

Advanced Life Insurance Mathematics

Computer lab on multiple state models

Katrien Antonio, Bavo Campo and Sander Devriendt

Workshop mortality dynamics | May, 2020

Prologue

Introduction

Workshop

 <https://github.com/katrienantonio/mortality-dynamics>

The workshop repo on GitHub, where we upload presentation, code, data sets, etc.


Us

 <https://katrienantonio.github.io/>

 katrien.antonio@kuleuven.be & bavo.decock@kuleuven.be & Sander Devriendt

 (Katrien) Professor in insurance data science

 (Bavo) PhD student in insurance data science

 (Sander) former PhD student, now with NBB (Brussels, Belgium)

Checklist

- ☑ Do you have a fairly recent version of R?

```
version$version.string
```

- ☑ Do you have a fairly recent version of RStudio?

```
RStudio.Version()$version  
## Requires an interactive session but should return something like "[1] '1.2.5001'"
```

- ☑ Have you installed the R packages listed in the software requirements?

Goals of the workshop

Goals of the workshop

This computer lab allows you to:

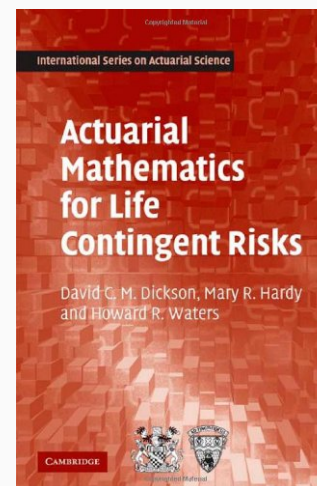
- numerically evaluate concepts discussed in ALIM classes
- improve intuition on these concepts
- translating the ALIM concepts to code
- spend time with R code provided.

You hereby focus on:

- multiple state models
- calculation of transition probabilities, premiums and policy values
- via numerical integration methods and a (basic) solver for a set of differential equations.

Outline of the workshop:

- Prologue
- Transition probabilities
- Premium calculations
- Policy values



Transition probabilities

The permanent disability model - Example 8.4

Section 8.5 in the Dickson et al. (2013, 2nd edition) book puts focus on the numerical evaluation of transition probabilities in multiple state models.

We start with Example 8.4 on the **permanent disability model**, as pictured on the right.

We specify:

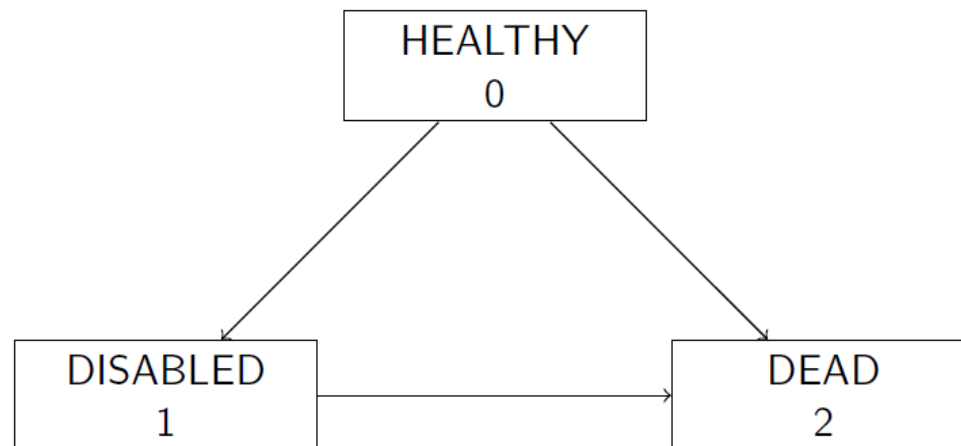
$$\mu_x^{01} = a_1 + b_1 \exp(c_1 x)$$

$$\mu_x^{02} = a_2 + b_2 \exp(c_2 x)$$

$$\mu_x^{12} = \mu_x^{02}.$$

The goal is then to calculate transition probabilities, e.g.

