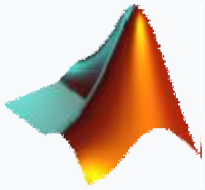
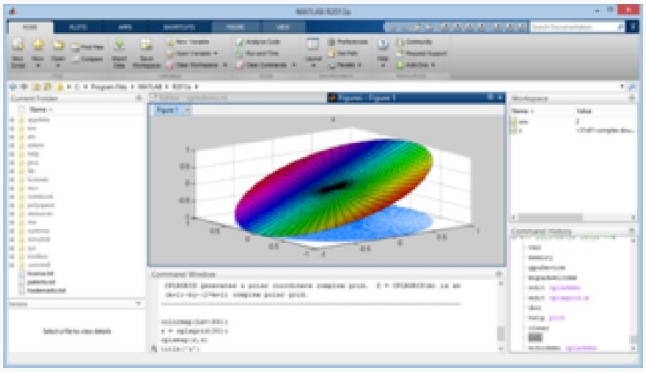
MATLAB

MATLAB



L-shaped membrane logo[1]



MATLAB R2013a running on Windows 8

Developer(s)	MathWorks
Initial release	1984; 35 years ago
Stable release	R2019a / March 20, 2019; 18 days ago
Preview release	None [±]
Written in	C, C++, Java
Operating system	Windows, macOS, and Linux[2]
Platform	IA-32, x86-64
Type	Numerical computing
License	Proprietary commercial software
Website	mathworks.com

MATLAB (*matrix laboratory*) is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting

of functionsand data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C#, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

As of 2018, MATLAB has more than 3 million users worldwide.[7] MATLAB users come from various backgrounds of engineering, science, and economics.

History

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s.[8] He designed it to give his students access to LINPACK and EISPACK without them having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematicscommunity. Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded MathWorks in 1984 to continue its development. These rewritten libraries were known as JACKPAC.[9] In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK.[10]

MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in image processing.[8]

Syntax

The MATLAB application is built around the MATLAB scripting language. Common usage of the MATLAB application involves using the Command Window as an interactive mathematical shell or executing text files containing MATLAB code.[11]

Variables[edit]

Variables are defined using the assignment operator, =. MATLAB is a weakly typed programming language because types are implicitly converted.[12] It is an inferred typed language because variables can be assigned without declaring their type, except if they are to be treated as symbolic objects,[13] and that their type can change. Values can come from constants, from computation involving values of other variables, or from the output of a function. For example:

```
>> x = 17
x =
 17

>> x = 'hat'
x =
hat

>> x = [3*4, pi/2]
x =
   12.0000    1.5708

>> y = 3*sin(x)
```

```
y =
   -1.6097    3.0000
```

Vectors and matrices[edit]

A simple array is defined using the colon syntax: *initial*`:`*increment*`:`*terminator*. For instance:

```
>> array = 1:2:9
array =
 1 3 5 7 9
```

defines a variable named `array` (or assigns a new value to an existing variable with the name `array`) which is an array consisting of the values 1, 3, 5, 7, and 9. That is, the array starts at 1 (the *initial* value), increments with each step from the previous value by 2 (the *increment* value), and stops once it reaches (or to avoid exceeding) 9 (the *terminator* value).

```
>> array = 1:3:9
array =
 1 4 7
```

the *increment* value can actually be left out of this syntax (along with one of the colons), to use a default value of 1.

```
>> ari = 1:5
ari =
 1 2 3 4 5
```

assigns to the variable named `ari` an array with the values 1, 2, 3, 4, and 5, since the default value of 1 is used as the incrementer.

Indexing is one-based,[14] which is the usual convention for matrices in mathematics, although not for some programming languages such as C, C++, and Java.

Matrices can be defined by separating the elements of a row with blank space or comma and using a semicolon to terminate each row. The list of elements should be surrounded by square brackets: []. Parentheses: () are used to access elements and subarrays (they are also used to denote a function argument list).

```
>> A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
A =
 16  3  2 13
  5 10 11  8
  9  6  7 12
  4 15 14  1
```

```
>> A(2,3)
ans =
 11
```

Sets of indices can be specified by expressions such as "2:4", which evaluates to [2, 3, 4]. For example, a submatrix taken from rows 2 through 4 and columns 3 through 4 can be written as:

```
>> A(2:4,3:4)
ans =
 11  8
 7  12
 14  1
```

A square identity matrix of size *n* can be generated using the function *eye*, and matrices of any size with zeros or ones can be generated with the functions *zeros* and *ones*, respectively.

```
>> eye(3,3)
ans =
 1  0  0
 0  1  0
 0  0  1

>> zeros(2,3)
ans =
 0  0  0
 0  0  0

>> ones(2,3)
ans =
 1  1  1
 1  1  1
```

