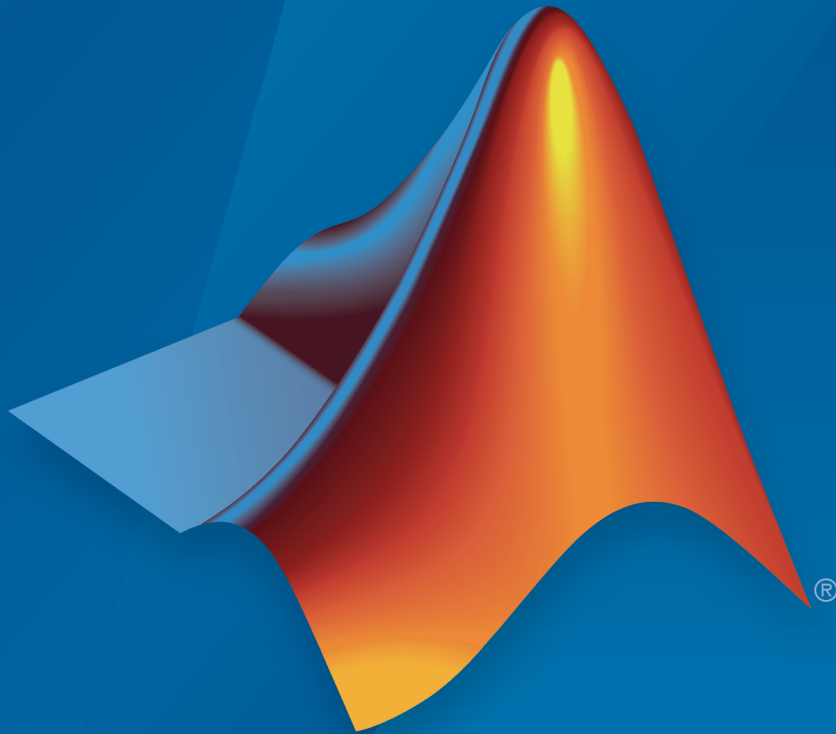


MATLAB®

Desktop Tools and Development Environment



MATLAB®

R2016b



How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

MATLAB[®] Desktop Tools and Development Environment

© COPYRIGHT 1984–2016 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

June 2004	First printing	New for MATLAB 7.0 (Release 14). Formerly part of Using MATLAB.
October 2004	Online only	Revised for Version 7.0.1 (Release 14SP1)
March 2005	Online only	Revised for Version 7.0.4 (Release 14SP2)
March 2005	Second printing	Revised for Version 7.0.4 (Release 14SP2)
June 2005	Third printing	Minor revision for Version 7.0.4 (Release 14SP2)
September 2005	Online only	Revised for Version 7.1 (Release 14SP3)
March 2006	Online only	Revised for Version 7.2 (Release 2006a)
September 2006	Online only	Revised for Version 7.3 (Release 2006b)
March 2007	Online only	Revised for Version 7.4 (Release 2007a)
September 2007	Online only	Revised for Version 7.5 (Release 2007b)
March 2008	Online only	Revised for Version 7.6 (Release 2008a)
October 2008	Online only	Revised for Version 7.7 (Release 2008b)
March 2009	Online only	Revised for Version 7.8 (Release 2009a)
September 2009	Online only	Revised for Version 7.9 (Release 2009b)
March 2010	Online only	Revised for Version 7.10 (Release 2010a)
September 2010	Online only	Revised for Version 7.11 (Release 2010b)
April 2011	Online only	Revised for Version 7.12 (Release 2011a)
September 2011	Online only	Revised for Version 7.13 (Release 2011b)
March 2012	Online only	Revised for Version 7.14 (Release 2012a)
September 2012	Online only	Revised for Version 8.0 (Release 2012b)
March 2013	Online only	Revised for Version 8.1 (Release 2013a)
September 2013	Online only	Revised for Version 8.2 (Release 2013b)
March 2014	Online only	Revised for Version 8.3 (Release 2014a)
October 2014	Online only	Revised for Version 8.4 (Release 2014b)
March 2015	Online only	Revised for Version 8.5 (Release 2015a)
September 2015	Online only	Revised for Version 8.6 (Release 2015b)
October 2015	Online only	Rereleased for Version 8.5.1 (Release 2015aSP1)
March 2016	Online only	Revised for Version 9.0 (Release 2016a)
September 2016	Online only	Revised for Version 9.1 (Release 2016b)

Startup and Shutdown

Start MATLAB on Windows Platforms	1-2
Ways to Start MATLAB	1-2
Speeding Up MATLAB Start Up on Windows Systems	1-2
Associating Files with MATLAB on Windows Platforms	1-4
Associate .mat Files with MATLAB	1-5
Start MATLAB on Linux Platforms	1-6
Start MATLAB on Mac Platforms	1-7
From Applications Folder	1-7
From Terminal Window	1-7
Exit MATLAB	1-9
Ways to Exit	1-9
Confirm Exiting	1-9
Running a Script When Exiting	1-10
Recovering Data After an Abnormal Termination	1-11
Error Log Reporting	1-12
Emailing Error Log Reports	1-12
When MATLAB Terminates Unexpectedly	1-13
Specifying Java Startup Options	1-14
MATLAB Startup Folder	1-16
Default Folder on Windows Platforms	1-16
Default Folder on Mac Platforms	1-17
Default Folder on Linux Platforms	1-17

userpath as Initial Working Folder	1-17
Changing the Startup Folder	1-18
Commonly Used Startup Options	1-19
Specify Startup Options	1-21
Startup Options from Operating System Prompt	1-21
Startup Options in Shortcut on Windows Systems	1-21
Startup Options in MATLAB Startup File	1-22
Passing Perl Variables on Startup	1-22
Startup and Calling Java Software from MATLAB	1-23
Toolbox Path Caching in MATLAB	1-24
About Toolbox Path Caching in MATLAB	1-24
Using the Cache File Upon Startup	1-24
Updating the Cache and Cache File	1-24
More Diagnostics with Toolbox Path Caching	1-26

Desktop

2

Change Fonts	2-2
Font Preferences	2-2
Help and Web Browser Fonts	2-3
Adding Fonts on Windows Systems	2-3
Fonts Custom Preferences	2-5
Change Color Settings	2-6
Changing Text and Background Colors in Desktop Tools	2-6
Changing Syntax Highlighting Colors	2-6
Changing Command Window Colors	2-7
Changing Code Analyzer Colors	2-7
Access Frequently Used Features	2-9
Optimize Desktop Layout for Limited Screen Space	2-11
Desktop Layout	2-11
Document Layout	2-13

Define Keyboard Shortcuts	2-15
Keyboard Shortcuts	2-15
Choose a Set of Keyboard Shortcuts	2-15
Compare Sets of Keyboard Shortcuts	2-18
Display Keyboard Shortcuts	2-19
Customize Keyboard Shortcuts	2-22
Evaluate and Resolve Keyboard Shortcut Conflicts	2-27
Examples of Creating, Modifying, and Deleting Keyboard Shortcuts	2-29
Delete a Set of Keyboard Shortcuts	2-32
Use Keyboard Shortcuts Settings Files Created on Other Systems	2-33
Keyboard Shortcut Restrictions	2-33
Set Print Options	2-36
Page Setup Options	2-36
Layout Options for Page Setup	2-36
Header Options for Page Setup	2-37
Fonts Options for Page Setup	2-37
Web Browsers and MATLAB	2-39
About Web Browsers and MATLAB	2-39
Display Pages in Web Browsers	2-41
Specify Proxy Server Settings for Connecting to the Internet	2-41
Specify the System Browser for Linux Platforms	2-42
Manage Your Licenses	2-43
Check for Software Updates	2-45
Macintosh Platform Conventions	2-46
Mouse Instructions and Macintosh Platforms	2-46
Navigating Within the MATLAB Root Folder on Macintosh Platforms	2-46
Preferences	2-48
Set Preferences for MATLAB	2-48
Where MATLAB Stores Preferences	2-49
Preferences MATLAB Uses When Multiple Releases Are Installed	2-49
General Preferences	2-51
Confirmation Dialog Boxes Preferences	2-52
Source Control Preferences	2-54

Keyboard Shortcuts Preferences	2-54
Colors Preferences	2-55
Colors Programming Tools Preferences	2-56
Comparison Preferences	2-57
Toolbars Preferences	2-58
Web Preferences	2-59

Entering Commands

3

Enter Statements in Command Window	3-2
Find Functions to Use	3-4
Format Output	3-7
Format Line Spacing in Output	3-7
Format Floating-Point Numbers	3-8
Wrap Lines of Code to Fit Window Width	3-9
Suppress Output	3-9
View Output by Page	3-9
Clear the Command Window	3-10
Stop Execution	3-11
Find Text in Command Window or History	3-12
Find Text in the Command Window	3-12
Find Text in the Command History Window	3-14
Create Shortcuts to Rerun Commands	3-15
Set Command Window Preferences	3-17
Set Keyboard Preferences	3-19
Check Syntax as You Type	3-22
Syntax Highlighting	3-22
Delimiter Matching	3-23
Tab Completion	3-23
Function Syntax Hints	3-26

Command History	3-28
What Is the Command History?	3-28
Use Command History Commands	3-29
Change the Command History Date Format	3-30
Command History Preferences	3-30

Help and Product Information

4

Ways to Get Function Help	4-2
MATLAB Code Examples	4-3
Standalone Examples	4-3
Inline Examples	4-5
Search Syntax and Tips	4-7
Bookmark and Share Page Locations	4-10
Bookmark Favorite Pages	4-10
View Page Locations	4-11
Contact Technical Support	4-12
Help Preferences	4-14
Japanese Documentation	4-16
Korean and Chinese Documentation	4-17
Information About Your Installation	4-18