$${}_{10}p_{60}^{01}.$$



We start from the following **sojourn probabilities** in the permanent disability model (cfr. the lecture sheets)

$${}_t p_{60}^{00} = \exp \left\{ - \int_0^t (\mu_{60+r}^{01} + \mu_{60+r}^{02}) dr \right\} = \exp \left\{ - \left((a_1 + a_2)t + \frac{b_1}{c_1} e^{60c_1} (e^{c_1 t} - 1) + \frac{b_2}{c_2} e^{60c_2} (e^{c_2 t} - 1) \right) \right\},$$

and

$${}_t p_{60}^{11} = \exp \left\{ - \int_0^t \mu_{60+r}^{12} dr \right\} = \exp \left\{ - \left(a_2 t + \frac{b_2}{c_2} e^{60c_2} (e^{c_2 t} - 1) \right) \right\}.$$

These expressions are then used to calculate ${}_t p_{60}^{01}$ as follows

$${}_{10} p_{60}^{01} = \int_0^{10} {}_t p_{60}^{00} \cdot \mu_{60+t}^{01} \cdot {}_{10-t} p_{60+t}^{11} dt.$$

We use **numerical integration** to evaluate this probability.

We define the necessary constants (see the book)

```
a1 = 4e-4
a2 = 5e-4
b1 = 3.4674e-6
b2 = 7.5858e-5
c1 = 0.138155
c2 = 0.087498
```

For ${}_t p_{60}^{00}$

```
p00 = function(t, age, a1, b1, c1, a2, b2, c2) {
  p1 = (a1 + a2) * t
  p2 = (b1 / c1) * exp(age * c1) * (exp(c1 * t) - 1)
  p3 = (b2 / c2) * exp(age * c2) * (exp(c2 * t) - 1)
  prob = exp(-(p1 + p2 + p3))
  return(prob)
}

p00(10, 60, a1, b1, c1, a2, b2, c2)
## [1] 0.5839526
```

Similarly, for ${}_t p_{60}^{11}$

```
p11 = function(t, age, a1, b1, c1, a2, b2, c2) {
  p1 = a2 * t
  p2 = (b2 / c2) * exp(age * c2) * (exp(c2 * t) - 1)
  prob = exp(-(p1 + p2))
  return(prob)
}

p11(10, 60, a1, b1, c1, a2, b2, c2)
## [1] 0.7897179
```

For the transition intensity μ_x^{01} we have

```
mu01 = function(age, a1, b1, c1, a2, b2, c2) {
  return(a1 + b1 * exp(c1 * age))
}
```

Putting it all together we define the function ${}_t p_x^{01}$ we wish to integrate

```
p01integrand = function(t, age, end,  
                        a1, b1, c1, a2, b2, c2) {  
  p1 = p00(t, age, a1, b1, c1, a2, b2, c2)  
  p2 = mu01(age + t, a1, b1, c1, a2, b2, c2)  
  p3 = p11(end - t, age + t, a1, b1, c1,  
          a2, b2, c2)  
  prob = p1 * p2 * p3  
  return(prob)  
}
```

`integrate(f, lower, upper, ...)` in R integrates the function `f` from `lower` to `upper`.

We use the `integrate()` function in R to perform the numerical integration.

We let t run from 0 to 10 and specify age $x = 60$.

```
integrate(  
  p01integrand,  
  lower = 0,  
  upper = 10,  
  age = 60,  
  end = 10,  
  a1,  
  b1,  
  c1,  
  a2,  
  b2,  
  c2  
)
```

```
## 0.2057653 with absolute error < 2.3e-15
```

