Ecological Dynamics
Lab 2: Structured population models

Background

- Key characteristics such as fertility and mortality often vary with age in complex organisms.
- Understanding such variation is often critical for managing or conserving natural populations.
- The effects of demographic and environmental stochasticity are ubiquitous and typically assessed via simulation.

Objectives

- Learn to construct a Leslie matrix from life table data.
- Use linear algebra to estimate key characteristics of structured population models.
- Test the sensitivity of structured population models to environmental stochasticity.

Instructions

- Launch RStudio, open a new script file and save it as `lab1-yourLastName.R`.
- Complete the activities below by adding the appropriate commands to your `R` script file.
- Embed your answers as comments in the `R` script file.
- At the end of your session, save your file onto a flash drive (files saved on lab computers are wiped when you log out).

Task 1: Modeling the dynamics of structured populations

1. The first step in modeling age-structured populations is often to load the data and compute fertility and survivorship schedules. To do so, begin by downloading the following dataset from the web:

```
life.table <- read.csv(file = "http://faraway.neu.edu/data/lifetable.csv")
```

   Once the data has been downloaded, compute the survivorship schedule $l(i) = \frac{S(i)}{S(0)}$, survivorship probabilities $P_i = \frac{l(i)}{l(i-1)}$ and the fertilities $F_i = b(i)P_i$ for each age class i. Note that subscripts i refer to age classes whereas (i) refer to ages (e.g., $b(i)$ refers to age i and age class $i+1$ whereas P_i refers to age class i and age $i-1$).

```
# Compute survivorship schedule
l <- life.table$S/life.table$S[1]
# Compute survivorship probabilities
P <- l[2:length(l)]/l[1:(length(l) - 1)]
# Compute fertilities
F <- c(life.table$b[1], life.table$b[2:NROW(life.table)] * P)
```

2. Now construct the Leslie matrix for this dataset. Remember that the Leslie matrix is a $k \times k$ square matrix, where k represents the number of age classes. The top row contains the fertilities F_i and the subdiagonal contains the survivorship probabilities P_i. All other entries are zeros.

```r
# Prepare Leslie matrix
A <- matrix(0, nrow = 8, ncol = 8, byrow = T)
# Fecundity values
A[1, ] <- F
# Survival probabilities
A[row(A) == col(A) + 1] <- P
# Print the Leslie matrix
A
```

```
##      [,1] [,2] [,3] [,4]       [,5]     [,6]      [,7]      [,8]
## [1,]  5.0  5.0  4.8 10.5 22.5000000 30.22222 4.3750000 0.4285714
## [2,]  0.5  0.0  0.0  0.0  0.0000000  0.00000 0.0000000 0.0000000
## [3,]  0.0  0.4  0.0  0.0  0.0000000  0.00000 0.0000000 0.0000000
## [4,]  0.0  0.0  0.5  0.0  0.0000000  0.00000 0.0000000 0.0000000
## [5,]  0.0  0.0  0.0  0.9  0.0000000  0.00000 0.0000000 0.0000000
## [6,]  0.0  0.0  0.0  0.0  0.8888889  0.00000 0.0000000 0.0000000
## [7,]  0.0  0.0  0.0  0.0  0.0000000  0.87500 0.0000000 0.0000000
## [8,]  0.0  0.0  0.0  0.0  0.0000000  0.00000 0.1428571 0.0000000
```

3. Use function `eigen` to compute the dominant eigenvalue and the corresponding eigenvector of the Leslie matrix. What do the eigenvalues and the eigenvectors, respectively, mean for this population?

```r
# Compute dominant eigenvalue (population growth rate)
(lambda1 <- Re(eigen(A)$values[1]))
```

```
## [1] 5.495723
```

```r
# Compute associated eigenvector (stable population structure)
(v <- Re(eigen(A)$vectors[, 1])/sum(Re(eigen(A)$vectors[, 1])))
```

```
## [1] 9.104803e-01 8.283535e-02 6.029077e-03 5.485245e-04 8.982840e-05
## [6] 1.452902e-05 2.313234e-06 6.013075e-08
```

Answer:

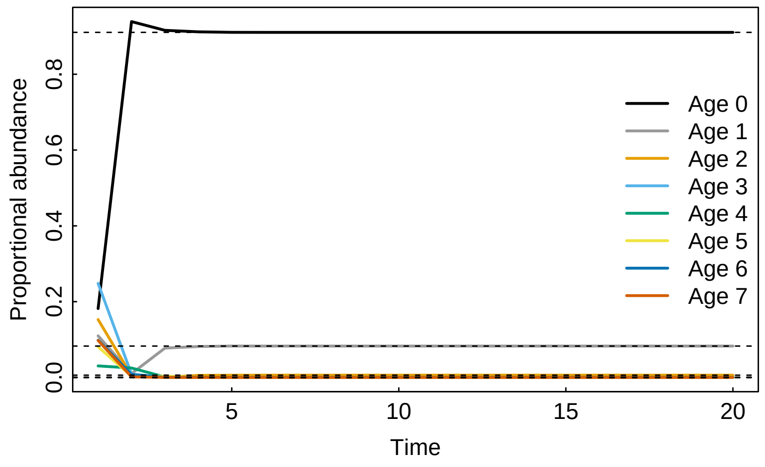
The dominant eigenvalue determines whether the population will grow ($\lambda > 1$) or decay ($\lambda < 1$) exponentially. Here, the dominant eigenvalue is greater than 1, which indicates the the population will growth exponentially. The eigenvector can be used to determine the stable population structure (i.e., the proportional abundance of each age once the population is growing exponentially). In this case, the eigenvector shows that the population size N will be dominated (> 90%) by individuals of age 0 (age class 1), with only 8% of individuals of age 1 (age class 2), and even fewer individuals of age 2-7.

4. Now it's time to simulate the model to verify our theoretical predictions (eigen-analysis). To do so, assume that the initial number of individuals of each age class comes from a random uniform distribution (see function `runif`). Now simulate the dynamics of the model for `t=20` time steps. Note that you will need to perform matrix multiplication for this step via the `%*%` operator in `R`.

```r
t <- 20
abund <- matrix(nrow = 8, ncol = t, NA)
set.seed(10)
abund[, 1] <- runif(8)
for (i in 2:t) {
    abund[, i] <- A %*% abund[, i - 1]
}
```

5. Plot the proportional abundance of each age as a function of time by using the `matplot` function (allows you to plot multiple lines at the same time). Do not forget to add axis labels and a legend to your figure. Also add the population structure predicted by the eigen-analysis as vertical dashed lines using function `abline`. Do your simulation results match your theoretical expectations?

```
matplot(1:t, t(abund)/colSums(abund), t = "l", lty = 1, lwd = 2, xlab = "Time",
    ylab = "Proportional abundance", col = cb[1:8])
legend(x = "right", legend = c(paste("Age", 0:7)), lty = 1, lwd = 2, col = cb[1:8],
    bty = "n")
abline(h = v, lty = 2)
```



Answer:

Yes, the vertical dashed lines are perfectly aligned with the proportional abundances of each age class observed during the final time steps of the simulations.

6. To estimate the population growth observed in your simulations empirically, use function `lm` to regress the log of total population size $\log(N)$ against time. This will give you an estimate of the growth rate of the population. Does this result match your theoretical expectation based on eigen-analysis?

```
pop.size <- colSums(abund[, ])
time <- 1:t
# Empirical estimate of growth R
coef(lm(log(pop.size) ~ time))[2]

##     time
## 1.714332

# Theoretical value of growth R
(R <- log(lambda1))

## [1] 1.70397
```

Answer:

The regression-based growth rate estimate is very similar to the one expected based on eigen-analysis.

Task 2: Simulating the effects of demographic and environmental stochasticity

For this section, you will estimate the effects of demographic and/or environmental stochasticity on the behavior of age-structured population models. To do so, you will need to download and install the `popbio` package:

```
## # Only install the package once:
## install.packages("popbio")
```

Note that this should only be done once per computer. Once the package is installed you can simply load it by issuing the following command:

```
# Load the package every time you need it in your R session:
library(popbio)
```