Workspace Browser and Variable Editor

5

Create and Edit Variables	5-2
View Workspace Contents	5-2
Create Variables	5-2

View Variable Contents	5-3
Edit Variable Contents	5-4
Navigate Variable Contents	5-10
Copy, Rename, and Delete Variables	5-11
Delete Variables	5-12
Display Statistics in the Workspace Browser	5-13
Save and Load Workspace Variables	5-15
View Contents of MAT-File	5-17
Save Variables to MATLAB Script	5-18
Save Structure Fields as Separate Variables	5-18
Workspace and Variable Preferences	5-20
Workspace Preferences	5-20
Variables Preferences	5-22

Managing Files in MATLAB

6

Find Files and Folders	6-2
Simple Search for File and Folder Names	6-2
Advanced Search for Files	6-2
Comparing Files and Folders	6-6
Comparing Files and Folders	6-6
Comparing Folders and Zip Files	6-8
Comparing Text and Live Scripts	6-11
Comparing Files with Autosave Version or Version on Disk ..	6-16
Comparing MAT-Files	6-17
Comparing Variables	6-20
Comparing Binary Files	6-20
Using Comparison Tool Features	6-21
Function Alternative for Comparing Files and Folders	6-24
Manage Files and Folders	6-25
MathWorks File Extensions	6-28
Files and Folders that MATLAB Accesses	6-29
Where Does MATLAB Look for Files?	6-29

Files and Folders You Should Add to the Search Path	6-29
When Multiple Files Have the Same Name	6-30
Locations of MathWorks Products	6-30
Current Folder Browser Preferences	6-32
Specify File Names	6-34
Construct Valid Path and File Names	6-34
Case Sensitivity of File Names	6-36
Create and Extract from Zip Archives	6-38
Create a Zip Archive	6-38
Add Files to a Zip Archive	6-39
Extract Files from a Zip Archive	6-39
Compare Zip Archive to Unzipped Files	6-40
What Is the MATLAB Search Path?	6-41
userpath Folder on the Search Path	6-41
MATLABPATH Environment Variable	6-42
Determine If Files and Folders Are on the Search Path	6-42
The Search Path Is Not the System Path	6-43
How MATLAB Stores the Search Path	6-44
Change Folders on the Search Path	6-45
For Current and Future Sessions	6-45
For the Current Session Only	6-47
Use Search Path with Different MATLAB Installations	6-48
Add Folders to the MATLAB Search Path at Startup	6-49
Use a startup.m File	6-49
Set the MATLABPATH Environment Variable	6-49
Assign userpath as Startup Folder (Macintosh or UNIX)	6-51
Path Unsuccessfully Set at Startup	6-52
Errors When Updating Folders on Search Path	6-54

Editor Preferences

7

Editor/Debugger Preferences	7-2
General Preferences for the Editor/Debugger	7-2
Editor/Debugger Display Preferences	7-3
Editor/Debugger Tab Preferences	7-4
Editor/Debugger Language Preferences	7-5
Editor/Debugger Code Folding Preferences	7-8
Editor/Debugger Backup Files Preferences	7-9
Editor/Debugger Autoformatting Preferences	7-10
Code Analyzer Preferences	7-12
Code Analyzer Preferences	7-12
Searching Messages in the Code Analyzer Preferences Dialog Box	7-13

Add-Ons

8

Get Add-Ons	8-2
Install an Add-On Manually	8-2
Manage Your Add-Ons	8-3
Change Add-Ons Installation Folder	8-3

Internationalization

9

Locale Settings for MATLAB Process	9-2
Default Locale Setting	9-3
Supported Character Set	9-3
Platform-Specific Localized Formats for Current Folder Browser	9-3
Limitations to International Character Support	9-4

Set Locale on Windows Platforms	9-5
Locale on Windows 10 Platforms	9-5
Locale on Windows 8 Platforms	9-5
Locale on Windows 7 Platforms	9-6
Set Locale on Mac Platforms	9-7
Set Locale on Linux Platforms	9-8
Unexpected Behavior on Mac Platforms	9-9
Characters Incorrectly Displayed on Windows Systems ...	9-10
datenum Might Not Return Correct Value	9-11
Numbers Display Period for Decimal Point	9-12
File or Folder Names Incorrectly Displayed	9-13
Script Compatibility	9-14
MATLAB Desktop Language Preference	9-15

Startup and Shutdown

- “Start MATLAB on Windows Platforms” on page 1-2
- “Associating Files with MATLAB on Windows Platforms” on page 1-4
- “Associate .mat Files with MATLAB” on page 1-5
- “Start MATLAB on Linux Platforms” on page 1-6
- “Start MATLAB on Mac Platforms” on page 1-7
- “Exit MATLAB” on page 1-9
- “Recovering Data After an Abnormal Termination” on page 1-11
- “Error Log Reporting” on page 1-12
- “When MATLAB Terminates Unexpectedly” on page 1-13
- “Specifying Java Startup Options” on page 1-14
- “MATLAB Startup Folder” on page 1-16
- “Commonly Used Startup Options” on page 1-19
- “Specify Startup Options” on page 1-21
- “Toolbox Path Caching in MATLAB” on page 1-24

Start MATLAB on Windows Platforms


In this section...

“Ways to Start MATLAB” on page 1-2

“Speeding Up MATLAB Start Up on Windows Systems” on page 1-2

Ways to Start MATLAB

There are several ways to start MATLAB on a Microsoft Windows platform. In these instructions, *Release* refers to your MATLAB release number, for example, R2016b.

- On Windows 10 systems, **Start > All apps > MATLAB Release**
- On Windows 8 systems, on the Start screen or the desktop click **MATLAB Release**.
- On Windows 7 systems, if you chose to have the installer put a shortcut to the MATLAB program on the Windows Start menu, select **Start > MATLAB Release**
- If you chose to have the installer create a shortcut, double-click the MATLAB shortcut  on your Windows desktop.
- Double-click a file with any of a number of file extensions in the Windows Explorer tool. The installer sets up associations between these file types and MathWorks® products during installation. For example, double-clicking a file with a `.m` extension starts MATLAB and opens the file in the MATLAB Editor. For more information, see “Associating Files with MATLAB on Windows Platforms” on page 1-4.
- From the Windows system prompt, type `matlab`.

After starting MATLAB, the desktop opens. Desktop components that were open when you last shut down MATLAB are opened on startup. You can specify other startup options, such the initial working folder—for more information, see “Specify Startup Options” on page 1-21 and “MATLAB Startup Folder” on page 1-16.

If you have trouble starting MATLAB, see “Troubleshooting Installation” in the Installation Guide.

Speeding Up MATLAB Start Up on Windows Systems

On Windows systems, the MathWorks Installer installs and configures a utility program that can speed up MATLAB startup, called the MATLAB Startup Accelerator.

For information about this program, including information about how to modify the configuration, see [Start the License Manager](#) in the “License Management” documentation.

Associating Files with MATLAB on Windows Platforms

When you install MATLAB on Windows platforms, the installer sets up associations between certain file types and MathWorks products. When you double-click one of these files in the Windows Explorer (file manager), Windows executes the open action specified by the file association. If Windows starts MATLAB, it opens the version of MATLAB associated with that file type.

By default, MATLAB associates the latest installed version to MATLAB file types. If you use the Windows file manager or the Control Panel to associate a MATLAB version to a file type, that version remains associated with the file type even when you install another MATLAB. If, instead, you always want to use the latest MATLAB version, you must manually reassociate the type with each new installation of MATLAB. If you want to return to the default behavior, uninstall every MATLAB that is manually associated to any MATLAB file type. Then reinstall the latest MATLAB release.

Sometimes double-clicking a MATLAB file in the Windows file manager opens a new instance of the version of MATLAB that is already open. To work around this issue, uninstall versions of MATLAB, release R2010a or earlier. Then reinstall MATLAB R2010b or later.

Alternatively, change the security setting on the `matlabroot\toolbox\local\pathdef.m` file to allow the `Users` group to write to the file. For more information, see <http://www.mathworks.com/matlabcentral/answers/93468-why-is-a-new-instance-of-matlab-opened-when-i-double-click-on-a-matlab-file-in-windows-explorer-even>.

More About

- “MathWorks File Extensions” on page 6-28
- “Associate `.mat` Files with MATLAB” on page 1-5

Associate .mat Files with MATLAB

To associate a .mat extension type with MATLAB R2010b or later, use the Microsoft Windows **Default Programs** control panel.

- 1 Open the Windows **Control Panel**. The **Control Panel** is typically available from the Windows Start menu, or refer to your Windows documentation.
- 2 From **View by: Category**, select **Programs**.
- 3 Select **Default Programs**.
- 4 Select **Set your default programs**. The **Programs** pane shows installed versions of MATLAB, R2010b and later.
- 5 Select a MATLAB release, then **Choose defaults for this program**.
- 6 Check the box next to the .mat entry.
- 7 Select **Save**.
- 8 Close all dialog boxes and menus, and exit the **Control Panel**.

If you want to use MATLAB R2010a or earlier, choose the **Associate a file type or protocol with a program** options from the **Default Programs** menu instead. Scroll down the list of file types to the .mat entry. Select **Change Program...** and choose a version of MATLAB from the **Recommended Programs** list. Do not use the **Browse** button.

For information about using these options, see your Windows documentation.

Note: By default, MATLAB associates the latest installed version to MATLAB file types. However, once you use the Windows Control Panel or the file manager to associate a MATLAB version to a file type, you must manually reassociate the type with each new installation of MATLAB. If you want to return to the default behavior, uninstall every MATLAB that is manually associated to any MATLAB file type. Then reinstall the latest MATLAB release.

