Built-in MATLAB Functions
Objectives

After studying this chapter you should be able to:

• Use a variety of common mathematical functions
• Understand and use trigonometric functions in MATLAB
• Compute and use statistical and data analysis functions
• Generate uniform and Gaussian random-number matrices
• Understand the computational limits of MATLAB
• Recognize and be able to use the special values and functions built into MATLAB
3.1 Using Built-in Functions

MATLAB uses function names consistent with most major programming languages

For example

• sqrt
• sin
• cos
• log
Function Input can be either scalars or matrices

```
>> x=9
x =
    9
>> sqrt(x)
ans =
    3
>>
```
Function Input can be either scalars or matrices

```
>> x=9
x =
    9
>> sqrt(x)
ans =
    3
>> x=[4, 9, 16]
x =
    4    9    16
>> sqrt(x)
ans =
    2    3    4
```
Using Predefined Functions

- Functions consist of
  - Name
  - Input argument(s)
  - Output

In MATLAB

```matlab
>>sqrt(x)
sqrt(4)
ans = 2
```

result
Some functions require multiple inputs

- Remainder function returns the remainder in a division problem
- For example the remainder of 10/3, is 1
Some functions return multiple results

- size function determines the number of rows and columns

```matlab
>> d = [1, 2, 3; 4, 5, 6];
>> f = size(d)
f =
    2  3
```
You can assign names to the output

```
>> d=[1,2,3; 4,5,6];
>> [rows,cols]=size(d)
rows = 2
cols = 3
```

The variable names are arbitrary – choose something that makes sense in the context of your problem.
Nesting Functions

```
>> x=2
x =
    2
>> g=sqrt(sin(x))
g =
    0.9536
>>
```
3.3 Elementary Math Functions

3.3.1 Common Computations

As in most computer languages, log(x) is the syntax for the natural log – there is no ln function defined in MATLAB.

- log(x) natural log
- log10(x) log base 10
3.3.2 Rounding Functions

- \text{round}(x)
- \text{fix}(x)
- \text{floor}(x)
- \text{ceil}(x)
```
>> fix(4.8)
ans =
        4
>> floor(4.8)
ans =
        4
>> ceil(4.8)
ans =
        5
```

```
>> fix(4.8)
ans = 
    4
>> floor(4.8)
ans = 
    4
>> ceil(4.8)
ans = 
    5
>> fix(-4.8)
ans = 
    -4
>> floor(-4.8)
ans = 
    -5
>> ceil(-4.8)
ans = 
    -4
```
3.3.3 Discrete Mathematics

- factor(x)
- gcd(x, y)  
  greatest common denominator
- lcm(x)  
  lowest common multiple
- rats(x)  
  represent x as a fraction
- factorial(x)
- nchoosek(n, k)
- primes(x)
- isprime(x)
3.4 Trigonometric Functions

- \( \sin(x) \)  \text{sine}  
- \( \cos(x) \)  \text{cosine}  
- \( \tan(x) \)  \text{tangent}  
- \( \arcsin(x) \)  \text{inverse sine}  
- \( \sinh(x) \)  \text{hyperbolic sine}  
- \( \text{asinh}(x) \)  \text{inverse hyperbolic sine}  
- \( \text{sind}(x) \)  \text{sine with degree input}  
- \( \text{asind}(x) \)  \text{inverse sin with degree output}
3.6 Random Numbers

- **rand(x)**
  - Returns an x by x matrix of random numbers between 0 and 1

- **rand(n,m)**
  - Returns an n by m matrix of random numbers

- These random numbers are evenly distributed
% MATLAB code and output

generate random numbers using rand(3)

ans =

0.8147    0.9134    0.2785
0.9058    0.6324    0.5469
0.1270    0.0975    0.9575

% Additional MATLAB commands

% sort rows of matrix x

sortrows(x,2)
sortrows(x,-2)

% clear command window

clc

% Display length of matrix x

length(x)

% Display number of elements in matrix x

numel(x)

rand(3)
```matlab
>> rand(3)
ans =
    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575

>> rand(1,3)
ans =
    0.9649    0.1576    0.9706
```
To generate random numbers between other bounds...

\[ x = (b - a) \cdot r + a \]

- \( a \) and \( b \) are the upper and lower bounds
- \( r \) is the array of random numbers
3.8 Computational Limits

