Document 2- R codes for the asymptotic analyses of a stage-structured matrix model for spotted hyenas infected with CDV

S. Benhaiem, L. Marescot

2018-10-11 15:09:32

This code presents the asymptotic analysis of the matrix model. For each epidemic period, we first load the input files containing the MECMR parameter estimates for that period and then the R script to construct the matrix model. It requires the packages ‘popbio’ 2.4.4 (Stubben and Milligan 2007) and ‘popdemo’ 1.3-0 (Stott et al. 2012).

This code is structured as follows:

a) Checking assumptions (irreducibility and ergodicity)
b) Asymptotic analyses (population’s growth rate ((λ)), R0, stable stage distribution and reproductive values)
   i) Pre-epidemic
   ii) Epidemic
   iii) Post-epidemic
   iv) Short-term population dynamics (Fig 4)

Note that parameter and submatrix names may differ between main text and R codes.

```
start_time <- Sys.time()

library(popdemo)
```

```
## Warning: package 'popdemo' was built under R version 3.5.1
```

```
library(popbio)

source('Input_90-92(pre-epidem).R')
source('1_Model construction.R')
M.finalpre<-M.final
NGMpre<-NGM
source('Input_93-94(epidem).R')
source('1_Model construction.R')
M.finalepidem<-M.final
NGMepidem<-NGM
source('Input_95-99(post-epidem).R')
source('1_Model construction.R')
M.finalpost<-M.final
NGMpost<-NGM
```

a) Checking assumptions (irreducibility and ergodicity)

Matrix models are termed ‘irreducible’ when their associated life cycles contain the transition rates to facilitate pathways from all states to all other states. Irreducible matrices are ‘ergodic’: the stable asymptotic growth rate is independent from the initial stage structure in the population projection. Both conditions should ideally be met for further analyses.

Pre-epidemic period: Is the matrix model irreducible?

```
isIrreducible(M.finalpre)
```

```
## [1] TRUE
```

Is the matrix model ergodic?

```
isErgodic(M.finalpre)
```

```
## [1] TRUE
```

Epidemic period: Is the matrix model irreducible?

```
isIrreducible(M.finalepidem)
```

```
## [1] TRUE
```

Is the matrix model ergodic?

```
isErgodic(M.finalepidem)
```

```
## [1] TRUE
```

Post-epidemic period: Is the matrix model irreducible?

```
isIrreducible(M.finalpost)
```

```
## [1] TRUE
```

Is the matrix model ergodic?

```
isErgodic(M.finalpost)
```

```
## [1] TRUE
```

b) Asymptotic analyses (population's growth rate ((λ)), R0, stable stage distribution and reproductive values)

For each epidemic period, we calculate 1) the population's growth rate ((λ)), which is the dominant eigenvalue of the final matrix, 2) The basic reproduction number R0, which is the dominant eigenvalue of the next generation matrix, 3) the stable stage distribution, which is the right eigenvector of the final matrix and 4) the reproductive values, which are the left eigenvector of the final matrix.

Note that the values of (λ) and R0 differ from the ones presented in the main text (Figure 1), as we presented there the mean + SD values estimated via stochastic analyses.

i) Pre-epidemic

(λ):

```
lambda(M.finalpre)
```

```
## [1] 1.099064
```

R0:

```
lambda(NGMpre)
```

```
## [1] 0.8413641
```

Stable-stage distribution:

```
rightpre<-as.data.frame(stable.stage(M.finalpre))
rightpre
```

```
##      stable.stage(M.finalpre)
## CLS              0.0603185784
## CLI              0.0363253727
## CHS              0.0720666478
## CHI              0.0191654840
## SLS              0.0483365030
## SLI              0.0007995175
## SLR              0.0228499253
## SHS              0.0574760886
## SHI              0.0043330267
## SHR              0.0120557573
## BLS              0.1005566201
## BLI              0.0021645219
## BLR              0.0745526419
## BHS              0.0679304870
## BHI              0.0071792410
## BHR              0.0690763385
## NBLS             0.1175436411
## NBLI             0.0025301744
## NBLR             0.0783197461
## NBHS             0.0798741749
## NBHI             0.0084415110
## NBHR             0.0581040007
```