1. Begin by using function `eigen.analysis` to determine sensitivity and elasticity of λ to each model parameter. Identify the most important parameter and justify your answer.

```
library(popbio)
ea <- eigen.analysis(A)
# Sensitivities
ea$sensitivities

##           [,1]       [,2]        [,3]        [,4]         [,5]
## [1,] 0.9092064 0.08271945 0.006020642 0.000547757 8.970272e-05
## [2,] 0.9014295 0.00000000 0.000000000 0.000000000 0.000000e+00
## [3,] 0.0000000 0.09279364 0.000000000 0.000000000 0.000000e+00
## [4,] 0.0000000 0.00000000 0.016436754 0.000000000 0.000000e+00
## [5,] 0.0000000 0.00000000 0.000000000 0.002741031 0.000000e+00
## [6,] 0.0000000 0.00000000 0.000000000 0.000000000 5.046940e-04
## [7,] 0.0000000 0.00000000 0.000000000 0.000000000 0.000000e+00
## [8,] 0.0000000 0.00000000 0.000000000 0.000000000 0.000000e+00
##              [,6]         [,7]         [,8]
## [1,] 1.450869e-05 2.309997e-06 6.004662e-08
## [2,] 0.000000e+00 0.000000e+00 0.000000e+00
## [3,] 0.000000e+00 0.000000e+00 0.000000e+00
## [4,] 0.000000e+00 0.000000e+00 0.000000e+00
## [5,] 0.000000e+00 0.000000e+00 0.000000e+00
## [6,] 0.000000e+00 0.000000e+00 0.000000e+00
## [7,] 1.157940e-05 0.000000e+00 0.000000e+00
## [8,] 0.000000e+00 1.801399e-07 0.000000e+00

# Elasticities
ea$elasticities

##           [,1]        [,2]        [,3]         [,4]         [,5]
## [1,] 0.8271945 0.075258024 0.005258467 0.0010465317 3.672512e-04
## [2,] 0.0820119 0.000000000 0.000000000 0.0000000000 0.000000e+00
## [3,] 0.0000000 0.006753881 0.000000000 0.0000000000 0.000000e+00
## [4,] 0.0000000 0.000000000 0.001495413 0.0000000000 0.000000e+00
## [5,] 0.0000000 0.000000000 0.000000000 0.0004488814 0.000000e+00
## [6,] 0.0000000 0.000000000 0.000000000 0.0000000000 8.163018e-05
## [7,] 0.0000000 0.000000000 0.000000000 0.0000000000 0.000000e+00
## [8,] 0.0000000 0.000000000 0.000000000 0.0000000000 0.000000e+00
##              [,6]         [,7]         [,8]
## [1,] 7.978657e-05 1.838928e-06 4.682598e-09
## [2,] 0.000000e+00 0.000000e+00 0.000000e+00
## [3,] 0.000000e+00 0.000000e+00 0.000000e+00
## [4,] 0.000000e+00 0.000000e+00 0.000000e+00
## [5,] 0.000000e+00 0.000000e+00 0.000000e+00
## [6,] 0.000000e+00 0.000000e+00 0.000000e+00
## [7,] 1.843610e-06 0.000000e+00 0.000000e+00
## [8,] 0.000000e+00 4.682598e-09 0.000000e+00
```

Answer:

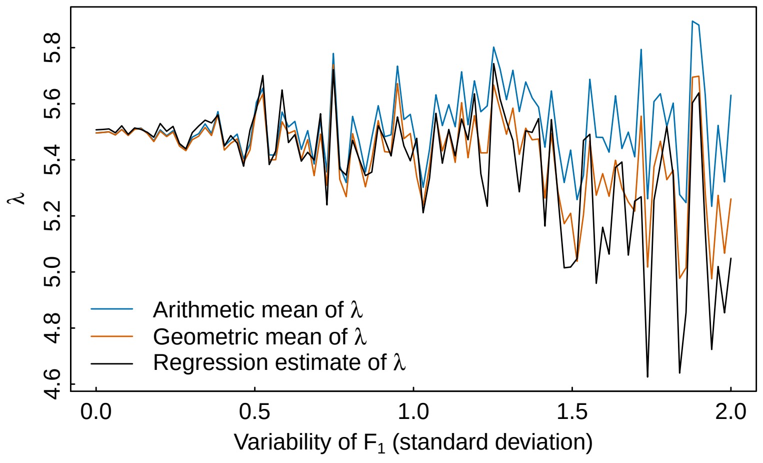
The most important parameter affecting λ appears to be F_1 with an elasticity of 0.83 and a sensitivity of 0.91. This means that a change in the value of this parameter will have the largest effect on λ.

