Brief review of last week
Additional ideas:
- Special arrays
- Changing an array
- Some array operators
- Character arrays
Element by element math operations
Textbook – rest of chapter 2, 3.1, 3.4, 3.5

Review of Last Week
- The fundamental data unit in Matlab
  - Rectangular collection of data
  - All variables are considered to be arrays

\[
\text{yield} = \begin{bmatrix}
4 & 5 & 3 & 9 \\
10 & 4 & 66 & 20 \\
18 & -3 & 2 & 0 \\
\end{bmatrix}
\]

- Data values organized into rows and columns

Size of an array
Construction:
- Brute force using brackets
- Concatenation of other arrays – L/R and U/D
- The colon operator
Addressing:
- Individual elements
- Subarrays

Some new tricks:
- \textit{end} specifies the last row or column
- A colon (:) specifies all of a row or column

\[
\text{yield}(2,:)
\]
\[
\text{yield(end,1)}
\]
- Use a single index for row and column vectors

\[
bob = \begin{bmatrix}
9 & 7 & 5 & 7 & 2
\end{bmatrix}
\]

Special Arrays
- Special predefined arrays:
  - All zeros: zeros(R,C) zeros(N)
  - All ones: ones(R,C) ones(N)
  - Zeros with ones on the diagonal: eye(R,C) eye(N)
  - Random numbers (within [0 1]): rand(R,C) rand(N)

Square versions

Command Window
\[
>> \text{eye}(3,4)
\]
ans =
\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
\end{bmatrix}
\]
\[
>> \text{rand}(2,5)
\]
ans =
\[
\begin{bmatrix}
6.9501 & 0.6068 & 0.8013 & 6.4865 & 9.8214 \\
6.2311 & 0.8956 & 0.7621 & 0.0165 & 9.9447
\end{bmatrix}
\]
- **Linspace** – like the colon operator, but definitely gets the last number on the list
  
  \[
  \text{linspace} \ (\text{start, last, number of values})
  \]
  
  - examples:
    
    \[
    \text{linspace}(0,10,6) \quad \rightarrow \quad [0 \ 2 \ 4 \ 6 \ 8 \ 10]
    \]
    \[
    \text{linspace}(0,1,4) \quad \rightarrow \quad [0 \ 0.333 \ 0.667 \ 1]
    \]
  
  - default for number is 100
    
    \[
    \text{linspace}(0,10) \quad \rightarrow \quad [0 \ 0.101 \ 0.202 \ldots \ 10]
    \]
    
    Note increment for 100 points, 99 intervals

### Changing an Array

Recall reading an array value:

\[
\text{z} = \begin{bmatrix}
0.4565 & 0.8214 & 0.6154 \\
0.0185 & 0.0447 & 0.7919
\end{bmatrix}
\]

\[
\text{>> } x = \text{test}(1,3)
\]

\[
x =
\begin{bmatrix}
0.6154
\end{bmatrix}
\]

- **To change a single value in an array**
  
  - Use addressing on the *left hand side* of =
  
  \[
  \text{test} =
  \begin{bmatrix}
0.4565 & 0.8214 & 0.6154 \\
0.0185 & 0.0447 & 0.7919
\end{bmatrix}
\]

  \[
  \text{>> } \text{test}(1,2) = 5
  \]

  \[
  \text{test} =
  \begin{bmatrix}
0.4565 & 5.0000 & 0.6154 \\
0.0185 & 0.0447 & 0.7919
\end{bmatrix}
\]

- **Can also be used to change a sub-array**

  \[
  \text{test} =
  \begin{bmatrix}
0.9218 & 0.1763 & 0.9355 \\
0.7202 & 0.3057 & 0.9669
\end{bmatrix}
\]

  \[
  \text{>> } \text{test}(1:2,1) = \text{zeros}(1,2)
  \]

  \[
  \text{test} =
  \begin{bmatrix}
0 & 0.1763 & 0.9355 \\
0 & 0.3057 & 0.9669
\end{bmatrix}
\]

- **Watch for addressing errors:**

  \[
  \text{test} =
  \begin{bmatrix}
0.4103 & 0.0579 & 0.8132 \\
0.3936 & 0.3529 & 0.0095
\end{bmatrix}
\]

  \[
  \text{>> } \text{test}(2,:) = [1 \ 2]
  \]

  \[
  ??? \text{ In an assignment A(matrix,:)} = \text{B}, \text{ the number}
  \text{ must be the same.}
  \]

  \[
  \text{>> } \text{test}(1,:) = [1 \ 2]
  \]

  \[
  ??? \text{ In an assignment A(:,matrix)} = \text{B}, \text{ the number}
  \text{ of columns in B must be the same.}
  \]

- **Other useful methods that do work:**

  \[
  \text{test} =
  \begin{bmatrix}
0.8462 & 0.4721 & 0.2913 & 0.9628 \\
0.6152 & 0.8361 & 0.3795 & 0.3693 \\
0.2026 & 0.6106 & 0.8310 & 0.4269
\end{bmatrix}
\]

  \[
  \text{>> } \text{test}(4,2) = 100
  \]

  \[
  \text{test} =
  \begin{bmatrix}
0.8462 & 0.4721 & 0.2913 & 90.020 \\
0.6152 & 0.8361 & 0.3795 & 0.3693 \\
0.2026 & 0.6106 & 0.8310 & 0.4269
\end{bmatrix}
\]

  Assigning values with too large an index just grows the array
Scalars work for sub-array replacement – they just scale up to the right size.