Avoiding the juggling with multiple functions, we can also define ${}_t p_x^{01}$ in a single step, as follows

```
p01_single ← function(x, a1, a2, b1, b2, c1, c2) {  
  p = exp(-((a1 + a2) * x + b1 / c1 * exp(60 * c1) * (exp(c1 * x) - 1) +  
            b2 / c2 * exp(60 * c2) * (exp(c2 * x) - 1))) *  
      exp(-(a2 * (10 - x) + b2 / c2 * exp((60 + x) * c2) * (exp(c2 * (10 - x)) - 1)))  
  p * (a1 + b1 * exp(c1 * (60 + x)))  
}  
  
integrate(  
  p01_single,  
  0,  
  10,  
  a1 = a1,  
  a2 = a2,  
  b1 = b1,  
  b2 = b2,  
  c1 = c1,  
  c2 = c2  
)  
## 0.2057653 with absolute error < 2.3e-15
```

Dickson et al. (2013, 2nd edition) discuss in Appendix B the trapezium rule and the Simpson rule to evaluate

$$I = \int_a^b f(x)dx,$$

for some function f .

With the **trapezium rule**:

let $h = (b - a)/n$

$$I \approx h \left(\frac{1}{2} f(a) + \sum_{j=1}^{n-1} f(a + j \cdot h) + \frac{1}{2} f(b). \right)$$

With the **Simpson rule**:

with $h = (b - a)/2n$ or $n = (b - a)/(2h)$

$$I \approx h/3 \left(f(a) + 4 \sum_{j=1}^n f(a + (2j - 1) \cdot h) + 2 \sum_{j=1}^{n-1} f(a + 2j \cdot h) + f(b). \right)$$

We now re-evaluate ${}_{10}p_{60}^{01} = \int_0^{10} {}_t p_{60}^{00} \cdot \mu_{60+t}^{01} \cdot {}_{10-t} p_{60+t}^{11} dt$.

```
h      = 1 / 12
a      = 0
b      = 10
age    = 60
grid   = seq(from = a, to = b, by = h)
ngrid  = length(grid)

f = p00(grid, age, a1, b1, c1, a2, b2, c2) * mu01(age + grid, a1, b1, c1, a2, b2, c2) *
    p11(10 - grid, age + grid, a1, b1, c1, a2, b2, c2)
```

With the trapezium rule

```
(inttrap = h * (0.5 * f[1] + sum(f[2:(ngrid - 1)]) + 0.5 * f[ngrid]))
## [1] 0.2057661
```

With the Simpson rule

```
n = (b - a) / (2 * h)
(intSimp =
  (h / 3) * (f[1] + 4 * sum(f[2 * 1:n]) + 2 * sum(f[2 * 1:(n - 1) + 1]) +
    f[ngrid]))
## [1] 0.2057653
```

The disability income model - Example 8.5

Section 8.5 in the Dickson et al. (2013, 2nd edition) book puts focus on the numerical evaluation of transition probabilities in multiple state models.

We now move on with Example 8.5 on the **disability income model**, as pictured on the right.

We specify:

$$\mu_x^{01} = a_1 + b_1 \exp(c_1 x)$$

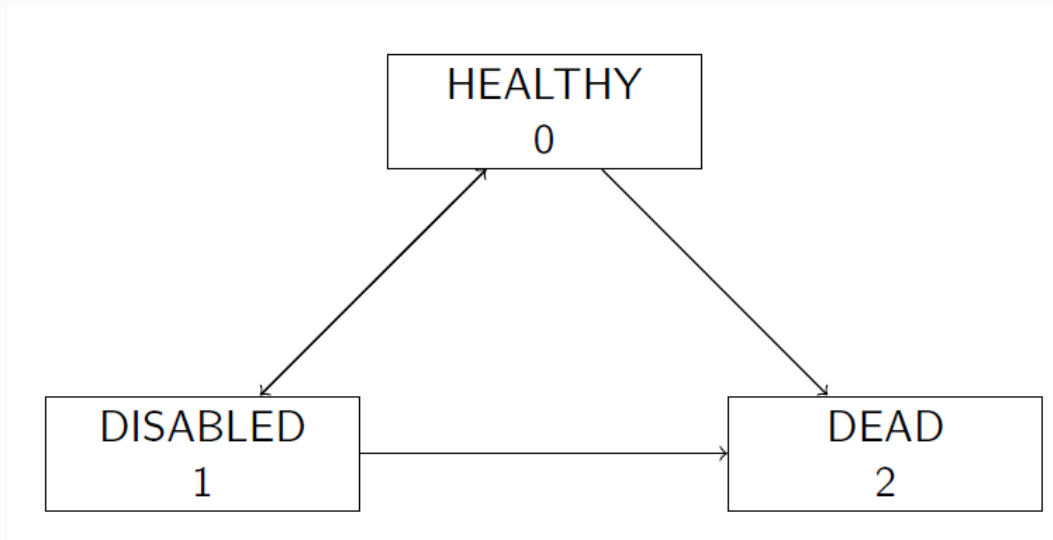
$$\mu_x^{02} = a_2 + b_2 \exp(c_2 x)$$

$$\mu_x^{10} = 0.1 \cdot \mu_x^{01}$$

$$\mu_x^{12} = \mu_x^{02}.$$

The goal is then to calculate

$${}_{10}p_{60}^{00}$$
$${}_{10}p_{60}^{01}.$$



We use the **Kolmogorov forward equations**:

$$\frac{d}{dt} {}_t p_x^{ij} = \sum_{k=0, k \neq j}^n \left({}_t p_x^{ik} \mu_{x+t}^{kj} - {}_t p_x^{ij} \mu_{x+t}^{jk} \right),$$

rewritten as

$${}_{t+h} p_x^{ij} = {}_t p_x^{ij} - h \sum_{k=0, k \neq j}^n \left({}_t p_x^{ij} \mu_{x+t}^{jk} - {}_t p_x^{ik} \mu_{x+t}^{kj} \right) + o(h).$$

For this particular application we get:

$${}_{t+h} p_{60}^{00} = {}_t p_{60}^{00} - h {}_t p_{60}^{00} (\mu_{60+t}^{01} + \mu_{60+t}^{02}) + h {}_t p_{60}^{01} \mu_{60+t}^{10} + o(h),$$

and

$${}_{t+h} p_{60}^{01} = {}_t p_{60}^{01} - h {}_t p_{60}^{01} (\mu_{60+t}^{12} + \mu_{60+t}^{10}) + h {}_t p_{60}^{00} \mu_{60+t}^{01} + o(h).$$

We define the transition intensities

```
mu01 ← function(a1, b1, c1, x) a1 + b1 * exp(c1 * x)
mu10 ← function(a1, b1, c1, x) 0.1 * mu01(a1, b1, c1, x)
mu02 ← function(a2, b2, c2, x) a2 + b2 * exp(c2 * x)
mu12 ← mu02
```

We define a function to evaluate the rhs of the expressions on the previous sheet:

```
## Kolmogorov equations
thP00 ← function(tP00, tP01, h, x)
  tP00 - h * tP00 * (mu01(a1, b1, c1, x) + mu02(a2, b2, c2, x)) + h * tP01 * mu10(a1, b1, c1, x)
thP01 ← function(tP00, tP01, h, x)
  tP01 - h * tP01 * (mu12(a2, b2, c2, x) + mu10(a1, b1, c1, x)) + h * tP00 * mu01(a1, b1, c1, x)
```

We use the above to evaluate $t p_{60}^{00}$ for $t = 1/12$, starting from ${}_0 p_{60}^{00} = 1$ and ${}_0 p_{60}^{01} = 0$.

```
thP00(1, 0, 1 / 12, 60) # Table 8.1, second row, column tP^{00}_{60}
## [1] 0.9975702
```

