

About This Document and Getting Help

This document is intended to be used with Matlab's extensive built in and online help system. Corrections and suggestions to tritt@msoe.edu or imas@msoe.edu are encouraged.

Scalar and Array (element wise) Arithmetic Operators

+ -	Addition and subtraction
.* ./ .\	Multiplication and division
.^	Power (exponentiation)
.'	Transpose
()	Grouping

Matrix (linear algebra) Arithmetic Operators

+ -	Addition and subtraction
* / \	Multiplication and division
^	Power (exponentiation)
'	[Conjugate] Transpose
()	Grouping

Relational Operators

<, <=, >, >=, == and ~=

Logical Operators (Also see *and*, *or*, *not*, *any* and *all* functions)

	Short-circuit OR
&&	Short-circuit AND
	Element wise OR
&	Element wise AND
~	NOT

Other Special Characters (for more information search help index for =)

[]	Used to form vectors & matrices. Use comas or spaces to separate elements. Use semicolons to separate rows. Empty matrices are allowed. For example, x = [1 2 3; 4 5 6].
{ }	Used in cell array assignments. For example, a{2,1} = [1 2; 3 4] or A(2,2) = {'Hello'}
()	Used to enclose vector and matrix subscripts, enclose function arguments and to group terms in arithmetic expressions.
=	Assignment operator is = (not ==, which compares for equality). Syntax is <i>target = expression</i> .
'	Matrix transpose, complex conjugate transpose (.' is the non-conjugate transpose). Delimits character (string) literals.
.	Decimal point. Structure field access.
...	Command line continuation.
,	Used to separate matrix subscripts and function arguments. Used for separating multiple statements on a line.
;	Used inside brackets to end rows. Used after an expression or statement to suppress printing or to separate statements on a line.
:	Used to create vectors using shortcut notation, array subscripting placeholder and for loop iterations.
%	Comment indicator.
!	Operating system command indicator.
@	Creates function handle.

Operator Precedence

The precedence rules for MATLAB operators are shown in this table, ordered from highest precedence level to lowest precedence level.

Parentheses ()
Array transpose (.'), power (.^), matrix & complex conjugate transpose ('), matrix power (^)
Unary plus (+), unary minus (-), logical negation (~)
Multiplication (.*), right division (./), left division (.\), matrix multiplication (*), matrix right division (/), matrix left division (\)
Addition (+), subtraction (-)
Colon operator (:)
Less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), equal to (==), not equal to (~=)
Element-wise AND (&)
Element-wise OR ()
Short-circuit AND (&&)
Short-circuit OR ()

Assignment & Sub-array Expressions

$x = 1.23 + 4.56i$, $x = [1 \ 2 \ 3]$, $x = [1 \ 2; 3 \ 4; 5 \ 6]$, $x = [0:2:10]$, $x = [-10: 0.2: 10]'$, etc. See also the zeros, ones and eye functions. Also, $\text{str}(1,:) = \text{'Some'}$ and $\text{str}(2,:) = \text{'More'}$ (sizes must match).

If $a = [1.1, 2.2, 3.3, 4.4, 5.5]$, then $a(3)$ is 3.3, $a([1 \ 4])$ is the array $[1.1 \ 4.4]$, $a(1:2:5)$ is the array $[1.1 \ 3.3 \ 5.5]$ and $a(3:\text{end})$ is the array $[3.3 \ 4.4 \ 5.5]$.

Control Constructs

```
if expression1
    statement(s)
elseif expression2
    statement(s)
else
    statement(s)
end
```

```
switch expression
    case case1, case2, etc.
        statement(s)
    case caseN, etc.
        statement(s)
    otherwise
        statement(s)
end
```

```
for index = start:increment:end
    statement(s)
end
```

```
while expression
    statement(s)
end
```

```
try
    statement(s)
catch
    statement(s)
end
```

Key Words

The following words have special meanings in Matlab and should never be used as variable names: *break, case, catch, classdef, continue, else, elseif, end, for, function, global, if, otherwise, parfor, persistent, return, spmd, switch, true, try* and *while*.

