   <- 0.999 # utility value health state: Well when using aspiring

## Costs
c_aspirin_use   <- 100 # yearly costs of using aspiring
c_post_mi   <- 8000 # yearly costs after having experienced a NON-FATAL MI
c_post_minor_stroke <- 2000 # yearly costs after having experienced a NON-FATAL minor stroke
c_post_major_stroke <- 20000 # yearly costs after having experienced a NON_FATAL major stroke
```

1.a. In making a health economic model we often have to combine evidence from different sources of literature. Assume that there was evidence that 235 out of 1,000 patients **who had a stroke and survived** moved into a post major stroke state (the remaining patients experienced a minor stroke). Based on this evidence, please define the parameter `p_minor_stroke` (probability of transiting to the "post-minor stroke" health state **after having experienced a NON-FATAL stroke** and `p_major_stroke` (probability of transiting to the "post-major stroke" health state **after having experienced a NON-FATAL stroke**.
Answer: `p_minor_stroke` and `p_major_stroke` should be calculated as follows:

```
p_post_major_stroke <- 235 / 1000 # probability to transit to "Post-major stroke" after a NON-FATAL str
p_post_minor_stroke <- 1 - p_post_major_stroke # probability to transit to "Post-minor stroke" after a
```

1.b. From other sources, you found that the incidence rates for myocardial infarction (MI), stroke, and death from other causes were respectively **400, 50, and 650 per 100,000 person-years**. Calculate the yearly incidence rates of MI, stroke, and death from other causes than MI and stroke in the placeholder of parameters `r_inc_mi`, `r_inc_stroke`, and `r_mort`.
Answer: `r_inc_mi`, `r_inc_stroke`, and `r_mort` should be calculated as follows:

```
r_inc_mi <- 400 / 100000 # TO CALCULATE (1.b.): yearly incidence rate MI
r_inc_stroke <- 50 / 100000 # TO CALCULATE (1.b.): yearly incidence rate stroke
r_mort <- 650 / 100000 # TO CALCULATE (1.b.): yearly rate of death
```

2. Based on the parameter values, determine the transition matrix of the "No aspirin" strategy, and call it `m_tp_comp` (for matrix transition probabilities of the comparator). Assume here that `r_mort` can be used to approximate the probability of 'Death other'.
   HINT: To do so, create a matrix of 7 by 7, and use the evidence `p_` and `r_` parameters to fill it in. Use the model structure to fill in the transition matrix. Once an individuals die, that person remains in that health state. The transition probabilities to the post-minor stroke health states are the product of experiencing a non-fatal stroke and of the consequences being minor or major.
   Answer: The transition probability should be filled in as follows:

```
m_tp_comp <- matrix(0,
                    ncol = n_hs,
                    nrow = n_hs,
```

```
                  dimnames = list(v_names_hs,
                                  v_names_hs))
## Filling the transition matrix
m_tp_comp["Well", "Well"] <- 1 - r_inc_mi - r_inc_stroke - r_mort # EXAMPLE! Calculate the remaining tr
m_tp_comp["Well", "Post-minor_stroke"] <- r_inc_stroke * (1 - r_fatal_stroke) * p_post_minor_stroke
m_tp_comp["Well", "Post-major_stroke"] <- r_inc_stroke * (1 - r_fatal_stroke) * p_post_major_stroke
m_tp_comp["Well", "Post-MI"] <- r_inc_mi * (1 - r_fatal_mi)
m_tp_comp["Well", "Death_stroke"] <- r_inc_stroke * r_fatal_stroke
m_tp_comp["Well", "Death_MI"] <- r_inc_mi * r_fatal_mi
m_tp_comp["Well", "Death_other"] <- r_mort

m_tp_comp["Post-minor_stroke", "Post-minor_stroke"] <- 1 - r_mort
m_tp_comp["Post-minor_stroke", "Death_other"] <- r_mort

m_tp_comp["Post-major_stroke", "Post-major_stroke"] <- 1 - r_mort
m_tp_comp["Post-major_stroke", "Death_other"] <- r_mort

m_tp_comp["Post-MI", "Post-MI"] <- 1 - r_mort
m_tp_comp["Post-MI", "Death_other"] <- r_mort

m_tp_comp["Death_stroke", "Death_stroke"] <- 1
m_tp_comp["Death_MI", "Death_MI"] <- 1
m_tp_comp["Death_other", "Death_other"] <- 1
kable(m_tp_comp,
      caption = "Transition matrix for the 'No aspirin' group",
      digits = 2)
```

Table 1: Transition matrix for the 'No aspirin' group

	Well	Post-minor_stroke	Post-major_stroke	Post-MI	Death_stroke	Death_MI	Death_other
Well	0.99	0.00	0.00	0.00	0	0	0.01
Post-minor_stroke	0.00	0.99	0.00	0.00	0	0	0.01
Post-major_stroke	0.00	0.00	0.99	0.00	0	0	0.01
Post-MI	0.00	0.00	0.00	0.99	0	0	0.01
Death_stroke	0.00	0.00	0.00	0.00	1	0	0.00
Death_MI	0.00	0.00	0.00	0.00	0	1	0.00
Death_other	0.00	0.00	0.00	0.00	0	0	1.00

2.a. Check whether the sum of all rows equals 1.