In a similar way, for $t p_{60}^{01}$ for $t = 1/12$

```
thP01(1, 0, 1 / 12, 60) # Table 8.1, second row, column tP^{01}_{60}
## [1] 0.001183657
```

```

## Reproduction table 8.1
Results = matrix(NA, 12 * 10 + 1, 2)
Results[1, ] = c(thP00(1, 0, 0, 60), thP01(1, 0, 0, 60))
for(i in 1:(12 * 10)) {
  Results[i + 1, ] = c(thP00(Results[i, 1], Results[i, 2], 1 / 12, 60 + (i - 1) * 1 / 12),
    thP01(Results[i, 1], Results[i, 2], 1 / 12, 60 + (i - 1) * 1 / 12))
}
head(Results)
##           [,1]      [,2]
## [1,] 1.0000000 0.000000000
## [2,] 0.9975702 0.001183657
## [3,] 0.9951243 0.002376098
## [4,] 0.9926623 0.003577357
## [5,] 0.9901841 0.004787465
## [6,] 0.9876898 0.006006455
tail(Results)
##           [,1]      [,2]
## [116,] 0.6091143 0.1925074
## [117,] 0.6048221 0.1945329
## [118,] 0.6005199 0.1965584
## [119,] 0.5962082 0.1985837
## [120,] 0.5918870 0.2006084
## [121,] 0.5875568 0.2026324

```

```

t = seq(0, 10, by = 1 / 12)
Table8.1 = cbind.data.frame(t = t,
                             mu01 = mu01(a1, b1, c1, 60 + t),
                             mu02 = mu02(a2, b2, c2, 60 + t),
                             mu10 = mu10(a1, b1, c1, 60 + t),
                             mu12 = mu12(a2, b2, c2, 60 + t),
                             thP00 = Results[, 1],
                             thP01 = Results[, 2])

head(round(Table8.1, 5))
##           t      mu01      mu02      mu10      mu12      thP00      thP01
## 1 0.00000 0.01420 0.01495 0.00142 0.01495 1.00000 0.00000
## 2 0.08333 0.01436 0.01506 0.00144 0.01506 0.99757 0.00118
## 3 0.16667 0.01453 0.01517 0.00145 0.01517 0.99512 0.00238
## 4 0.25000 0.01469 0.01527 0.00147 0.01527 0.99266 0.00358
## 5 0.33333 0.01485 0.01538 0.00149 0.01538 0.99018 0.00479
## 6 0.41667 0.01502 0.01549 0.00150 0.01549 0.98769 0.00601
tail(round(Table8.1, 5))
##           t      mu01      mu02      mu10      mu12      thP00      thP01
## 116 9.58333 0.05228 0.03393 0.00523 0.03393 0.60911 0.19251
## 117 9.66667 0.05288 0.03418 0.00529 0.03418 0.60482 0.19453
## 118 9.75000 0.05349 0.03442 0.00535 0.03442 0.60052 0.19656
## 119 9.83333 0.05410 0.03467 0.00541 0.03467 0.59621 0.19858
## 120 9.91667 0.05473 0.03492 0.00547 0.03492 0.59189 0.20061
## 121 10.00000 0.05535 0.03517 0.00554 0.03517 0.58756 0.20263