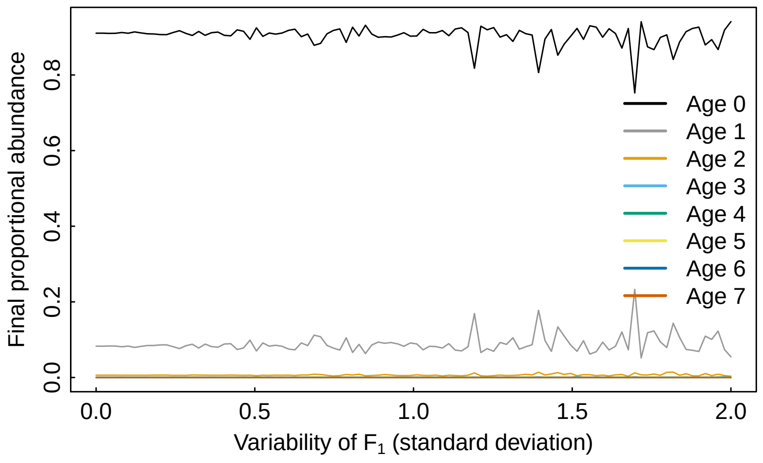
2. We are now going to conduct an experiment by making F_1 vary over time and across simulations. To do so, we will assume that F_1 comes from a normal distribution with a mean of 5 and a standard deviation that varies from 0 to 2 in 100-linearly spaced increments (see function `rnorm`). For each of the 100 standard deviation values, we will draw $t = 50$ random values of F_i. Each one of these values will be used for a single time step t of the simulation. Assume that all other Leslie matrix parameters remain the same. We will need to keep track of multiple metrics. First, we will compute the (i) arithmetic and (ii) geometric means of the λ values obtained via eigen-analysis at each time step for each simulation, along with the (iii) regression-based estimate of λ. Second, we will need to record the final proportional abundance at the end of each simulation. Note that you will have to create a function to compute the geometric mean, which is defined as: $\left(\prod_{i=1}^{N} \lambda_i\right)^{1/N}$.

```r
# Compute the geometric mean
geom.mean <- function(x) {
    return(prod(x)^(1/length(x)))
}
# Reset Leslie matrix
set.seed(1)
A <- matrix(0, nrow = 8, ncol = 8, byrow = T)
# Fecundity values
A[1, ] <- F
# Survival probabilities
A[row(A) == col(A) + 1] <- P
# Total time steps
t <- 50
# Total number of standard deviations
stds <- seq(from = 0, to = 2, len = 100)
abund <- matrix(nrow = 8, ncol = t, NA)
abund[, 1] <- runif(8)
lambdas.temp <- numeric(t - 1)
lambdas <- matrix(nrow = length(stds), ncol = 3)
v <- matrix(nrow = NROW(A), ncol = length(stds), NA)
# Launch simulations
for (j in 1:length(stds)) {
    # F1 is normally distributed with mean of F[1] and specific sd
    F.norm <- rnorm(mean = F[1], sd = stds[j], n = t)
    for (i in 2:t) {
        A[1, 1] <- F.norm[i - 1]
        abund[, i] <- A %*% abund[, i - 1]
        eig <- eigen(A)
        lambdas.temp[i - 1] <- max(Re(eig$values))
    }
    mod.reg <- lm(log(colSums(abund[, ])) ~ I(1:t))
    lambdas[j, ] <- c(mean(lambdas.temp), geom.mean(lambdas.temp), exp(coef(mod.reg)[2]))
    v[, j] <- t(abund[, NCOL(abund)])/sum(abund[, NCOL(abund)])
}
```

3. On a 2-row x 1-column figure, plot the three estimates of λ (y-axis) as a function of the standard deviation of F_1 in the top panel and the final proportional abundance (y-axis) as a function of the standard deviation of F_1 in the bottom panel. Did environmental stochasticity affect the behavior of the model and, if so, how?

```
par(mfrow = c(2, 1))
matplot(stds, lambdas, t = "l", lty = 1, col = c(cb["blue"], cb["red"],
    cb["black"]), xlab = expression(paste("Variability of ", F[1], " (standard deviation)")),
    ylab = expression(lambda))
legend(x = "bottomleft", legend = c(expression(paste("Arithmetic mean of ",
    lambda)), expression(paste("Geometric mean of ", lambda)), expression(paste("Regression estimate of ",
    lambda))), lty = 1, col = c(cb["blue"], cb["red"], cb["black"]), bty = "n")
matplot(stds, t(v), t = "l", lty = 1, col = cb[1:8], ylab = "Final proportional abundance",
    xlab = expression(paste("Variability of ", F[1], " (standard deviation)")))
legend(x = "right", legend = c(paste("Age", 0:7)), lty = 1, lwd = 2, col = cb[1:8],
    bty = "n")
```



