

Lab 10: Quick Reference to MATLAB Commands

Simple mathematics and expressions:

+	addition	sum(x)	gives the sum of variable x
-	subtraction	abs(x)	returns the absolute value of variable x
*	multiplication	mean(x)	gives the arithmetic mean of variable x
/	right division	std(x)	gives the standard deviation
^	power		
.*	element by element multiplication		

Working with variables in the memory:

;	semicolon at end of line suppresses printing to screen
who	List of the variables that are in the memory
clear	Purges the memory to start a new session
clc	Clears the input window and starts over at top
<variable name>	Gives the values of that variable that are in the memory
arrow-keys	Let you scroll through the commands you have used in the session
	Note that Matlab is case-sensitive! a ≠ A

Variable and matrix handling:

D=[0.8 0.9 0.7]	Produces a 1 x 3 vector named 'D'.
A=[0 0 0.0756; 0.63 0 0; 0 0.9 0.9]	Produces a 3 x 3 matrix named "A". The semicolons separate each row, each of which has three values.
[X, D]=eig(A)	Calculates the right eigenvectors and eigenvalues of a matrix. "X" is an arbitrary name for a matrix of eigenvectors, and "D" produces a diagonal matrix of eigenvalues (e.g., [vec, val]=eig(A) gives matrices "vec" and "val" for matrix "A")
B=A(row, column)	Creates a new variable where you can pull out a cell, row or column out of a matrix. Use ":" as a wildcard to indicate all rows or columns (e.g., A(:,3) pulls out the 3rd column of matrix A)
C=A'	Creates a new matrix named "C" where the values of array A have been transposed across a diagonal axis

What does it all mean?

Eigenvalue: largest positive, real number in D (for A)	Lambda (λ)
Right eigenvectors: associated column of numbers in X (for A)	Stable age distribution (unscaled)
Left eigenvectors: the right eigenvectors associated with lambda (again), but for C which is the transposed matrix of A	Reproductive value (unscaled)

Lab 10: Estimating Sensitivity and Elasticity with Matlab

Exploratory steps

Startup Matlab. The opening screen of MATLAB will have two or more windows. On the right is the Command Window where you can enter your commands. At the cursor symbol (`>>`) type the following:

Example 1. Simple multiplication.

```
mat=[0 0 0.108 0.108; 0.630 0 0 0; 0 0.8 0 0; 0 0 0.9 0.9]
b = mat*3
mat*3
who
clear b ans
mat^50
clear
```

creates a projection matrix named 'mat'
creates a matrix b where mat is multiplied by 3
gives the answer to the same in variable 'ans'
shows the variables in the memory
gets rid of variables b and ans
raises the matrix to the 50th power
dumps all the variables

Example 2. Matrix multiplication

```
syms a b c d e f g h
X=[a b;c d]
Y=[e f;g h]
X*Y
X.*Y
clear
```

creates eight symbolic variables named a to h
creates two matrices named X and Y
multiplies X and Y using the rules of matrix multiplication
multiplies X and Y on an element by element basis
dumps all the variables

Note the subtle difference in the last two commands. Without a period, calculations follow matrix multiplication rules. With a period, only values that are in the same position in the matrix are multiplied.

Calculating lambda (λ), stage age (w), reproductive value (v), sensitivity (S) and elasticity (E) with the Power method

Several papers in the reader used the "power method" to estimate λ , w and v (e.g., Crouse et al. 1987:1416). Let's reanalyse the Killer Whale matrix in Matlab.

```
mat=[0 0.0043 0.1132 0; 0.9775 0.9111 0 0; 0 0.0736 0.9534 0; 0 0 0.0452 0.9804]
```

Raise the matrix to higher powers several times, and watch the cell values change. Why do cells with zero progressively fill with values?

Try: `mat^2`, `mat^3`, `mat^4`, etc.