Replacing with a null matrix is the same as deleting – but it only works for entire rows or columns.

### Some Array Operators

- **Transpose (single quote symbol `'`)**
  - switches rows and columns

```
>> test'
ans =
  1 2 3 4
  5 6 7 8
  9 10 11 12
```

- **Size** – the number of rows and columns
- **Length** – the larger of these two

```
>> size(test)
ans =
  3 4

>> size(bob)
ans =
  1 4
```

- **Diag** – matrix ↔ vector operator for diagonal elements

```
>> diag([1 2 3])
ans =
  1   0   0
  0   2   0
  0   0   3
```

- **Reshape**
  - resize fixed set of elements

```
>> reshape(test,3,5)
ans =
  1 0 6 3 11
  5 2 10 7 4
  12
```

Note – the set of elements is exactly the same.

Note column-wise approach to re-ordering the values.
**Character Arrays**

- Rows of the array are strings of alphanumeric characters, one array entry per character
- Enter using a single quotation mark (') at each end of the string

```matlab
>> test = 'Tehan'
test = Tehan
>> size(test)
ans =
    1     4
```

**Element by Element Math Operations**

- For arrays of identical sizes, addition is defined term by term:
  - the command \( F = A + B \) means \( F(r,c) = A(r,c) + B(r,c) \)
  - for all row and column pairs \( r,c \) "element-by-element" addition

- For example:

```matlab
A = 1 2 3
   4 5 6
   7 8 9
B = 10 11 12
    13 14 15
    16 17 18
>> F = A + B
F =
    11 13 15
    18 20 22
    25 28 31
```

- Notes:
  - Arrays must be of identical sizes
  - One can be a scalar (it is "sized up")
  - Subtraction is identical

- The other basic math operations work element by element using the `.*` notation (with \( A, B \) the same sizes):
  - multiplication
    \( F = A \times B \rightarrow F(r,c) = A(r,c) \times B(r,c) \)
  - division
    \( F = A / B \rightarrow F(r,c) = A(r,c) / B(r,c) \)
  - exponentiation:
    \( F = A \times B \rightarrow F(r,c) = A(r,c)^B(r,c) \)

```matlab
>> a = [1 2 3; 4 5 6]
a =
    1     2     3
    4     5     6
>> b = [4 5 6; 1 2 3]
b =
    4     5     6
    1     2     3
>> a.*b
ans =
    4    10    18
    4    10    18
```

- For multi-row alphanumeric arrays, each row must have the same number of characters
  - `name = ['Marty'; 'James'; 'Bob']`
  - Use 2 quotation marks in a row to get 1
    - 'Mary's' \( \rightarrow \) Mary's (a 1 by 6 array)
  - Also, there are some built-in arrays
    - `y = date` \( \rightarrow \) y = 05-Jan-2004 (a 1 by 11 array)
One could be scalar: \[ a = [1 \ 2 \ 3] \quad b = 2 \]

\[
\begin{array}{c}
\text{\texttt{a.*b}}
\end{array}
\begin{array}{c}
\text{\texttt{b.*a}}
\end{array}
\begin{array}{c}
2 \ 4 \ 6
\end{array}
\begin{array}{c}
2 \ 4 \ 6
\end{array}
\]

\[
\begin{array}{c}
\text{\texttt{a./b}}
\end{array}
\begin{array}{c}
\text{\texttt{b./a}}
\end{array}
\begin{array}{c}
\begin{array}{c}
0.5000
1.0000
1.5000
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
2.0000
1.0000
0.6667
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\text{\texttt{b.'b}}
\end{array}
\begin{array}{c}
\text{\texttt{b.'a}}
\end{array}
\begin{array}{c}
\begin{array}{c}
1 \ 4 \ 9
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
2 \ 4 \ 6
\end{array}
\end{array}
\]

Built-in functions also work element-by-element:
- log and exp
- trigonometric
- etc.

\[
\begin{array}{c}
\text{\texttt{b.* [4 \ 9 \ 25; 1 \ 2 \ 10]}}
\end{array}
\begin{array}{c}
\text{\texttt{b.*b}}
\end{array}
\begin{array}{c}
\begin{array}{c}
4 \ 9 \ 25
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
1 \ 2 \ 10
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\text{\texttt{log(b)}}
\end{array}
\begin{array}{c}
\text{\texttt{log(b.*b)}}
\end{array}
\begin{array}{c}
\begin{array}{c}
2.0000
3.0000
5.6000
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
1.0000
1.4142
3.1053
\end{array}
\end{array}
\]