More About

- “Associating Files with MATLAB on Windows Platforms” on page 1-4

Start MATLAB on Linux Platforms

To start MATLAB on Linux platforms, type `matlab` at the operating system prompt.

If you did not set up symbolic links in the installation procedure, type `matlabroot/bin/matlab`, where *matlabroot* is the name of the folder in which you installed MATLAB.

After starting MATLAB, the desktop opens. Desktop components that were open when you last shut down MATLAB are opened on startup.

If the `DISPLAY` environment variable is not set or is invalid, the desktop does not display. If you have trouble starting MATLAB, see “Troubleshooting Installation” topics in the Installation Guide.

You can specify the initial working folder and other options — for more information, see “MATLAB Startup Folder” on page 1-16 and “Specify Startup Options” on page 1-21.

To execute a MATLAB script, for example `hello.m`, via a remote ssh login, at the command shell type:

```
ssh local.foo.com matlab -nodisplay -nojvm < hello.m
```

The `ssh` command pipes `hello.m` to MATLAB running on the remote host, `local.foo.com`. The `-nodisplay` option starts MATLAB without the desktop.

See Also

`matlab` (Linux)

Start MATLAB on Mac Platforms

In this section...

“From Applications Folder” on page 1-7

“From Terminal Window” on page 1-7

From Applications Folder

This example shows how to start MATLAB from the Applications folder.

Double-click the MATLAB icon in the `Applications/matlabroot` folder, where *matlabroot* is the name of the folder in which you installed MATLAB.

The desktop opens, including components that were open when you last shut down MATLAB.

The default startup folder is `userhome/Documents/MATLAB`.

MATLAB automatically adds the `userpath` folder to the top of its search path.

If MATLAB fails to start due to a problem with required system components such as Java[®] software, diagnostics run automatically and advise you of the problem, with suggestions to correct it.

From Terminal Window

This example shows how to start MATLAB from a terminal window.

Open a Terminal window.

Navigate to your MATLAB installation folder,

```
/Applications/matlabroot/MATLAB_Release.app/bin
```

where *matlabroot* is the name of the folder in which you installed MATLAB, and *Release* is your MATLAB release number, for example, R2013b.

Start MATLAB.

```
./matlab
```

The desktop opens, including components that were open when you last shut down MATLAB.

The default startup folder is the MATLAB installation folder.

MATLAB automatically adds the `userpath` folder to the top of its search path.

If MATLAB fails to start due to a problem with required system components such as Java software, diagnostics run automatically and advise you of the problem, with suggestions to correct it.

See Also

`matlab` (Mac)

More About

- “MATLAB Startup Folder” on page 1-16
- “Troubleshooting Installation”

Exit MATLAB

In this section...



“Ways to Exit” on page 1-9

“Confirm Exiting” on page 1-9

“Running a Script When Exiting” on page 1-10

Ways to Exit

At any time, do one of the following:


- Click the Close box  in the MATLAB desktop.
- Click  on the left side of the desktop title bar and select **Close**.
- Type `quit` or `exit` at the command prompt.

MATLAB closes after:

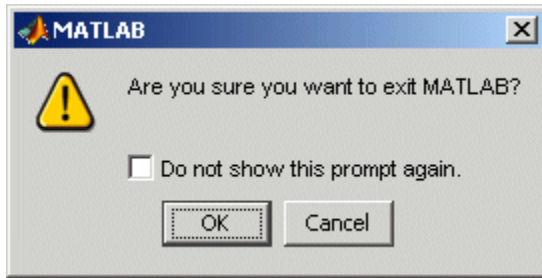
- Prompting you to confirm exiting, if that preference is specified.
- Prompting you to save any unsaved files
- Running a script when exiting, if the `finish.m` script exists in the current folder or on the search path.

Confirm Exiting

To set a preference that displays a confirmation dialog box when you exit:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**.
- 2 Select **MATLAB > General > Confirmation Dialogs**.
- 3 Select the **Confirm before exiting MATLAB** check box and click **OK**.

MATLAB then displays the following dialog box when you exit.



You can also display your own exit confirmation dialog box using a `finish.m` script, as described in the following section.

Running a Script When Exiting

When MATLAB exits, it runs the script `finish.m`, if `finish.m` exists in the current folder or anywhere on the search path. You create the file `finish.m` containing statements such as saving the workspace or displaying a confirmation dialog box. There are two sample files in `matlabroot/toolbox/local` that you can use as the basis for your own `finish.m` file:

- `finishesav.m` — Includes a `save` function so the workspace is saved to a MAT-file.
- `finishdlg.m` — Displays a confirmation dialog box that allows you to cancel quitting.

See Also

`exit` | `finish` | `quit`

More About

- “Confirmation Dialog Boxes Preferences” on page 2-52
- “Recovering Data After an Abnormal Termination” on page 1-11
- “Error Log Reporting” on page 1-12
- “When MATLAB Terminates Unexpectedly” on page 1-13

Recovering Data After an Abnormal Termination

If MATLAB terminates unexpectedly, you might lose information. After you start MATLAB again, you can try the following suggestions to recover some of the information. Some of these suggestions refer to actions you need to take during the session before MATLAB terminates. If you did not take those actions, consider regularly performing them to help you recover from any future abnormal terminations you might experience.

- Use the Command History or the file on which it is based, `history.m`, to run statements from the previous session. You might be able to recreate data as it was before the termination.
- If you used the `diary` function or `-logfile` startup option for the session in which MATLAB terminated unexpectedly, you might be able to recover output.
- If you saved the workspace to a MAT-file during the session, you can recover it by loading the MAT-file.
- If you were editing a file in the Editor when MATLAB terminated unexpectedly, and you had the backup feature enabled, you should be able to recover changes you made to files you had not saved. To recover, open the file in the Editor.
- If you were in a Simulink® session when a segmentation violation occurred, and you have the Simulink **Autosave Options** preference selected, the last autosave file for the model reflects the state of the autosave data before the segmentation violation. Because Simulink models might be corrupted by a segmentation violation, a model is not autosaved after a segmentation violation occurs. To recover the file, open the model.

See Also

`diary` | `load` | `save`

Related Examples

- “Save and Load Workspace Variables” on page 5-15
- “Save Files”

More About

- “Commonly Used Startup Options” on page 1-19

Error Log Reporting

Upon startup, if MATLAB detects an error log generated by a serious problem during the *previous* session, an Error Log Reporter prompts you to email the log to MathWorks for analysis. The error log contains the stack trace and information about the MATLAB software configuration. If the problem occurs repeatedly, make note of what seems to cause it, look for information about it in the MathWorks Bug Reports database, and if the problem is reproducible, submit a Service Request via http://www.mathworks.com/support/contact_us/ts/help_request_1.html.

Emailing Error Log Reports

There are some situations where the Error Log Reporter does not open, for example, when you start MATLAB with a `-r` option or run in deployed mode. It also does not open if you selected the **Never Send** option the last time the Error Log Reporter opened. If you experience abnormal termination but do not see the Error Log Reporter on subsequent startups, you can instead email the reports. To locate the error log reports, type:

```
dir(fullfile(tempdir, 'matlab_crash_dump.*.*'))
```

Copy the contents of the file into the body of an email message and send to segv@mathworks.com. After you send the log file, delete it or move it to another location. If you do not delete it, the Error Log Reporter might detect it on the next startup and prompt you to send it, even if you already did.

See Also

`tempdir`

When MATLAB Terminates Unexpectedly

In the event MATLAB experiences a segmentation violation (segv) or other serious problem, the MATLAB System Error dialog box opens to notify you about the problem. When this occurs, the internal state of MATLAB is unreliable and not suitable for further use. Exit as soon as possible and then restart. However, you might want to first try to save your work in progress.

To exit and restart without trying to save your work, follow these steps:

- 1 If you want to view the stack trace for the problem, click **Details**.
- 2 Click **Close** to terminate MATLAB.
- 3 Restart MATLAB. If the Error Log Reporter dialog box opens, select the option to send a report to MathWorks.

To try to save your work in progress before exiting and restarting MATLAB, follow these steps:

- 1 If you want to view the stack trace for the problem, click **Details**.
- 2 Click **Attempt to Continue**. MATLAB tries to return to the Command Window or tool you were using.

The Command Window displays the message **Please exit and restart MATLAB** to the left of the prompt, which reminds you to discontinue use.

- 3 From the Command Window or tool, try to save the workspace and unsaved files.

Caution Because the internal state of MATLAB might be corrupted, do not save existing files to the same file name. Instead, specify a new file name. The information in the new file might be corrupted or incomplete.

- 4 Exit MATLAB immediately after saving because any further usage would be unreliable.
- 5 Restart MATLAB. If the Error Log Reporter dialog box opens, select the option to send a report to MathWorks.

Specifying Java Startup Options

You can specify custom Java startup options by creating a `java.opts` file, a text file containing one option per line. For example, use the `-Dproperty=value` command to assign a *value* to a system *property*.

Put the `java.opts` file in one of the following folders:

- MATLAB startup folder on page 1-16, if starting MATLAB from a command prompt.
- If there is no `java.opts` file in the startup folder, MATLAB checks the `matlabroot/bin/arch` folder. `matlabroot` is the output of the `matlabroot` function and `arch` is the output of the `computer('arch')` function.

A `java.opts` file in this location applies to all users, but individual users might not have permissions to modify files there.

Do not use a `java.opts` file in the following situations:

- To enable the use of the Java debugger, use the `matlab -jdb` command. For information, see `matlab (Linux)` or `matlab (Mac)`.
- To adjust the Java heap size on desktop versions of MATLAB, use “Java Heap Memory Preferences”.
- To modify the static Java class path, create a `javaclasspath.txt` file. For information, see “Static Path”.

To modify the library path, create a `javalibrarypath.txt` file. For information, see “Locating Native Method Libraries”.

