Math142 Homework2

Chen Li – 705426669

4/12/2020

Problem 1

a

Start from growth rate:

$$R(t) = \frac{N(t + \Delta t - N(t))}{\Delta t N(t)}$$

if we measure the population instaneously,

$$R(t) = \lim_{\Delta t \to 0} \frac{N(t + \Delta t - N(t))}{\Delta t N(t)} = \frac{1}{N(t)} \lim_{\Delta t \to 0} \frac{N(t + \Delta t - N(t))}{\Delta t} = \frac{1}{N} \frac{dN}{dt}$$

For a constant growth rate,

$$\frac{dN}{dt} = R_0 N$$

b Solve ODE

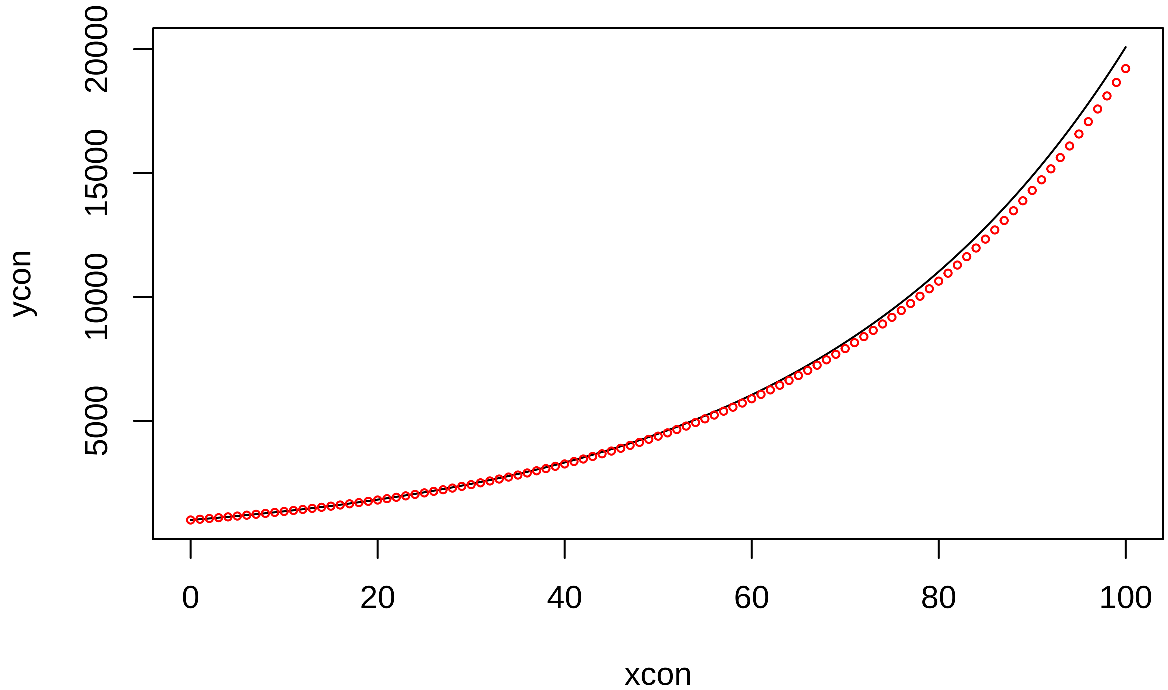
$$\frac{dN}{dt} = R_0 N$$
$$\frac{dN}{N} = R_0 dt$$
$$log(N) = R_0 t + C_1$$
$$N = C_2 e^{R_0 t}$$

Plug in initial condition we get:

$$N(t) = 1000e^{0.03t}$$

c

```
xcon <- seq(0, 100, len=10000)
ycon <- 1000 * exp(0.03*xcon)
xdis <- seq(0, 100)
ydis <- 1000*1.03^(xdis)
plot(xcon, ycon,type="l",col=1)
points(xdis,ydis,col=2,cex=0.5)
```



I realize the discrete is closer to contineous one than I thought, yet the distance is getting larger, And discrete is growing slower.

problem 2

(a)

```r
L <- rbind(c(1,1),c(1/2,3/2))
eig <- eigen(L)
eigv <- eig[[1]]
eigV <- eig[[2]] / rep(apply(eig[[2]], 2, min),each=2) #simplify eigenvactor
eigv
```

```
## [1] 2.0 0.5
```

```r
eigV
```

```
##      [,1] [,2]
## [1,]    1  1.0
## [2,]    1 -0.5
```

The population will grow without bound, since largest eigenvalue is 2 greater than 1.

