Matrix population models

NRES 470/670

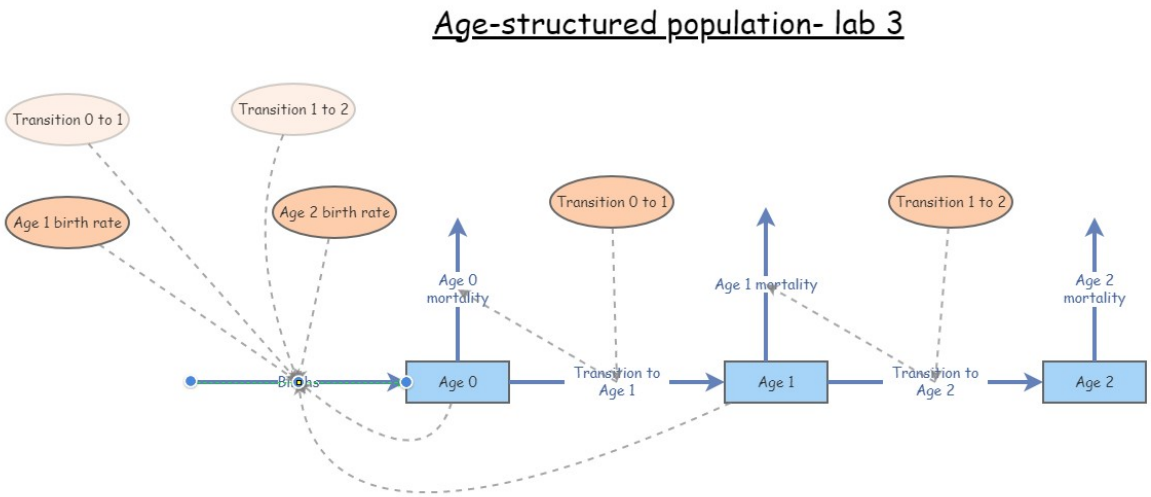
Spring 2023

Matrix population models

First of all, this lecture is full of R code (R makes it easy to run matrix population models!). If you want to follow along in R, you can find the R script here. I recommend right-clicking on the link, saving the script to a designated folder, and loading up the script in RStudio.

Why matrices?

Reason 1: simplify!



You might recognize this InsightMaker model from Lab 3. This represents an age-structured population with only three age classes. Imagine if there were five age classes, or 10? What if you could jump from (e.g.) stage 3 to stage 5? Or from stage 5 back to stage 3? How many lines would you have to draw, how many equations would you have to put in all the different flows? It would be tedious, and you could easily run into errors that would be hard to uncover!

Consider the teasel example from our textbook. It's possible to implement this model in InsightMaker, but it would be tedious and prone to error. And this is far from the most complicated populations out there (although notice that plants can do some things that vertebrate animals can't do- for instance go backwards in developmental stage. With matrix models, there is an easier way!

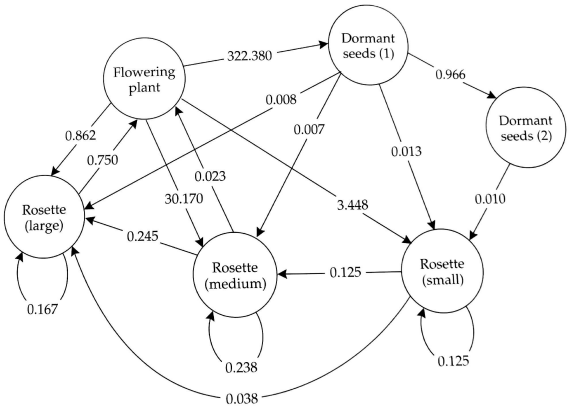


Figure 3.6 Transition matrix and loop diagram for teasel (*Dipsacus sylvestris*). Transitions are shown for dormant first-year and second-year seeds [seed (1) and seed (2)], small, medium, and large rosettes [ros (s), ros (m), ros (l)], and flowering plants. (Data from Caswell 1989.)

