Leslie matrix

Marc Girondot

13 septembre 2022

Load Packages

Package popbio is explained here:

Stubben C, Milligan B (2007) Estimating and analyzing demographic models using the popbio package in R. Journal of Statistical Software 22: 1-23

```
suppressPackageStartupMessages(library(popbio))
```

Population dynamics of *Rattus norvegicus*

Model definition

```
A <- matrix(c(0, 0.3, 0.8, 0.7, 0.4, 0.1,
              0.6, 0, 0, 0, 0, 0,
              0, 0.9, 0, 0, 0, 0,
              0, 0, 0.9, 0, 0, 0,
              0, 0, 0, 0.8, 0, 0,
              0, 0, 0, 0, 0.6, 0), nrow = 6, byrow = TRUE)
A
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]  0.0  0.3  0.8  0.7  0.4  0.1
## [2,]  0.6  0.0  0.0  0.0  0.0  0.0
## [3,]  0.0  0.9  0.0  0.0  0.0  0.0
## [4,]  0.0  0.0  0.9  0.0  0.0  0.0
## [5,]  0.0  0.0  0.0  0.8  0.0  0.0
## [6,]  0.0  0.0  0.0  0.0  0.6  0.0
```

```
# initial population vector N
N0 <- matrix(c(15, 9, 13, 5, 0, 0), ncol = 1)
N0
```

```
##      [,1]
## [1,]   15
## [2,]    9
## [3,]   13
## [4,]    5
## [5,]    0
## [6,]    0
```

Run the model

8 generations

```
N1 <- A %*% N0
N1
```

```
##      [,1]
## [1,] 16.6
## [2,]  9.0
## [3,]  8.1
## [4,] 11.7
## [5,]  4.0
## [6,]  0.0
```

```
N2 <- A %*% N1
N2
```

```
##       [,1]
## [1,] 18.97
## [2,]  9.96
## [3,]  8.10
## [4,]  7.29
## [5,]  9.36
## [6,]  2.40
```

```
N3 <- A %*% N2
N4 <- A %*% N3
N5 <- A %*% N4
N6 <- A %*% N5
N7 <- A %*% N6
N8 <- A %*% N7
N8
```

```
##           [,1]
## [1,] 21.966141
## [2,] 12.752105
## [3,] 11.202776
## [4,]  9.654380
## [5,]  7.225148
## [6,]  4.328510
```

The same for 100 generations using the package popbio.