The values in the cell get progressively smaller but the proportions eventually stabilize. The relative difference between the cells also stabilizes. Enter the following commands and observe the change in the values. This value is the eigenvalue or lambda, watch how it stabilizes.

```
(mat^5)./(mat^4)
(mat^7)./(mat^6)
(mat^10)./(mat^9)
```

Capture this value into a scalar named lambda.

```
temp=(mat^250)/(mat^249)      Ratio of two matrices raised to high power  
lamb=temp(1,1)                Pulls element in row 1 column 1 into variable lamb
```

It is possible to calculate the left eigenvector or the reproductive values (RV or \mathbf{v}) by summing the columns of a stabilized matrix. These values are then scaled so that RV for the first node is one.

```
temp=mat^250                  Calculates stabilized matrix  
temp=sum(temp)                Sums down columns  
rvw=temp/temp(1,1)           Scales by the first cell  
rvw=rvw'                      Transposes to a column vector
```

rvw should be a vector of reproductive values, compare it to pages 1447 to 1448 of Brault and Caswell

Doing the same thing for the transposed matrix gives the right eigenvector or the stable age distribution (SAD or \mathbf{w}). These values are then scaled so that they sum to one.

```
temp=mat^250                  Calculates stabilized matrix  
temp=temp'                    Transposes stabilized matrix  
temp=sum(temp)                Sums down columns (rows of original stabilized matrix)  
age=temp/sum(temp)            Scales by sum  
age=age'                      Transposes to a column vector
```

age should be a vector of stable age distribution values, compare it to pages 1447 to 1448 of Brault and Caswell

The next step is to use the estimates of scaled RV (\mathbf{v}) and scaled age (\mathbf{w}) to calculate the sensitivity matrix. The formula for sensitivity is:

$$\frac{v_i w_i}{\langle \mathbf{v}, \mathbf{w} \rangle}$$

How do we implement this in Matlab? We can continue with the estimates of λ , \mathbf{v} , and \mathbf{w} for the Killer Whale. First, we calculate the numerator as weighted reproductive value multiplied by the transposed matrix of the stable age distribution:

```
num=rvw*age'
```

The denominator in brackets “ $\langle \rangle$ ” indicates a scalar product:

```
den=sum(rvw.*age)
```

The final calculation is thus:

```
sens=num./den
```

sens should be the matrix of sensitivity values, compare it to pages 1447 to 1448 of Brault and Caswell.

How about the elasticity values? The formula for elasticity is given below where a_{ij} indicates the cells in the original matrix.

$$\frac{a_{ij} \cdot v_i w_j}{\lambda \langle v_i w_j \rangle}$$

How can we do these calculations in Matlab? Let's calculate the first part of the equation
`firstbit=mat./lamb`

The final calculation is thus:
`elas=firstbit.*sens`

`sens` should be the matrix of elasticity values, compare it to pages 1447 to 1448 of Brault and Caswell.

This exercise should illustrate the ease with which matrix operations can be conducted within Matlab. These are somewhat simpler than using the same power method in an Excel spreadsheet.

Want to try the same calculations for a different matrix in your reader? Try one of the matrices below.

```
% Lesser Kestrel;
% Hiraldo, F., J.J. Negro, J.A. Donazar, and P. Gaona. 1996. A demographic
model for a population;
% of the endangered Lesser Kestrel in southern Spain. Journal of Applied
Ecology 33:1085-1095;
% Data from Table 6, matrix from Eqn 2;
mat = [0.1222 0.2939;
       0.7101 0.7101];

% Loggerhead sea turtle;
% Crouse, D.T., D.B. Crowder, and H. Caswell. 1987. A stage-based population
model...
% loggerhead sea turtles and implications for conservation. Ecology 68:
1412-1423.
% Matrix of values from Table 4;
mat=[ 0           0           0   127.0000   4.0000   80.0000;
      0.6747     0.7370     0     0         0         0;
      0          0.0486     0.6610  0         0         0;
      0          0          0.0147  0.6907   0         0;
      0          0          0          0.0518  0         0;
      0          0          0          0         0.8091  0;
      0          0          0          0         0         0.8091  0.8089];

%Table III of McDonald and Caswell (1993) Current Ornith 10:139-185.
mat=[0.0110     0           0   0.5030   0.5820   0.8200;
     0.1560     0           0     0         0         0;
     0          0.1310     0.0790  0         0         0;
     0.1690     0           0     0         0         0;
     0          0.5090     0.6610  0         0         0;
     0.0140     0           0   0.6400   0.7400   0.8200];
```