Transposing a vector or a matrix is done either by the function *transpose* or by adding prime after a dot to the matrix. Without the dot MATLAB will perform conjugate transpose.

```
>> A = [1 ; 2],  B = A.', C = transpose(A)
A =
     1
     2
```

```
B =
     1     2
C =
     1     2

>> D = [0 3 ; 1 5], D.'
D =
     0     3
     1     5
ans =
     0     1
     3     5
```

Most MATLAB functions can accept matrices and will apply themselves to each element. For example, `mod(2*J,n)` will multiply every element in "J" by 2, and then reduce each element modulo "n". MATLAB does include standard "for" and "while" loops, but (as in other similar applications such as R), using the vectorized notation often produces code that is faster to execute. This code, excerpted from the function *magic.m*, creates a magic square *M* for odd values of *n* (MATLAB function `meshgrid` is used here to generate square matrices I and J containing 1:n).

```
[J,I] = meshgrid(1:n);
A = mod(I + J - (n + 3) / 2, n);
B = mod(I + 2 * J - 2, n);
M = n * A + B + 1;
```

Structures

MATLAB has structure data types.[15] Since all variables in MATLAB are arrays, a more adequate name is "structure array", where each element of the array has the same field names. In addition, MATLAB supports dynamic field names[16] (field look-ups by name, field manipulations, etc.). Unfortunately, MATLAB JIT does not support MATLAB structures, therefore just a simple bundling of various variables into a structure will come at a cost.[17]

Functions

When creating a MATLAB function, the name of the file should match the name of the first function in the file. Valid function names begin with an alphabetic character, and can contain letters, numbers, or underscores. Functions are often case sensitive.

Function handles

MATLAB supports elements of lambda calculus by introducing function handles,[18] or function references, which are implemented either in .m files or anonymous[19]/nested functions.[20]

Classes and object-oriented programming

MATLAB supports object-oriented programming including classes, inheritance, virtual dispatch, packages, pass-by-value semantics, and pass-by-reference semantics.[21] However, the syntax and calling conventions are significantly different from other languages. MATLAB has value classes and reference classes, depending on whether the class has *handle* as a super-class (for reference classes) or not (for value classes).[22]

Method call behavior is different between value and reference classes. For example, a call to a method

```
object.method();
```

can alter any member of *object* only if *object* is an instance of a reference class.

An example of a simple class is provided below.

```
classdef hello
    methods
        function greet(this)
            disp('Hello!')
        end
    end
end
```

When put into a file named `hello.m`, this can be executed with the following commands:

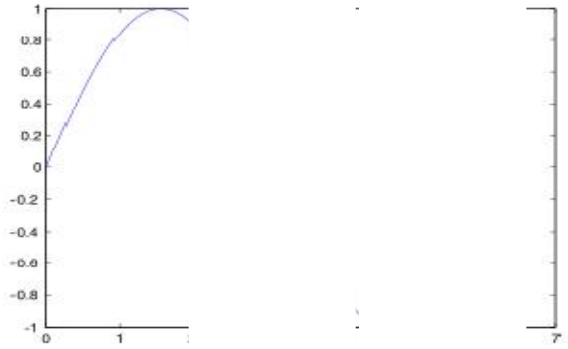
```
>> x = hello;
>> x.greet();
Hello!
```

Graphics and graphical user interface programming

MATLAB supports developing applications with graphical user interface (GUI) features. MATLAB includes GUIDE[23] (GUI development environment) for graphically designing GUIs.[24] It also has tightly integrated graph-plotting features. For example, the function *plot* can be used to produce a graph from two vectors *x* and *y*. The code:

```
x = 0:pi/100:2*pi;
y = sin(x);
plot(x,y)
```

produces the following figure of the sine function:



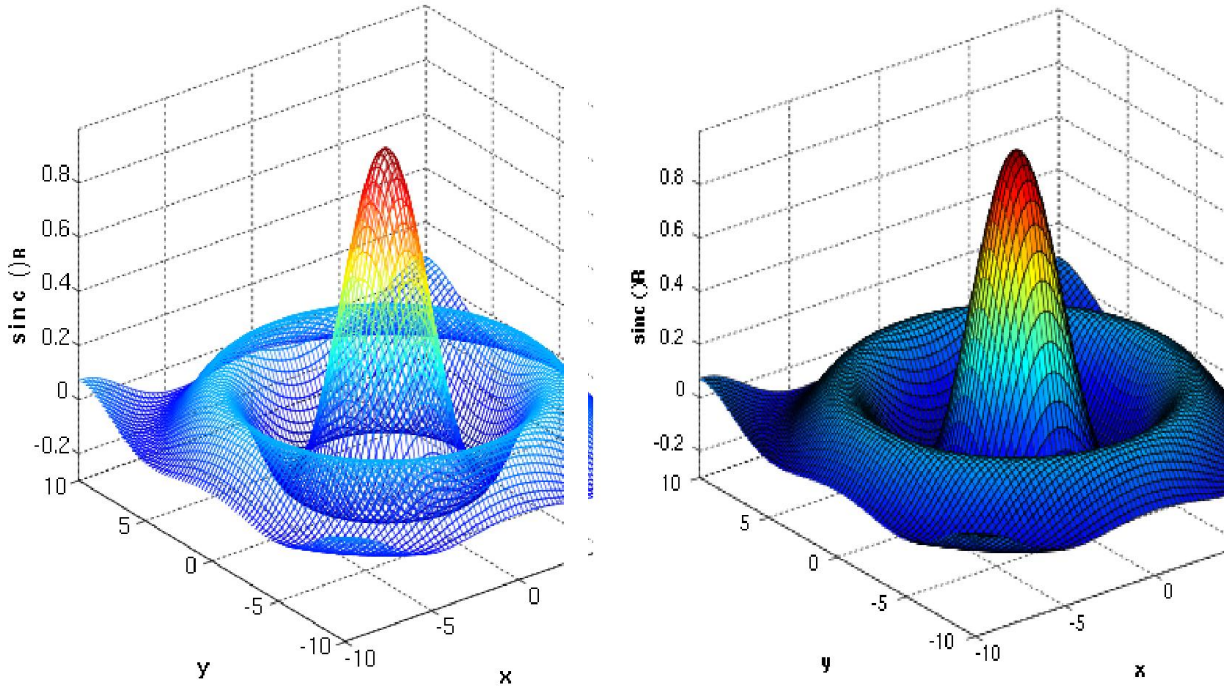
A MATLAB program can produce three-dimensional graphics using the functions *surf*, *plot3* or *mesh*.

```
[X,Y] = meshgrid(-10:0.25:10,-
10:0.25:10);
f = sinc(sqrt((X/pi).^2+(Y/pi).^2));
mesh(X,Y,f);
axis([-10 10 -10 10 -0.3 1])
xlabel('{\bfx}')
ylabel('{\bfy}')
zlabel('{\bfsinc} ({\bfR})')
hidden off
```

This code produces a **wireframe** 3D plot of the two-dimensional unnormalized sinc function:

```
[X,Y] = meshgrid(-10:0.25:10,-
10:0.25:10);
f = sinc(sqrt((X/pi).^2+(Y/pi).^2));
surf(X,Y,f);
axis([-10 10 -10 10 -0.3 1])
xlabel('{\bfx}')
ylabel('{\bfy}')
zlabel('{\bfsinc} ({\bfR})')
```

This code produces a **surface** 3D plot of the two-dimensional unnormalized sinc function:



In MATLAB, graphical user interfaces can be programmed with the GUI design environment (GUIDE) tool.[25]

Interfacing with other languages

MATLAB can call functions and subroutines written in the programming languages C or Fortran.[26] A wrapper function is created allowing MATLAB data types to be passed and returned. MEX files (MATLAB executables) are the dynamically loadable object files created by compiling such functions.[27][28] Since 2014 increasing two-way interfacing with Python was being added.[29][30]

Libraries written in Perl, Java, ActiveX or .NET can be directly called from MATLAB,[31][32] and many MATLAB libraries (for example XML or SQL support) are implemented as wrappers around Java or ActiveX libraries. Calling MATLAB from Java is more complicated, but can be done with a MATLAB toolbox[33] which is sold separately by MathWorks, or using an undocumented mechanism called JMI (Java-to-MATLAB Interface),[34][35] (which should not be confused with the unrelated Java Metadata Interface that is also called JMI). Official MATLAB API for Java was added in 2016.[36]

As alternatives to the MuPAD based Symbolic Math Toolbox available from MathWorks, MATLAB can be connected to Maple or Mathematica.[37][38]

Libraries also exist to import and export MathML.[39]

License

MATLAB is a proprietary product of MathWorks, so users are subject to vendor lock-in.[40][41] Although MATLAB Builder products can deploy MATLAB functions as library files which can be used with .NET[42] or Java[43] application building environment, future development will still be tied to the MATLAB language.