The population vital rates for pretty much any age-structured or stage-structured population can be represented as a **transition matrix** (or, projection matrix), which summarizes all the information about survival, birth rates, and transitions between stages! (and the fact that a life history like teasel can be represented by a transition matrix illustrates the generality of this concept!)

For example, the teasel vital rates can be summarized in this matrix:

```
# Teasel example ---------------------------------
# Teasel example from Gotelli: summarizing a complex life history!

teasel <- read.csv("teaselmatrix1.csv", header=T)      # read in the teasel transition matrix from Gote
teasel <- teasel[,-1]                                  # remove the row names
teasel_matrix <- as.matrix(teasel)                     # convert to a matrix (from a data frame)
colnames(teasel_matrix) <- names(teasel)               # assign row and column names
rownames(teasel_matrix) <- names(teasel)
teasel_matrix                                          # print the matrix
```

```
##           seed1 seed2  ros1  ros2  ros3 flowering
## seed1     0.000  0.00 0.000 0.000 0.000   322.380
## seed2     0.966  0.00 0.000 0.000 0.000     0.000
## ros1      0.013  0.01 0.125 0.000 0.000     3.448
## ros2      0.007  0.00 0.125 0.238 0.000    30.170
## ros3      0.008  0.00 0.000 0.245 0.167     0.862
## flowering 0.000  0.00 0.000 0.023 0.750     0.000
```

Stage-structure vs age-structure In the previous module, we talked about 'age-structured populations'. What we meant by that is that the population vital rates (e.g., b and d) varied by age.

Sometimes, it's convenient to classify individuals within a certain age range as belonging to a particular life-history stage. For example, we might classify the life history of a grizzly bear like this:

Age 0-1: newborn
Age 1-2: yearling
Age 2-5: subadult
Age 6+: adult

This can simplify our models considerably. For example, consider a species like a sea turtle, with up to 75 or 100 years of life. You could build a model in which you have 100 [Stocks], one for each year of life.

–OR–

You could have 5 [Stocks] representing age ranges in which sea turtles tend to have consistent(ish) vital rates. For example, we might divide the sea turtle life history into the following stages:

Age 0-1: hatchling
Age 1-5: young juvenile
Age 5-10: older juvenile
Age 10-17: subadult
Age 18+: adult

By using stages, we have simplified our model from having 100 stocks (with even more associated flows/transitions) to a model with only 5 stocks- and we are still accurately representing how vital rates change with age (i.e., the model is still *biologically realistic*).

Matrix population models can represent age-structured and stage-structured models with equal simplicity and elegance.

The term 'Leslie Matrix' refers to an age-structured matrix population model.

When a matrix is used to represent a stage-structured population, it is often called a 'Lefkovitch" Matrix.

Reason 2: modeling age-structured population dynamics!

In one of the questions in Lab 3, you were asked to use a life table to predict the number of births in a population one time step into the future. As you probably realized, this is not as straightforward as it sounds!

Life tables are great for summarizing survivorship schedules and other aspects of age-structured populations. But life tables are not designed to model the abundance dynamics of age- or stage- structured populations!

You know what *is* great for projecting age-structured abundance into the future?

MATRICES of course!

For example, let's project a teasel population 1 year into the future:

First of all, we need to begin with a teasel population **vector**...

```
# Summarize initial age-structured abundance as a matrix with one column

Initial_teasel <- matrix(c(1000,1500,200,300,600,25),ncol=1)          # initial population size (populat
rownames(Initial_teasel) <- rownames(teasel_matrix)                   # add row and column names
colnames(Initial_teasel) <- "Abundance"
Initial_teasel
```

```
##           Abundance
## seed1          1000
## seed2          1500
## ros1            200
## ros2            300
## ros3            600
## flowering        25
```

Then all we need to do is 'matrix-multiply' this **vector** of abundances by the **transition matrix** from above! Each time we do this multiplication step, we advance one year! It's that easy!

NOTE: matrix multiplication (percent-asterisk-percent in R) is not the same as standard multiplication (asterisk in R).

