heemod

N Green

17/01/2020

R Markdown

```
# heemod TB population Markov model
# N Green
# Imperial College London
#
# see: https://cran.r-project.org/web/packages/heemod/vignettes/d_non_homogeneous.html

# NOTE:
# transitions happen at the beginning of each year (equivalent to transition happening at
# the end + ignoring the first year) with method = "beginning".
# Since with this method the first year is actually the second,
# costs should be discounted from the start with the argument first = TRUE in discount().


library(heemod)
library(purrr)
library(dplyr)

# age-dependent probability of death, TB and QoL weighting
pdeath_QoL <-
  read.csv("C:/Users/ngreen1/Google Drive/MSc-MPH/projects/2017/LTBI-TST_Manabu/R/raw data/pdeath_QoL.cs
head(pdeath_QoL)
```

```
##   age    pDeath pDeath_TB QOL_weight
## 1  35 0.0008065     0.012       0.91
## 2  36 0.0008710     0.012       0.91
## 3  37 0.0009280     0.012       0.91
## 4  38 0.0010540     0.012       0.91
## 5  39 0.0011160     0.012       0.91
## 6  40 0.0012320     0.012       0.91
```

```
# probabilistic realisations of starting state probabilities
load(file = "data/init_states.RData")
head(init_states)
```

```
##   noLTBI completeTx incompleteTx      noTx activeTB dead
## 1  0.643 0.04429927  0.068075534 0.2446252        0    0
## 2  0.683 0.02576789  0.019090692 0.2721414        0    0
## 3  0.672 0.04943117  0.030844729 0.2477241        0    0
## 4  0.742 0.05223975  0.001835084 0.2039252        0    0
## 5  0.726 0.02743526  0.010678055 0.2358867        0    0
## 6  0.650 0.05570397  0.024181029 0.2701150        0    0
```

```r
# define the model heemod parameters
param <- define_parameters(
  age_init = 34,                   # starting age
  age = age_init + markov_cycle,   # increment age annually

  # transition probabilities
  pReact_comp = 0.0006779,         # TB after completed LTBI treatment
  pReact_incomp = 0.0015301,       # TB after LTBI treatment dropout
  pReact = 0.0019369,              # TB after no treatment

  TB_cost = 4925.76,               # cost of TB treatment (Â£)
  d = 0.035,                       # annual discount factor

  # match prob death to age
  pdeath = look_up(data = pdeath_QoL,
                   value = "pDeath",
                   age = age),
  pdeathTB = look_up(data = pdeath_QoL,
                     value = "pDeath_TB",
                     age = age),