See Also

`matlab (Linux)` | `matlab (Mac)`

Related Examples

- “Java Heap Memory Preferences”
- “Static Path”
- “Locating Native Method Libraries”

More About

- “MATLAB Startup Folder” on page 1-16

MATLAB Startup Folder

By default, MATLAB sets the initial working folder (startup folder) based on the method you use to start MATLAB. To identify the startup folder, type `pwd` at the command line immediately after starting MATLAB and before typing any other commands. You can change the startup folder using the `userpath` function or the General Preferences panel. For convenience, make this folder a folder that you frequently use.

Note: If a `startup.m` file changes the current folder, this value overrides the initial working folder value. Do not add `cd` statements to `startup.m`. For more information about user-defined options, see `startup`.

In this section...

“Default Folder on Windows Platforms” on page 1-16

“Default Folder on Mac Platforms” on page 1-17

“Default Folder on Linux Platforms” on page 1-17

“`userpath` as Initial Working Folder” on page 1-17

“Changing the Startup Folder” on page 1-18

Default Folder on Windows Platforms

How You Start MATLAB	Startup Folder
Double-click the MATLAB shortcut on your Windows desktop	The startup folder is set to the <code>userpath</code> value, whose default value is <code>Documents\MATLAB</code> . The <code>userpath</code> folder is automatically added to the search path. If MATLAB does not find a valid <code>userpath</code> value, the startup folder is <code>C:\Windows\System32</code> .
Double-click a file type associated with MATLAB	The folder in which the file resides is the startup folder. The <code>userpath</code> folder is automatically added to the search path.
In a DOS window	The folder in which you ran the <code>matlab</code> command is the startup folder. The <code>userpath</code> folder is automatically added to the search path.

Default Folder on Mac Platforms

How You Start MATLAB	Startup Folder
Double-click the MATLAB application	<p>The startup folder is the value returned when you enter <code>userpath</code>, which by default is <code>userhome/Documents/MATLAB</code>. MATLAB automatically adds the <code>userpath</code> folder to the top of its search path upon startup. To specify a different folder for <code>userpath</code>, and for other options, use the <code>userpath</code> function.</p> <p>You can specify that the <code>userpath</code> be the startup folder by setting the value of the environment variable <code>MATLAB_USE_USERWORK</code> to 1.</p>
Start MATLAB in a shell	The startup folder is the MATLAB installation folder.

Default Folder on Linux Platforms

On Linux platforms, the default startup folder is the folder from which you started MATLAB.

To specify the `userpath` as the startup folder, set the value of the environment variable `MATLAB_USE_USERWORK` to 1 before startup. By default, `userpath` is `userhome/Documents/MATLAB`, and MATLAB automatically adds the `userpath` folder to the top of the search path upon startup. To specify a different folder for `userpath`, and for other options, use the MATLAB `userpath` function.

`userpath` as Initial Working Folder

Using `userpath` as the startup folder offers these benefits:

- You can store the MATLAB files you work with in one, appropriately named location, such as `Documents/MATLAB`.
- Your MATLAB files are readily available at startup, because the current folder is always the same.
- You can always run your files because MATLAB automatically adds the `userpath` folder to the top of the search path.
- The first time you run a new version of MATLAB, MATLAB automatically creates the `userpath` folder if it does not exist.

- When you upgrade to a newer version of MATLAB, MATLAB automatically continues to use the same startup folder and your existing files.
- The default `userpath` utilizes the benefits provided by the standard location in the Windows and Macintosh environments for storing personal files. Files in the `Documents/MATLAB` folder are available to you when you use other machines. Because each user has their own `Documents/MATLAB` folder, other users, even those using your machine, cannot access files in your `Documents/MATLAB` folder.

Changing the Startup Folder

Starting in R2014b, you can change the startup folder using the General Preferences panel. On the **Home** tab, in the **Environment** section, click **Preferences**. Select **MATLAB > General**. Choose an option for the **Initial working folder**.

By default, the initial working folder is set to **Location based on MATLAB startup rules**.

See Also

`startup` | `userpath`

Related Examples

- “Specify Startup Options” on page 1-21
- “Assign `userpath` as Startup Folder (Macintosh or UNIX)” on page 6-51

More About

- “General Preferences” on page 2-51
- “Determine If Files and Folders Are on the Search Path” on page 6-42

Commonly Used Startup Options

The following table lists commonly used startup options for the `matlab` command. For a complete list of options, refer to the input arguments for `matlab` (Windows), `matlab` (Mac), or `matlab` (Linux).

Platform	Option	Description
All	<code>-c licensefile</code>	Set <code>LM_LICENSE_FILE</code> to <code>licensefile</code> . It can have the form <code>port@host</code> .
All	<code>-h</code> or <code>-help</code>	Display startup options (without starting MATLAB).
All	<code>-logfile</code> <code>"logfilename"</code>	Automatically write output from MATLAB to the specified log file.
Windows platforms	<code>-minimize</code>	Start MATLAB with the desktop minimized. Any desktop tools or documents that were undocked when MATLAB was last closed are not minimized at startup.
Mac and Linux platforms	<code>-nojvm</code>	Start MATLAB without loading the JVM™ software. This minimizes memory usage and improves initial startup speed, but restricts functionality. With <code>nojvm</code> , you cannot use the desktop, figures, or any tools that require Java software. For example, you cannot set preferences if you start MATLAB with the <code>nojvm</code> option. However, you can start MATLAB once <i>without</i> the <code>nojvm</code> option, set the preference, and quit MATLAB. MATLAB remembers that preference when you start it again, even if you use the <code>nojvm</code> option.
All	<code>-nosplash</code>	Start MATLAB without displaying its splash screen.
All	<code>-r "statement"</code>	Automatically run the specified statement immediately after MATLAB starts. This is sometimes referred to as calling MATLAB in batch mode. Files you run must be in the startup folder for MATLAB or on the search path. Do not include path names or file extensions. Enclose the statement in double quotation marks (" <i>statement</i> "). To separate multiple statements, use semicolons or commas.
All	<code>-singleCompThread</code>	Limit MATLAB to a single computational thread. By default, Windows uses the multithreading capabilities of the computer on which it is running.

See Also

matlab (Linux) | matlab (Mac) | matlab (Windows)

Specify Startup Options

In this section...

“Startup Options from Operating System Prompt” on page 1-21

“Startup Options in Shortcut on Windows Systems” on page 1-21

“Startup Options in MATLAB Startup File” on page 1-22

“Passing Perl Variables on Startup” on page 1-22

“Startup and Calling Java Software from MATLAB” on page 1-23

Startup Options from Operating System Prompt

You can specify startup options (also called command flags or command-line switches) that instruct the MATLAB program to perform certain operations when you start it. On all platforms, specify the options as arguments to the `matlab` command when you start at the operating system prompt. For example, the following starts MATLAB and suppresses the display of the splash screen.


```
matlab -nosplash
```

On Windows platforms, you can precede a startup option with either a hyphen (-) or a slash (/). For example, `-nosplash` and `/nosplash` are equivalent.

Startup Options in Shortcut on Windows Systems

You can add selected startup options (also called command flags or switches for the command line) to the target path for your shortcut on the Windows platform for MATLAB.

To use startup options for the MATLAB shortcut icon on a Windows platform, follow these steps:

- 1 Right-click the shortcut icon for MATLAB  and select **Properties** from the context menu. The Properties dialog box for MATLAB opens to the **Shortcut** pane.
- 2 In the **Target** field, after the target path for "`matlab.exe`", add the startup option, and click **OK**.

This example runs the MATLAB `results` script or function after startup, where `results.m` is in the startup folder or on the MATLAB search path. The text in the **Target** field is similar to the following:

```
"C:\Program Files\MATLAB\R2010b\bin\matlab.exe" -r "results"
```

Include the statement, but not the option (-r) in double quotation marks.

Use semicolons or commas to separate multiple statements. This example changes the format to `short`, and then runs the MATLAB code file `results`:

```
"... matlab.exe" -r "format('short');results"
```

Separate multiple options with spaces. This example starts MATLAB without displaying the splash screen, and then runs the MATLAB code file `results`:

```
"... matlab.exe" -nosplash -r "results"
```

Startup Options in MATLAB Startup File

The `startup.m` file is a file you create to specify startup options. Create the `startup.m` file in a folder on the MATLAB search path. Use `startup.m` to modify the default search path, predefine variables in your workspace, or define defaults for graphics objects. For example, the following statement adds the user-defined folder `/home/myname/mytools` to the search path.

```
addpath /home/myname/mytools
```

To change the current folder on startup to `mytools`, set the **Initial working folder** value, described in “General Preferences” on page 2-51, to:

```
/home/myname/mytools
```

At startup, MATLAB automatically executes the file `matlabrc.m` and, if it exists on the MATLAB search path, `startup.m`. The file `matlabrc.m`, which is in the `matlabroot/toolbox/local` folder, is reserved for use by MathWorks and by system administrators on multiuser systems. To locate the `startup.m` file, type:

```
which startup
```

If MATLAB finds a `startup.m` file, it displays the path to the file.

Passing Perl Variables on Startup

You can pass Perl variables to MATLAB on startup by using the `-r` option of the `matlab` function. For example, assume a MATLAB function `test` that takes one input variable:

```
function test(x)
```

To pass a Perl variable instead of a constant as the input parameter, follow these steps. This command starts MATLAB and runs `test` with the input argument `10`.

- 1 Create a Perl script such as

```
#!/usr/local/bin/perl
$val = 10;
system('matlab -r "test(' . ${val} . ')");
```

- 2 Invoke the Perl script at the prompt using a Perl interpreter.

For more information, see the `-r` option in `matlab` (Windows), `matlab` (Mac), or `matlab` (Linux).