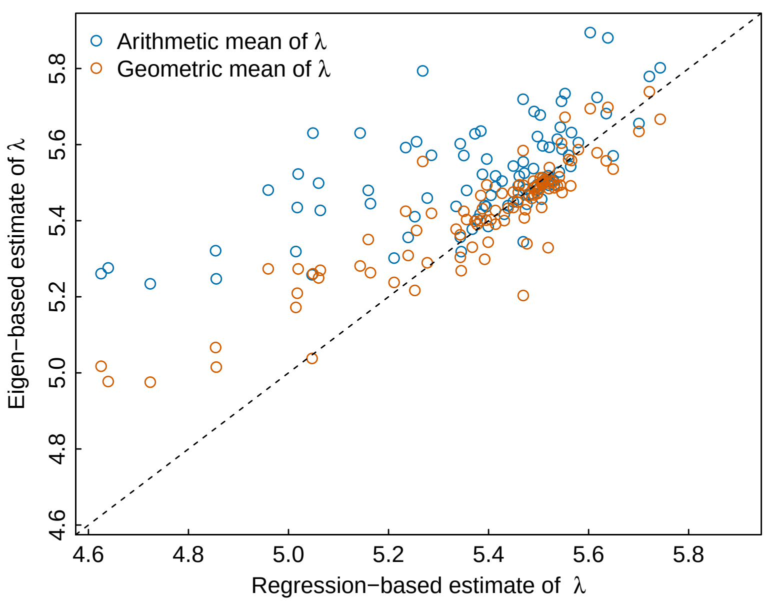


Answer:

Increasing the stochasticity of F_1 typically reduced the growth rate of the population λ, but it did not affect the stable age structure (i.e., the final proportional abundance of all age classes). Age class 1 continued to dominate the population for all levels of stochasticity.

4. To focus on the different estimates of λ, plot both the geometric mean of λ in red and the arithmetic mean of λ in blue against the regression-based estimate of λ (x-axis). Add the 1:1 line (dashes), a legend and the proper axis labels. Using this figure, compare the arithmetic and geometric mean estimates of λ to the empirical, regression-based estimate. Can you explain the patterns based on the model?

```r
plot(lambdas[, 3], lambdas[, 1], col = cb["blue"], ylab = expression(paste("Eigen-based estimate of ",
    lambda)), xlab = expression(paste("Regression-based estimate of  ",
    lambda)), xlim = c(range(lambdas)), ylim = range(lambdas))
points(lambdas[, 3], lambdas[, 2], col = cb["red"])
legend(x = "topleft", legend = c(expression(paste("Arithmetic mean of ",
    lambda)), expression(paste("Geometric mean of ", lambda))), pch = 1,
    col = c(cb["blue"], cb["red"]), bty = "n")
abline(a = 0, b = 1, lty = 2)
```



Answer:

The geometric mean of λ is typically much closer to the regression-based estimate of λ (1:1 line) than the arithmetic mean of λ. This is because growth is described by a recursive (multiplicative) process whereby variation gets compounded over time. Hence, the best approximation for the time-averaged growth rate will be the geometric mean. The discrepancy between the geometric mean and the regression-based estimate of λ is due to the fact that the former assumes the the population has reached its stable age structure, which is impossible when the system is perturbed via environmental stochasticity affecting F_1 at each time step.

5. To confirm these graphical results, use function `lm` to regress the arithmetic and geometric means of λ against the regression-based estimate. Compare the slopes and the goodness-of-fit R^2 to determine which mean is a better approximation of the regression-based estimate.

```
# Regression of arithmetic mean vs. lm estimate
summary(lm(lambdas[, 1] ~ lambdas[, 3]))

##
## Call:
## lm(formula = lambdas[, 1] ~ lambdas[, 3])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.20276 -0.06924 -0.02756  0.06440  0.31756
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.60987    0.25777  14.004  < 2e-16 ***
## lambdas[, 3]  0.35425    0.04783   7.406 4.58e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1053 on 98 degrees of freedom
## Multiple R-squared:  0.3589,Adjusted R-squared:  0.3523
## F-statistic: 54.86 on 1 and 98 DF,  p-value: 4.577e-11

# Regression of geometric mean vs. lm estimate
summary(lm(lambdas[, 2] ~ lambdas[, 3]))

##
## Call:
## lm(formula = lambdas[, 2] ~ lambdas[, 3])
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.264659 -0.020859  0.001426  0.030235  0.212057
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.08385    0.16878   12.35   <2e-16 ***
## lambdas[, 3]  0.61876    0.03132   19.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06892 on 98 degrees of freedom
## Multiple R-squared:  0.7993,Adjusted R-squared:  0.7973
## F-statistic: 390.4 on 1 and 98 DF,  p-value: < 2.2e-16
```

Answer:

Both the slope (closer to 1) and the R^2 values are higher for the geometric mean than the arithmetic mean indicating that the geometric mean is a better approximation of the growth rate λ observed in the simulations.