  # match QoL weight to age
  QoL = look_up(data = pdeath_QoL,
                value = "QOL_weight",
                age = age)
)

# create transition matrix
mat_trans <- define_transition(
  state_names = c(
    "noLTBI",
    "completeTx",
    "incompleteTx",
    "noTx",
    "activeTB",
    "dead"
  ),

  # from-to probability matrix
  # C represent complements
  C, 0, 0, 0, 0,             pdeath,
  0, C, 0, 0, pReact_comp,   pdeath,
  0, 0, C, 0, pReact_incomp, pdeath,
  0, 0, 0, C, pReact,        pdeath,
  C, 0, 0, 0, 0,             pdeathTB,
  0, 0, 0, 0, 0,             1
)

# define starting state populations
init_states <- select(.data = init_states,
                      noLTBI,
                      completeTx,
                      incompleteTx,
```

```
                      noTx)

init_states <- data.frame(init_states, activeTB = 0, dead = 0)


# define cost and utility values associated with each state

noLTBI <- define_state(
  cost = 0,
  utility = discount(QoL, d, first = TRUE)
)

completeTx <- define_state(
  cost = 0,
  utility = discount(QoL, d, first = TRUE)
)

incompleteTx <- define_state(
  cost = 0,
  utility = discount(QoL, d, first = TRUE)
)

noTx <- define_state(
  cost = 0,
  utility = discount(QoL, d, first = TRUE)
)

activeTB <- define_state(
  cost = discount(TB_cost, d, first = TRUE),
  utility = discount(QoL - 0.15, d, first = TRUE)
)

dead <- define_state(
  cost = 0,
  utility = 0
)

# combine all of the model elements to form
# a 'stratgey' consisting of a transition
# matrix and states states with properties attached
strat <- define_strategy(
  transition = mat_trans,
  noLTBI = noLTBI,
  completeTx = completeTx,
  incompleteTx = incompleteTx,
  noTx = noTx,
  activeTB = activeTB,
  dead = dead
)

# run a single simulation
res_mod <-
  run_model(
```

```r
    init = 1000 * init_states[1, ], # initial population sizes
    method = "end",
    strat,
    parameters = param,
    cycles = 66,                    # number of time steps
    cost = cost,
    effect = utility
  )


# run multiple simulations
# using the sample of starting state probabilities

res_mod <- list()

for (i in 1:nrow(init_states)) {

  res_mod[[i]] <-
    run_model(
      # init = c(674.0588764, # hard-code values
      #          168.0253748,
      #          42.42724895,
      #          115.4884998,
      #          0,0),
      init = 1000 * init_states[i, ],
      method = "end",
      strat,
      parameters = param,