Reproductive values:

```
ReproValuespre <- as.data.frame(reproductive.value(M.finalpre))
ReproValuespre
```

```
##      reproductive.value(M.finalpre)
## CLS                       1.0000000
## CLI                       0.7721931
## CHS                       1.9472853
## CHI                       1.4281903
## SLS                       1.2275836
## SLI                       1.2275836
## SLR                       1.2275836
## SHS                       2.2704470
## SHI                       2.2704470
## SHR                       2.2704470
## BLS                       1.9446977
## BLI                       1.9446977
## BLR                       1.9446977
## BHS                       3.8923289
## BHI                       3.8923289
## BHR                       3.8923289
## NBLS                      1.5207977
## NBLI                      1.5207977
## NBLR                      1.5207977
## NBHS                      2.9253673
## NBHI                      2.9253673
## NBHR                          2.9253673
```

ii) Epidemic

(λ):

```
lambda(M.finalepidem)
```

```
## [1] 0.9614995
```

R0:

```
lambda(NGMepidem)
```

```
## [1] 5.83354
```

Stable-stage distribution:

```
rightepidem<-as.data.frame(stable.stage(M.finalepidem))
rightepidem
```

```
##      stable.stage(M.finalepidem)
## CLS                 1.782793e-03
## CLI                 8.166456e-02
## CHS                 4.099603e-03
## CHI                 8.954384e-02
## SLS                 5.299528e-04
## SLI                 6.769462e-04
## SLR                 4.719065e-02
## SHS                 3.875630e-04
## SHI                 2.141395e-03
## SHR                 5.174377e-02
## BLS                 4.127034e-05
## BLI                 7.908617e-05
## BLR                 2.084285e-01
## BHS                 3.389525e-06
## BHI                 3.189068e-05
## BHR                 1.667652e-01
## NBLS                1.733372e-04
## NBLI                3.321652e-04
## NBLR                1.993191e-01
## NBHS                2.930318e-05
## NBHI                2.757019e-04
## NBHR                1.447600e-01
```

Reproductive values:

```
ReproValuesepidem <- as.data.frame(reproductive.value(M.finalepidem))
ReproValuesepidem
```

```
##      reproductive.value(M.finalepidem)
## CLS                          1.0000000
## CLI                          0.8535953
## CHS                          1.6536877
## CHI                          1.5490872
## SLS                          1.4771673
## SLI                          1.4771673
## SLR                          1.4771673
## SHS                          2.6807328
## SHI                              2.6807328
## SHR                              2.6807328
## BLS                              2.0432398
## BLI                              2.0432398
## BLR                              2.0432398
## BHS                              4.0243682
## BHI                              4.0243682
## BHR                              4.0243682
## NBLS                             1.7766077
## NBLI                             1.7766077
## NBLR                             1.7766077
## NBHS                             3.3536892
## NBHI                             3.3536892
## NBHR                             3.3536892
```

ii) Post-epidemic

(λ):

```r
lambda(M.finalpost)
```

```
## [1] 1.017958
```

R0:

```r
lambda(NGMpost)
```

```
## [1] 2.671083
```

Stable-stage distribution:

```r
rightpost<-as.data.frame(stable.stage(M.finalpost))
rightpost
```

```
##      stable.stage(M.finalpost)
## CLS                0.018337760
## CLI                0.067621378
## CHS                0.035824077
## CHI                0.061114312
## SLS                0.013573212
## SLI                0.001388742
## SLR                0.040534907
## SHS                0.019684864
## SHI                0.008916139
## SHR                0.036634317
## BLS                0.017616961
## BLI                0.002547518
## BLR                0.170183176
## BHS                0.006170841
## BHI                0.004381201
## BHR                0.151089192
## NBLS               0.023105993
## NBLI               0.003341265
## NBLR               0.167378231
## NBHS               0.011764996
## NBHI               0.008352963
## NBHR               0.130437955
```