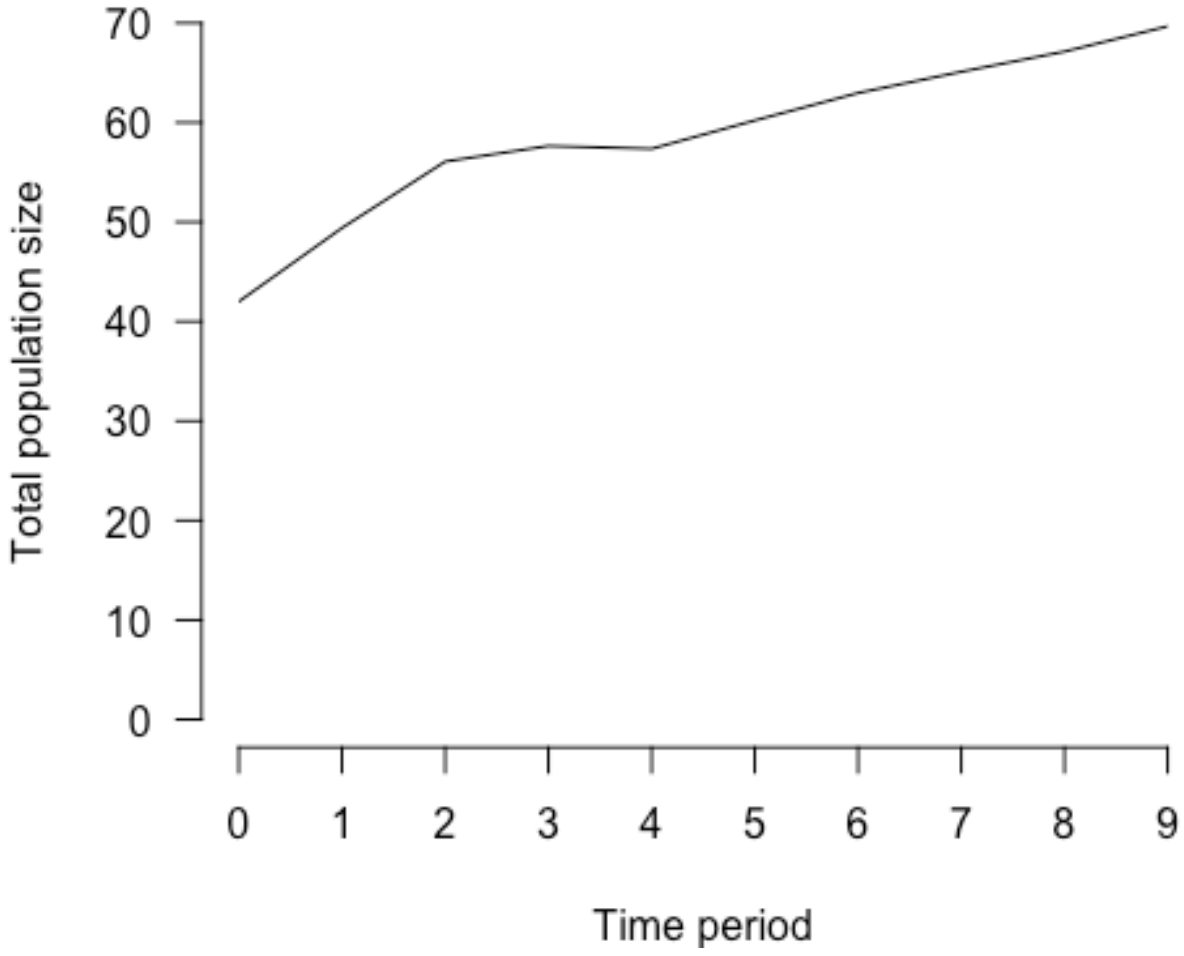
```
N <- pop.projection(A, N0, 100)
```

Plot the results

Total population size

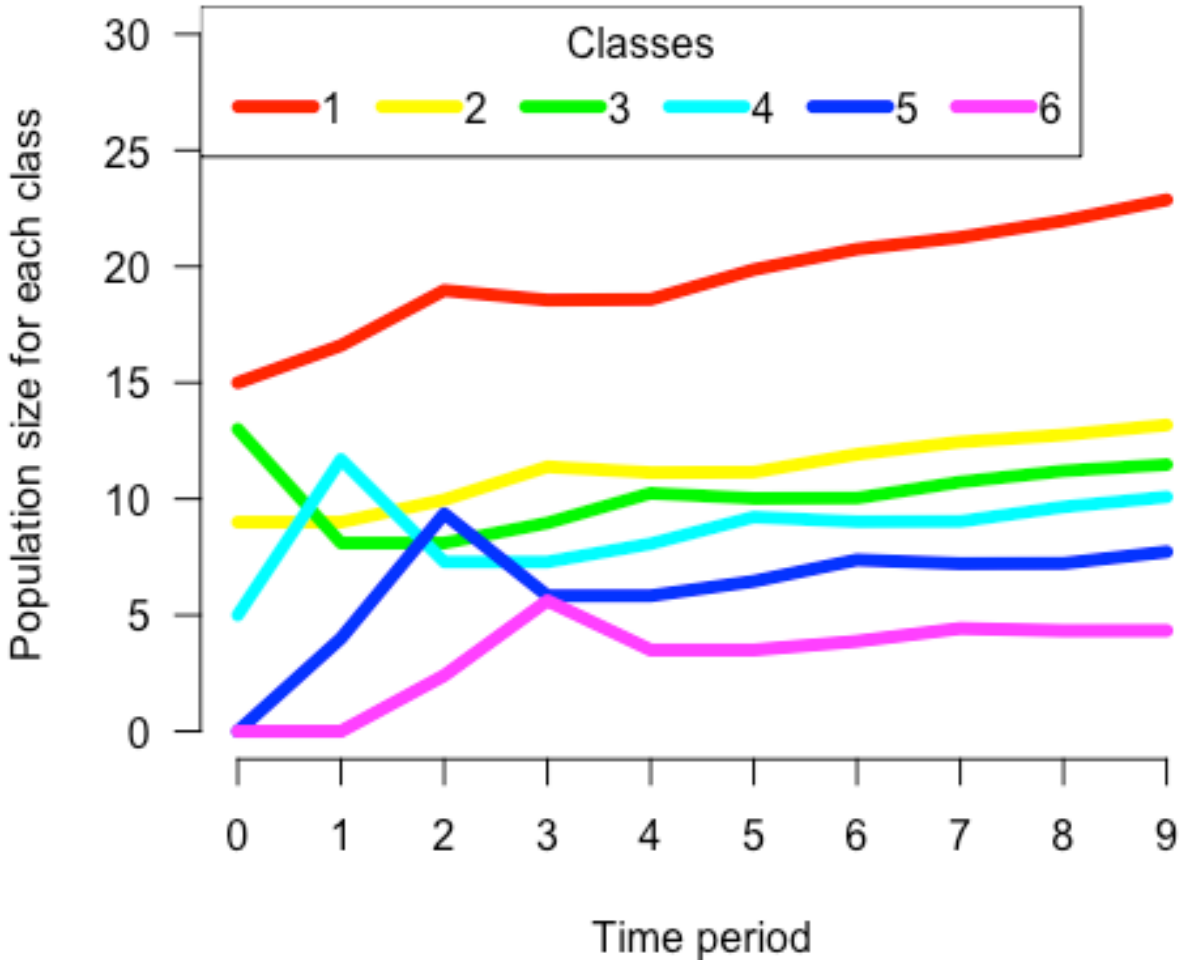
```
par(mar=c(4, 4, 1, 1))
plot(1:10, N$pop.sizes[1:10], type="l", xlab="Time period",
```

```
     ylab="Total population size", las=1, bty="n",
     ylim=c(0, 70), xaxt="n")
axis(1, at=1:10, labels = as.character(0:9))
```