      cycles = 66,
      cost = cost,
      effect = utility
    )
}


###########
# results #
###########

res_mod[[1]]
```

```
## 1 strategy run for 66 cycles.
##
## Initial state counts:
##
## noLTBI = 643
## completeTx = 44.2992721299233
## incompleteTx = 68.0755335064717
## noTx = 244.625194363605
## activeTB = 0
## dead = 0
##
## Counting method: 'end'.
```

```
##
## Values:
##
##      cost  utility
## I 62739.3 18900.67
```

```r
# extract the cost and utility values
c1 <- map_df(res_mod, "run_model")$cost
h1 <- map_df(res_mod, "run_model")$utility

get_counts(res_mod[[1]])
```

```
## # A tibble: 396 x 4
##    .strategy_names markov_cycle state_names count
##    <chr>                  <int> <chr>       <dbl>
##  1 I                          1 noLTBI       643
##  2 I                          2 noLTBI       642.
##  3 I                          3 noLTBI       643.
##  4 I                          4 noLTBI       643.
##  5 I                          5 noLTBI       642.
##  6 I                          6 noLTBI       642.
##  7 I                          7 noLTBI       642.
##  8 I                          8 noLTBI       642.
##  9 I                          9 noLTBI       642.
## 10 I                         10 noLTBI       641.
## # ... with 386 more rows
```

```r
get_values(res_mod[[1]])
```

```
##     markov_cycle .strategy_names value_names       value
## 1              1               I        cost    0.000000
## 2              2               I        cost 2795.769776
## 3              3               I        cost 2694.172518
## 4              4               I        cost 2596.099603
## 5              5               I        cost 2501.453976
## 6              6               I        cost 2409.954523
## 7              7               I        cost 2321.657828
## 8              8               I        cost 2236.336172
## 9              9               I        cost 2153.994737
## 10            10               I        cost 2074.430343
## 11            11               I        cost 1997.642736
## 12            12               I        cost 1923.381158
## 13            13               I        cost 1851.604428
## 14            14               I        cost 1782.312318
## 15            15               I        cost 1715.343104
## 16            16               I        cost 1650.671596
## 17            17               I        cost 1588.062876
## 18            18               I        cost 1527.462285
## 19            19               I        cost 1468.888781
## 20            20               I        cost 1412.237940
## 21            21               I        cost 1357.362057
## 22            22               I        cost 1304.188468
## 23            23               I        cost 1252.628015
```