Each toolbox is purchased separately. If an evaluation license is requested, the MathWorks sales department requires detailed information about the project for which MATLAB is to be evaluated. If granted (which it often is), the evaluation license is valid for two to four weeks. A student version of MATLAB is available as is a home-use license for MATLAB, Simulink, and a subset of Mathwork's Toolboxes at substantially reduced prices.

It has been reported that European Union (EU) competition regulators are investigating whether MathWorks refused to sell licenses to a competitor.[44] The regulators dropped the investigation after the complainant withdrew its accusation and no evidence of wrongdoing was found.[45]

Alternatives

See also: list of numerical analysis software and comparison of numerical analysis software

MATLAB has a number of competitors.[46] Commercial competitors include Mathematica, TK Solver, Maple, and IDL. There are also free open source alternatives to MATLAB, in particular GNU Octave, Scilab, FreeMat, and SageMath, which are intended to be mostly compatible with the MATLAB language; the Julia programming language also initially used MATLAB-like syntax. Among other languages that treat arrays as basic entities (array programming languages) are APL, Fortran 90 and higher, S-Lang, as well as the statistical languages R and S. There are also libraries to add similar functionality to existing languages, such as IT++ for C++, Perl Data Language for Perl, ILNumerics for .NET, NumPy/SciPy/matplotlib for Python, SciLua/Torch for Lua, SciRuby for Ruby, and Numeric.js for JavaScript.

GNU Octave is unique from other alternatives because it treats incompatibility with MATLAB as a bug (see MATLAB Compatibility of GNU Octave), therefore, making GNU Octave a superset of the MATLAB language.

Release history[edit]

Version[47]	Release name	Number	Bundled JVM	Year	Release date	Notes
MATLAB 1.0				1984		
MATLAB 2				1986		
MATLAB 3				1987		
MATLAB 3.5				1990		Ran on DOS but needed at least a 386 processor; version 3.5m needed math coprocessor
MATLAB 4				1992		Ran on Windows 3.1x and Macintosh
MATLAB 4.2c				1994		Ran on Windows 3.1x, needed a math coprocessor
MATLAB 5.0	Volume 8			1996	December 1996	Unified releases across all platforms

MATLAB 5.1	Volume 9			1997	May 1997	
MATLAB 5.1.1	R9.1					
MATLAB 5.2	R10			1998	March 1998	Last version working on classic Macs
MATLAB 5.2.1	R10.1					
MATLAB 5.3	R11			1999	January 1999	
MATLAB 5.3.1	R11.1				November 1999	
MATLAB 6.0	R12	12	1.1.8	2000	November 2000	First release with bundled Java virtual machine (JVM)
MATLAB 6.1	R12.1		1.3.0	2001	June 2001	
MATLAB 6.5	R13	13	1.3.1	2002	July 2002	
MATLAB 6.5.1	R13SP1			2003		
MATLAB 6.5.2	R13SP2					Last release for IBM/AIX, Alpha/TRU64, and SGI/IRIX[48]
MATLAB 7	R14	14	1.4.2	2004	June 2004	Introduced anonymous and nested functions[49] Re-introduced for Mac (under Mac OS X)
MATLAB 7.0.1	R14SP1				October	

					2004	
MATLAB 7.0.4	R14SP2		1.5.0	2005	March 7, 2005	Support for memory-mapped files[50]
MATLAB 7.1	R14SP3		1.5.0		September 1, 2005	First 64-bit version available for Windows XP 64-bit
MATLAB 7.2	R2006a	15	1.5.0	2006	March 1, 2006	
MATLAB 7.3	R2006b	16	1.5.0		September 1, 2006	HDF5-based MAT-file support
MATLAB 7.4	R2007a	17	1.5.0_07	2007	March 1, 2007	New `bsxfun` function to apply element-by-element binary operation with singleton expansion enabled[51]
MATLAB 7.5	R2007b	18	1.6.0		September 1, 2007	Last release for Windows 2000 and PowerPC Mac; License Server support for Windows Vista;[52] new internal format for P-code
MATLAB 7.6	R2008a	19	1.6.0	2008	March 1, 2008	Major enhancements to object-oriented programming abilities with a new class definition syntax,[53] and ability to manage namespaces with packages[54]
MATLAB 7.7	R2008b	20	1.6.0_04		October 9, 2008	New Map data structure:[55] upgrades to random number generators[56]
MATLAB 7.8	R2009a	21	1.6.0_04	2009	March 6, 2009	First release for Microsoft 32-bit & 64-bit Windows 7, new external interface to .NET Framework[57]
MATLAB 7.9	R2009b	22	1.6.0_12		September 4, 2009	First release for Intel 64-bit Mac, and last for Solaris SPARC; new use for the tilde operator (~) to ignore arguments in function calls[58][59]
MATLAB 7.9.1	R2009bSP1		1.6.0_12	2010	April 1, 2010	bug fixes.
MATLAB 7.10	R2010a	23	1.6.0_12		March 5, 2010	Last release for Intel 32-bit Mac
MATLAB 7.11	R2010b	24	1.6.0_17		September 3, 2010	Add support for enumerations[60]
MATLAB 7.11.1	R2010bSP1		1.6.0_17	2011	March 17, 2011	bug fixes and updates
MATLAB 7.11.2	R2010bSP2		1.6.0_17		April 5, 2012[61]	bug fixes
MATLAB 7.12	R2011a	25	1.6.0_17		April 8, 2011	New `rng` function to control random number generation[62][63][64]
MATLAB 7.13	R2011b	26	1.6.0_17		September 1, 2011	Access-change parts of variables directly in MAT-files, without loading into memory;[65]increased maximum local workers with Parallel Computing Toolbox from 8 to 12[66]
MATLAB 7.14	R2012a	27	1.6.0_17	2012	March 1, 2012	Last version with 32-bit Linux support.[67]
MATLAB 8	R2012b	28	1.6.0_17		September 11, 2012	First release with Toolstrip interface;[68] MATLAB Apps.[69] redesigned documentation system