```

Premium calculations

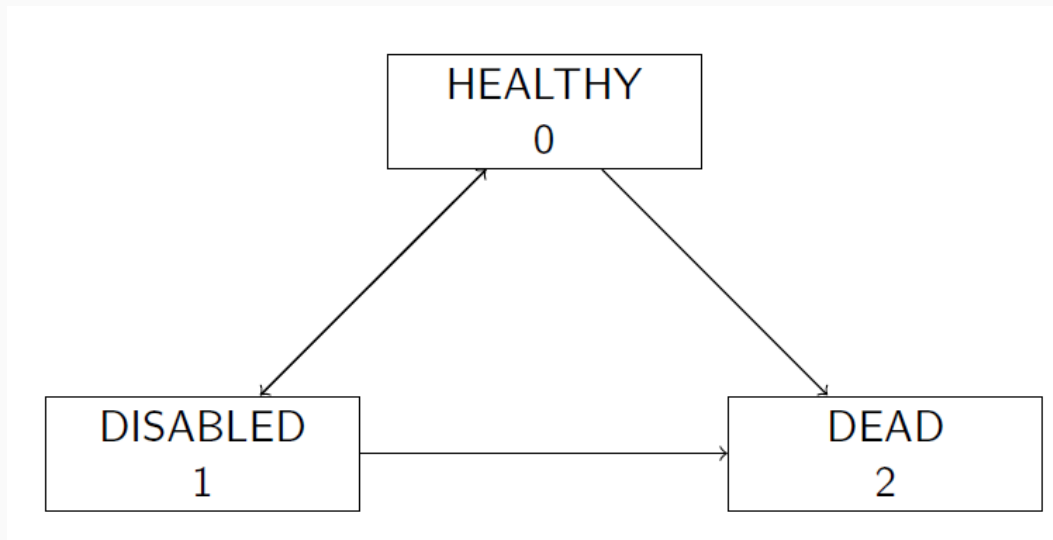
Premium calculations - Example 8.6

In this example we value:

- a 10-year disability income product
- issued to a healthy (60)
- interest rate of 5% per year.

Part (a) of this exercise considers:

- premiums payable continuously in the healthy state
- benefit of 20 000 EUR per year payable continuously in the disabled state
- death benefit of 50 000 EUR on death.



We initialize some given constants

```
i      = 0.05
delta = log(1 + i)
h      = 1/12
a      = 0
b      = 10
n      = (b - a) / (2 * h)
grid   = seq(0, 10, by = h)
ngrid  = length(grid)
```

Following the Dickson et al. (2013) book we use the [trapezium](#) and [Simpson's rule](#) to numerically evaluate the integrals.

```
# discounting
dis      = numeric(ngrid)
dis      = exp(-delta * grid)
```

We extract the variables from our reproduction of Table 8.1 in the book

```
list2env(Table8.1[, c("mu02", "mu12", "thP00", "thP01")], envir = .GlobalEnv)
## <environment: R_GlobalEnv>
```

For the **EPV of the premiums** we evaluate

$$P \cdot \bar{a}_{60:\overline{10}|}^{00} = P \int_0^{10} e^{-\delta t} {}_t p_{60}^{00} dt,$$

with P the annual premium rate.

The premiums are continuously payable while in the healthy state.

The book reports a value of $P \cdot 6.5714$.

Let's check this in R!

```
## EPV of premium income ##
thP00dis = dis * thP00

# using trapezium rule
(intprem =
  h * (0.5 * thP00dis[1] +
    sum(thP00dis[2:(ngrid - 1)]) +
    0.5 * thP00dis[ngrid]))
## [1] 6.571398

# using Simpson's rule
(intprem =
  (h / 3) * (thP00dis[1] +
    4 * sum(thP00dis[2 * 1:n]) +
    2 * sum(thP00dis[2 * 1:(n - 1) + 1]) +
    thP00dis[ngrid]))
## [1] 6.571382
```

For the **EPV of the sickness benefit** we evaluate

$$20\,000 \cdot \bar{a}_{60:\overline{10}|}^{01} = 20\,000 \int_0^{10} e^{-\delta t} {}_tP_{60}^{01} dt.$$

The sickness benefits are continuously payable while in the sickness state.

The book reports a value of **20 000 · 0.66359**.

Let's check this in **R**!

```
## EPV of sickness benefit ##
thP01dis = dis * thP01

# using trapezium rule
(intsick = h * (0.5 * thP01dis[1] +
               sum(thP01dis[2:(ngrid - 1)]) +
               0.5 * thP01dis[ngrid]))

## [1] 0.6635877

# using Simpson's rule
(intsick = (h / 3) * (thP01dis[1] +
                    4 * sum(thP01dis[2 * 1:n]) +
                    2 * sum(thP01dis[2 * 1:(n - 1) + 1]) +
                    thP01dis[ngrid]))

## [1] 0.6635908
```


The **EPV of the death benefit** becomes

$$50\,000 \cdot \bar{A}_{60:\overline{10}|}^{02} \\ = 50\,000 \int_0^{10} e^{-\delta t} \left({}_tP_{60}^{00} \cdot \mu_{60+t}^{02} + {}_tP_{60}^{01} \cdot \mu_{60+t}^{12} \right) dt.$$

The death benefit is paid upon transition into the dead state.