Population size for each class

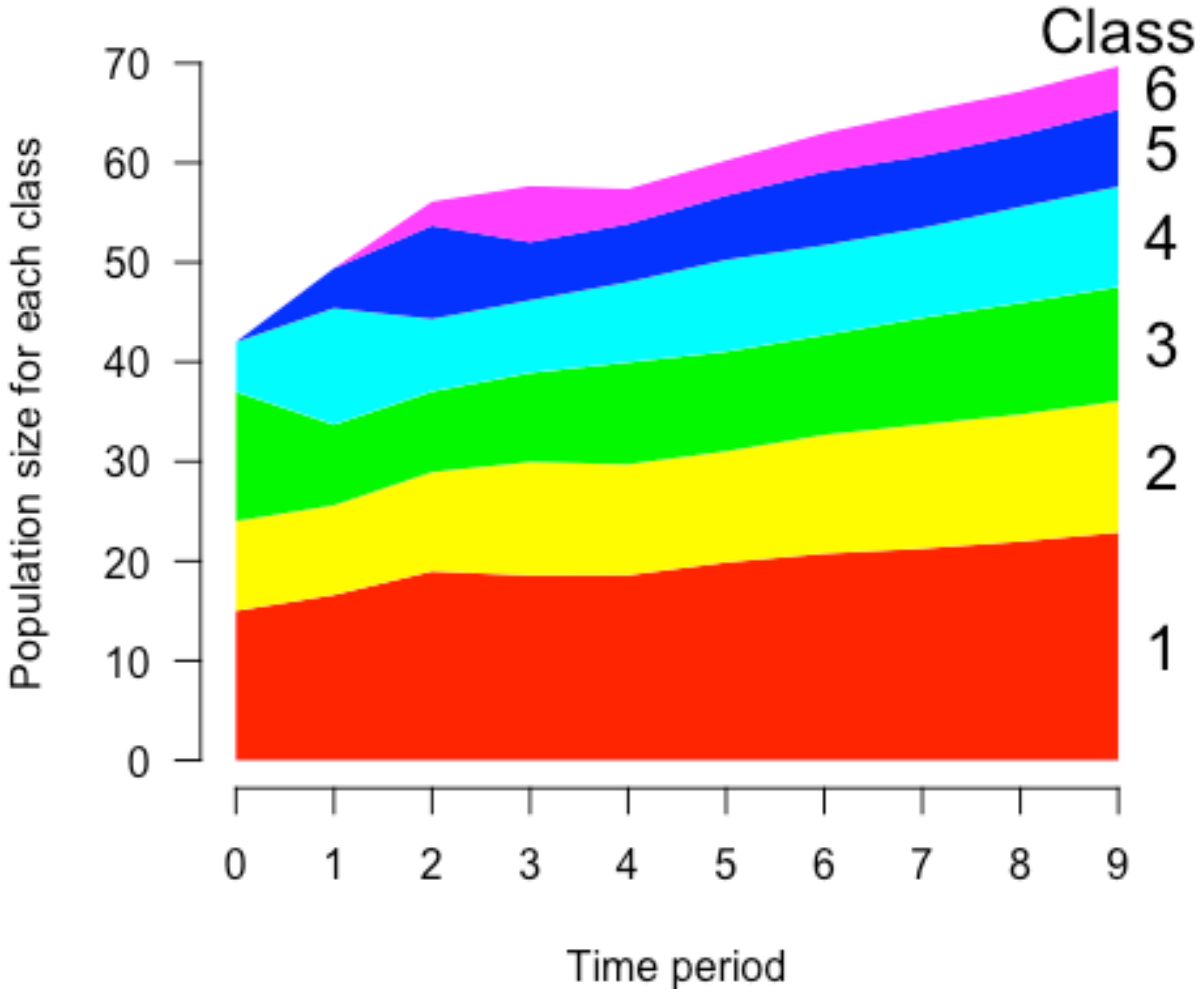
```
par(mar=c(4, 4, 1, 1))
plot(1:10, N$pop.sizes[1:10], type="n", xlab="Time period",
     ylab="Population size for each class", las=1, bty="n",
     ylim=c(0, 30), xaxt="n")
axis(1, at=1:10, labels = as.character(0:9))

for (i in 1:6) {
  lines(x=1:10, y=N$stage.vectors[i, 1:10], col=rainbow(6)[i], lwd=5)
}

legend("topleft", legend = as.character(1:6), lty=1, col=rainbow(6), horiz =
TRUE,
       lwd=5, title = "Classes", cex = 1, x.intersp = 0.2)
```