Startup and Calling Java Software from MATLAB

When MATLAB starts, it constructs the class path for Java software using `javaclasspath.txt` and `javalibrarypath.txt` files. For more information, see “Java Class Path” and “Locating Native Method Libraries”.

For information about memory allocation for Java objects, see “Java Heap Memory Preferences”.

See Also

`matlab` (Linux) | `matlab` (Mac) | `matlab` (Windows)

More About

- “Commonly Used Startup Options” on page 1-19

Toolbox Path Caching in MATLAB

In this section...

“About Toolbox Path Caching in MATLAB” on page 1-24

“Using the Cache File Upon Startup” on page 1-24

“Updating the Cache and Cache File” on page 1-24

“More Diagnostics with Toolbox Path Caching” on page 1-26

About Toolbox Path Caching in MATLAB

For performance reasons, MATLAB caches toolbox folder information across sessions. The caching features are mostly transparent to you. However, if MATLAB does not see the latest versions of your MATLAB code files or if you receive warnings about the toolbox path cache, you might need to update the cache.

Using the Cache File Upon Startup

Upon startup, MATLAB gets information from a cache file to build the toolbox folder cache. Because of the cache file, startup is faster, especially if you run MATLAB from a network server or if you have many toolbox folders. When you end a session, MATLAB updates the cache file.

MATLAB does not use the cache file at startup if you clear the **Enable toolbox path cache** check box in **General Preferences**. Instead, it creates the cache by reading from the operating system folders, which is slower than using the cache file.

Updating the Cache and Cache File

How the Toolbox Path Cache Works

MATLAB caches (essentially, stores in a known files list) the names and locations of files in *matlabroot*/toolbox folders. These folders are for files provided with MathWorks products that should not change except for product installations and updates. Caching those folders provides better performance during a session because MATLAB does not actively monitor those folders.

We strongly recommend that you save any MATLAB code files you create and any files provided by MathWorks that you edit in a folder that is *not* in the *matlabroot* / toolbox folder tree. If you keep your files in *matlabroot* / toolbox folders, they might be overwritten when you install a new version of MATLAB.

When to Update the Cache

When you add files to *matlabroot* / toolbox folders, the cache and the cache file need to be updated. MATLAB updates the cache and cache file automatically when you install toolboxes or toolbox updates using the installer for MATLAB. MATLAB also updates the cache and cache file automatically when you use MATLAB tools, such as when you save files from the MATLAB Editor to *matlabroot* / toolbox folders.

When you add or remove files in *matlabroot* / toolbox folders by some other means, MATLAB might not recognize those changes. For example, when you:

- Save new files in *matlabroot* / toolbox folders using an external editor
- Use operating system features and commands to add or remove files in *matlabroot* / toolbox folders

MATLAB displays this message:

```
Undefined function or variable
```

Update the cache so MATLAB recognizes the changes you made in *matlabroot* / toolbox folders.

Steps to Update the Cache

To update the cache and the cache file,

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > General**.

The **General Preferences** pane is displayed.

- 2 Click **Update Toolbox Path Cache** and click **OK**.

Function Alternative

To update the cache, use `rehash toolbox`. To also update the cache file, use `rehash toolboxcache`. For more information, see `rehash`.

More Diagnostics with Toolbox Path Caching

To display information about startup time when you start MATLAB, select the **Enable toolbox path cache diagnostics** check box in **General Preferences**.

Desktop

- “Change Fonts” on page 2-2
- “Fonts Custom Preferences” on page 2-5
- “Change Color Settings” on page 2-6
- “Access Frequently Used Features” on page 2-9
- “Optimize Desktop Layout for Limited Screen Space” on page 2-11
- “Define Keyboard Shortcuts” on page 2-15
- “Set Print Options” on page 2-36
- “Web Browsers and MATLAB” on page 2-39
- “Manage Your Licenses” on page 2-43
- “Check for Software Updates” on page 2-45
- “Macintosh Platform Conventions” on page 2-46
- “Preferences” on page 2-48
- “Web Preferences” on page 2-59

Change Fonts


In this section...

“Font Preferences” on page 2-2

“Help and Web Browser Fonts” on page 2-3

“Adding Fonts on Windows Systems” on page 2-3

Font Preferences

Change the font for desktop tools using the **Fonts Preferences** dialog box. Access this dialog box on the **Home** tab, in the **Environment** section, by clicking  **Preferences**. Select **MATLAB > Fonts**.

The default font that MATLAB uses for a particular tool depends upon its content:

- Code tools, such as the Command Window and Editor, use a monospaced font to preserve vertical alignment.
- Text-based tools, such as the Current Folder browser, use your system’s font.
- A few specific tools, including the Profiler, use a custom proportional font.

You can change the font for the group of code tools, for the group of text-based tools, or for individual tools. To change the font for an individual tool, or to move a tool from one group to another, click **Custom Fonts** and set the preferences for that tool.

This table describes the factory defaults for each group of tools. Refer to this table to restore fonts to their original state.

Font Group	Factory Defaults	Default Font Group Tools
Desktop code font	Monospaced, Plain, 10 point	Command History Command Window Editor (and Shortcuts Editor)
Desktop text font	Your system's current font	Current Folder browser (and Path browser) Workspace browser Variables editor

Font Group	Factory Defaults	Default Font Group Tools
		Function Browser
Custom fonts	SansSerif, Plain, 10 point	Profiler (and Code Analyzer messages, Function Browser help, and Supplemental Software help)

Note: For the Profiler, you can change the font type and size, but not the style (for example, bold or italic).

UNIX^{®1} systems include a preference to apply antialiasing: **Use antialiasing to smooth desktop fonts.** Select this preference for a smoother desktop appearance. You must restart MATLAB for the preference to take effect. This option is not provided on Microsoft Windows or Apple Macintosh platforms, because MATLAB follows the operating system's font settings on these platforms.

Help and Web Browser Fonts

To adjust the font size in the Help browser or MATLAB web browser, right-click the page and select **Zoom In** or **Zoom Out**. You cannot change the font type or style.

Adding Fonts on Windows Systems

MATLAB determines the set of fonts for the Preferences dialog box from your system settings on the first use of fonts within a session.

If, during a MATLAB session, you install a font that MATLAB can use, restart MATLAB to include it in the list. A common reason to install additional fonts is to read files created in different languages. For details on adding fonts to your system, refer to the Microsoft Windows help.

If MATLAB cannot display a particular font, it excludes that font from the list. The criteria for compatible fonts are as follows:

- For desktop components (such as the Command Window), figure windows, and uicontrols — Fonts compatible with TrueType and Microsoft OpenType[®] fonts

1. UNIX is a registered trademark of The Open Group in the United States and other countries.

- For graphics objects, such as `xlabel`, `ylabel`, `title`, and `text` — TrueType and Microsoft OpenType fonts

MATLAB looks for fonts in the following locations:


- The operating system's standard location (see your system administrator for details)
- The `/jre/lib/fonts` folder where Java software is installed on your system

Related Examples

- “Set Print Options” on page 2-36

Fonts Custom Preferences

You can override font settings for individual desktop tools, as described in the table that follows. Desktop tools otherwise use the settings that the “Font Preferences” on page 2-2 specify.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Fonts > Custom**, and then set options as described in this table:

Preference	Usage
Desktop tools	Select the desktop tool for which you want to view or customize fonts, such as the Command Window or Editor.
Font to use	<p>Indicates the font currently being used in the selected desktop tool. Use one of these fonts to change it.</p> <ul style="list-style-type: none"> • Desktop code Uses the characteristics of the desktop code font, as described in “Font Preferences” on page 2-2. • Desktop text Uses the characteristics of the desktop text font, as described in “Font Preferences” on page 2-2. • Custom Uses the type, style, and size you specify in the fields. <div style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <div style="display: flex; align-items: center; margin-bottom: 5px;"> <div style="border: 1px solid #ccc; padding: 2px 5px; margin-right: 5px;">SansSerif</div> <div style="border-left: 1px solid #ccc; border-right: 1px solid #ccc; border-bottom: 1px solid #ccc; width: 10px; height: 10px; margin-left: 5px;"></div> </div> <div style="display: flex; align-items: center;"> <div style="border: 1px solid #ccc; padding: 2px 5px; margin-right: 5px;">Plain</div> <div style="border-left: 1px solid #ccc; border-right: 1px solid #ccc; border-bottom: 1px solid #ccc; width: 10px; height: 10px; margin-left: 5px;"></div> <div style="border: 1px solid #ccc; padding: 2px 5px; margin-left: 5px;">10</div> <div style="border-left: 1px solid #ccc; border-right: 1px solid #ccc; border-bottom: 1px solid #ccc; width: 10px; height: 10px; margin-left: 5px;"></div> </div> </div> <p>For the Profiler, you can change the font type and size, but changes to the font style (for example, bold or italic) have no effect.</p>

Change Color Settings

In this section...

“Changing Text and Background Colors in Desktop Tools” on page 2-6

“Changing Syntax Highlighting Colors” on page 2-6


“Changing Command Window Colors” on page 2-7

“Changing Code Analyzer Colors” on page 2-7

Changing Text and Background Colors in Desktop Tools

To change the colors that MATLAB uses for text and background in desktop tools, follow these steps:

Note: The colors you specify also apply to the Import Wizard, but do not apply to the Help display pane or the web browser.

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Colors**.
- 2 Clear **Use system colors**.

System colors are the text and background colors that your platform (for example, Microsoft Windows) uses for other applications.
- 3 Select the colors you want to use from the **Text** and **Background** color palettes.

When you choose a color, the **Syntax Highlighting sample** and **Command Window sample** areas in the dialog box update to show you how it looks.

Tip If you use a gray background color, a selection in an inactive window is not visible.

- 4 Click **OK**.


