. **Lesson Two Student Notes**

Working with Matrices

MATLAB provides four functions that generate basic matrices. They are **zeros**, **ones**, **rand**, and **randn**.
The MATLAB function call that creates an n-by-n array filled with zeros is **zeros(n)**.
The MATLAB function call that creates an m-by-n array filled with zeros is **zeros(m,n)**.
The MATLAB function call that creates an array filled with zeros with the same dimensions as matrix A is **zeros(size(A))**.
The MATLAB function call that creates an n-by-n array filled with ones is **ones(n)**.
The MATLAB function call that creates an m-by-n array filled with ones is **ones(m,n)**.
The MATLAB function call that creates an array filled with ones with the same dimensions as matrix A is **ones(size(A))**.
The **rand** function generates arrays of random numbers whose elements are uniformly distributed in the interval **(0,1)**.
rand(n) will generate an n-by-n array of uniformly distributed random numbers in the range **(0,1)**.
rand(m,n) will generate an m-by-n array of uniformly distributed random numbers in the range **(0,1)**.
rand(size(A)) will generate an array of uniformly distributed random numbers that has the same dimensions as the matrix A.
The **randn** function generates arrays of random numbers whose elements are normally distributed with mean **0** and standard deviation **1**.
randn(m,n) will generate an m-by-n array of normally distributed random numbers with mean **0** and standard deviation **1**.

The **load function** reads binary files containing matrices generated by earlier MATLAB sessions, or reads text files containing numeric data. The text file should be organized as a rectangular table of numbers, separated by blanks, with one row per line, and an equal number of elements in each row.
You can create your own matrices using M-files, which are text files containing MATLAB code. Use the MATLAB Editor or another text editor to create a file containing the same statements you would type at the MATLAB command line. Save the file under a name that ends in .m
Concatenation is the process of joining small matrices to make bigger ones. In fact, you made your first matrix by concatenating its individual elements. The pair of square brackets, [], is the concatenation operator.
For example: if P is the matrix

$$\begin{bmatrix} 0.70 & 0.15 & 0.15 \\ 0.20 & 0.80 & 0.15 \\ 0.10 & 0.05 & 0.70 \end{bmatrix}$$

Then [P-eye(3), zeros(3,1)] would be the 3 by 4 matrix:

$$\begin{bmatrix} -0.3 & 0.15 & 0.15 & 0 \\ 0.20 & -0.2 & 0.15 & 0 \\ 0.10 & 0.05 & -0.3 & 0 \end{bmatrix}$$

You can delete a rows or columns by assigning them to [].
For example if we start A as the matrix:

$$A = \begin{bmatrix} -0.3 & 0.15 & 0.15 & 0 \\ 0.20 & -0.2 & 0.15 & 0 \\ 0.10 & 0.05 & -0.3 & 0 \end{bmatrix}$$

then
A (: , end) = []
leaves A as
A =

$$\begin{bmatrix} -0.3 & 0.15 & 0.15 \\ 0.20 & -0.2 & 0.15 \\ 0.10 & 0.05 & -0.3 \end{bmatrix}$$

If we continue the assignment to [] using only a single subscript then the operation behaves as if the original matrix is first reshaped into a single row vector. So that with A as previously
A([1, 5,9]) = []
now leaves
A =
[0.15, 0.15, 0.20, 0.15, 0.10, 0.05]

Linear Algebra Notes

The **matrix multiplication** A B requires that the number of columns in A must be the same as the number of rows in B, i.e. if A is m by p the B must be p by n. Alternately we could say that the length of the rows of A must be the same as the length of the columns of B. The result of multiplying an m by p matrix A times a p by n matrix B is an m by n matrix C in which the (i,j) element is the sum of the products of the elements of the ith row of A and the jth column of B. For example

$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} * \begin{bmatrix} 5 \\ 6 \end{bmatrix}$ yields the 2 by 1 product $\begin{bmatrix} 17 \\ 39 \end{bmatrix}$

Matrix multiplication is not commutative, i.e. generally $A*B \neq B*A$.

The common method of **solving a system of linear equations** is to write the augmented matrix for the system and then use elementary row operations to reach an equivalent system of equations in **row reduce echelon form**. This method is called the Gauss-Jordan elimination method. The MATLAB function that does the reduction to reduced row echelon form is rref(). The solutions to the system of equations can be read from the reduced row echelon form.

The **inverse of a square matrix** A is a matrix B for which A*B = I = B*A where I is the identity matrix. If a square matrix A has an inverse we call it invertible or **nonsingular** otherwise it is **singular** or noninvertible. The inverse matrix can also be used to solve systems of equations:

If A*X = C and A is an n by n invertible matrix then X = B*C where B is the inverse of A.

If square matrix A has a scalar λ and a vector X for which

$A*X = \lambda X$

Then we say λ is an **eigenvalue** for A with the associated **eigenvector** X. In MATLAB eig(A) returns a column vector of the eigenvalues of A.

For example: if P is the matrix

$$\begin{bmatrix} 0.70 & 0.15 & 0.15 \\ 0.20 & 0.80 & 0.15 \\ 0.10 & 0.05 & 0.70 \end{bmatrix}$$

and X is the vector $\begin{bmatrix} 7000 \\ 10000 \\ 4000 \end{bmatrix}$

then P*X is $\begin{bmatrix} 7000 \\ 10000 \\ 4000 \end{bmatrix} = X$

then one is an eigenvalue of P with corresponding eigenvector $\begin{bmatrix} 7000 \\ 10000 \\ 4000 \end{bmatrix}$.

More About Matrices and Arrays

Matrix operations from Linear Algebra

det(A) returns the determinant of the square matrix A.
rref(A) returns the matrix that is row equivalent to A and is in reduced row echelon form.
inv(A) returns the inverse matrix of a nonsingular matrix A.
eig(A) returns a column vector of the eigenvalues of square matrix A.
[name1, name2] = eig(A) returns the eigenvectors as the columns of name1, and the corresponding eigenvalues as the diagonals of the diagonal matrix name2.

Array Operator Notes

Arithmetic operations on arrays are done element by element. This means that addition and subtraction are the same for arrays and matrices, but that multiplicative operations are different. MATLAB uses a dot, or decimal point, as part of the notation for multiplicative array operations.
The list of array operators includes

+	Addition
-	Subtraction
.*	Element-by-element multiplication
./	Element-by-element division
.\	Element-by-element left division
.^	^Element-by-element power
.'	Unconjugated array transpose

Suppose n is the column vector n = (0:9)';
Then pows = [n n.^2 2.^n]
builds a table of squares and powers of 2.
The elementary math functions operate on arrays element by element.
so
x = (1:0.1:2)';
and
logs = [x log10(x)]
builds a table of logarithms.

Matrices and scalars can be combined in several different ways. For example, a scalar is subtracted from a matrix by subtracting it from each element.
With **scalar expansion**, MATLAB assigns a specified scalar to all indices in a range. For example, B(1:2,2:3) = 0
zeroes out a portion of B.

Logical Subscripting

The logical vectors created from logical and relational operations can be used to reference subarrays. Suppose X is an ordinary matrix and L is a matrix of the same size that is the result of some logical operation. Then X(L) specifies the elements of X where the elements of L are nonzero.

The find Function

The find function determines the indices of array elements that meet a given logical condition. In its simplest form, find returns a column vector of indices. Transpose that vector to obtain a row vector of indices.