Cumulative population size for each class

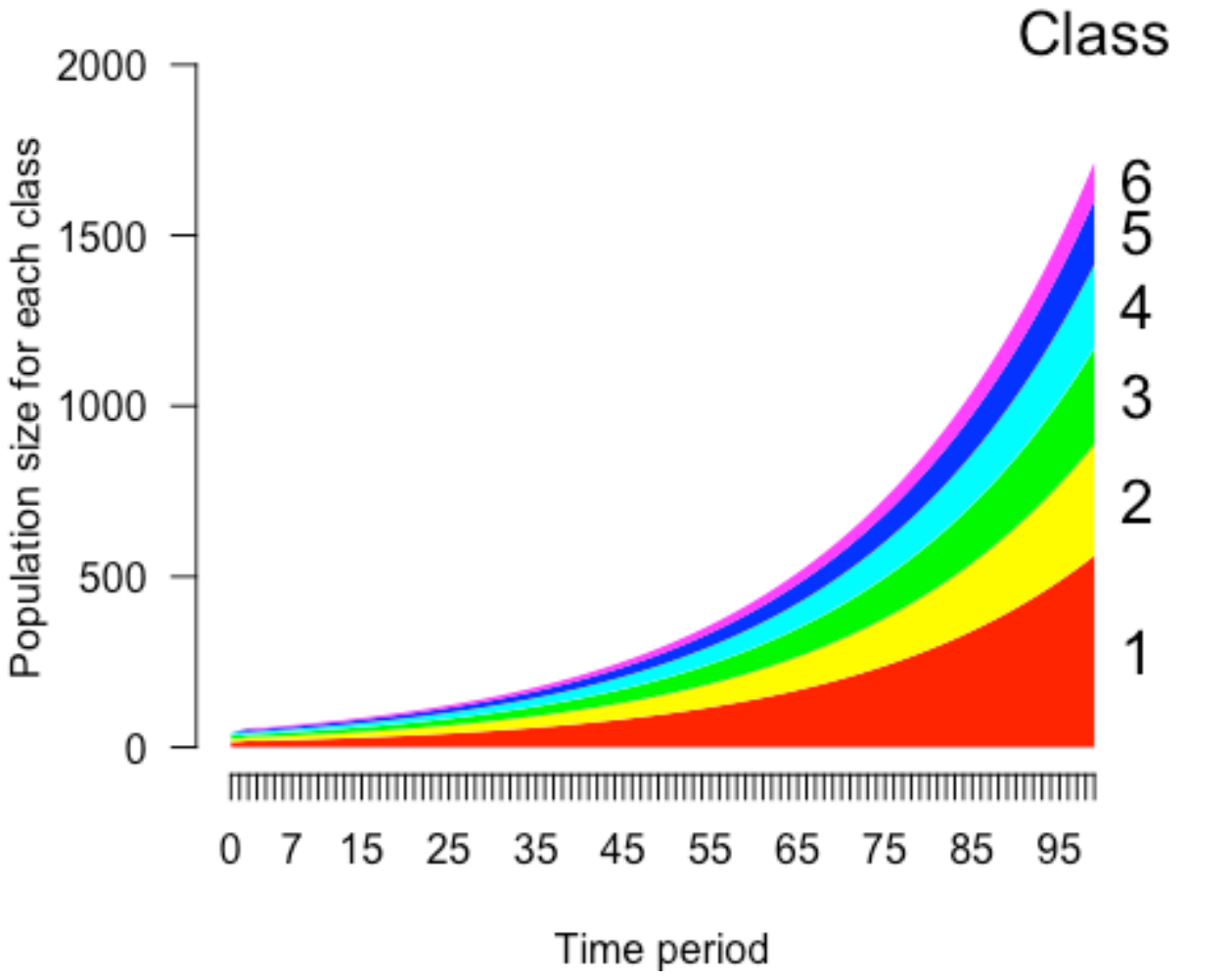
```
library(HelpersMG)

## Le chargement a nécessité le package : MASS

## Le chargement a nécessité le package : ggplot2

## Le chargement a nécessité le package : rlang

## Le chargement a nécessité le package : coda

## Le chargement a nécessité le package : Matrix

## Welcome in package HelpersMG version 5.6

## No update is available

par(mar=c(4, 4, 1, 2))
plot(1:10, N$pop.sizes[1:10], type="n", xlab="Time period", ylab="Population
size for each class", las=1, bty="n",
     ylim=c(0, 70), xaxt="n")
```

```
axis(1, at=1:10, labels = as.character(0:9))

nx <- rep(0, 10)
px <- NULL
par(xpd=TRUE)

for (i in 1:6) {
  polygon(x=c(1:10, 10:1), y=c(nx, rev(nx+N$stage.vectors[i, 1:10])), col=rai
nbow(6)[i], border = NA)
  px <- nx[10]+ N$stage.vectors[i, 10]/2
  text(x=ScalePreviousPlot(x=1.05, y=0)$x, y=px, labels=as.character(i), cex=
1.5)
  nx <- nx + N$stage.vectors[i, 1:10]
}

text(x=ScalePreviousPlot(x=1, y=1.05)$x, y=ScalePreviousPlot(x=1, y=1.05)$y,
labels="Class", cex=1.5)
```