Changing Syntax Highlighting Colors

In the Command Window, Command History, Editor, and Shortcuts callback area, MATLAB conveys syntax information using different colors. This feature, known as

syntax highlighting, helps you to identify syntax elements, such as `if/else` statements at a glance. The Editor also provides syntax highlights colors for other languages.


In the Command Window, only the MATLAB input you type is highlighted. The output from running MATLAB functions is not highlighted.

To change syntax highlighting colors, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Editor/Debugger > Language**.
- 2 From the **Language** drop-down menu, select the language for which you want to change syntax highlighting colors.
- 3 In the **Syntax highlighting** section, select **Enable syntax highlighting**.
- 4 Change the colors.
 - If you set the **Language** to MATLAB, click the **Set syntax colors** link, and then change the colors under **MATLAB syntax highlighting colors**.
 - If you did not set the **Language** to MATLAB, change the colors under **Syntax highlighting**.
- 5 Click **OK**.

Changing Command Window Colors

To change the colors that MATLAB uses for errors, warnings and hyperlinks in the Command Window, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Colors**.
- 2 Select the colors you want to use from the **Error text** and **Warning text**, and **Hyperlinks** color palettes.

When you choose a color, the **Command Window sample** area in the dialog box updates to show you how it looks.


- 3 Click **OK**.

Changing Code Analyzer Colors

Code Analyzer helps you to identify potential problems and refine your MATLAB code. By default, the Editor indicates:

- Code for which there are warnings, by underlining that code with an orange wavy line and placing an orange line in the message bar.
- Code for which there are errors, by underlining that code with a red wavy line and placing a red line in the message bar.
- Code that MATLAB can fix automatically (autofix), by highlighting that code in tan.

To change Code Analyzer colors:

- 1** On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Colors > Programming Tools**.
- 2** Under **Code analyzer colors**, select the colors you want for warnings, autofix highlighting, or both.
- 3** Decide if you want autofix highlights to appear in the Editor.

Clear **Autofix highlight** if you do not want autofix highlights to appear in the Editor; select **Autofix highlight** if you do.

- 4** Click **Apply**.
- 5** Decide if you want to change the color that the Code Analyzer uses for errors.
 - If you do not, go to step 6.
 - If you do, then:
 - a** In the left navigation pane, click **Colors**.
 - b** Under **MATLAB syntax highlighting colors**, change the color for **Errors**.

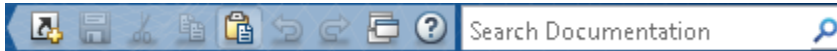
In addition to changing the color of Code Analyzer indicators for errors, this action also changes the color for errors in the Command Window, Command History window, Editor, and Shortcuts callback area.

- 6** Click **OK**.

For more information, see “Automatically Check Code in the Editor — Code Analyzer”.

Access Frequently Used Features


The quick access toolbar provides access to frequently used operations. This toolbar is always visible, even when you navigate between different MATLAB Toolstrip tabs.



You can change the location of the quick access toolbar. On the **Home** tab, in the **Environment** section, click **Layout**, and then select an option for the **Quick Access Toolbar**.



To add a Toolstrip button to the quick access toolbar, right-click the button, and then select **Add to Quick Access Toolbar**.

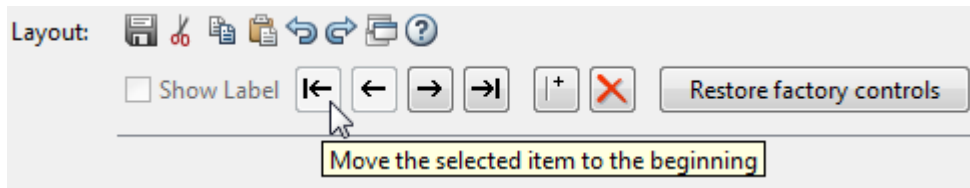
To add, remove, or arrange buttons on the quick access toolbar, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Then, select **MATLAB > Toolbars**.
- 2 From the **Toolbar** drop-down menu, select **Quick Access**.

The controls for the selected toolbar appear in the **Layout** and **Controls** sections of the Toolbars Preferences pane.

- 3 In the **Controls** list, select or clear the check box for controls that you want to display or remove from the toolbar, respectively.
- 4 Under **Layout**, rearrange the controls and separator bars on the selected toolbar, by doing either of the following:
 - Drag the icon for a control or separator bar to another position.
 - Select a **Layout** icon, and then click one of the **Layout** buttons below the layout icons.

For instance, to move the MATLAB desktop **Cut** icon to the beginning of the toolbar, select the **Cut** icon , and then click .



- 5 Click **Apply** or **OK**.

Optimize Desktop Layout for Limited Screen Space

In this section...






“Desktop Layout” on page 2-11






“Document Layout” on page 2-13

Desktop Layout

You can close, minimize, and undock desktop tools to optimize the desktop layout. Once you design a layout you like, you can save it for reuse.

This table shows how you can optimize the MATLAB desktop layout on your screen.

Action	Procedure
Minimize or restore the MATLAB Toolstrip	At the upper right corner of the Toolstrip, click  or  .
Open or hide a tool	On the Home tab, in the Environment section, click Layout . Then, under Show , select, or clear desktop tools you want to show or hide. You also can use a function to open desktop tools. For example, to open the Editor, use <code>edit</code> . To open the Profiler, use <code>profile</code> with the <code>viewer</code> option.
Maximize a tool	Do one of the following: <ul style="list-style-type: none"> • Double-click the title bar in that tool. • On the title bar of a docked tool, click , and then select Maximize.
Minimize a tool	On the title bar of a docked tool, click  , and then select Minimize  The button for the tool appears along the edge of the MATLAB desktop indicated by the arrow in the Minimize icon. Move the button to a different edge of the desktop by dragging it.

Action	Procedure
Use a minimized tool	Click the button for the tool to open the tool temporarily on the desktop. When you finish using the tool, click another tool.
Restore a tool as it appeared before maximizing or minimizing	Do one of the following: <ul style="list-style-type: none"> • Double-click the title bar of the maximized tool, or the button of the minimized tool. • On the title bar of the tool, click , and then select Restore. • Click the Restore button  on the title bar in that tool.
Move a tool	Drag a tool by its title bar to a new location. The status bar indicates where the tool moves if you release the mouse.
Close a tool	On the title bar of a docked tool, click  , and then select Close .
Show or hide title bars	On the Home tab, in the Environment section, click Layout . Then, under Show , select, or clear Panel Titles .
Show or hide a toolbar in a figure window	From the View menu, select the toolbar of interest.
Undock tools to move them outside the desktop	Do one of the following: <ul style="list-style-type: none"> • Drag a tool by its title bar to a new location outside of the MATLAB desktop. • On the title bar of the tool, click , and then select Undock.
Move undocked tools back to the desktop	At the upper right of the tool panel, click  , and then select Dock .

Action	Procedure
Manage a desktop arrangement <ul style="list-style-type: none"> • Save an arrangement • Use an arrangement • Rename or delete a saved arrangement 	On the Home tab, in the Environment section, click Layout , and then select an option. <hr/> Note: MATLAB stores the arrangements you save as XML files in the preferences folder for MATLAB. The layout last used in a session is <code>MATLABDesktop.xml</code> . The <code>MATLABDesktop.xml</code> file loads when you start MATLAB and is overwritten when you close MATLAB.





When you end a session, MATLAB saves the current desktop arrangement. The next time you start MATLAB, the desktop appears as you left it. However, tools such as the Help browser, web browser, and Variables editor do not reopen automatically, even if they were open when you ended the last session. You can use startup options to specify tools that you want to open on startup. For more information, see “Specify Startup Options” on page 1-21.

Document Layout

When you open MATLAB documents, they open in the associated tool, such as the Editor or Variables editor. The Editor and Variables editor appear in the position they occupied when last used. Entries for undocked documents appear on the Windows task bar, or the equivalent for your platform. Click the task bar entry for a document to make that document active.

This table shows how to optimize the layout of documents within a tool.

Action	Procedure
Move or hide document tabs	On the View tab, in the Document Tabs section, click Tabs Position ▾ , and then select an option.
Reorder documents	Drag a document tab to a different position. To alphabetize names of documents, in the Document Tabs section of the View tab, select Alphabetize .

Action	Procedure
Arrange or tile documents	<p>In the Editor and Variables editor, select the View tab. In the Tiles section, click a tile option.</p> <p>In a Figure panel, Help browser, or web browser, select a tile option, , , , or , on the right side of the toolbar.</p>
Move a tiled document	Drag the document tab to another tile. If you drag it to a tile that already contains a document, the document you are dragging covers up the other document.
Undock a document	Right-click the document tab and then select Undock .
Close and save the document currently displaying	Click × .
Close a document in the Editor without saving	Click Ctrl +× .

Define Keyboard Shortcuts

In this section...

“Keyboard Shortcuts” on page 2-15

“Choose a Set of Keyboard Shortcuts” on page 2-15

“Compare Sets of Keyboard Shortcuts” on page 2-18

“Display Keyboard Shortcuts” on page 2-19

“Customize Keyboard Shortcuts” on page 2-22

“Evaluate and Resolve Keyboard Shortcut Conflicts” on page 2-27

“Examples of Creating, Modifying, and Deleting Keyboard Shortcuts” on page 2-29

“Delete a Set of Keyboard Shortcuts” on page 2-32

“Use Keyboard Shortcuts Settings Files Created on Other Systems” on page 2-33

“Keyboard Shortcut Restrictions” on page 2-33

Keyboard Shortcuts

To access desktop features quickly, you can use keyboard shortcuts. Press **Alt** to display tooltips on MATLAB Toolstrip buttons, indicating what keys to press to access those features. For example, pressing **Alt** followed by **H** accesses the **Home** tab and displays tooltips for the features available on that tab. You cannot customize these shortcuts.