The book reports a value of **$50\,000 \cdot 0.16231$** .

```
## EPV of death benefit ##
fdis = dis * (thP00 * mu02 + thP01 * mu12)

# using trapezium rule
(intdeath = h * (0.5 * fdis[1] +
                sum(fdis[2:(ngrid - 1)]) +
                0.5 * fdis[ngrid]))

## [1] 0.1623143

# using Simpson's rule
(intdeath = (h / 3) * (fdis[1] +
                    4 * sum(fdis[2 * 1:n]) +
                    2 * sum(fdis[2 * 1:(n - 1) + 1]) +
                    fdis[ngrid])))

## [1] 0.1623145
```

Putting the actuarial equivalence relation together and solving for **P** we find

```
(P = (20000 * intsick + 50000 * intdeath) / intprem)
## [1] 3254.649
```

Policy values

Policy values - Example 8.7

In this example we work with the **disability income model**.

Suppose that $x = 40$, $n = 20$, $\delta = 0.04$, $B = 100\,000$ and $S = 500\,000$ and

$$\mu_x^{01} = a_1 + b_1 \exp(c_1 x)$$

$$\mu_x^{10} = 0.1 \cdot \mu_x^{01}$$

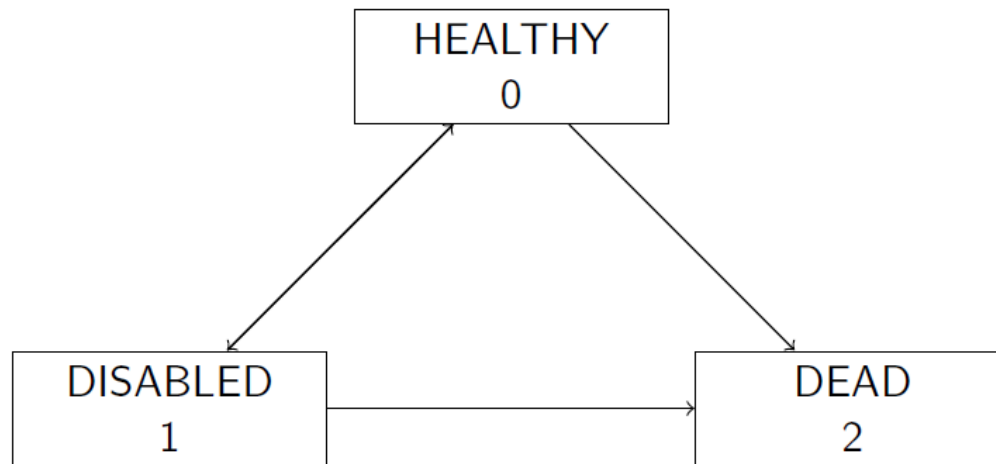
$$\mu_x^{20} = a_2 + b_2 \exp(c_2 x)$$

$$\mu_x^{12} = \mu_x^{02}.$$

We now calculate

$${}_{10}V^{(0)}, {}_{10}V^{(1)} \text{ and } {}_0V^{(0)}$$

for $n = 20$ given that $P = 5\,500$.



For policy value calculations in multiple state models we use **Thiele's differential equations**.

