

## Lab 11: Using Analytical Formulae to Estimate Matrix Properties in Matlab

**Objectives:** In the exercises we have completed so far, we have used the power method to estimate basic matrix properties. This has proven to be cumbersome for two reasons: 1) it is unclear how many times the matrix must be raised to higher powers to obtain the asymptotic values and 2) if  $\lambda$  is particularly low then the cell values may run out of decimal places leading to overflow. The goals of this exercise are threefold: to introduce the use of Matlab files (i.e., m-files), to present analytical methods for doing the matrix calculations, and to demonstrate how the lower-level elasticities can be calculated.

**I. m-files.** Files with the extension “.m” are recognized by Matlab as files written with Matlab programming tools. They are saved as ASCII files and can be edited with any DOS editor. These files save retyping the text of a particular program.

Startup Matlab. Click on File | New | m-file to create a new m-file.

Add a line of documentation to this file to remind you what it is. Type this on the first line:

```
% Killer Whale matrix from Brault and Caswell;
```

It should appear in green type

Next, enter the projection matrix for the killer whale:

```
mat = [0 0.0043 0.1132 0; 0.9775 0.911 0 0; 0 0.0736 0.9534 0; 0  
0 0.0452 0.9804]
```

Click on File | Save to save the file in the work directory. Name it “killer.m”. Close this window and go back to Matlab. Make sure that the Current Directory on the line above the Command Window reads: “c:\matlabR12\work”. At the command line prompt >> type:

```
killer
```

and the matrix should appear in the command window.

**II. Matrix calculations.** Now we are ready to begin the matrix calculations.

Go back to File | Open and retrieve the killer.m file.

### Step 1. Calculation of population growth rate, damping ratio and convergence time

Leave a blank line below the matrix and in the next two lines type the following commands:

```
[wmat, dmat] = eig(mat)
```

```
vmat = conj(inv(wmat))
```

These two Matlab commands produce a matrix of eigenvalues (dmat) and matrices of the eigenvectors (wmat and vmat). Save this m-file, return to the Command Window and type:

```
killer
```

Start with calculation of the population growth rate. Lambda ( $\lambda$ ) is the dominant eigenvalue for a given matrix, and there may be other eigenvalues for the same matrix.  $\lambda$  is sneaky and can appear anywhere on the diagonal of matrix dmat. Subdominant eigenvalues are sometimes complex numbers with an imaginary component (i.e.,  $\sqrt{-1}$ ). Go back to the m-file and type:

```

lambdiag = diag(dmat);
lambvec = sort(abs(lambdiag));
dim = size(lambvec,1);
lambda1 = lambvec(dim)
lambda2 = lambvec(dim-1)
rho = lambda1 / lambda2
t20 = log(20) / log(rho)

```

### What do these 7 commands do?

Line 1 pulls out the diagonal vector out of the dmat matrix.

Line 2 takes the absolute values of all the elements of the vector and then sorts them, lowest to highest, in ascending order.

Line 3 counts up the number of elements in the vector

Line 4 pulls out the last value, which is the dominant eigenvalue or  $\lambda$  because the values have been sorted in ascending order.

Line 5 pulls out the second to last value which is the subdominant eigenvalue.

Line 6 calculates rho or  $\rho$  which is a metric known as the ‘the damping ratio’. The damping ratio is a measure of how fast the population might be expected to converge to the stable age distribution. Rho can be sensitive to the size of the matrix.

Line 7 calculates t20 or  $t_{20}$  which is the ‘time to convergence’. This is a measure of how many years it would take for the contribution of lambda1 to be 20 times as great as that of lambda2 (20 is a somewhat arbitrary value). In Matlab, log(x) is the function used to calculate natural logarithms.

### Step 2. Calculation of stable age distribution and reproductive values

Now we are ready to calculate the stable age distribution and reproductive values from the eigenvector matrices. The trick here is that the correct eigenvectors are in the matrix column that corresponds to whatever element the lambda1 value is in. Depending on the matrix this column might be in a different position. Type the following in the m-file.

```

imax = find(lambdiag==max(lambdiag));
rawage = wmat(:,imax);
sad = rawage./sum(rawage)
rawrv = vmat(imax,:)' ;
rv = rawrv / rawrv(1,1)

```

Save the file and type `killer` in the Command Window to run the new version. The command lines do not print anything to the screen if there is a semicolon on the end. Remove the semicolons if you want to see the intermediate calculations for imax, rawage and rawrv.

### What do these 5 commands do?

Line 1 determines which element of the diagonal vector the value of lambda1 is in and then names it imax. imax is a reference number and is not the value of lambda1.

Line 2 uses imax to pull column with the corresponding eigenvector out of the wmat matrix. The colon symbol pulls out all the values in the appropriate column.

Line 3 divides the vector on a cell by cell basis by the sum to obtain the stable stage distribution.

Line 4 again uses imax to pull the row with the corresponding eigenvector out of the vmat matrix. The quotation mark at the end transposes the vector to a column.

Line 5 divides this vector through by the first value to obtain the scaled reproductive values where the reproductive value of the youngest node,  $rv(1,1)$ , is set to one.

### Step 3. Calculation of sensitivity and elasticity matrices

The last step is to calculate the sensitivity and elasticity matrices. We did something like this last week after we calculated  $sad$  and  $rv$  with the power method. Type the following in the m-file.

```
sens = (rv*sad') ./ (sum(rv.*sad))  
elas = (mat./lambda1) .*sens
```

What do these 2 commands do?