Here's how we can do this in R!

```
# Project the population at time 1

Year1 <- teasel_matrix %*% Initial_teasel   # note: the '%*%' denotes 'matrix multiplication' in R. We'
Year1
```

```
##           Abundance
## seed1       8059.50
## seed2        966.00
## ros1         139.20
## ros2         857.65
## ros3         203.25
## flowering    456.90
```

Pretty simple!

To compute teasel abundance in year 2 of our simulation, we can simply repeat:

```
# Project the population at time 2

thisYear <- Year1
nextYear <- teasel_matrix %*% thisYear
nextYear  # now we get the (age structured) population size at time 2!
```

```
##             Abundance
## seed1     147295.4220
## seed2       7785.4770
## ros1        1707.2247
## ros2       14062.6102
## ros3         702.3908
## flowering    172.1635
```

We could use this strategy to simulate abundance for ten years (or 20, or 30, or 10000)...

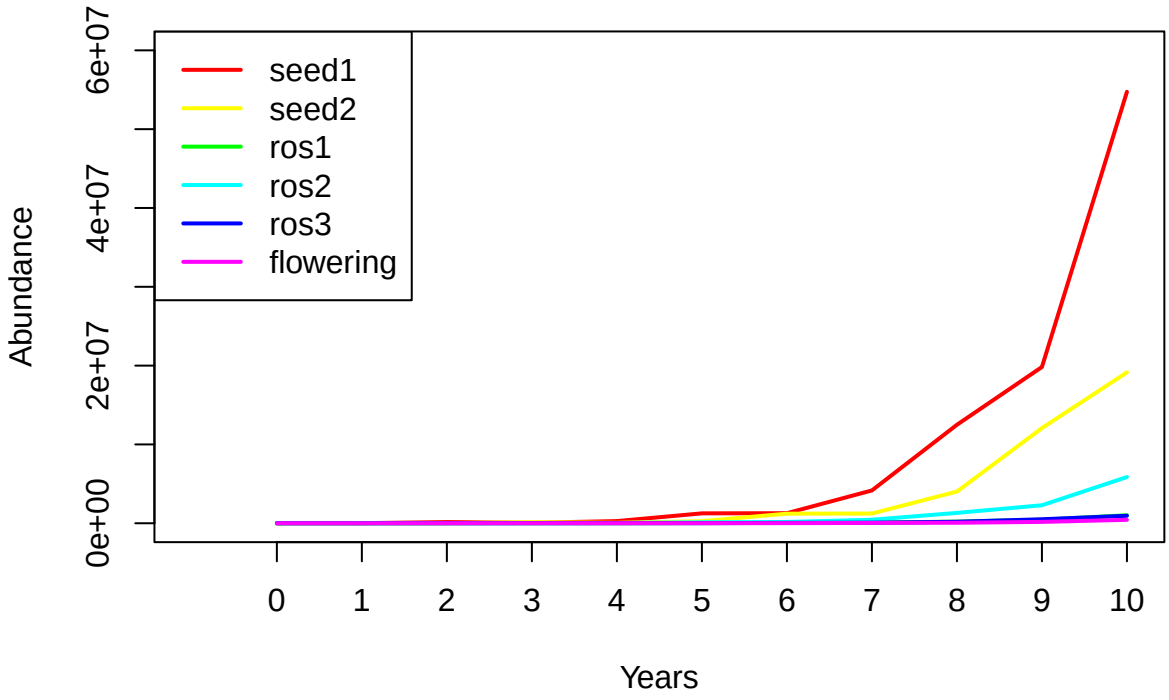
Notice the use of a **for loop** here!

```
# Use a FOR loop to project the population dynamics for the next 10 years!

nYears <- 10
```

```
tenYears <- matrix(0,nrow=6,ncol=nYears+1)          # initialize storage array for recording age struct
rownames(tenYears) <- rownames(Initial_teasel)      # assign row and column names
colnames(tenYears) <- seq(0,10)
tenYears[,1] <- Initial_teasel                      # initialize the simulated abundances

# run the for loop!

for(t in 2:(nYears+1)){    # here we use 't' as our looping variable, but we could choose any name we w
  tenYears[,t] <-  teasel_matrix %*% tenYears[,t-1]     # perform matrix multiplication for each year o
}

tenYears
```

```
##              0       1           2           3          4           5          6          7
## seed1     1000 8059.50 147295.4220  55502.0530 274098.158 1254742.541 1274599.05 4160519.75 12493783
## seed2     1500  966.00   7785.4770 142287.3777  53614.983  264778.821 1212081.29 1231262.68  4019062
## ros1       200  139.20   1707.2247   2799.7179   5425.969   18197.711   34866.57   77547.56   209719
## ros2       300  857.65  14062.6102   9785.5436  28718.972  126857.393  160533.59  440850.62  1312972
## ros3       600  203.25    702.3908   4889.4070   4390.907   13317.225   46750.08   68459.45   186131
## flowering   25  456.90    172.1635    850.2331   3892.123    3953.716   12905.64   38754.83    61484
##                    9         10
## seed1     19821259.9 54739267.1
## seed2     12068994.7 19147337.1
## ros1        440822.1  1018930.3
## ros2       2281135.7  5859547.7
## ros3        505712.0   948267.5
## flowering   169797.3   431750.1
```

