

Table A.2: **Array operators**

Character	Description
.*	Array multiplication
./	Array (right) division
.^	Array power
.\	Array (left) division
.'	Array (nonconjugated) transpose

Table A.3: **Relational and logical operators**

Character	Description
<	Less than
≤	Less than or equal to
>	Greater than
≥	Greater than or equal to
==	Equal to
~=	Not equal to
&	Logical or element-wise AND
	Logical or element-wise OR
&&	Short-circuit AND
	Short-circuit OR

Table A.4: **Managing workspace and file commands**

Command	Description
<code>cd</code>	Change current directory
<code>clc</code>	Clear the Command Window
<code>clear (all)</code>	Removes all variables from the workspace
<code>clear x</code>	Remove x from the workspace
<code>copyfile</code>	Copy file or directory
<code>delete</code>	Delete files
<code>dir</code>	Display directory listing
<code>exist</code>	Check if variables or functions are defined
<code>help</code>	Display help for MATLAB functions
<code>lookfor</code>	Search for specified word in all help entries
<code>mkdir</code>	Make new directory
<code>movefile</code>	Move file or directory
<code>pwd</code>	Identify current directory
<code>rmdir</code>	Remove directory
<code>type</code>	Display contents of file
<code>what</code>	List MATLAB files in current directory
<code>which</code>	Locate functions and files
<code>who</code>	Display variables currently in the workspace
<code>whos</code>	Display information on variables in the workspace

Table A.5: **Predefined variables and math constants**

Variable	Description
<code>ans</code>	Value of last variable (answer)
<code>eps</code>	Floating-point relative accuracy
<code>i</code>	Imaginary unit of a complex number
<code>Inf</code>	Infinity (∞)
<code>eps</code>	Floating-point relative accuracy
<code>j</code>	Imaginary unit of a complex number
<code>NaN</code>	Not a number
<code>pi</code>	The number π (3.14159...)

Table A.6: **Elementary matrices and arrays**

Command	Description
<code>eye</code>	Identity matrix
<code>linspace</code>	Generate linearly space vectors
<code>ones</code>	Create array of all ones
<code>rand</code>	Uniformly distributed random numbers and arrays
<code>zeros</code>	Create array of all zeros

Table A.7: **Arrays and Matrices: Basic information**

Command	Description
<code>disp</code>	Display text or array
<code>isempty</code>	Determine if input is empty matrix
<code>isequal</code>	Test arrays for equality
<code>length</code>	Length of vector
<code>ndims</code>	Number of dimensions
<code>numel</code>	Number of elements
<code>size</code>	Size of matrix

Table A.8: **Arrays and Matrices: operations and manipulation**

Command	Description
<code>cross</code>	Vector cross product
<code>diag</code>	Diagonal matrices and diagonals of matrix
<code>dot</code>	Vector dot product
<code>end</code>	Indicate last index of array
<code>find</code>	Find indices of nonzero elements
<code>kron</code>	Kronecker tensor product
<code>max</code>	Maximum value of array
<code>min</code>	Minimum value of array
<code>prod</code>	Product of array elements
<code>reshape</code>	Reshape array
<code>sort</code>	Sort array elements
<code>sum</code>	Sum of array elements
<code>size</code>	Size of matrix

Table A.9: **Arrays and Matrices: matrix analysis and linear equations**

Command	Description
<code>cond</code>	Condition number with respect to inversion
<code>det</code>	Determinant
<code>inv</code>	Matrix inverse
<code>linsolve</code>	Solve linear system of equations
<code>lu</code>	LU factorization
<code>norm</code>	Matrix or vector norm
<code>null</code>	Null space
<code>orth</code>	Orthogonalization
<code>rank</code>	Matrix rank
<code>rref</code>	Reduced row echelon form
<code>trace</code>	Sum of diagonal elements

Appendix B

Release notes for Release 14 with Service Pack 2

B.1 Summary of changes

MATLAB 7 Release 14 with Service Pack 2 (R14SP2) includes several new features. The major focus of R14SP2 is on *improving* the quality of the product. This document doesn't attempt to provide a complete specification of every single feature, but instead provides a brief introduction to each of them. For full details, you should refer to the MATLAB documentation (Release Notes).