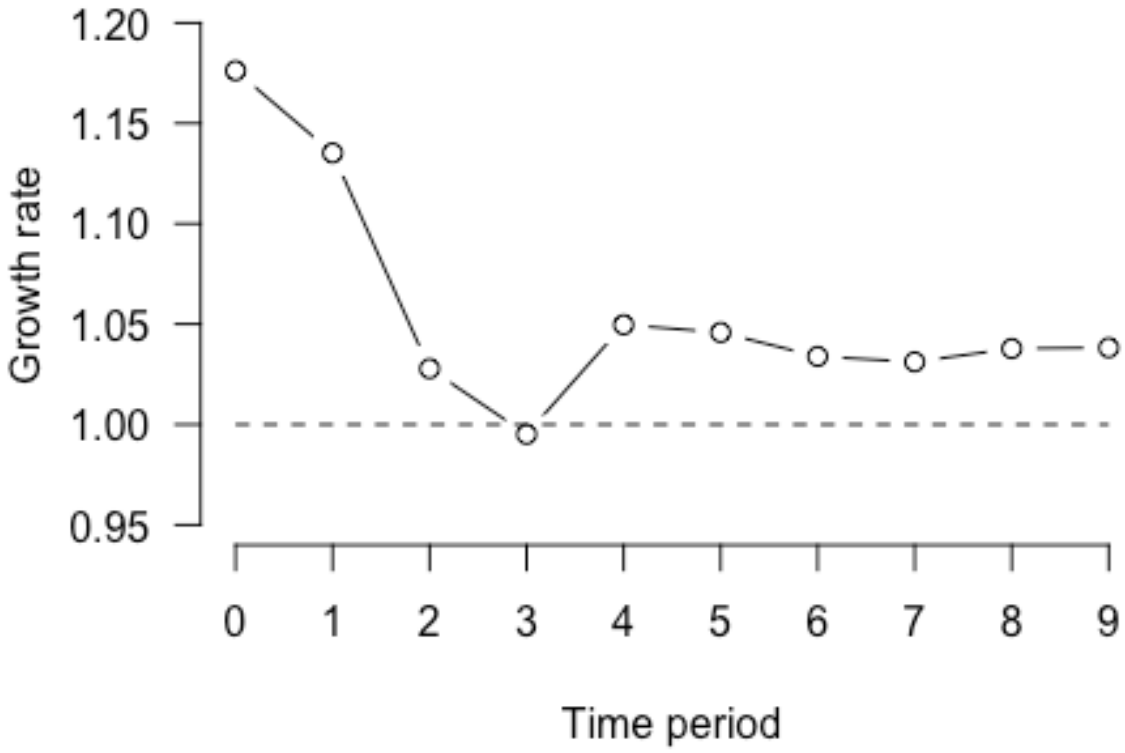

```
library(HelpersMG)
par(mar=c(4, 4, 1, 2))
```

```
plot(1:100, N$pop.sizes[1:100], type="n", xlab="Time period", ylab="Populatio
n size for each class", las=1, bty="n",
     ylim=c(0, 2000), xaxt="n")
axis(1, at=1:100, labels = as.character(0:99))

nx <- rep(0, 100)
px <- NULL
par(xpd=TRUE)

for (i in 1:6) {
  polygon(x=c(1:100, 100:1), y=c(nx, rev(nx+N$stage.vectors[i, 1:100])), col=
rainbow(6)[i], border = NA)
  px <- nx[100]+ N$stage.vectors[i, 100]/2
  text(x=ScalePreviousPlot(x=1.05, y=0)$x, y=px, labels=as.character(i), cex=
1.5)
  nx <- nx + N$stage.vectors[i, 1:100]
}

text(x=ScalePreviousPlot(x=1, y=1.05)$x, y=ScalePreviousPlot(x=1, y=1.05)$y,
labels="Class", cex=1.5)
```