Reproductive values:

```
ReproValuespost <- as.data.frame(reproductive.value(M.finalpost))
ReproValuespost
```

```
##      reproductive.value(M.finalpost)
## CLS                        1.0000000
## CLI                        0.7346883
## CHS                        1.8868305
## CHI                        1.4166803
## SLS                        1.2256260
## SLI                        1.2256260
## SLR                        1.2256260
## SHS                        2.3633425
## SHI                        2.3633425
## SHR                        2.3633425
## BLS                        1.7773824
## BLI                        1.7773824
## BLR                        1.7773824
## BHS                        3.7592258
## BHI                        3.7592258
## BHR                        3.7592258
## NBLS                       1.4863734
## NBLI                       1.4863734
## NBLR                       1.4863734
## NBHS                       2.9944584
## NBHI                       2.9944584
## NBHR                       2.9944584
```

Summary table for λ, R0 and stable stage distribution for each epidemic period:

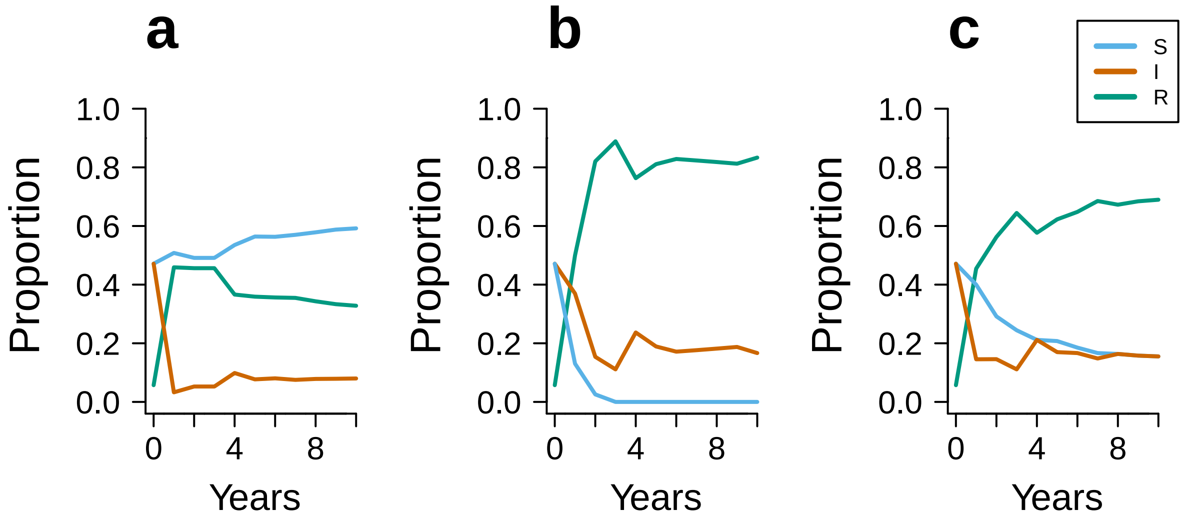
Population indicator	Pre-epidemic	Epidemic	Post-epidemic
λ	1.0990637	0.9614995	1.0179581
R0	0.8413641	5.8335396	2.6710826
Stable stage:Low-ranking	0.5442972	0.5402183	0.5256291
Stable stage:High-ranking	0.4557028	0.4597817	0.4743709
Stable stage:Susceptible	0.6041027	0.0070472	0.1460787
Stable stage:Infected	0.0809388	0.1747456	0.1576635
Stable stage:Recovered	0.3149584	0.8182072	0.6962578
Stable stage:Cubs	0.1878761	0.1770908	0.1878761
Stable stage:Subadults	0.1458508	0.1026703	0.1026703
Stable stage:Breeders	0.3214599	0.3753494	0.3753494
Stable stage:Non-Breeders	0.3448132	0.3448896	0.3448896