Finally, we can plot out the abundance of each stage over 10 years!



So projection is easy with matrices! What more reason do we need to convince ourselves of why matrix population models are worth knowing? Well, how about this...

Reason 3: Matrix algebra 'tricks'!

Fortunately, we can compute two key population properties from the transition matrix using some mathematical tricks:

1. We can compute **Lambda** from the transition matrix in one step
2. We can compute **Stable Stage Distribution (SSD) from the transition matrix in one step

Lambda There is a clear similarity between the finite population growth equation:

$N_{t+1} = \lambda \cdot N_t$,

where N is abundance (as always), t is time, often in years but could be any time units, and λ is the multiplicative growth rate over the time period $t \rightarrow t+1$

... and the matrix population growth equation:

$\mathbf{N}_{t+1} = \mathbf{A} \cdot \mathbf{N}_t$,

where **N** is a **vector** of abundances (abundance for all stages), and **A** is the **transition matrix**, which we have seen before.

Both equations describe discrete exponential growth or decline!

Note that N in the first equation is a **scalar** – that is, it is just a 'naked' number.

In contrast:

N in the second equation represents an age-structured **vector (a bundle of numbers organized in a single line)**: a set of abundances structured by age or stage class.

Similarly, the finite population growth rate, λ is a scalar,

In fact, the concept of **Lambda**, or the multiplier used to model discrete population growth, applies to matrix population models as well!

Recall that when a population is at *stable stage distribution (SSD)*, it grows in a discrete exponential growth pattern- this rate of exponential growth can be described by a single parameter – **Lambda**!

A is a **matrix (a bundle of numbers organized in rows and columns)** known as the transition matrix in Population Ecology.

In one step, you can compute λ from **A**!!

Let's do this in R!

What is the growth rate λ for the teasel population. If you recall, it looked like it was growing, so it should be above 1...

```
# Matrix "tricks" for population ecology ----------------------------

# Use the transition matrix to compute Lambda, or the finite rate of population growth!

library(popbio)      # load the 'popbio' package in R!
Lambda <- lambda(teasel_matrix)
Lambda
```

```
## [1] 2.32188
```

```
#   as.numeric(round(eigen(teasel_matrix)$values[1],2))  # this is an alternative method- if you don't
```

You don't have to understand the math here- but I do want you to understand how simple that was- just one line of code and we computed the annual rate of growth from the teasel transition matrix!

Stable Stage Distribution (SSD) Here's another trick:

In one step, you can compute **stable stage distribution (SSD)** from **A**!!

Let's do this in R!