An action can have multiple keyboard shortcuts. All defined shortcuts work, but only one appears on the desktop Toolstrip tooltip.

You can:

- Choose from a set of shortcuts that install with MATLAB.
- Create customized sets of shortcuts.
- Use a set of shortcuts copied from another system

Choose a Set of Keyboard Shortcuts

By default, MATLAB uses keyboard shortcut settings that correspond to the platform on which you are running. To choose different keyboard shortcut settings, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 Click the down arrow in the **Active settings** field, and make a selection from the drop-down list, as summarized in this table.

Settings File	Option to Select	Details
Installed with MATLAB	<ul style="list-style-type: none"> • On Mac, Mac Default Set • On all other systems, Windows Default Set or Emacs Default Set 	For a description of the files that install with MATLAB, see “Installed Settings Files for Keyboard Shortcuts” on page 2-16.
Previously added	The file name	No additional information.
On your system, but not in the drop-down list	Browse	“Browse to Keyboard Shortcuts Settings Files” on page 2-16.

- 3 Click **Apply**.


Installed Settings Files for Keyboard Shortcuts

The following table lists the keyboard shortcuts settings files installed with MATLAB.

Operating System	Keyboard Shortcut Settings Files Installed with MATLAB
Windows	<ul style="list-style-type: none"> • Windows Default Set (Default) • Emacs Default Set
UNIX	<ul style="list-style-type: none"> • Emacs Default Set (Default) • Windows Default Set
Macintosh	<ul style="list-style-type: none"> • Macintosh Default Set (Default)

Browse to Keyboard Shortcuts Settings Files

Browse to use a keyboard shortcuts settings file that is on your system, but not an **Active settings** choice in the Keyboard Shortcuts Preferences dialog box. This situation typically arises when you copy a settings file from another system to a folder other than the `prefdir` folder. To browse to a settings file and make it your active settings file, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 In the **Active settings** field, click the down arrow, and then select **Browse**.
- 3 In the Open dialog box, navigate to the folder containing the settings file.
- 4 Select the settings file, and then click **Open**.
- 5 In the Keyboard Shortcuts preferences pane, click **OK**.

The settings file you selected in step 4 is now the active settings file for MATLAB.

Future MATLAB sessions provide this settings file as a choice in the **Active settings** drop-down menu.

Use Keyboard Shortcut Settings Files from File Exchange

Download keyboard shortcut settings files from File Exchange when you want to do either of the following:


- Restore the MATLAB default keyboard shortcuts that were in place for MATLAB Version 7.9 (R2009a) and earlier releases.
- Find and download keyboard shortcuts that others created and uploaded to File Exchange.

Follow these steps:

- 1 Search the File Exchange Web site for the keyboard shortcut set that you want to use. Files tagged with **keyboard shortcuts configurable** include:
 - MATLAB Desktop R2009a Non-Default Keyboard Shortcut sets
 - MATLAB Desktop R2009a Default Keyboard Shortcut sets
- 2 Click the name of the file submission to view a description of the file.
- 3 Click the **Download Submission** button and save the **.ZIP** file to your computer.
- 4 In the MATLAB Current Folder browser, navigate to the location of your saved file. Right-click the downloaded **.ZIP** file, and then select **Extract**.

MATLAB creates a subfolder with the same name as the **.ZIP** file and extracts the files from that **.ZIP** file into the newly created folder.

- 5 In the Current Folder browser, expand the newly created folder, and then double-click the settings file you want to use.

A keyboard key icon  preceding a file name indicates a valid keyboard shortcut settings file.

- 6 In the Keyboard Shortcuts Preferences dialog box, review the settings, and then click **OK**.

The newly downloaded settings file is now in effect.

Compare Sets of Keyboard Shortcuts

Compare sets of keyboard shortcuts to:



- Upgrade MATLAB from a version before Version 7.9 (R2009b).

MATLAB 7.9 made keyboard shortcuts consistent across the desktop. Therefore, you might find that shortcuts you used before Version 7.9 are different.

- See how a set of keyboard shortcuts you found on File Exchange differs from your current set of keyboard shortcuts.
- See how a set of keyboard shortcuts differs from the default set.

Steps for Comparing Keyboard Shortcuts

To compare your current set of keyboard shortcuts to another set:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 Click the Actions button .
- 3 From the drop-down menu, choose the set of keyboard shortcuts to which you want to compare the current set.
- 4 The Comparison Tool opens and displays the two keyboard shortcut sets side-by-side.

Read the Results of Comparing Sets of Keyboard Shortcuts

When you compare keyboard shortcut sets, they appear in the Comparison Tool as follows:

- One set displays on the left side of the tool and the other set displays on the right side of the tool.

- Each column header displays the name of the keyboard shortcut set contained within the column.
- Highlighting identifies rows that differ:
 - Rows that exist in one file, but not the other, appear in green highlighting.
 - Rows that appear in both files, but that differ in content appear in pink highlighting.
- When multiple desktop tools support the same keyboard shortcut for a single desktop action, there is a row for each tool. For example, if both the MATLAB desktop and the Editor support the keyboard shortcut **Ctrl+W** for closing a selected window, a column of the Comparison Tool might appear like this:

<u>51</u>	Close	MATLAB Desktop	Ctrl+W	Closes the selected window
<u>52</u>	Close	MATLAB Editor	Ctrl+W	Closes the selected window

- When there are multiple keyboard shortcuts for the same action in a single tool, there is a row for each keyboard shortcut. For example, if there are two different keyboard shortcuts in the Editor for applying a code analyzer autofix, a column of the Comparison Tool might appear like this:

Autofix Message	MATLAB Editor	Alt+Enter	Applies the suggested autofix	<u>11</u>
Autofix Message	MATLAB Editor	Shift+F9	Applies the suggested autofix	<u>12</u>

- On Macintosh platforms, the textual format of keyboard shortcuts is slightly different from other platforms, and also differs from the representation shown on MATLAB desktop menus. These differences are due to the Macintosh platform displaying shortcuts using symbols. For instance, the Macintosh platform uses the symbol **⌘** for a keyboard key. Because the Comparison Tool represents symbols as text, it specifies the symbol **⌘** as CMD.

See also “Using Comparison Tool Features” on page 6-21.



Display Keyboard Shortcuts

The following sections describe the various ways you can display keyboard shortcuts:

- “List All Keyboard Shortcuts in a Set” on page 2-20
- “Display Keyboard Shortcuts on Menus” on page 2-20
- “Display Keyboard Shortcuts in the Preferences Dialog Box” on page 2-20

List All Keyboard Shortcuts in a Set

You can copy all the keyboard shortcuts from a keyboard shortcuts set and paste them in a text file or spreadsheet application, such as Microsoft Excel®. To create a list of keyboard shortcuts for easy browsing and future reference, follow these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 Click the Actions button .
- 3 From the drop-down menu, choose **Copy to Clipboard**.
- 4 Open a spreadsheet application or a text editor.

For the best formatting use a spreadsheet application.

- 5 Paste in the data from the clipboard.


In Microsoft Excel, for example, choose **Home > Paste**.

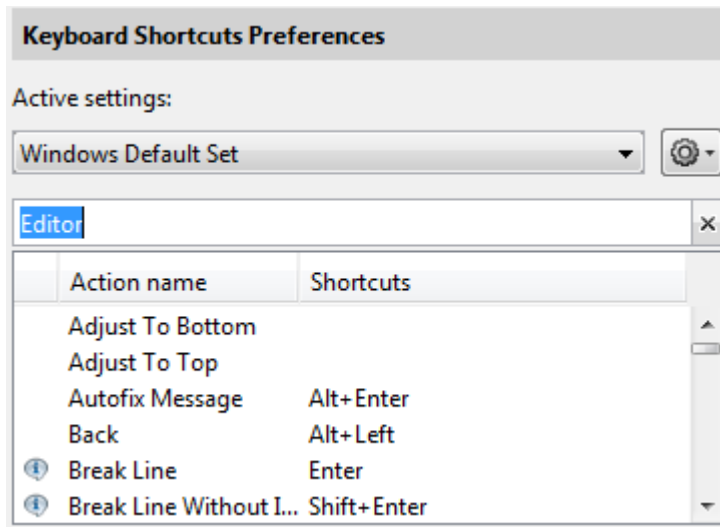
Display Keyboard Shortcuts on Menus

If no keyboard shortcut appears on the menu, one does not currently exist for that action. To create a keyboard shortcut for an action, follow the steps in “Customize Keyboard Shortcuts” on page 2-22.

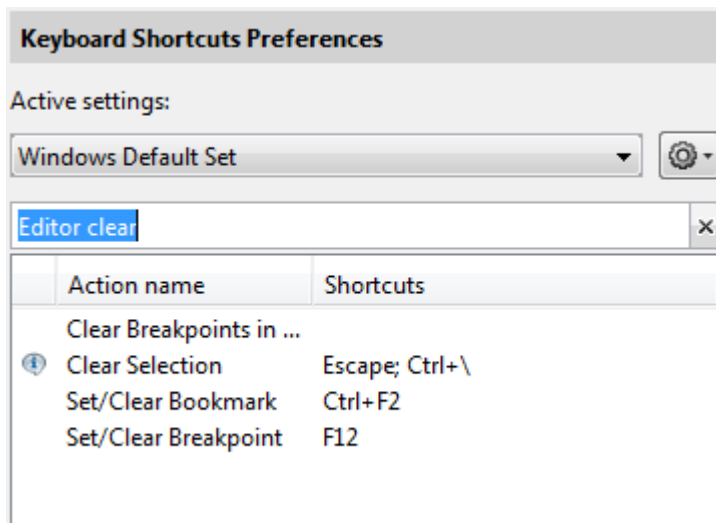
Display Keyboard Shortcuts in the Preferences Dialog Box

To identify a keyboard shortcut when there is no menu option for an action, use the **Keyboard Shortcuts Preferences** pane:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 In the filter field, type the name of the tool for which you want to list the keyboard shortcuts. For example, type **Editor** to see the keyboard shortcuts currently defined for actions you can perform in the Editor.