(b) Calculat N_k by hand and compare with computer stimulation

Solve $\begin{pmatrix} 1 & 1 \\ 1/2 & 3/2 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix}$, we get $\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} 3 \\ -2 \end{pmatrix}$ So we have $N_0 = 3\begin{pmatrix} 1 \\ 1 \end{pmatrix} - 2\begin{pmatrix} 1 \\ -0.5 \end{pmatrix}$, The we have N_k from recurrence relation, $N_k = L^k N_0 = L^k(3\begin{pmatrix} 1 \\ 1 \end{pmatrix} - 2\begin{pmatrix} 1 \\ -0.5 \end{pmatrix}) = 3L^k\begin{pmatrix} 1 \\ 1 \end{pmatrix} - 2L^k\begin{pmatrix} 1 \\ -0.5 \end{pmatrix} = 3 \times 2^k \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 2 \times 0.5^k \begin{pmatrix} 1 \\ -0.5 \end{pmatrix}$.

When k = 4, $N_4 = 3 \times 2^4 \begin{pmatrix} 1 \\ 1 \end{pmatrix} - 2 \times 0.5^4 \begin{pmatrix} 1 \\ -0.5 \end{pmatrix} = \begin{pmatrix} 47.875 \\ 48.0625 \end{pmatrix}$

```
# running with computer
L %*% L %*% L %*% L %*%  matrix(c(1, 4))
```

```
##         [,1]
## [1,] 47.8750
## [2,] 48.0625
```

Which gives exact the same answer.

(c)

as $k \to$ inf, only term with largest eigenvalue exist, so stable population is $\vec{N} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$

Problem 3

(a) Write down a Leslie matrix model

$$\begin{pmatrix} N_j^{k+1} \\ N_a^{k+1} \end{pmatrix} = \begin{pmatrix} 0 & b \\ s & 1-m \end{pmatrix} \begin{pmatrix} N_j^k \\ N_a^k \end{pmatrix}$$

(b) find long term polulation

$\begin{vmatrix} -\lambda & 0.9 \\ 0.4 & 0.9-\lambda \end{vmatrix} = (-\lambda)(0.9-\lambda) - 0.36 = \lambda^2 - 0.9\lambda - 0.36$

$= (\lambda - 1.2)(\lambda + 0.3) = 0$

$\lambda_1 = 1.2$, $\lambda_2 = -0.3$

↑ large

~~E~~ ~~E~~ find $E_{1.2}$:

$\left(\begin{array}{cc|c} -1.2 & 0.9 & 0 \\ 0.4 & -0.3 & 0 \end{array}\right)$ $E_{1.2} = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$

So. long term growth factor is 1.2.

as $k \to$ inf, only term with largest eigenvalue exist, so stable population is $\vec{N} = \begin{pmatrix} 3/7 \\ 4/7 \end{pmatrix}$

```
L <- rbind(c(0,0.9),c(0.4,0.9))
eig <- eigen(L)
eigv <- eig[[1]]
eigV <- eig[[2]] / c(-0.2, -0.2, eig[[2]][4], eig[[2]][4])
eigv
```

```
## [1]  1.2 -0.3
```

```
eigV
```

```
##      [,1] [,2]
## [1,]    3   -3
## [2,]    4    1
```