What is the stable age distribution for the teasel population. If you recall, the first seed stage looked like it dominated in the figure above.

```
# Compute stable age distribution from the transition matrix!

library(popbio)    # ... and it's even easier if we use the 'popbio' package...
SAD <- stable.stage(teasel_matrix)
SAD      # stable age distribution as a percentage of the total population
```

```
##       seed1       seed2        ros1        ros2        ros3   flowering
## 0.636901968 0.264978062 0.012174560 0.069281759 0.012076487 0.004587164
```

```
# #abs(as.numeric(round(eigen(teasel_matrix)$vectors[,1],3)))  # alternative- doesn't use 'popbio'
# SAD/sum(SAD)
```

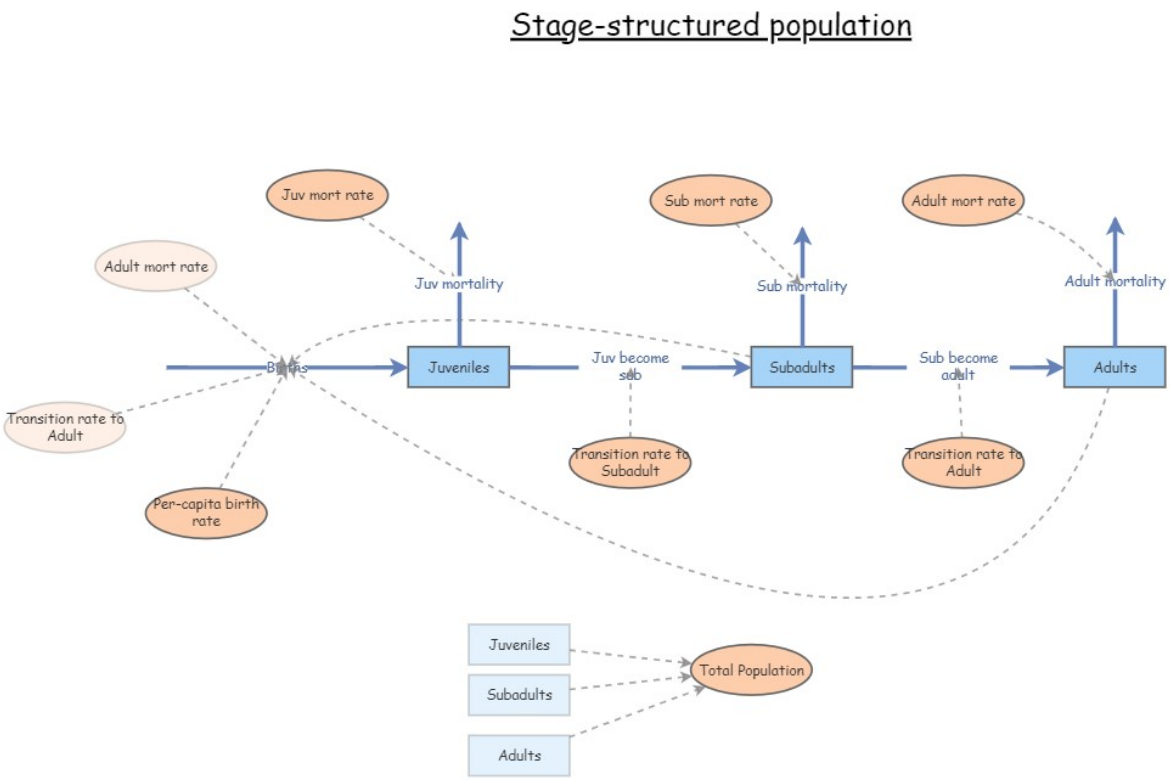
This vector represents the *relative abundances in each age class at the stable age distribution*!

Q: Does a stage-structured population grow at the rate of λ per time step if it is NOT at stable age distribution? [PointSolutions]

To answer this question, you may find it helps to load an stage-structured model in InsightMaker like this one.