The following key points may be relevant:

1. **Spaces before numbers** - For example: `A* .5`, you will typically get a mystifying message saying that *A* was previously used as a variable. There are two workarounds:

(a) Remove all the spaces:

`A*.5`

(b) Or, put a zero in front of the dot:

`A * 0.5`

2. **RHS empty matrix** - The right-hand side must literally be the empty matrix `[]`. It cannot be a variable that has the value `[]`, as shown here:

```
rhs = [];  
A(:,2) = rhs  
??? Subscripted assignment dimension mismatch
```

3. **New format option** - We can display MATLAB output using two *new* formats: `short eng` and `long eng`.

- `short eng` – Displays output in *engineering* format that has at least 5 digits and a power that is a multiple of three.

```
>> format short eng
>> pi
ans =
    3.1416e+000
```

- `long eng` – Displays output in *engineering* format that has 16 significant digits and a power that is a multiple of three.

```
>> format long eng
>> pi
ans =
    3.14159265358979e+000
```

4. **Help** - To get help for a *subfunction*, use

```
>> help function_name>subfunction_name
```

In previous versions, the syntax was

```
>> help function_name/subfunction_name
```

This change was introduced in R14 (MATLAB 7.0) but was not documented. Use the MathWorks Web site search features to look for the latest information.

5. **Publishing** - Publishing to L^AT_EX now respects the image file type you specify in preferences rather than always using EPSC2-files.

- The Publish image options in Editor/Debugger preferences for Publishing Images have changed slightly. The changes prevent you from choosing invalid formats.
- The files created when publishing using cells now have more natural extensions. For example, JPEG-files now have a .jpg instead of a .jpeg extension, and EPSC2-files now have an .eps instead of an .epsc2 extension.
- Notebook will no longer support Microsoft Word 97 starting in the next release of MATLAB.

6. **Debugging** - Go directly to a subfunction or using the enhanced **Go To** dialog box. Click the **Name** column header to arrange the list of function alphabetically, or click the **Line** column header to arrange the list by the position of the functions in the file.

B.2 Other changes

1. There is a new command `mlint`, which will scan an M-file and show inefficiencies in the code. For example, it will tell you if you've defined a variable you've never used, if you've failed to pre-allocate an array, etc. These are common mistakes in EA1 which produce runnable but inefficient code.
2. You can comment-out a block of code without putting `%` at the beginning of each line. The format is

```
%{  
  
    Stuff you want MATLAB to ignore...  
  
%}
```

The delimiters `%{` and `%}` must appear on lines by themselves, and it may not work with the comments used in functions to interact with the help system (like the H1 line).

3. There is a new function `linsolve` which will solve $Ax = b$ but with the user's choice of algorithm. This is in addition to left division $x = A \backslash b$ which uses a default algorithm.
4. The `eps` constant now takes an optional argument. `eps(x)` is the same as the old `eps*abs(x)`.
5. You can break an M-file up into named cells (blocks of code), each of which you can run separately. This may be useful for testing/debugging code.
6. Functions now optionally end with the `end` keyword. This keyword is mandatory when working with nested functions.

B.3 Further details

1. You can *dock* and *un-dock* windows from the main window by clicking on an icon. Thus you can choose to have all Figures, M-files being edited, help browser, command window, etc. All appear as panes in a single window.
2. Error messages in the command window resulting from running an M-file now include a clickable link to the offending line in the editor window containing the M-file.
3. You can customize figure interactively (labels, line styles, etc.) and then automatically generate the code which reproduces the customized figure.