iv) Short-term population dynamics

```
#Initial values from E-surge's excel file

initialvalues <-c(0.12639981,0.12639981,# CL
      0.161066754,0.161066754, # CH
      0.002133319,0.002133319,0.002133319,# SAL
      0.00533333, 0.00533333, 0.00533333,# SAH
      0.005333321, 0.005333321, 0.005333321,# BL
      0.007629848, 0.007629848, 0.007629848, # BH
      0.010133336, 0.010133336, 0.010133336,# NBL
      0.012266641, 0.012266641, 0.012266641)# NBH


# Modifying the stage.vector.plot function to enhance figure
stage.vector.plot.mod <-
function (stage.vectors, proportions = TRUE,
            ylim = NULL, xlab = "Years", ylab = NULL, col = rainbow(8),
            ...)
  {
    p <- stage.vectors
    n <- dim(p)[1]
    if (is.null(n)) {
      stop("stage.vectors should be a matrix with two or more stages")
    }
    x <- colnames(p)
    if (is.null(x)) {
      x <- 0:(dim(p)[2] - 1)
    }
    if (length(col) < n) {
      col <- rep(col, n)
    }
    if (proportions) {
      if (is.null(ylab)) {
        ylab <- "Proportion in stage class"
      }
      p <- prop.table(p, 2)
      if (is.null(ylim)) {
        ylim = c(min(p, na.rm = TRUE), max(p, na.rm = TRUE))
      }
      plot(x, p[1, ], type = "n", ylim = ylim, xlab = xlab,
           ylab = ylab, ...)
    }
    else {
      if (is.null(ylab)) {
        ylab <- "Number in stage class"
      }
      if (is.null(ylim)) {
        ylim = c(floor(min(p, na.rm = TRUE)), ceiling(max(p,
                                                          na.rm = TRUE)))
      }
      plot(x, p[1, ], type = "n", ylim = ylim, xlab = xlab,
           ylab = ylab, ...)
    }
    y <- sort(apply(p, 1, mean, na.rm = TRUE), index.return = TRUE,
              decreasing = TRUE)
    for (i in y$ix) {
      lines(x, p[i, ], lty = 1, col = col[i], lwd = 2)
    }
    leg.names <- paste(names(y$x), "")
    if (leg.names[1] == " ") {
      leg.names <- paste("row", y$ix, "")
    }
  }

# The function transdyn is to get the SIR states for each period for the short term projections

transdyn<-function(M.final,n,horizon)
{
  p<-pop.projection(M.final,n,horizon)# Project over 11 years

 all <-as.data.frame(p$stage.vectors)
 colnames(all) <- paste0(c(0:10)) # from 0 to x-1

## We create a new dataframe with 3 rows, S,I,R, which are the sum of the detailed states
# first empty table

 sir <- matrix(0, ncol = 11, nrow = 3) #col = x
 colnames(sir) <- paste0(c(0:10)) # from 0 to x-1
 rownames(sir) <- c("S","I","R")

#then we fill it
  for (i in names(all)){
  sir[1,i] <- round(sum(all[c(1,3,5,8,11,14,17,20),i]),digits=2) # Susceptible
  sir[2,i] <- round(sum(all[c(2,4,6,9,12,15,18,21),i]),digits=2) # Infected
  sir[3,i] <- round(sum(all[c(7,10,13,16,19,22),i]),digits=2) # Recovered
  }
  return(sir)
}

horizon<-11 # Number of projected years

sirpre<-transdyn(M.finalpre,initialvalues,horizon)

sirepidem<-transdyn(M.finalepidem,initialvalues,horizon)

sirpost<-transdyn(M.finalpost,initialvalues,horizon)
```

Figure 4 - Short-term dynamics (first ten years) of the proportion of different infection states and their convergence to a stable stage distribution (using the time invariant initial state vector projection from the MECMR model), for susceptible (blue), infected (red) and recovered (green) females (across all demographic and social states) during a) pre-epidem, b) epidem, c) post-epidem.

```
## Warning in if (axes) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (frame.plot) localBox(...): the condition has length > 1 and
## only the first element will be used

## Warning in if (axes) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (frame.plot) localBox(...): the condition has length > 1 and
## only the first element will be used

## Warning in if (axes) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (frame.plot) localBox(...): the condition has length > 1 and
## only the first element will be used
```



```
## Time difference of 1.27 secs
```

References (R packages)

Stubben, C. & Milligan, B. Estimating and analyzing demographic models using the popbio package in R. J. Stat. Softw 22, 1-23 (2007).

Stott, I., Hodgson, D. J. & Townley, S. popdemo: an R package for population demography using projection matrix analysis. Methods Ecol. Evol. 3, 797-802 (2012).