```
## 24          24          I      cost 1202.657685
## 25          25          I      cost 1154.094094
## 26          26          I      cost 1107.023286
## 27          27          I      cost 1061.252042
## 28          28          I      cost 1016.691493
## 29          29          I      cost  973.447557
## 30          30          I      cost  931.402075
## 31          31          I      cost  890.504538
## 32          32          I      cost  850.762316
## 33          33          I      cost  812.160065
## 34          34          I      cost  774.833496
## 35          35          I      cost  738.342870
## 36          36          I      cost  702.646295
## 37          37          I      cost  667.662025
## 38          38          I      cost  633.435905
## 39          39          I      cost  599.927304
## 40          40          I      cost  566.751238
## 41          41          I      cost  534.205724
## 42          42          I      cost  502.145143
## 43          43          I      cost  470.828332
## 44          44          I      cost  440.103590
## 45          45          I      cost  410.024029
## 46          46          I      cost  380.448985
## 47          47          I      cost  351.321854
## 48          48          I      cost  322.365674
## 49          49          I      cost  294.060165
## 50          50          I      cost  266.105785
## 51          51          I      cost  238.872856
## 52          52          I      cost  212.372972
## 53          53          I      cost  186.726035
## 54          54          I      cost  162.356251
## 55          55          I      cost  139.261868
## 56          56          I      cost  117.704591
## 57          57          I      cost   97.860681
## 58          58          I      cost   79.882174
## 59          59          I      cost   64.069459
## 60          60          I      cost   50.160404
## 61          61          I      cost   38.227267
## 62          62          I      cost   28.538763
## 63          63          I      cost   20.760109
## 64          64          