Intro to MATLAB

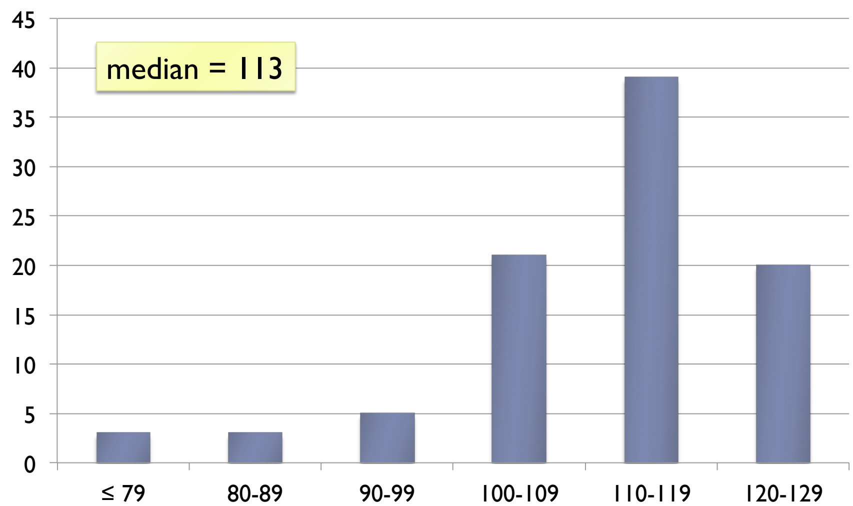
ENGR 151, Lecture 17: 10 Nov 14

Announcements

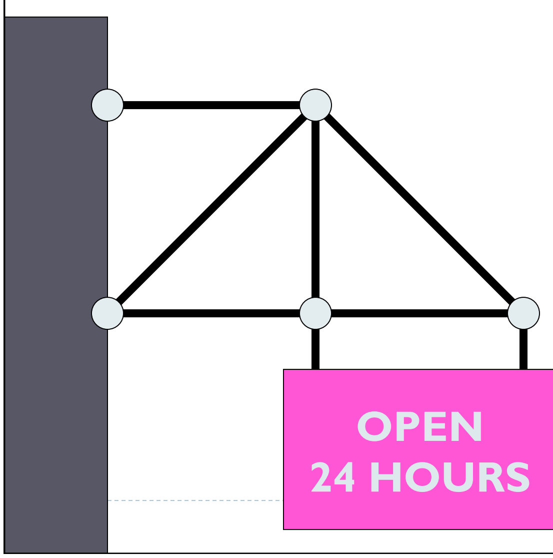
- Project 6 due Wednesday 11PM
- Project 7 out shortly thereafter
- Exam 2 results…

```
score = raw_score + 20;
```

Midterm Exam #2 Raw Scores (out of 130)

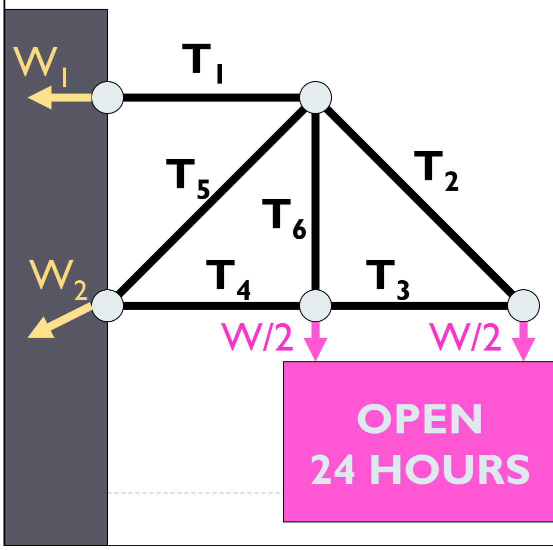


Consider a General Statics Problem



- A sign is suspended away from a building by a network of struts and joints.
- All angles are 45 or 90 degrees.

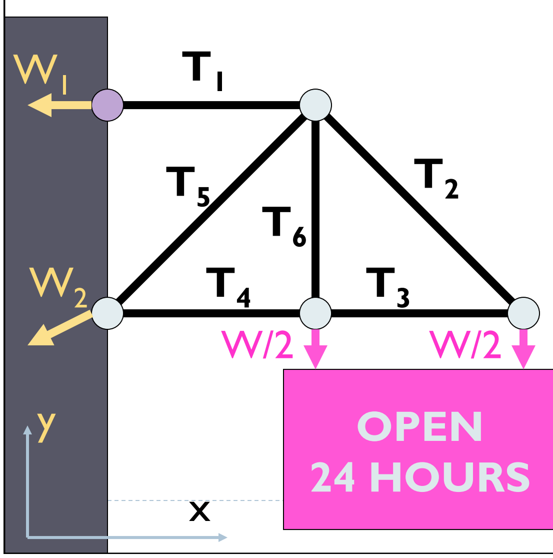
General Statics Problem: Forces



- Tension across each strut
- Sign exerts force on joints to which it is attached
- Wall exerts forces via joints attached to wall