```
kable(rowSums(m_tp_comp))
```

	x
Well	1
Post-minor_stroke	1
Post-major_stroke	1
Post-MI	1

	x
Death_stroke	1
Death_MI	1
Death_other	1

3. Assume all 100,000 individuals (defined as the `n_ind` object) start in the "Well" health state (as defined in the `v_start_hs` vector), and perform the cohort simulation for the "No aspirin" group using the transition matrix `m_tp_comp`. Remember from previous assignment that you need the following elements to perform the matrix multiplication needed to perform the cohort simulation. Most of these elements are already defined in the `Assignment_start.R` script:

- `n_cycles`: the number of cycles to simulate, assume 10 years (yearly cycles)
- `n_ind`: the number of individuals to simulate, assume 100,000
- `m_hs_comp`: a matrix to store the number of individuals in each health state during each cycle for the comparator. This matrix has `n_cycles` + 1 row (because we need to account for the start position), and as much columns as health states 75). Numerate the rows from 0 to `n_cycles` and name the column with the names of the health states ("Well", "Post-minor_stroke", "Post-major_stroke", "Post-MI", "Death_stroke", "Death_MI", "Death_other").
- `v_start_hs`: a vector determining the starting position of the individuals over the different cycles (assume all individuals start in the "Well" health state)
  Once you have determined these parameters, use `v_start_hs` to determine the starting position of the individuals in the simulation.
  Answer: The cohort simulation `m_hs_comp` should be filled as follows:

```
# Define cohort simulation matrix
m_hs_comp <- matrix(0,
                    ncol = n_hs,
                    nrow = n_cycles + 1,
                    dimnames = list(c(0:n_cycles),
                                    v_names_hs)
                    )
kable(cbind(Cycle = c(0:n_cycles), m_hs_comp),
      caption = "Start empty cohort simulation",
      digits = 0,
      row.names = F
)
```

Table 3: Start empty cohort simulation

Cycle	Well	Post-minor_stroke	Post-major_stroke	Post-MI	Death_stroke	Death_MI	Death_other
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0

Cycle	Well	Post-minor_stroke	Post-major_stroke	Post-MI	Death_stroke	Death_MI	Death_other
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0

```
# Define the starting position of the individuals in each health state
m_hs_comp[1, ] <- v_start_hs

# Perform the matrix multiplication (for loop) to fill in the cohort simulation
for(cycle in 1:n_cycles){

  m_hs_comp[cycle + 1, ] <- m_hs_comp[cycle, ] %*% m_tp_comp # matrix multiplication

}
kable(cbind(Cycle = c(0:n_cycles), m_hs_comp),
      caption = "Filled cohort simulation",
      digits = 0,
      row.names = F
      )
```

Table 4: Filled cohort simulation

Cycle	Well	Post-minor_stroke	Post-major_stroke	Post-MI	Death_stroke	Death_MI	Death_other
0	100000	0	0	0	0	0	0
1	98900	27	8	300	15	100	650
2	97812	53	16	595	30	199	1295
3	96736	79	24	884	45	297	1935
4	95672	104	32	1169	59	393	2570
5	94620	129	40	1448	73	489	3201
6	93579	154	47	1723	88	584	3826
7	92549	178	55	1992	102	677	4447
8	91531	201	62	2257	115	770	5063
9	90525	225	69	2517	129	861	5674
10	89529	247	76	2772	143	952	6281

3.a. Check whether all rows sum up to 100,000 to ensure you do not 'loose' or 'add' any person to the cohort simulation. Sometimes, due to rounding errors, you may seem that the sum does not exactly match the number of individuals, so round your sums to 5 decimals.

```
#3.a. Check whether all rows sum up to 100,000 to ensure you do not 'loose' or 'add' any person to the
kable(cbind("Total per cycle" = round(rowSums(m_hs_comp), 0),
      "Check whether total = 100,000"= round(rowSums(m_hs_comp), 5) == n_ind
      ),
      caption = "Check row sums is equal to number of individuals"
      )
```

Table 5: Check row sums is equal to number of individuals

	Total per cycle	Check whether total = 100,000
0	100000	1
1	100000	1
2	100000	1
3	100000	1
4	100000	1
5	100000	1
6	100000	1
7	100000	1
8	100000	1
9	100000	1
10	100000	1

References