Predefined Special Values and Built-in Functions

Special values include: *true, false, pi, i, j, Inf, NaN, clock, date, eps* and *ans*.

Selected functions: *sin, cos, tan* (arguments in radians), *asin, acos, atan, atan2, sqrt, double, fix, max, min, mod* (remainder), *log* (natural), *log10* (base 10 logarithm), *str2double, str2num, size, mean, std*, etc.

String and Cell String Functions and Formatting

strcmp – Compares two strings. Returns true (1) if they are the same. Needed because the equality operator (==) only works for strings of equal length and is not recommended.

Others: *strcmpi, strncmp, strncmpi, strcat, findstr, strrep, strtok, isletter, isspace, upper, lower, deblank, int2str, num2str, sprintf* and *sscanf*.

Console Input and Output

input(prompt) and *input(prompt, 's')* – Prompts user for input and returns entered value as a numerical value, variable name or string (2nd form).

disp(x) – Displays *x* without displaying its name.

fprintf(controlString, data, ...) – Displays control string and data formatted based on imbedded codes. Typical codes include *%s* for strings, *%8.2d* for decimal values, etc. Control strings can include escape sequence special character representation (like *\n* for newline).

Formatted File Input and Output

fid = open('filename') – Opens the specified file. Returns a file identifier (stored in *fid* in this case).

a = fscanf(fid, formatSpec) – Reads and returns all data from file specified by *fid* based on specifications in *formatSpec*.

feof(fid) – Returns 1 (true) if the end-of-file indicator for *fid* has been set (the end of the file has been reached).

fprintf(fid, controlString, data, ...) – File version of *fprintf* described in *Console Input and Output* section.

close(fid or 'all') – Close the specified file or files.

Handle Graphics

Storing a handle: *hPlot = plot(x, y)*; Getting a property value: *curColor = get(hPlot, 'Color')*; Setting a property value: *set(hPlot, 'Color', [.5 .5 .5])* and *set(hPlot, 'Color', 'yellow')*.

Color Specification

RGB Value	[0 0 0]	[1 0 0]	[0 1 0]	[0 0 1]	[1 1 0]	[1 0 1]	[0 1 1]	[1 1 1]
Short String	K	r	g	b	y	m	c	w
Long String	Black	red	green	blue	yellow	magenta	cyan	white

Simple GUI Input and Output

`caAns = inputdlg('prompt')` – Displays a modal dialog box with the supplied prompt(s). Returns user inputs in a cell array. Many forms and options possible.

`nChoice = menu('title', 'opt1', ...)` – Displays a modal menu box with the indicated title and options. Returns integer value corresponding to the selected option.

`[x, y] = ginput()` – Enables user to select points on a figure using the mouse.

`msgbox('message')` – Displays a by default non-Modal message box containing the *message*.

Cell Arrays

Content (indirect) indexing: `a{1,1} = [1 2 3; 4 5 6]`, `a{1,2} = 'Hello World'`; Cell indexing: `a(1,1) = {[1 2 3; 4 5 6]}`; `a(1,2) = {'Hello World'}`; If a cell array contains a reference to an array, braces and parentheses are used together. For example, `a{1, 1}(1, 2)` means element (1, 2) of the array referenced by the element (1, 2) of cell array *a*. Cell arrays of strings (*cellstrs*) have largely replaced character matrices and are often used as function arguments.

Structures

Pre-allocation: `part(10) = struct('number', [], 'count', [], 'descript', [])`; Assignment: `part(6).number = 123;`, `part(2).count = int16(4)`; and `part(3).descript = '#10x1 Screw'`; Access: `order = part(1)`; and `fprintf('Description: %s', part(3).descript)`;

Classes

Created using the `classdef` keyword and class definition files or in `@ClassName` folders. See *Matlab > User Guide > Object-Oriented Programming* help document and the *DocPolynom* sample class.