At each joint the forces must exactly balance.
→ Otherwise there would be acceleration

Balance of Forces



$$T_1 + W_{1x} = 0$$
$$W_{1y} = 0$$
$$-T_1 - T_5/\sqrt{2} + T_2/\sqrt{2} = 0$$
$$-T_5/\sqrt{2} - T_6 - T_2/\sqrt{2} = 0$$
$$W_{2x} + T_4 + T_5/\sqrt{2} = 0$$
$$W_{2y} + T_5/\sqrt{2} = 0$$
$$T_3 - T_4 = 0$$
$$-W/2 + T_6 = 0$$
$$-T_3 - T_2/\sqrt{2} = 0$$
$$-W/2 + T_2/\sqrt{2} = 0$$

Convert to a Matrix

Q1: Triangular?
Q2: How to solve?

$$
\begin{pmatrix}
-1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\
0 & -1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{pmatrix}
=
\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ W/2 \end{pmatrix}
$$

Making a Matrix Lower Triangular

- Idea: transform a matrix to lower-triangular form

$$
\begin{pmatrix} 1 & 2 & 0 \\ 3 & 0 & 0 \\ -1 & 8 & 3 \end{pmatrix}
\begin{pmatrix} a \\ b \\ c \end{pmatrix}
=
\begin{pmatrix} 7 \\ 2 \\ 0 \end{pmatrix}
$$

swap rows 1 and 2

$$
\begin{pmatrix} 3 & 0 & 0 \\ 1 & 2 & 0 \\ -1 & 8 & 3 \end{pmatrix}
\begin{pmatrix} a \\ b \\ c \end{pmatrix}
=
\begin{pmatrix} 2 \\ 7 \\ 0 \end{pmatrix}
$$

Apply to Statics Problem

pick swaps to get as close to lower-triangular as possible

$$\begin{pmatrix} -1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & -1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ W/2 \end{bmatrix}$$

Almost Lower-Triangular

can get close, but not quite there

$$\begin{pmatrix} 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix} = \begin{bmatrix} 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Another Trick

- Given two equations:

$$2x + 3y + z = 5$$
$$x + y = 2$$

multiply one by a scalar and add them to get a new equation:

$$2x + 3y + z = 5$$
$$-2(x + y = 2)$$

$$\begin{array}{rcr} 2x + 3y + z & = & 5 \\ -2x - 2y & = & -4 \\ \hline y + z & = & 1 \end{array}$$

Exercise

- Which operation would make progress in transforming the given matrix to lower-triangular form?

$$\begin{pmatrix} 1 & 8 & 0 \\ 3 & 5 & 6 \\ -1 & 2 & 3 \end{pmatrix}$$

A Add row 1 to row 3

B Add row 3 to row 1

C Add (– 2) times row 3 to row 2

D Add (– 4) times row 3 to row 1

E None of the above

Almost Lower-Triangular

Add row 6 to row 3

$$\begin{pmatrix} 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix} = \begin{bmatrix} 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Almost Lower-Triangular (cont.)

Subtract row 5 from row 3

$$\begin{pmatrix} 0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix} = \begin{bmatrix} 0 \\ W/2 \\ W/2 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

-1

Almost Lower-Triangular (cont.)

Swap rows 1 and 3

$$
\begin{pmatrix}
0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -2/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix}
=
\begin{bmatrix} 0 \\ W/2 \\ W/2 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
$$

Almost Lower-Triangular (cont.)

Add twice row 2 to row 1

$$
\begin{pmatrix}
1 & -2/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix}
=
\begin{bmatrix} W/2 \\ W/2 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
$$

~~Almost~~ Lower-Triangular

$$
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1/\sqrt{2} & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 1/\sqrt{2} & 0 & 0 & -1/\sqrt{2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1/\sqrt{2} & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1/\sqrt{2} & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ W_{1x} \\ W_{1y} \\ W_{2x} \\ W_{2y} \end{bmatrix}
=
\begin{bmatrix} 3W/2 \\ W/2 \\ 0 \\ 0 \\ 0 \\ W/2 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
$$

Gaussian Elimination

- A systematic version of the process we just went through.
- Pseudocode to transform any square matrix to lower-triangular form:
  - For each row *i*, starting from last and moving up:
    - If element on the diagonal `A[i][i]` is zero, swap with row $j < i$ such that `A[j][i]` is nonzero.
    - Divide all elements in row *i* by `A[i][i]`.
    - Subtract `A[j][i]*row[i]` from each row $j < i$.
    - Now every row $j < i$ has a 0 value in column *i* .

MATLAB and Matrices

- Whereas we can implement matrix operations in C++, MATLAB provides functionality to solve directly just these kinds of problems.
- Given a matrix A and a vector b, find the x such that

  Ax = b

  - In C++, write program for Gaussian elimination, solving equations in lower-triangular form
  - In MATLAB, evaluate expression:

```
x = A\b
```

MATLAB Poll

- How much MATLAB experience do you have?