MATLAB 8.1	R2013a	29	1.6.0_17	2013	March 7, 2013	New unit testing framework[70]
MATLAB 8.2	R2013b	30	1.7.0_11		September 6, 2013[71]	Built in Java Runtime Environment (JRE) updated to version 7;[72] New table data type[73]
MATLAB 8.3	R2014a	31	1.7.0_11	2014	March 7, 2014[74]	Simplified compiler setup for building MEX-files; USB Webcams support in core MATLAB; number of local workers no longer limited to 12 with Parallel Computing Toolbox
MATLAB 8.4	R2014b	32	1.7.0_11		October 3, 2014	New class-based graphics engine (a.k.a. HG2);[75] tabbing function in GUI;[76] improved user toolbox packaging and help files;[77] new objects for time-date manipulations;[78] Git-Subversion integration in IDE;[79] big data abilities with MapReduce (scalable to Hadoop);[80]new `py` package for using Python from inside MATLAB,[81] new engine interface to call MATLAB from Python;[82] several new and improved functions: `webread` (RESTful web services with JSON/XML support), `tcpclient` (socket-based connections), `histcounts`, `histogram`, `animatedline`, and others
MATLAB 8.5	R2015a	33	1.7.0_60	2015	March 5, 2015	Last release supporting Windows XP and Windows Vista
MATLAB 8.5	R2015aSP1		1.7.0_60		October 14, 2015	
MATLAB 8.6	R2015b	34	1.7.0_60		September 3, 2015	New MATLAB execution engine (a.k.a. LXE);[83] `graph` and `digraph` classes to work with graphs and networks;[84] MinGW-w64 as supported compiler on Windows;[85] Last version with 32-bit support
MATLAB 9.0	R2016a	35	1.7.0_60	2016	March 3, 2016	Live Scripts: interactive documents that combine text, code, and output (in the style of Literate programming);[86] App Designer: a new development environment for building apps (with new kind of UI figures, axes, and components);[87] pause execution of running programs using a Pause Button
MATLAB 9.1	R2016b	36	1.7.0_60		September 15, 2016	define local functions in scripts;[88] automatic expansion of dimensions (previously provided via explicit call to `bsxfun`); `tall` arrays for Big data;[89] new `string` type;[90] new functions to encode/decode JSON;[91] official MATLAB Engine API for Java[36]
MATLAB 9.2	R2017a	37	1.7.0_60	2017	March 9, 2017	MATLAB Online: cloud-based MATLAB desktop accessed in a web browser;[92] double-quoted strings; new `memoize` function for Memoization; expanded object properties validation;[93]mocking framework for unit testing;[94] MEX targets 64-bit by default; new `heatmap` function for creating heatmap charts[95]
MATLAB 9.3	R2017b	38	1.8.0_121		September 21, 2017	
MATLAB 9.4	R2018a	39	1.8.0_14	2018	March	

			4		15, 2018[96]	
MATLAB 9.5	R2018b	40	1.8.0_152		September 12, 2018	
MATLAB 9.6	R2019a	41	1.8.0_181	2019	March 20, 2019	

The number (or release number) is the version reported by Concurrent License Manager program FLEXlm.

For a complete list of changes of both MATLAB and official toolboxes, consult the MATLAB release notes.[97]