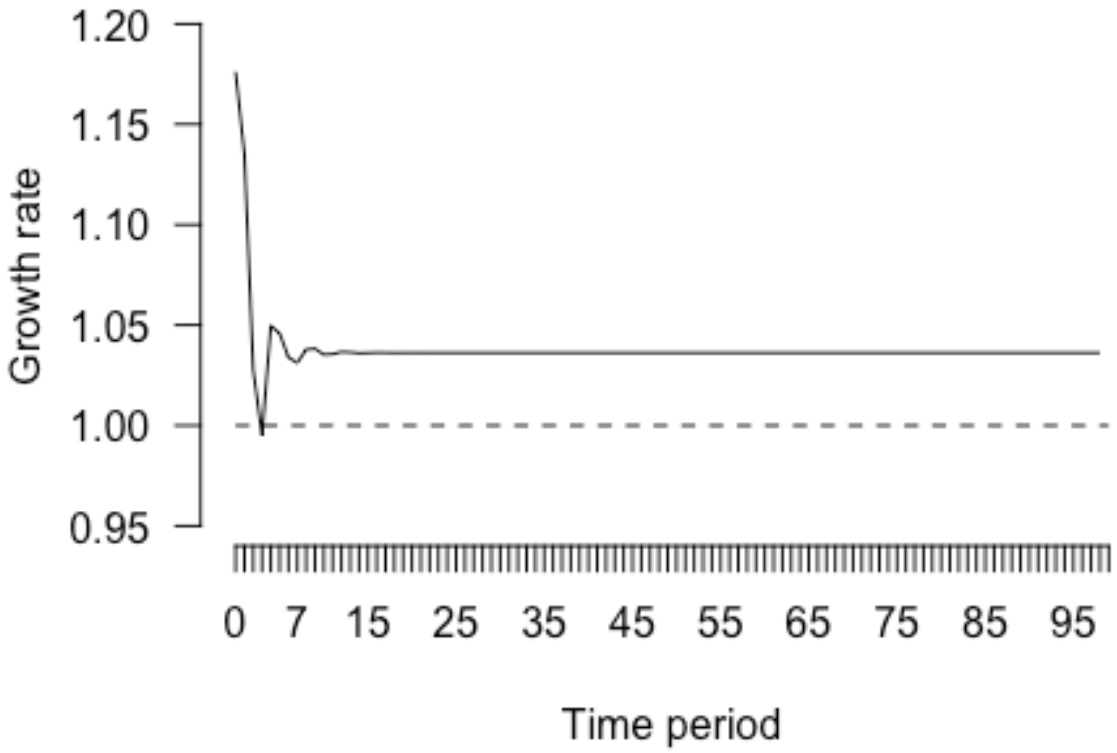


Growth rate

```
plot(1:10, N$pop.changes[1:10], type="b", xlab="Time period", ylab="Growth ra
te", las=1, bty="n",
     ylim=c(0.95, 1.2), xaxt="n")
axis(1, at=1:10, labels = as.character(0:9))
segments(x0=1, x1=10, y0=1, y1=1, lty=2)
```




```
plot(1:100, N$pop.changes[1:100], type="l", xlab="Time period", ylab="Growth
rate", las=1, bty="n",
     ylim=c(0.95, 1.2), xaxt="n")
axis(1, at=1:100, labels = as.character(0:99))
segments(x0=1, x1=100, y0=1, y1=1, lty=2)
```



Test if the matrix is primitive

```
pA <- A
for (k in 1:99) {
  pA <- pA %*% A
  if (all(as.vector(pA) != 0)) {
    message(paste0("C'est bon pour ",k, "\n"))
    break
  }
}
```

```
## C'est bon pour 7
```

Eigenvalues

Mannually

```
eigenA <- eigen(A)
pos <- which.max(Re(eigenA$values))
lambda <- Re(eigenA$values)[pos]
v <- Re(eigenA$vectors[, pos])
eigenA$vectors[, pos]
```

```
## [1] 0.7196588+0i 0.4167035+0i 0.3619253+0i 0.3143480+0i 0.2426889+0i
## [6] 0.1405240+0i

abs(eigenA$vectors[, pos])/sum(abs(eigenA$vectors[, pos]))

## [1] 0.32773612 0.18976879 0.16482252 0.14315559 0.11052169 0.06399529
```

Using popbio

The damping ratio measures the rate of convergence to a stable stage distribution and is determined by the ratio of the dominant eigenvalue to the second largest eigenvalue.

```
library(popbio)
eigen.analysis(A, zero = FALSE)

## $lambda1
## [1] 1.036217
##
## $stable.stage
## [1] 0.32773612 0.18976879 0.16482252 0.14315559 0.11052169 0.06399529
##
## $sensitivities
##           [,1]       [,2]       [,3]       [,4]        [,5]        [,6]
## [1,] 0.29033293 0.16811125 0.14601200 0.12681782 0.097908302 0.056691769
## [2,] 0.50141324 0.29033293 0.25216688 0.21901799 0.169090429 0.097908302
## [3,] 0.48052567 0.27823841 0.24166226 0.20989427 0.162046560 0.093829696
## [4,] 0.29518064 0.17091822 0.14844997 0.12893531 0.099543085 0.057638356
## [5,] 0.12829772 0.07428813 0.06452250 0.05604062 0.043265545 0.025052015
## [6,] 0.02801854 0.01622356 0.01409087 0.01223854 0.009448628 0.005471032
##
## $elasticities
##           [,1]       [,2]      [,3]       [,4]        [,5]        [,6]
## [1,] 0.0000000 0.04867067 0.1127270 0.08566976 0.037794513 0.005471032
## [2,] 0.2903329 0.00000000 0.0000000 0.00000000 0.000000000 0.000000000
## [3,] 0.0000000 0.24166226 0.0000000 0.00000000 0.000000000 0.000000000
## [4,] 0.0000000 0.00000000 0.1289353 0.00000000 0.000000000 0.000000000
## [5,] 0.0000000 0.00000000 0.0000000 0.04326554 0.000000000 0.000000000
## [6,] 0.0000000 0.00000000 0.0000000 0.00000000 0.005471032 0.000000000
##
## $repro.value
## [1] 1.00000000 1.72702851 1.65508498 1.01669708 0.44189863 0.09650487
##
## $damping.ratio
## [1] 1.582039
```