A. None
B. Have tried it out a little
C. Have written simple programs
D. Have written a substantial program
E. Expert

C++ vs. MATLAB

- Which is *not* a way that MATLAB differs from C++?

A. MATLAB is interpreted rather than compiled
B. MATLAB does not require variable type declarations
C. MATLAB does not have integer data types
D. MATLAB is proprietary whereas C++ is an open standard
E. None of the above

C++ vs. MATLAB

Compiled	Interpreted
Fast	Slower
Strongly typed	Weakly typed
Predefined libraries	More math libraries
Variety of data structures	Matrix based
Graphing external	Integrated graphing
Open standard	Proprietary (MathWorks)

Array as Fundamental Data Type

- To MATLAB all data is some kind of array.
- Scalars: arrays with one element (zero dimensions)
  - Elements typically a floating-point or character type
- Vectors: sequence of scalars (one dimension: row or column)
- Matrices: two or more dimensions (rows, columns, …)

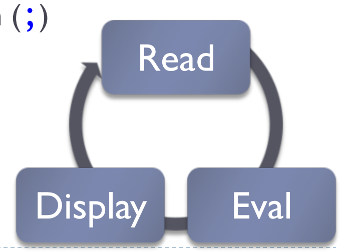
MATLAB Workspace

- "main" scope for MATLAB functions and variable names
- Names introduced (e.g., initial assignment) and referenced from command window or script file
- Lifetime of names: from introduction until `clear` command
- Use `who` command to display active variables in workspace

Read/Eval/Display Loop

- The MATLAB interpreter processes input (from keyboard or scripts) as follows:

1. read the next statement
   - a command, assignment, or expression
2. execute the statement, evaluating expressions as needed
3. if an assignment or expression stmt, display the result
   - unless statement ends in semicolon (;)



To Create a New Variable

- Variable names (identifiers) must start with a letter which can be followed by letters, numbers and the _ character.
- Simply assign some value to the variable and MATLAB creates it
  - do **not** need to declare a data type

```
var = 45.8;
x = 2.0; y = 3.0;
complex = 0.5 - 0.5 * i;
```

- Brackets construct arrays

```
list = [ 1.0  2.0  3.0 ];
```

MATLAB Expressions

- As in C++, build compound expressions from:
  - atomic expressions (identifiers, literals)
  - operators
  - functions
- Examples:
  - `23.4 + 8`
  - `a = 4; a / 3`
  - `pi / 2`
  - `sin(pi/2)`
  - `b = [ 10 20 30 ]; b(2)*9`

Evaluate/Display of Arithmetic Expressions

- Evaluating `23.4 + 8` in the interpreter produces the result

```
ans = 31.4000
```

Why this many digits?

A. The default display of fractional numbers is fixed-point with 4 decimal places
B. The default display of numbers is 6 significant figures
C. The default precision for reals is 4 decimal digits
D. The maximum precision for reals is 4 decimal digits
E. None of the above

Scalar Operations

- all binary, infix:

Addition	a + b
Subtraction	a – b
Multiplication	a * b
Division	a / b
Left Division	a \ b
Exponentiation	a ^ b

Scalar Functions

- MATLAB offers a large number of predefined mathematical functions
  - already seen: `sin`, …
- More examples
  - `mod(11, 3)` → 2
  - `floor(11.9)` → 11
  - `sqrt(39)` → 6.2450
  - `log(2.7)` → 0.9933
  - `exp(log(2.7))` → 2.7000
  - `factorial(6)` → 720

Creating Matrices

- Matrices are specified in row order, separated by commas or spaces.
- Rows can be separated by semicolons (;) or new lines.

```
a = [1.0, 3.0, 5.0; 2.0, 4.0, 6.0]
b = [1.0  3.0  5.0
     2.0  4.0  6.0 ]
```

Transpose

- A single quote denotes the (postfix) transpose operator:

```
v = [0 1 2 3]
w = v'
```

- Could equivalently define w using:

```
w = [  0; 1; 2; 3]
```

or

```
w = [0
     1
     2
     3 ]
```

or

```
w = [0 1 2 3]'
```

Matrices

- Any number of rows and columns
- Must have same number of elements in every row:

```
b = [ 10 15 20; 6 9 ]    error!
```

Accessing Array Elements

- Parentheses denote index operator ( )
  - unlike C++, MATLAB indices start at 1
  - Example:

```
b = [1.0  3.0  5.0
     2.0  4.0  6.0 ]
b(1,3) ➔ 5.0
b(2,2) = 12 ➔
             b = [1.0  3.0  5.0
                  2.0  12.0  6.0 ]
```

1-d Matrix Indexing

- Define a matrix

```
A = [ 1 2 3 4 5; 6 7 8 9 10 ]
```

- Reference as column-major vector using a single index:

```
A(5)  → 3
A(6)  → 8
```

Assigning to Non-Existent Indices

- MATLAB automatically extends the array
- For example:

```
c = [ 3 ];      → [ 3 ]
c(2) = 7;       → [ 3 7 ]
c(4) = 2;       → [ 3 7 0 2 ]
c(2,3) = 5;     → [ 3 7 0 2
                    0 0 5 0 ]
```