- MATLAB’s computational range on most computers is:
  - $10^{-308}$
  - $10^{308}$
- When you divide by 0, the computer returns Inf
Check the limits on your computer with these commands

- realmax
- realmin
- intmax
- intmin
```matlab
>> realmax
ans =
    1.7977e+308
>> realmin
ans =
    2.2251e-308
>> intmax
ans =
    2147483647
>> intmin
ans =
   -2147483648
>> 
```
When using very large or very small numbers the result may depend on the order of operation.

```
>> 2.5e200*2e200*1e-100
ans =
     Inf
>> 2.5e200*1e-100*2e200
ans =
     5.0000e+300
>>
```
3.9 Special Values and Miscellaneous Functions

- $\pi$
- $i,j$
- Inf
- NaN
- clock
- date
- eps
- ans

Hint: The function $i$ is the most common of these functions to be unintentionally renamed by MATLAB users.
3.2 Using the Help Feature

- There are functions for almost anything you want to do
- Use the help feature to find out what they are and how to use them
  - From the command window
  - From the help selection on the menu bar
From the Command Window

```
>> help sqrt
SQRT    Square root.
    SQRT(X) is the square root of the elements of X. Complex
results are produced if X is not positive.

See also sqrtm.

Overloaded functions or methods (ones with the same name in other dir

    help sym/sqrt.m

Reference page in Help browser

doc sqrt
```
From the Help Menu
Rate of Change

time, hour

Rate of temperature change, degrees/hour

Mathematics

Arrays and Matrices

Linear Algebra

Matrix analysis, linear equations, eigenvalues, singular values, logarithms, exponentials, factorization

Polynomials

Multiplication, division, evaluation, roots, derivatives, integration, eigenvalue problem, curve fitting, partial fraction expansion

Interpolation and Computational Geometry

Interpolation, Delaunay triangulation and tessellation, convex hulls, Voronoi diagrams, domain generation

Cartesian Coordinate System Conversion

Conversions between Cartesian and polar or spherical coordinates

Nonlinear Numerical Methods

Differential equations, optimization, integration

Specialized Math

Airy, Bessel, Jacobi, Legendre, beta, elliptic, error, exponential integral, gamma functions

Sparse Matrices

Elementary sparse matrices, operations, reordering algorithms, linear algebra, iterative methods, tree operations
**Fixed-Point Toolbox**

**sqrt**

Square root of `fi` object

**Syntax**

\[
\begin{align*}
c &= \text{sqrt}(a) \\
c &= \text{sqrt}(a,T) \\
c &= \text{sqrt}(a,F) \\
c &= \text{sqrt}(a,T,F)
\end{align*}
\]

**Description**

This function computes the square root of a `fi` object using a bisection algorithm.

- `c = \text{sqrt}(a)` returns the square root of `fi` object `a` with the same `fimath` object as `a`. Intermediate quantities are also calculated using the `fimath` object of `a`. The `numeric type` object of `c` is determined automatically for you using an internal rule.

- `c = \text{sqrt}(a,T)` returns the square root of `fi` object `a` with `numeric type` object `T` and the same `fimath` object as `a`. Intermediate quantities are calculated using the `fimath` object of `a`. See Data Type Propagation Rules.

- `c = \text{sqrt}(a,F)` returns the square root of `fi` object `a` with `fimath` object `F`. Intermediate quantities are also calculated using `fimath` object `F`. The `numeric type` object of `c` is determined automatically for you using an internal rule. When `a` is a built-in `double` or `single` data type, this syntax is equivalent to `c = \text{sqrt}(a)` and the `fimath` object `F` is ignored.

- `c = \text{sqrt}(a,T,F)` returns the square root `fi` object `a` with `numeric type` object `T` and `fimath` object `F`. Intermediate quantities are also calculated using `fimath` object `F`. See Data Type Propagation Rules.

`sqr t` does not support complex, negative-valued, or [Slope Bias] inputs.

**Internal Rule**

For syntaxes where the `numeric type` object of the output is not specified as an input to the `sqrt` function, it is automatically calculated according to the following internal rule:

\[
\begin{align*}
\text{sign}_c &= \text{sign}_a \\
\text{WL}_c &= \text{ceil}\left(\frac{\text{WL}_a}{2}\right)
\end{align*}
\]
The windowed help function can also be accessed using the doc command.