4. `feval` is no longer needed when working with function handles, but still works for backward compatibility. For example, `x=@sin; x(pi)` will produce `sin(pi)` just like `feval(x,pi)` does, but faster.
5. You can use function handles to create anonymous functions.
6. There is support for nested functions, namely, functions defined within the body of another function. This is in addition to sub-functions already available in version 6.5.
7. There is more support in arithmetic operations for numeric data types other than double, e.g. `single`, `int8`, `int16`, `uint8`, `uint32`, etc.

Finally, please visit our webpage for other details:

<http://computing.mccormick.northwestern.edu/matlab/>

Appendix C

Main characteristics of MATLAB

C.1 History

- Developed primarily by Cleve Moler in the 1970's
- Derived from FORTRAN subroutines LINPACK and EISPACK, linear and eigenvalue systems.
- Developed primarily as an interactive system to access LINPACK and EISPACK.
- Gained its popularity through word of mouth, because it was not officially distributed.
- Rewritten in C in the 1980's with more functionality, which include plotting routines.
- The MathWorks Inc. was created (1984) to market and continue development of MATLAB.

According to Cleve Moler, three other men played important roles in the origins of MATLAB: J. H. Wilkinson, George Forsythe, and John Todd. It is also interesting to mention the authors of LINPACK: Jack Dongara, Pete Steward, Jim Bunch, and Cleve Moler. Since then another package emerged: LAPACK. LAPACK stands for Linear Algebra Package. It has been designed to supersede LINPACK and EISPACK.

C.2 Strengths

- MATLAB may behave as a *calculator* or as a *programming language*
- MATLAB combine nicely calculation and graphic plotting.
- MATLAB is relatively easy to learn

- MATLAB is interpreted (not compiled), errors are easy to fix
- MATLAB is optimized to be relatively fast when performing matrix operations
- MATLAB does have some object-oriented elements

C.3 Weaknesses

- MATLAB is not a *general* purpose programming language such as C, C++, or FORTRAN
- MATLAB is designed for scientific computing, and is not well suitable for other applications
- MATLAB is an interpreted language, slower than a compiled language such as C++
- MATLAB commands are specific for MATLAB usage. Most of them do not have a direct equivalent with other programming language commands

C.4 Competition

- One of MATLAB's competitors is **Mathematica**, the *symbolic* computation program.
- MATLAB is more convenient for *numerical analysis* and *linear algebra*. It is frequently used in *engineering community*.
- Mathematica has superior symbolic manipulation, making it popular among *physicists*.
- There are other competitors:
 - **Scilab**
 - **GNU Octave**
 - **Rlab**

Bibliography

- [1] The MathWorks Inc. *MATLAB 7.0 (R14SP2)*. The MathWorks Inc., 2005.
- [2] S. J. Chapman. *MATLAB Programming for Engineers*. Thomson, 2004.
- [3] C. B. Moler. *Numerical Computing with MATLAB*. Siam, 2004.
- [4] C. F. Van Loan. *Introduction to Scientific Computing*. Prentice Hall, 1997.
- [5] D. J. Higham and N. J. Higham. *MATLAB Guide*. Siam, second edition edition, 2005.
- [6] K. R. Coombes, B. R. Hunt, R. L. Lipsman, J. E. Osborn, and G. J. Stuck. *Differential Equations with MATLAB*. John Wiley and Sons, 2000.
- [7] A. Gilat. *MATLAB: An introduction with Applications*. John Wiley and Sons, 2004.
- [8] J. Cooper. *A MATLAB Companion for Multivariable Calculus*. Academic Press, 2001.
- [9] J. C. Polking and D. Arnold. *ODE using MATLAB*. Prentice Hall, 2004.
- [10] D. Kahaner, C. Moler, and S. Nash. *Numerical Methods and Software*. Prentice-Hall, 1989.
- [11] J. W. Demmel. *Applied Numerical Linear Algebra*. Siam, 1997.
- [12] D. Houcque. Applications of MATLAB: Ordinary Differential Equations. *Internal communication, Northwestern University*, pages 1–12, 2005.