An MPM with recurrent removal (Hastings *et al.* 2006) - eq. 3

Bram D'hondt

2025-12-19

Subject

Hastings, A., Hall, R. J., & Taylor, C. M. (2006). A simple approach to optimal control of invasive species. Theoretical population biology, 70(4), 431-435.

$$N_T = L^T N_0 - \sum_{i=1}^{T} L^{T+1-i} H_i$$

Equation 3 in Hastings *et al.* (2006) extends the base formula for matrix population model projections with a scenario of recurrent removal. The equation is then used for the optimization of population control, through linear programming techniques. Indeed, such removal-based management is the standard approach to invasive species control, but in practice, calculations are almost never made. This explains my interest in developing on it (here).

Introduction

This is a preparatory document that develops the above formula step by step, using a simple example. This not only serves as an illustration, but it also acts as a reference to check whether the developed function, the scripting of which is way more abstract, is correct.

First, I avoid using *popdemo* functions, to stay closer to the core formulation. Then, I provide the *popdemo* alternative, which is heavily used in further development. Note how the *popdemo::project()* function has the huge advantage that, for a given time span, all intermediate values are provided. Also, by using the option *return.vec = FALSE/TRUE*, one can switch between output per stage or summed over stages.

As a time span, we opt for five steps (years), i.e. $T = 5$.

L, N_0 and H

L is the transition matrix. In this example, we assume three stages: seedlings, isolates and meadows. (In the R script, these stages can easily be added as row names.)

$$L = \begin{bmatrix} 0 & 0.1 & 0.1 \\ 1 & 1 & 1 \\ 0 & 0.3 & 1.2 \end{bmatrix}$$

N_0 represents the initial population. Of course, the stages follow the above order.

$$N_0 = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \end{bmatrix}$$

H is the area removed . Removal is performed at the **start of the year**. This is important to understand, because for one thing, it means that you will nowhere see the exact value of H in the 'ghost matrix' (see below). The first value you will encounter directly is LH, i.e. the end-of-year 'ghost' of what was newly removed at the start of the year.

$$H = \begin{bmatrix} 0 \\ 0.1 \\ 0 \end{bmatrix}$$

Without removal

If there would be **no removal**, an MPM's base formula gives the population extent.

$$N_T = L^T N_0$$

Without *popdemo*, this can be scripted with the following code, the results of which are presented just below (N_0 (summed) to N_5).

```
N1 <- sum(L%^%1 %*% N0) # expm package
N2 <- sum(L%^%2 %*% N0) # matrix power (L^2 = element-wise, L%^%2 = L x L)
N3 <- sum(L%^%3 %*% N0)
N4 <- sum(L%^%4 %*% N0)
N5 <- sum(L%^%5 %*% N0)
```

```
## [1] 0.30000 0.47000 0.78500 1.32400 2.24250 3.80365
```

With *popdemo*, the same results can be obtained through the *project()* function. Note how N_0 is automatically included in the output, and how the results can be shown per stage.

```
Nt <- project(L, vector = N0, time = 5, return.vec = T)
```

```
print(Nt)        # across stages
```

```
## 1 deterministic population projection over 5 time intervals.
##
## [1] 0.30000 0.47000 0.78500 1.32400 2.24250 3.80365
```

```
print(t(Nt@vec)) # per stage
```

```
##             [,1] [,2]  [,3]  [,4]   [,5]    [,6]
## seedlings  0.1 0.02 0.045 0.074 0.1250 0.21175
## isolates   0.1 0.30 0.470 0.785 1.3240 2.24250
## meadows    0.1 0.15 0.270 0.465 0.7935 1.34940
```

With removal

Let's now include **removal**. The equation at the start of this document states that from this total projection, we now have to substract a contingent of "what could have been", i.e. that which would have resulted from the removed portion ($-L^T H$), for each of the time steps that have passed ($sum_{i=1}^T$). I refer to the removed cohorts and how they would have propagated as 'ghosts', their yearly numbers collected in the *ghost matrix*.