I      cost   14.777203
## 65          65          I      cost   10.152295
## 66          66          I      cost    6.720050
## 67           1          I   utility  879.227053
## 68           2          I   utility  848.724483
## 69           3          I   utility  819.304004
## 70           4          I   utility  790.858282
## 71           5          I   utility  763.303962
## 72           6          I   utility  736.663978
## 73           7          I   utility  710.871244
## 74           8          I   utility  685.932177
## 75           9          I   utility  661.786899
## 76          10          I   utility  638.439800
## 77          11          I   utility  575.208045
```

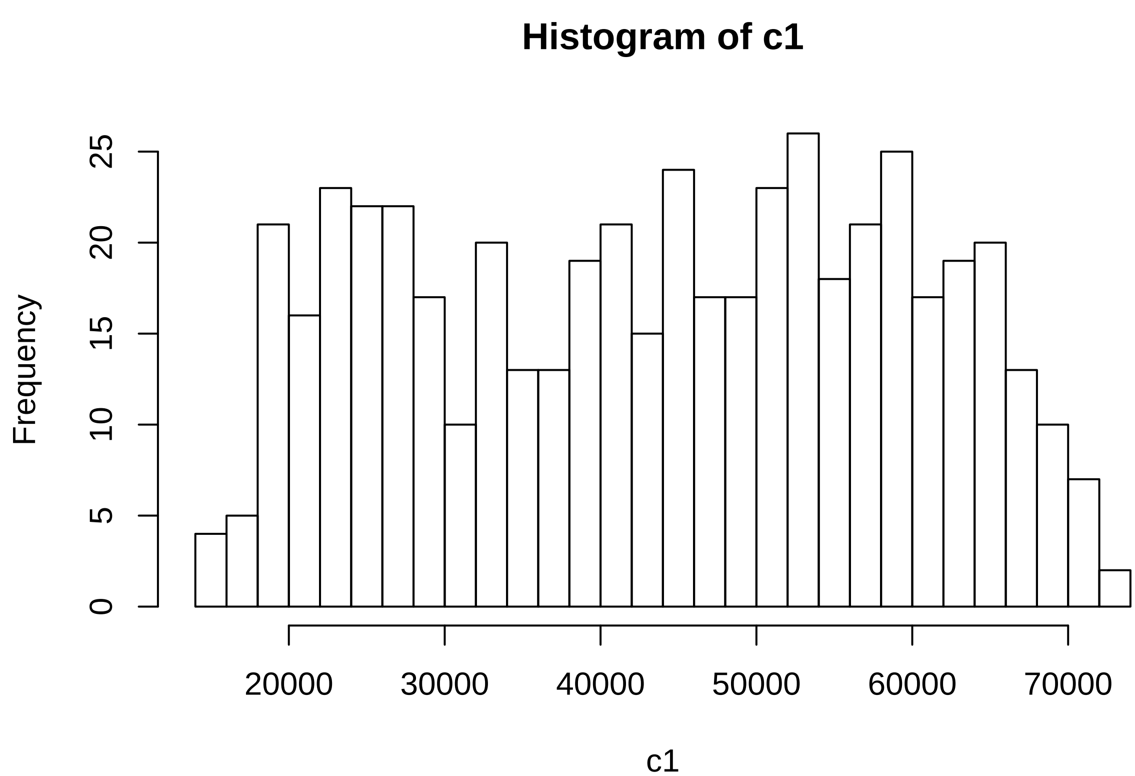
```
## 78            12                I     utility  554.729804
## 79            13                I     utility  534.922528
## 80            14                I     utility  515.741419
## 81            15                I     utility  497.182389
## 82            16                I     utility  479.178165
## 83            17                I     utility  461.715353
## 84            18                I     utility  444.802694
## 85            19                I     utility  428.411635
## 86            20                I     utility  412.500293
## 87            21                I     utility  373.691000
## 88            22                I     utility  359.559078
## 89            23                I     utility  345.832962
## 90            24                I     utility  332.462205
## 91            25                I     utility  319.473575
## 92            26                I     utility  306.813466
## 93            27                I     utility  294.458043
## 94            28                I     utility  282.439678
## 95            29                I     utility  270.725985
## 96            30                I     utility  259.303999
## 97            31                I     utility  241.972384
## 98            32                I     utility  231.391989
## 99            33                I     utility  221.138738
## 100           34                I     utility  211.088924
## 101           35                I     utility  201.231673
## 102           36                I     utility  191.544620
## 103           37                I     utility  182.041830
## 104           38                I     utility  172.712621
## 105           39                I     