That is: for $i = 0, 1, \dots, n$ and $0 \leq t \leq n$

$$\frac{d}{dt} {}_tV^{(i)} = \delta_t \cdot {}_tV^{(i)} - B_t^{(i)} - \sum_{j=0, j \neq i}^n \mu_{x+t}^{ij} \left(S_t^{(ij)} + {}_tV^{(j)} - {}_tV^{(i)} \right).$$

Applied to the specific context of this example, and after some rewriting (see the sheets), we find (approximately)

$${}_{t-h}V^{(0)} = {}_tV^{(0)}(1 - \delta h) - Ph + h\mu_{x+t}^{01}({}_tV^{(1)} - {}_tV^{(0)}) + h\mu_{x+t}^{02}(S - {}_tV^{(0)}),$$

and

$${}_{t-h}V^{(1)} = {}_tV^{(1)}(1 - \delta h) + Bh + h\mu_{x+t}^{10}({}_tV^{(0)} - {}_tV^{(1)}) + h\mu_{x+t}^{12}(S - {}_tV^{(1)}).$$

With the above recursions, we work backwards!

Some constants

```
x = 40  
n = 20  
d = 0.04  
B = 100000  
S = 500000  
P = 5500
```

and

```
tV0 = 0  
tV1 = 0  
h = 1/12
```

For the transition intensities

```
a1 = 4e-4  
a2 = 5e-4  
b1 = 3.4674e-6  
b2 = 7.5858e-5  
c1 = 0.138155  
c2 = 0.087498  
mu01 ← function(x) a1 + b1 * exp(c1 * (x))  
mu10 ← function(x) 0.1 * mu01(x)  
mu02 ← function(x) a2 + b2 * exp(c2 * (x))  
mu12 ← mu02
```

Starting from ${}_nV^{(0)} = 0$ and ${}_nV^{(1)} = 0$ with $n = 20$, can we find the ${}_tV^{(0)}$ and ${}_tV^{(1)}$ at $t = 10$ and $t = 0$?

We use a step size $h = 1/12$ and divide a 10-year period in 120 steps.

We reason backwards!

Now we solve Thiele's differential equations with the Euler method and a step size $h = 1/12$

```
for(i in seq_len(120 * 2)) {  
  tV0[i + 1] = tV0[i] * (1 - d * h) - P * h + h * mu01(x + n - ((i - 1) * h)) * (tV1[i] - tV0[i]) +  
    h * mu02(x + n - ((i - 1) * h)) * (S - tV0[i])  
  tV1[i + 1] = tV1[i] * (1 - d * h) + B * h + h * mu10(x + n - ((i - 1) * h)) * (tV0[i] - tV1[i]) +  
    h * mu12(x + n - ((i - 1) * h)) * (S - tV1[i])  
}
```

Starting from (at time $n = 20$)

```
tV0[1]; tV1[1]  
## [1] 0  
## [1] 0
```

Ultimately, we retrieve (for $P = 5500$) at time $n = 10$ and $n = 0$

```
tV0[120 + 1] ; tV1[120 + 1]  
## [1] 18083.95  
## [1] 829731.3  
tV0[120 * 2 + 1]  
## [1] 3815.348
```

Now you can rerun the code and find the results corresponding to $P = 6000$!

Alternatively, we search the equivalence premium P for which ${}_0V^{(0)} = 0$.

```
## Finding the equivalence premium by defining a function and using the function uniroot ##
f <- function(P, decades = 2) {
  for(i in seq_len(120 * decades)) {
    tV0[i + 1] = tV0[i] * (1 - d * h) - P * h + h * mu01(x + n - ((i - 1) * h)) * (tV1[i] - tV0[i]) +
      h * mu02(x + n - ((i - 1) * h)) * (S - tV0[i])
    tV1[i + 1] = tV1[i] * (1 - d * h) + B * h + h * mu10(x + n - ((i - 1) * h)) * (tV0[i] - tV1[i]) +
      h * mu12(x + n - ((i - 1) * h)) * (S - tV1[i])
  }
  return(tV0[120 * decades + 1])
}
```

We search the premium P , say between 5 000 and 6 000, for which the above function becomes zero.

```
(P = uniroot(f, c(5e3, 6e3))$root)
## [1] 5796.594
```

Assuming ${}_0V^{(0)}$ is a linear function of P the book retrieved a value of 5796.59 for the equivalence premium.

Thanks!



Slides created with the R package `xaringan`.

Course material available via

 <https://github.com/katrienantonio/mortality-dynamics>