MATRIX INDICES:

Matrices can be represented conveniently in MATLAB by writing individual rows, separated by semicolons (this use of the semicolon is different of the use of the same character at the end of a line). For example, typing

>> A = [1 2 3u4 5 6u7 8 9];

creates a 3&3 matrix in MATLAB in which the first row has elements 1, 2, 3, the second row has elements 4, 5, 6, and so on (note the use of the brackets [ ]). If, instead, we write the same line without the final semicolon, we would get

>> A = [1 2 3u4 5 6u7 8 9]

A =

1 2 3

4 5 6

7 8 9

One of MATLAB's most powerful index operators is the colon operator | : }. It is used in a number of different forms. For example, writing

>> 1:5

produces the row of integers

1 2 3 4 5

If we wanted to create an 1 x N matrix, X, containing the sequence 1, 2,.....,N for an

integer value of N, we would write

>> X = 1:N;

Alternatively, we could use the fact that a:b is the same as colon(a,b) and write

>> X = colon(1,N);

Nonuniform spacing can be obtained by specifying an increment. For example,

>> 100: -10:50

produces the sequence

100 90 80 70 60 50

The same sequence would be produced by the statement colon(100,10,50).

When used in matrix indexing, the colon operator selects parts of a matrix. For example, the following statement extracts the third column of A:

>> C3 = A(:,3)

C3 =

3

6

9

Similarly, we extract the second row as follows:

>> R2 = A(2,:)

R2 =

4 5 6

Finally, the following statement extracts the top two rows:

>> T2 = A(1:2,1:3)

T2 =

1 2 3

4 5 6

Note in the first example how using the colon in A(:,3) signified all the rows of the array. In other words, A(:,3) is the same as A(1:3,3).

The colon notation can be used also in combination with other operators. For example,

>> S = sum(B(1:k,j))u

computes the sum of the first k elements of the jth column of B and stores it as s.

GENERATING MATRICES:

The colon operator is an important tool used in a number of different applications

throughout the rest of the book.

GENERATING MATRICES:

MATLAB provides four functions that generate the following basic matrices:

-zeros(M,N) generates an M £N matrix of zeros.

- ones(M,N) generates an M £N matrix of ones.

- rand(M,N) generates an M £ N matrix whose entries are uniformly-distributed random numbers in the interval [0:0; 1:0].

-randn(M,N) generates an M£N matrix whose numbers are normally-distributed (i.e., Gaussian) numbers with mean 0 and variance 1.

***[Margin Note: See Chapter 5 for additional discussions on random numbers].

For example:

>> A = 5*ones(3,3)

A =

5 5 5

5 5 5

5 5 5

>> B = rand(2,4)

B =

0.2311 0.4860 0.7621 0.0185

0.6068 0.8913 0.4565 0.8214

MATRIX CONCATENATION:

Matrix concatenation is the process of joining small matrices to create larger matrices.

The concatenation operator is the pair of square brackets, [ ]. Earlier in this section,

when we wrote

>> A = [1 2 3u4 5 6u7 8 9]

all we did was concatenate three rows (separated by semicolons) to create a 3&3 matrix.

Note that individual elements are separated by a space. Separating the elements by 6 commas would yield the same result. Note also the order in which the elements were concatenated. The group of three elements before the first semicolon formed the first row, the next group formed the second row, and so on. Clearly the number of elements between semicolons have to be equal. This concept is applicable also when the elements themselves are matrices. For example, consider the 2 & 2 matrix

>> B= [1 2u3 4] u

The statement

>> C = [B BuB+4 B1]

in which B + 4 and B ¡ 1 indicate adding 4 and subtracting 1 from all elements of B,

respectively, yields the result

C =

1 2 1 2

3 4 3 4

5 6 0 1

7 8 2 3

Note how matrix B was duplicated twice in the top half of C, corresponding to the two

occurrences of B before the first semicolon in the statement C = [B BuB+4 B1].