utility  163.447144
## 106           40                I     utility  154.331660
## 107           41                I     utility  136.008177
## 108           42                I     utility  127.751249
## 109           43                I     utility  119.626485
## 110           44                I     utility  111.649013
## 111           45                I     utility  103.781322
## 112           46                I     utility   96.008548
## 113           47                I     utility   88.255520
## 114           48                I     utility   80.653575
## 115           49                I     utility   73.121352
## 116           50                I     utility   65.760865
## 117           51                I     utility   58.576148
## 118           52                I     utility   51.601109
## 119           53                I     utility   44.953942
## 120           54                I     utility   38.635842
## 121           55                I     utility   32.721061
## 122           56                I     utility   27.260682
## 123           57                I     utility   22.299453
## 124           58                I     utility   17.923894
## 125           59                I     utility   14.063995
## 126           60                I     utility   10.742899
## 127           61                I     utility    8.039178
## 128           62                I     utility    5.862318
## 129           63                I     utility    4.183370
## 130           64                I     utility    2.881664
## 131           65                I     utility    1.912731
```

```
## 132           66                I      utility    1.230143
```

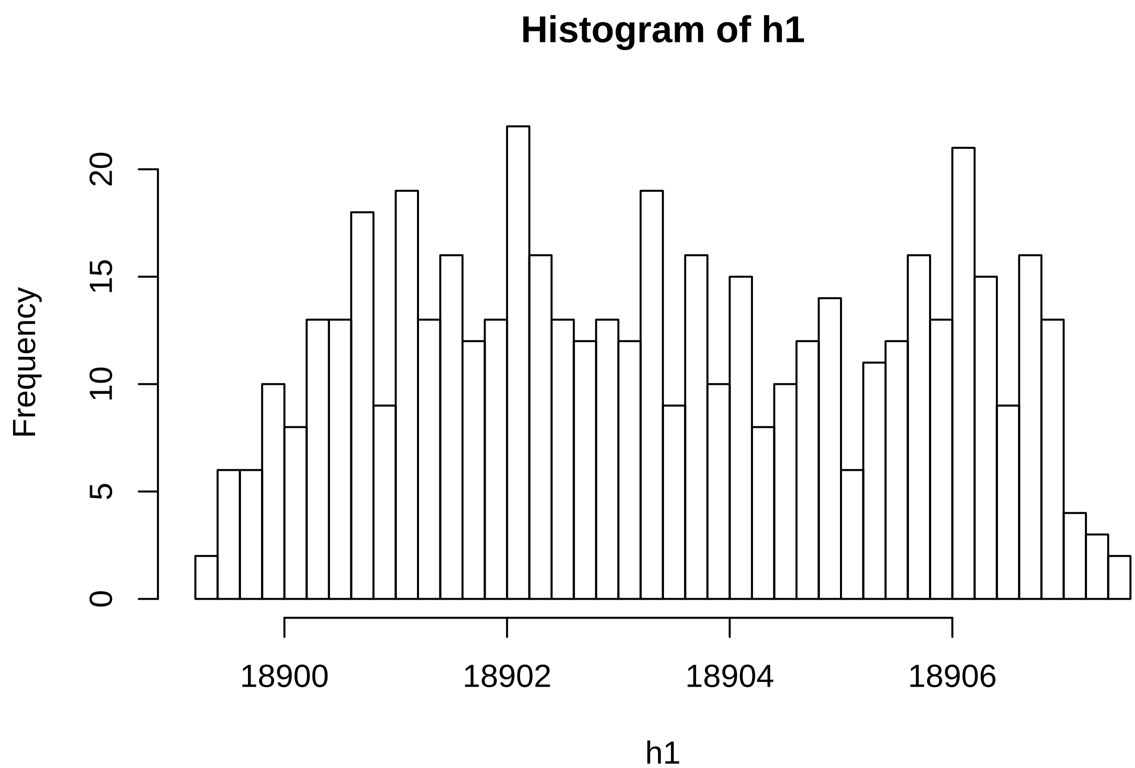
```r
summary(res_mod[[4]])
```

```
## 1 strategy run for 66 cycles.
##
## Initial state counts:
##
## noLTBI = 742
## completeTx = 52.2397525531171
## incompleteTx = 1.83508364108649
## noTx = 203.925163805796
## activeTB = 0
## dead = 0
##
## Counting method: 'end'.
##
## Values:
##
##    cost  utility
## I 44682 18903.23
```

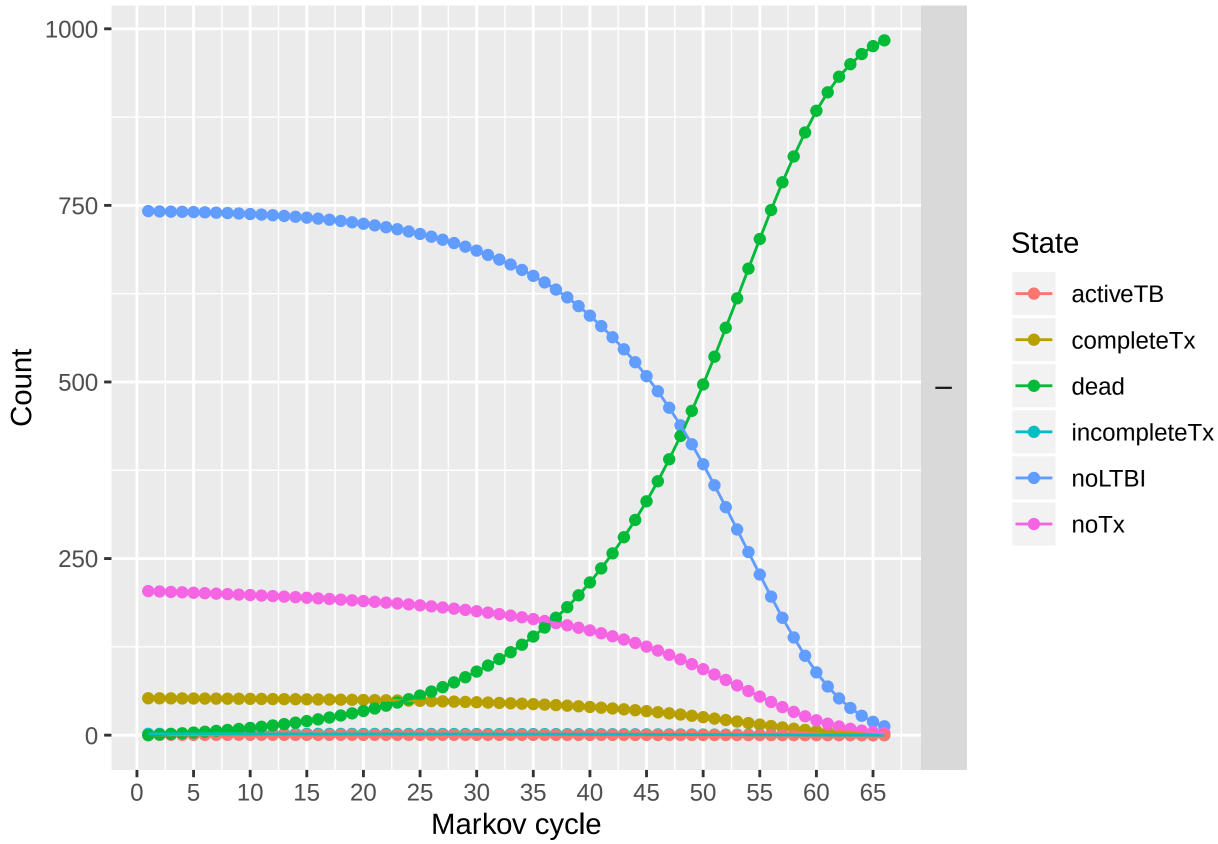
```r
# plots
hist(c1, breaks = 30)
```



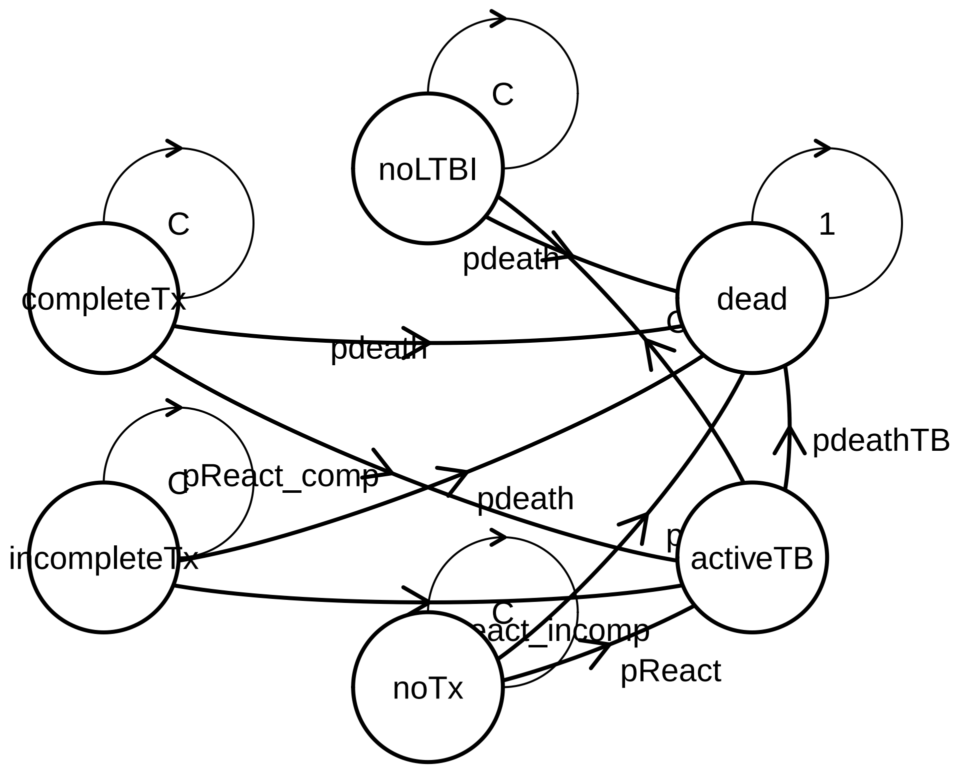
```r
hist(h1, breaks = 30)
```




```r
plot(res_mod[[4]])
```




```
# state-edge graph
plot(mat_trans, arr.type = "simple")
```


noLTBI
C
completeTx
C
dead
1
pdeath
pdeath
incompleteTx
C
pReact_comp
pdeath
pdeathTB
activeTB
noTx
C
pReact_incomp
pReact