Mechanics of matrix population models

Let's take a look at a basic stage-structured population – specifically this one - we used this model for running the 'supplementation' example in the 'age structured populations' lecture. In InsightMaker it looks something like this:



Let's convert the vital rates to a three-stage **projection matrix**. Projection matrices are **square matrices** where the number of rows and columns are equal to the number of life stages. In this case, that means three! Let's make a blank matrix for now:

```
# Demo -------------------------
# In class demo: convert an insightmaker model to a matrix projection model

# First, we specify a blank transition matrix
```

```
TMat <- matrix(0,nrow=3,ncol=3)                    # create a blank matrix with 3 rows and 3 columns
stagenames <- c("Juveniles","Subadults","Adults")  # name the rows and columns
rownames(TMat) <- stagenames
colnames(TMat) <- stagenames
TMat                                               # now we have an all-zero transition matrix.
```

```
##           Juveniles Subadults Adults
## Juveniles         0         0      0
## Subadults         0         0      0
## Adults            0         0      0
```

You can read the **elements** of a transition matrix as follows:

"The per-capita rate of transition from *(col name)* this year to *(row name)* next year is *(value of element)*"

NOTE: the top row in the transition matrix represents **fecundities (f)**. These elements are best interpreted as follows:

"The per-capita production of newborns next year by *(col name)* in the population this year is *(value of element)*".

These matrix elements account for both survival (surviving to the next year) and birth rate (birth rate one year from now). In life table terms:

$f_t = g(t) * b(t+1)$

Now we can start filling in this matrix. Let's begin with the top left **element** of the matrix (a **fecundity** term, since it's part of the top row). This represents the per-capita production of Juveniles (col) next year by Juveniles (row) this year. What is the value of this element?

Let's update our transition matrix:

```
# fill in the top left element of the matrix

TMat[1,1] <- 0
TMat
```

```
##           Juveniles Subadults Adults
## Juveniles         0         0      0
## Subadults         0         0      0
## Adults            0         0      0
```

How about the second row, first column. This represents the per-capita transition from Juveniles (col) this year to Subadults (row) next year. That is, the transition rate from juvenile to subadult. The value from our model is 0.3.

Let's update our transition matrix:

```
# update the second row, first column

TMat[2,1] <- 0.3
TMat
```

```
##           Juveniles Subadults Adults
## Juveniles       0.0         0      0
## Subadults       0.3         0      0
## Adults          0.0         0      0
```

If we keep going, we get the following matrix. See if you can understand what this matrix is saying about the transitions from and two the three life stages.

```
# and keep filling it in...

TMat[,1] <- c(0,0.3,0)          # fill in the entire first column of the transition matrix
TMat[,2] <- c(0.1*4,0.4,0.1)        # fill in the entire second column of the transition matrix
TMat[,3] <- c(0.85*4,0,0.85)         # fill in the entire third column of the transition matrix
TMat
```

```
##           Juveniles Subadults Adults
## Juveniles       0.0       0.4   3.40
## Subadults       0.3       0.4   0.00
## Adults          0.0       0.1   0.85
```

Now we can run a 40-year projection and compare it with the InsightMaker model. It should look the same!!

First we must specify the initial abundances in each stage:

```
# specify initial abundance vector

InitAbund <- c(40,0,0)
names(InitAbund) <- colnames(TMat)
InitAbund
```

```
## Juveniles Subadults    Adults
##        40         0         0
```

So we are starting with only Juveniles...