Below, I script this first without *popdemo*, one then again with *popdemo::project()*. Ensure how the output is identical. In the next step, we will build upon the latter approach to make a general formulation.

```
total  = sum(L%^%5 %*% N0)         # total (actual + removed) - as above

ghost1 = sum(L%^%(5+1-1) %*% H)    # removed in year 1 (five years ago)
ghost2 = sum(L%^%(5+1-2) %*% H)    # removed in year 2 (four years ago)
ghost3 = sum(L%^%(5+1-3) %*% H)    # removed in year 3 (three years ago)
ghost4 = sum(L%^%(5+1-4) %*% H)    # removed in year 4 (two years ago)
ghost5 = sum(L%^%(5+1-5) %*% H)    # removed in year 5 (previous year)

total-sum(ghost1, ghost2, ghost3, ghost4, ghost5)
```

```
## [1] 1.453128
```

```
total.  = project(L, vector = N0, time = 5, return.vec = F)     # removed in ...
ghost.1 = project(L, vector = H,  time = 5+1-1, return.vec = F) # ... year 1 (five years ago)
ghost.2 = project(L, vector = H,  time = 5+1-2, return.vec = F) # ... year 2 (four years ago)
ghost.3 = project(L, vector = H,  time = 5+1-3, return.vec = F) # ... year 3 (three years ago)
ghost.4 = project(L, vector = H,  time = 5+1-4, return.vec = F) # ... year 4 (two years ago)
ghost.5 = project(L, vector = H,  time = 5+1-5, return.vec = F) # ... year 5 (previous year)

tail(total.,1)-(tail(ghost.1,1)+tail(ghost.2,1)+tail(ghost.3,1)+tail(ghost.4,1)+tail(ghost.5,1))
```

```
## [1] 1.453128
```

Generic formulation

The formulation below constructs a full ghost matrix

```
mpm_removal <- function(start.vec, trans.mat, removal, T){

  # "total population" matrix
  total <- project(trans.mat,
                   vector = start.vec,
                   time = T,
                   return.vec = F) # yields vector of length T+1

  # "ghost population" matrix
  ## template
  ghost <- matrix(data = 0,
                  nrow = T+1,
                  ncol = T+1) # all zeros, same length of 'total'
  ## populate
  for (i in 1:T)
  {
    propagation <- project(trans.mat,
                            vector = removal,
                            time = T-i+1,
                            return.vec = F)
    ghost[i,(i+1):ncol(ghost)] <- tail(propagation, -1) # copy in values, apart from first
  }
  ## names
  colnames(ghost) <- paste0("n", 0:(ncol(ghost) - 1))

  # "net population" matrix
  net <- total - colSums(ghost)

  # output
  return(list(total = total,
              ghost = ghost,
              net   = net))
  }
```

((with H fixed over time, note that the same series of numbers appears in each row, so there exist alternative (easier) ways of coding. The current way is preferred as it allows for time-varying H.))

E.g., this function applied to the above example, yields the following output.

```
A <- mpm_removal(start.vec = N0,
                 trans.mat = L,
                 removal = H,
                 T = 5)
```

```
## $total
## 1 deterministic population projection over 5 time intervals.
##
## [1] 0.30000 0.47000 0.78500 1.32400 2.24250 3.80365
##
## $ghost
##      n0   n1    n2     n3      n4       n5
## [1,]  0 0.14 0.219 0.3608 0.60596 1.024762
## [2,]  0 0.00 0.140 0.2190 0.36080 0.605960
## [3,]  0 0.00 0.000 0.1400 0.21900 0.360800
## [4,]  0 0.00 0.000 0.0000 0.14000 0.219000
## [5,]  0 0.00 0.000 0.0000 0.00000 0.140000
## [6,]  0 0.00 0.000 0.0000 0.00000 0.000000
##
## $net
## 1 deterministic population projection over 5 time intervals.
##
## [1] 0.300000 0.330000 0.426000 0.604200 0.916740 1.453128
```