Wichita State University
College of Engineering
Biomedical Engineering

Introduction to Arrays

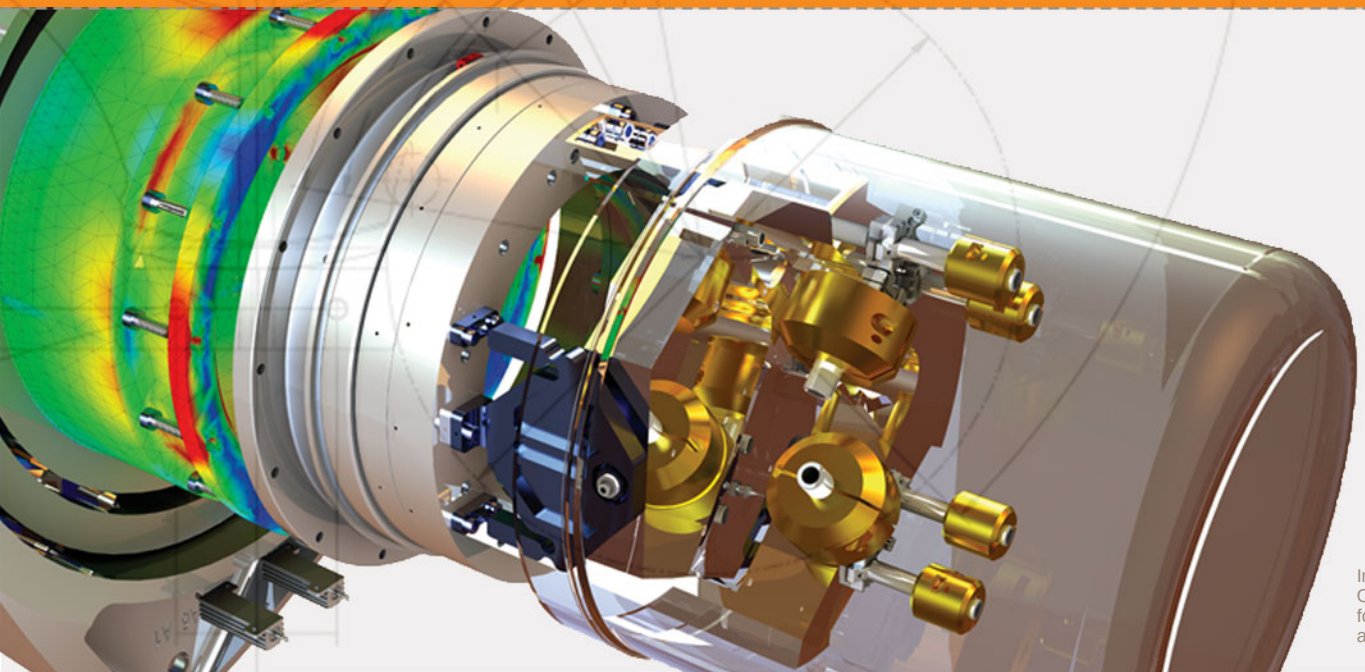


Image courtesy of National Optical Astronomy Observatory, operated by the Association of Universities for Research in Astronomy, under cooperative agreement with the National Science Foundation.

Arrays

- An array is a collection of like elements.
- There are many engineering applications that use arrays.
- MATLAB® is an acronym for **Mat**rix **Lab**oratory. (A matrix is a two-dimensional array)
- MATLAB® stores data in arrays and performs all numerical computations using array operations. Therefore, to use MATLAB® effectively as a computing tool, one must understand arrays and operations with arrays.

1-d Arrays: Vectors

A vector is a one-dimensional array.

Examples:

A row vector with 4 elements $\mathbf{x} = [\ 0 \quad -1.5 \quad 4 \quad 7\ ]$

A column vector with 3 elements $\mathbf{y} = \begin{pmatrix} 5 \\ 2.9 \\ 3 \end{pmatrix}$

2-d Arrays: Matrices

A matrix is a two-dimensional array (like a table).

Examples:

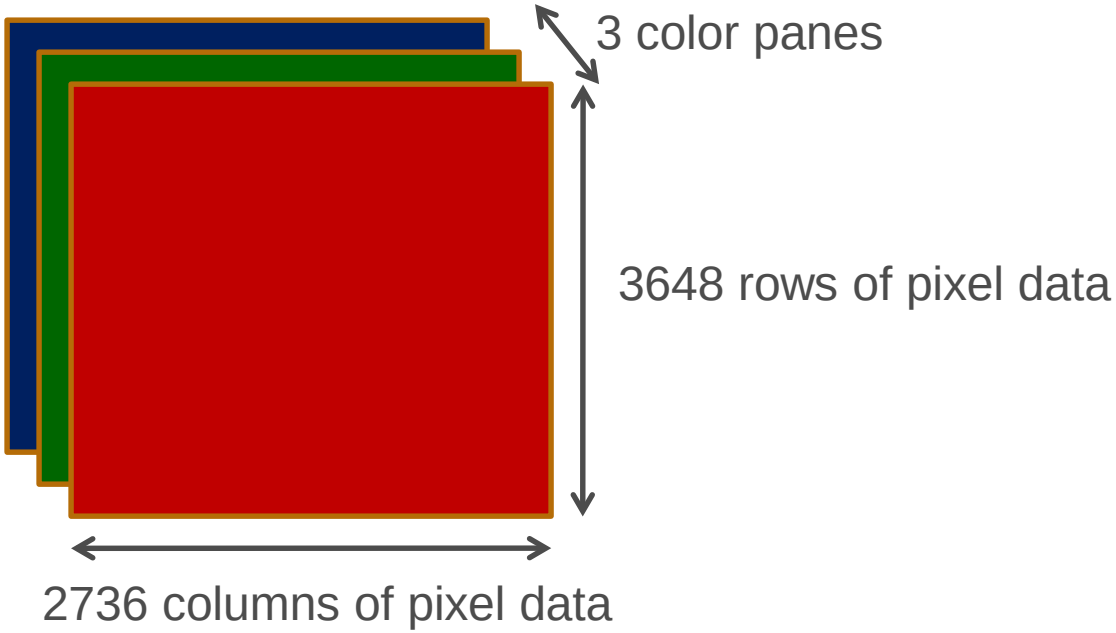
$$\mathbf{A} = \begin{pmatrix} 2.3 & 4.5 & -7.5 \\ 3.2 & -5.1 & 10 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} 76 & 75 & 84 & 82 \\ 92 & 95 & 89 & 88 \\ 67 & 65 & 70 & 75 \end{pmatrix}$$

A has 2 rows and 3 columns (2x3 Matrix)

B has 3 rows and 4 columns (3x4 Matrix)

Multi-Dimensional Arrays

Arrays can have more than two dimensions. For example, a 3648 x 2736 jpg color image imported into MATLAB® would be a 3-dimensional array of size 3648 x 2736 x 3 where the 3rd dimension represents the RGB panes as illustrated below.



CREATING 1-D ARRAYS (VECTORS)

Creating Vectors in MATLAB®

```
>> a = [2.3  7.5  4.3  6]
% Creates a single row of numbers
      a =   2.3000    7.5000    4.3000    6.0000

>> b = [2.3;  7.5;  4.3;  6]
% Creates a single column of numbers
      b =
            2.3000
            7.5000
            4.3000
            6.0000
```

Creating Vectors: Other Options

STAR INC MAXIMUM

```
>> t = -1:0.5:3

t =

   -1   -0.5   0   0.5   1.0   1.5   2.0   2.5   3.0
```

START

INCREMENT = 0.5

MAX.

Note: If you leave out the increment (middle value), MATLAB will set increment = 1

```
>> t = -1:0.6:3

t =

      -1   -0.4   0.2   0.8   1.4   2.0   2.6
```

Creating Vectors: Other Options

START

END

Number of Points

```
>> t = linspace(-1,3,7)

t =

 -1.0000 -0.3333  0.3333  1.0000  1.6667  2.3333  3.0000
```

START

7 Data Points

END

ENTRY BY ENTRY MATH OPERATORS

Example

Suppose we create a vector, t, as follows:

```
>> t = 0:2:10
     t =     0     2     4     6     8    10
```

What if we wanted to square each entry in the vector, t?

```
>> t^2
```

```
Error using  ^
Inputs must be a scalar and a square matrix.
To compute elementwise POWER, use POWER (.^)
instead.
```

Arithmetic Operators

MATLAB® stores data in arrays and performs all numerical computations using array operations.

All of the arithmetic operators: + – * / ^
perform array operations and must follow the rules for arrays.

[0 2 4 6 8 10] * [0 2 4 6 8 10]

(1 x 6) * (1 x 6)

Inner Matrix
Dimensions are
not the same.
Invalid Multiplication

Entry by Entry Operators

MATLAB® provides entry by entry operators that allow the user to perform computations on each entry in an array.

The entry by entry operators are: .* .^ ./

Back to Example

```
>> t = 0:2:10
     t =     0     2     4     6     8    10
```

How do we square each entry in the vector, t?

```
>> t.^2

ans =

     0     4    16    36    64   100
```

Your Turn ...

Try these commands (one at a time) in MATLAB. Explain what each command does.

```
>> x = [5  -3  7  -10]
>> y = [1;  3;  -17]
>> t = 0:0.1:2
>> z = linspace(0,2,11)
>> q = [1  -2  3  9]; q^3
>> q = [1  -2  3  9]; q.^3
>> a = [1 2 3]; b = [4 5 6]; a*b
>> a = [1 2 3]; b = [4 5 6]; a.*b
>> a = [1 2 3]; b = [4 5 6 7]; a.*b
```

Test Your Understanding