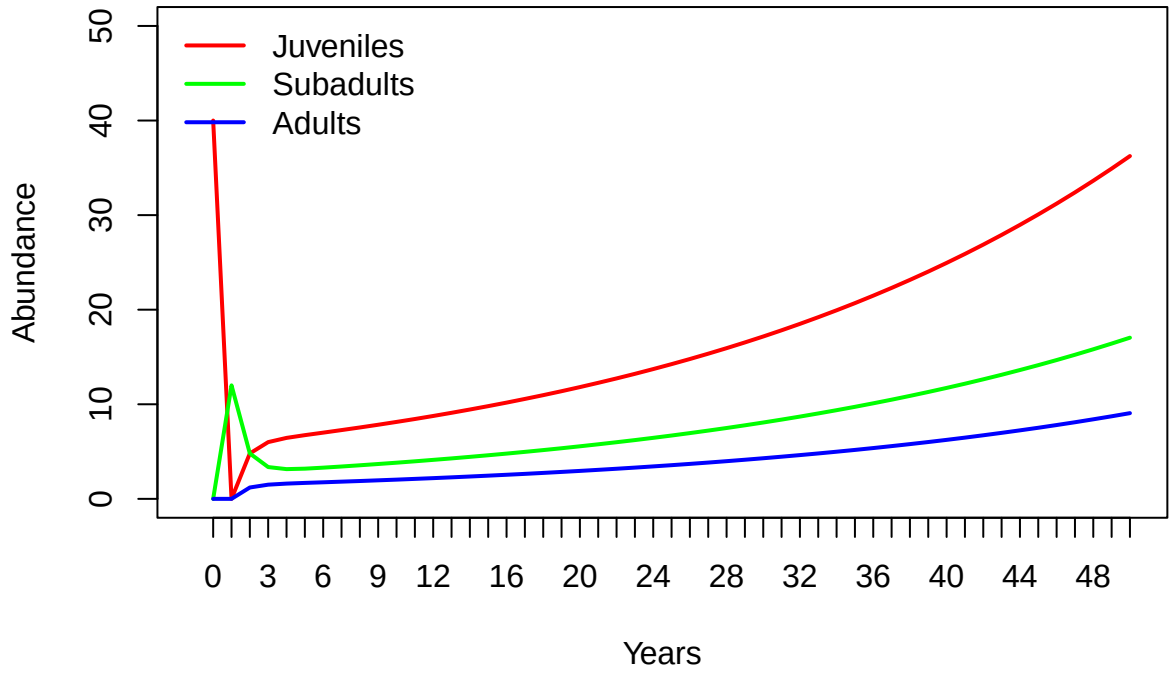
```
# Run the model for 50 years (using for loop)

nYears <- 50
allYears <- matrix(0,nrow=nrow(TMat),ncol=nYears+1)
rownames(allYears) <- rownames(TMat)
colnames(allYears) <- seq(0,nYears)
allYears[,1] <- InitAbund

for(t in 2:(nYears+1)){
  allYears[,t] <-  TMat %*% allYears[,t-1]
}

allYears
```

```
##            0  1   2    3     4       5        6        7        8        9       10       11       1
## Juveniles 40  0 4.8 6.00 6.444 6.73500 7.001070 7.269637 7.546811 7.834194 8.132444 8.442032 8.76340
## Subadults  0 12 4.8 3.36 3.144 3.19080 3.296820 3.419049 3.548511 3.683448 3.823637 3.969188 4.12028
## Adults     0  0 1.2 1.50 1.611 1.68375 1.750267 1.817409 1.886703 1.958549 2.033111 2.110508 2.19085
##                 14       15        16        17        18        19        20        21        22
## Juveniles 9.443309 9.802795 10.175966 10.563343 10.965466 11.382897 11.816219 12.266036 12.732977 13
## Subadults 4.439956 4.608975  4.784429  4.966561  5.155627  5.351891  5.555625  5.767116  5.986657  6
## Adults    2.360827 2.450699  2.543992  2.640836  2.741366  2.845724  2.954055  3.066509  3.183244  3
##                   24        25        26        27        28        29        30        31        32
```

```
## Juveniles 13.720861 14.243184 14.785391 15.348238 15.932512 16.539027 17.168631 17.822203 18.500655
## Subadults  6.451130  6.696711  6.951640  7.216273  7.490981  7.776146  8.072166  8.379456  8.698443
## Adults     3.430215  3.560796  3.696348  3.837060  3.983128  4.134757  4.292158  4.455551  4.625164
##                  34        35        36        37        38        39        40        41        42
## Juveniles 19.936024 20.694944 21.482755 22.300556 23.149489 24.030739 24.945536 25.895157 26.880929
## Subadults  9.373310  9.730131 10.100536 10.485041 10.884183 11.298520 11.728630 12.175113 12.638592
## Adults     4.984006  5.173736  5.370689  5.575139  5.787372  6.007685  6.236384  6.473789  6.720232
##                  44        45        46        47        48        49        50
## Juveniles 28.96648 30.069168 31.213835 32.402077 33.635552 34.915984 36.245158
## Subadults 13.61915 14.137605 14.675793 15.234468 15.814410 16.416430 17.041367
## Adults     7.24162  7.517292  7.803459  8.100519  8.408888  8.728996  9.061289
```