Problem 4

(a) explain Leslie matrix

- For $N_{k+1}^{(F)}$:
  - There are $(1-m)N_k^{(F)}$ F survived, and t of survived transformed So there are $(1-m)(1-t)N_k^{(F)}$ left as F.
  - There are $(1-m)N_k^{(T)}$ T survived and u of them transform to F. So there are $(1-m)uN_k^{(F)}$ transform form T to F.
  - No $N_k^{(J)}$ become F in day k+1.
- For $N_{k+1}^{(J)}$:
  - No F in day k become J in day k+1.
  - There are $(1-m)N_k^{(J)}$ J survived, and c of survived transformed So there are $(1-m)(1-c)N_k^{(J)}$ left as J.
  - There are $(1-m)N_k^{(T)}$ T survived and each of them produce b J. So there are $(1-m)bN_k^{(T)}$ J are produced by T.
- For $N_{k+1}^{(J)}$:
  - There are $(1-m)N_k^{(F)}$ F survived, and t of survived transformed to T. So there are $(1-m)tN_k^{(F)}$ transform from F to T.
  - There are $(1-m)N_k^{(J)}$ J survived, and c of survived transformed to T. So there are $(1-m)cN_k^{(J)}$ transform form J to T.
  - There are $(1-m)N_k^{(T)}$ T survived, and u of survived transformed. So there are $(1-m)(1-u)N_k^{(J)}$ left as T.

(b) explore Leslie matrix

(i) find eigenvalue

```
L <- rbind(c(0.6,0,0.7),c(0,0.4,0.5),c(0.3,0.5,0.2))
eig <- eigen(L)
stableV <- eig[[2]][,1] / sum(eig[[2]][,1])
names(stableV) <- c("F", "J", "T")
eig
```

```
## eigen() decomposition
## $values
## [1]  1.0504880  0.5021306 -0.3526186
##
## $vectors
##            [,1]       [,2]       [,3]
## [1,] -0.7764170  0.8197641 -0.5220384
## [2,] -0.3840705 -0.5611153 -0.4719745
## [3,] -0.4996665 -0.1146141  0.7104336
```

So largest eigenvalue is `1.050488`, larger than 1, so the population will grow without bound.

Here is the stable fraction:

```
stableV
```

```
##         F         J         T
## 0.4676777 0.2313463 0.3009760
```

(ii)

```
N0 <- c(0, 100, 0)
population_at_k_day <- function(k) {
  result <- N0
  for (i in seq(k)) {
    result <- L %*% result
  }
  return(result)
}
population_array <- vapply(1:10, population_at_k_day, FUN.VALUE = numeric(3))
rownames(population_array) <- c("F", "J", "T")
population_array <- cbind(N0, population_array)
colnames(population_array) <- 0:10
```

The table for population from day 0 to day 10

```
population_array
```

```
##     0  1  2    3     4      5       6        7        8        9       10
## F   0  0 35 42.0 51.10 55.650 59.9900 63.59640 67.16458 70.71128 74.36790
## J 100 40 41 31.4 31.06 30.274 31.1096 32.15984 33.58304 35.15645 36.87768
## T   0 50 30 37.0 35.70 38.000 39.4320 41.43820 43.44648 45.63019 47.91765
```

Growth factor matrix

```
growth_factor_matrix <- population_array[,9:11] / population_array[,8:10]
colnames(growth_factor_matrix) <- c("7 to 8", "8 to 9", "9 to 10")
rownames(growth_factor_matrix) <- c("F", "J", "T")
growth_factor_matrix
```

```
##     7 to 8   8 to 9  9 to 10
## F 1.056107 1.052806 1.051712
## J 1.044254 1.046852 1.048959
## T 1.048464 1.050262 1.050130
```

Fraction cell

```
fraction_matrix <- population_array[,9:11] / rep(apply(population_array[,9:11],2,sum),each=3)
colnames(fraction_matrix) <- 8:10
rownames(fraction_matrix) <- c("F", "J", "T")
fraction_matrix
```

```
##           8         9        10
## F 0.4657929 0.4667475 0.4672430
## J 0.2329016 0.2320590 0.2316972
## T 0.3013055 0.3011935 0.3010598
```

As we can see, the fraction of cells are converging to the theoretic value we calculated in part i.