Line 1 calculates the sensitivity matrix using the familiar formula that you have seen in lecture and last week's lab exercise:

$$\frac{\partial \lambda}{\partial a_{ij}} \text{ or } s_{ij} = \frac{v_i w_j}{\langle \mathbf{v}, \mathbf{w} \rangle}$$

The sensitivity matrix is calculated by multiplying the two vectors and then dividing through by the scalar product. Remember the syntax for matrix operations: a period before a division (/) or multiplication (\*) operator means the calculations are done on an element by element basis. If no period is included, the rules of matrix multiplication are used instead.

Line 2 calculates the elasticity matrix by dividing the original matrix through by the rate of population change and then multiplying it on an element by element basis with the sensitivity matrix.

$$\frac{\partial \ln \lambda}{\partial \ln a_{ij}} \text{ or } e_{ij} = \frac{a_{ij}}{\lambda} s_{ij}$$

Save and close the killer.m file.

**III. Lower-level elasticities.** You wrestled with estimation of the partial derivatives for the killer whale matrix in your last assignment. As it turns out, it is possible to do these calculations in Matlab.

### Step 1. Define the symbols for the variables

Create a new m-file and name it 'kestrel.m'

Put whatever headers you want at the start of the file by putting a % sign at the start of the line. Calculation of partial derivatives in Matlab requires use of symbolic notation. Type the following:

```
syms b co ca so sa;  
mat = [co*b*so ca*b*so; sa sa]
```

The first line creates the symbolic variables and the second then casts the variables into a matrix.

Recall that for the kestrel paper, the five demographic rates included:  $b$  = number of female offspring per female,  $co$  = probability of breeding among juveniles,  $ca$  = probability of breeding among adults,  $so$  = survival of juveniles and  $sa$  = survival of adults

## Step 2. Calculation of the partial derivatives

Where  $\frac{\partial \lambda}{\partial a_{ij}}$  is the sensitivity of the matrix elements,  $\frac{\partial \lambda}{\partial x}$  is the sensitivity of a demographic rate

$x$  that may appear in one or more of each of the matrix elements. The lower-level sensitivities for demographic rate  $x$  are calculated as:

$$\frac{\partial \lambda}{\partial x} = \sum_{i,j} \frac{\partial \lambda}{\partial a_{ij}} \frac{\partial a_{ij}}{\partial x} \text{ where } \frac{\partial \lambda}{\partial a_{ij}} \text{ is the usual matrix of sensitivity values } \frac{\partial a_{ij}}{\partial x} \text{ is the partial}$$

derivative of each matrix cell with respect to  $x$ .

To calculate partial derivatives in preparation for estimation of lower-level sensitivity values, type:

```
PD_b = diff (mat, b)
PD_co = diff (mat, co)
PD_ca = diff (mat, ca)
PD_so = diff (mat, so)
PD_sa = diff (mat, sa)
```

The syntax of these five lines uses the `diff(X,N)` function of Matlab to take the partial derivative of matrix `mat` with respect to variable  $x$ . The variable names are arbitrary but are named PD for partial derivative.

Next type:

```
b = 0.9321;
co = 0.3847;
ca = 0.9250;
so = 0.3409;
sa = 0.7107;
```

These five lines set values for the five demographic rates. Next type:

```
mat = subs(mat);
PD_b = subs(PD_b);
PD_co = subs(PD_co);
PD_ca = subs(PD_ca);
PD_so = subs(PD_so);
PD_sa = subs(PD_sa);
```

These six lines use the `subs(x)` of Matlab to replace the symbolic notation in the matrices with the actual values of those five rates. The first line substitutes the parameter values into the original matrix; the next five lines substitute the values into all of the partial derivative matrices as well.

Now you are ready to do all the standard calculations that you just did for the Killer Whale matrix. Go to File | Open and retrieve the `killer.m` file. Cut and paste all of the code that you just prepared for killer whales into the `kestrel.m` file. Be sure to discard the first line of `killer.m` that defines the killer whale matrix as `mat`.

### Step 3. Calculation of lower-level metrics

The final step is to calculate the lower level sensitivities and elasticities. Type the following at the bottom of the m-file.

```
S_b = sum(sum(PD_b.*sens))
S_co = sum(sum(PD_co.*sens))
S_ca = sum(sum(PD_ca.*sens))
S_so = sum(sum(PD_so.*sens))
S_sa = sum(sum(PD_sa.*sens))
```

What are these commands doing? The lower-level sensitivities are the product of the partial derivative matrix for each vital rate and the sensitivity matrix. Nested within the brackets, the partial derivative matrices are multiplied on a cell by cell basis with the sensitivity matrix. The inner summation function pools values in all of the columns, the outer summation function then pools the sums of the columns to produce a scalar value.

```
E_b=b/lambda1*S_b
E_co=co/lambda1*S_co
E_ca=ca/lambda1*S_ca
E_so=so/lambda1*S_so
E_sa=sa/lambda1*S_sa
```

The lower-level elasticities are calculated just like a matrix element: the individual vital rates are divided by lambda1 (the dominant eigenvalue), and then multiplied by the lower-level sensitivity value for the demographic rate.