Now let's plot it out!

```
# and plot out the results!

plot(1,1,pch="",ylim=c(0,50),xlim=c(0,nYears+1),xlab="Years",ylab="Abundance",xaxt="n")
cols <- rainbow(3)
for(s in 1:3){
  points(allYears[s,],col=cols[s],type="l",lwd=2)
}
axis(1,at=seq(1,nYears+1),labels = seq(0,nYears))
legend("topleft",col=cols,lwd=rep(2,3),legend=rownames(allYears),bty="n")
```



Does this look the same as the InsightMaker results?

Limitations of matrix population models

Matrix population models are great, but they have some limitations too.

What about density-dependence?

In some ways, while introducing a new level of realism in our models – age-structure – we have been ignoring another type of realism that we introduced in earlier lectures- **density-dependence** (both positive and negative)!

Which vital rates are density-dependent? All? Some? It depends? Are the data available?

How do you incorporate density-dependence into a matrix population model?

How do you incorporate predator-prey dynamics into a matrix population model?

ANSWER: use computer programming (e.g., R or InsightMaker!)

Overview of matrix multiplication

(we will do this on the whiteboard!)

In-class exercise: loggerhead conservation!

First, please review the introduction to loggerhead sea turtles here.

Next, download the Excel document for this example here. We will work through the Excel example in class.

Using your Excel spreadsheet, investigate four management scenarios for the loggerhead turtle population:

1. Improve fecundity (labeled "Sub Fecundity" and "Adult Fecundity" in the IM model) via *nest-site protection*!
    - Double fecundity (improve to 2X the current value)

2. Improve hatchling survival (labeled "hatchling growth" in the IM model) via *nest monitoring*
    - Improve *to* 100%

3. Improve large juvenile survival (labeled "Lg Juv Surv" in the IM model) using *Turtle Excluder Devices (TEDs)*
    - Add 0.25 to the existing large juvenile survival

4. Improve adult/subadult survival (labeled "Sub Surv" and "Adult Surv" in IM) by *restricting longline fisheries*.
    - Increase by 10% (add 0.1 to the existing subadult and adult survival)

PointSolutions: Which of the four management recommendations would have the greatest positive effect on this population?

Figure 1: turtle excluder device

translate life history description into matrix population model

If we have time, we may review the final question from Lab 4:

Translate the following paragraph into a matrix population model. Remember a matrix population model has two components- an **initial abundance vector** and a **transition matrix**.

We assumed that the red-tailed hawk life history could be described in terms of three major life stages: hatchling (first year of life), juvenile (largely individuals in their second year of life), and

adult (generally the third year of life and beyond). We assumed that adult females experienced an average of 15% mortality each year. Juvenile female mortality was set at 30% per year. Approximately 10% of juvenile females remain in the juvenile phase each year, and all other survivors transition to the adult stage. Finally, hatchlings had a 21% chance of surviving and transitioning to become juveniles. Adults are the primary reproductive stage, and produce an average of 6 new hatchlings each year, half of which are female. Juveniles that fail to transition to the Adult stage tend to produce only 1.8 new hatchlings each year on average, half of which are female. We ran a female-only matrix population model, and we initialized the population with 800 hatchlings, 150 juveniles, and 50 adults.

For more on matrix population models, the bible of this subject is this book by Hal Caswell.

And finally, check this out- this is a database of thousands of stage matrices for plants and animals around the world:

COMADRE and COMPADRE databases

–go to next lecture–