- 3 Narrow the list of **Action names** that the preferences pane displays by adding text to describe the action. For example, add **clear**, if you want to find the keyboard shortcut for clearing selected text in the Editor. Keep the description text short to increase the likelihood of the filter returning the action you seek.




- 4 Select the action name of interest. In this example, select **Clear Selection**.
- 5 View the table labeled **Shortcuts for Clear Selection**. It indicates that the **Escape** key is the current keyboard shortcut for the **Clear Selection** action in the Editor.

Shortcuts for Clear Selection

Shortcut	Tools with shortcut
⌨ Escape	MATLAB Editor
⌨ Ctrl+\	Command Window

+ -


Customize Keyboard Shortcuts

You can customize or view keyboard shortcuts for MATLAB desktop tools. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**. If you have an active Internet connection, you can watch the Customizable Keyboard Shortcuts video for an overview.

The following sections provide details:

- “Steps for Customizing Keyboard Shortcuts” on page 2-22
- “Filter Keyboard Shortcut Actions” on page 2-25
- “Specify Keystrokes for a Keyboard Shortcut” on page 2-26
- “Evaluate and Resolve Keyboard Shortcut Conflicts” on page 2-27
- “Examples of Creating, Modifying, and Deleting Keyboard Shortcuts” on page 2-29
- “Display Keyboard Shortcuts” on page 2-19

Steps for Customizing Keyboard Shortcuts

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 In the **Active settings** field, choose the file that contains the set of keyboard shortcuts that you want to customize.

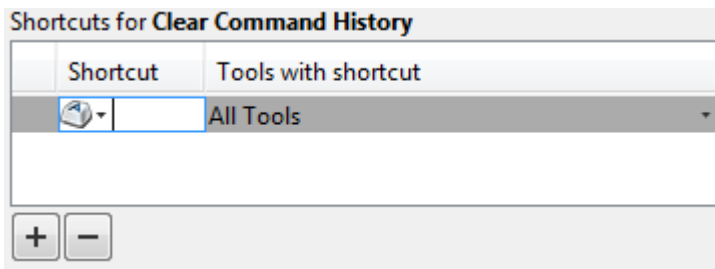
Typically, the first time you modify keyboard shortcuts, you begin with the default settings for your platform. For details, see “Choose a Set of Keyboard Shortcuts” on page 2-15.

- 3 Under **Action name**, select the action for which you want to define or modify a keyboard shortcut. An action is the operation for which you want to customize the shortcut, such as **Clear Command History**.

For tips on finding the action you want, see “Filter Keyboard Shortcut Actions” on page 2-25.


- 4 Click the Add button .

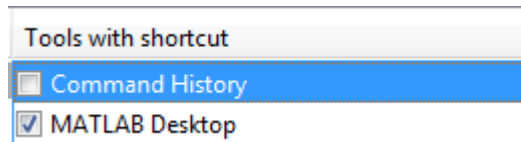
An editable field opens under the **Shortcut** column.





- 5 Type the shortcut that you want to use for the action you selected in Step 3. Alternatively, you can choose a shortcut from the drop-down menu.

For details, see “Specify Keystrokes for a Keyboard Shortcut” on page 2-26.

- 6 Assign the shortcut to the tool or tools with which you want to use it. For example, in the **Tools with shortcut** column:
- Click the down arrow  for the list of desktop tools to which you can assign a shortcut. Not all actions are available with all desktop tools.
 - Select a check box to assign the shortcut to a tool. Clear a check box to remove it.



- 7 Evaluate and resolve any conflicts, indicated by the informational  and error  icons.


For more information, see “Evaluate and Resolve Keyboard Shortcut Conflicts” on page 2-27.

8 Click **Apply**.

- The keyboard shortcut becomes available immediately.
- If a changed shortcut corresponds to a menu option that previously displayed no keyboard shortcut, MATLAB reflects the new keyboard shortcut on the menu.

Restore Default Keyboard Shortcut Sets

If you modify keyboard shortcuts, and then decide you do not want to keep the changes, you can restore the default shortcuts. To restore the default state of a keyboard shortcut:

- 1 Click the **Actions** button .
- 2 Select **Undo Modifications to Windows Default Set (modified)** or **Undo Modifications to Emacs Default Set (modified)**, as appropriate for your system.
- 3 Click **OK**.

Note: Undoing modifications reverts all keyboard shortcuts changes that you made to the set. You cannot undo modifications on a shortcut-by-shortcut basis.

Save Keyboard Shortcuts to a Settings File

Save keyboard shortcuts to a settings file to:

- Save changes you make to a default settings file, such as the `Windows default` set, to a new set.

MATLAB preserves changes you make to the default sets across sessions. However, if you undo modifications to a default keyboard shortcut set (as described in “Restore Default Keyboard Shortcut Sets” on page 2-24) you lose all changes, unless you first save them to a new set.



- Copy the keyboard shortcuts settings file to another system running MATLAB and use it there.
- Overwrite a settings file that you previously saved.

You cannot overwrite the default settings files that install with MATLAB. MATLAB saves modifications that you make to a default set using the name of the default set appended with the text `(modified)`. For instance, `Windows default (modified)`.

- Share a keyboard shortcuts settings file with others.

For example, you can submit your file to the File Exchange repository. Click this link to go directly to the page where you can submit your file: [MATLAB Central File Exchange — Submit New File](#).

To save a keyboard shortcuts settings file, follow these steps:

- 1 Open the Keyboard Shortcuts Preferences dialog box. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 Click the **Actions** button , and then select **Save As**.
- 3 In the Save dialog box, navigate to the folder where you want to save the file, specify the file name, and then click **Save**.

MATLAB saves the file as an `.xml` file in the folder that you specified.

Filter Keyboard Shortcut Actions

Use the filter field to see the list of actions for which you can customize or define a keyboard shortcut:

- 1 Type all or part of any one of the following:

- An action name, for example, **Delete**.

MATLAB displays only the action names or desktop menus that contain the text you specify.

- The name of a desktop tool or menu, for example, **File** or **Command Window**.

MATLAB displays a list of the action names associated with the tool or menu you specify. In addition, the list includes any action names that contain the name of the tool or menu. For example, if you specify **Command History**, the list of action names includes **Next History Command**, which is a **Command Window** action.

- A keyboard shortcut, for example, **Ctrl+R**

MATLAB displays only the action names that have the shortcut you specify. Be aware of the following:

- You can enter most keyboard shortcuts by either pressing keystrokes or typing the key names.

For example, to enter **Ctrl+S**, use the keystroke (by pressing the **Ctrl** key and the **S** key). Or, type **Ctrl+S** character by character (**C-t-r-l-+-Y**).


- If using keystrokes for a keyboard shortcut does not work, try typing the characters instead. You *must* type some keyboard shortcuts character by character, such as shortcuts including the **Tab**, **Backspace**, or **Delete** keys.
 - Type **numpad** to refer to the number pad that is on the far right of some keyboards.
 - Type **Up** or **Down** to refer to the **Up arrow** or **Down arrow** keypad keys, respectively.
- 2 Verify that an **Action name** performs the action you expect:
 - a Hover the mouse pointer over the **Action name**. For example, **Remove Next Word**.
 - b View the tooltip that appears.


Action name	Shortcuts
Remove	
Remove Next Word	Ctrl+Delete
Remove Previous Word	Ctrl+Backspace

Deletes the next word

Specify Keystrokes for a Keyboard Shortcut

A *keystroke* can be a single key or the combination of a modifier (**Alt**, **Shift**, or **Ctrl**) and another key. When you create a keyboard shortcut, specify the keystrokes for the shortcut as follows:

- 1 Click the **Add** button .
- 2 Specify the number of keystrokes you want to use for the shortcut:
 - To use the default number of keystrokes, which is one keystroke, skip to step 3.
 - To specify multiple keystrokes, or to specify explicitly one keystroke follow these steps:


- a Click the down arrow next to the key icon  in the **Shortcuts** field.
- b Choose **Limit to 1 keystroke**, **Limit to 2 keystrokes**, or **Limit to 3 keystrokes**.

For instance, **Ctrl+F** is one keystroke, **Ctrl+Y**, **Shift+Z** is two keystrokes, and **Ctrl+Y**, **Shift+Z**, **F9** is three keystrokes.

3 Specify the keystrokes by doing one of the following:

- Type the keystrokes, by pressing the keys, *not* by typing the key names character by character.


For example, press the **Ctrl** key and the **Y** key. Do not type **C-t-r-l-+-Y**.



- Choose a keystroke, such as the **Tab** key, by clicking the down arrow next to the key icon  in the **Shortcuts** field. Then, choose the key name.



The listed keys already have a defined action within dialog boxes. For example, the **Tab** key navigates from one field to the next in dialog boxes.

Evaluate and Resolve Keyboard Shortcut Conflicts


Conflicts arise when two or more different actions have the same shortcut. There is no requirement that you resolve keyboard shortcut conflicts. However, if the same shortcut specifies two different actions, the shortcuts can be confusing to use.

View keyboard shortcut conflicts — On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.

The Keyboard Shortcuts preferences pane indicates conflicts using informational  and error  icons.

-  —An informational icon indicates that two different actions in two different tools have the same shortcut. For information on resolving these conflicts, see “Actions in Different Tools Have the Same Shortcut — Evaluating Conflicts” on page 2-28.
-  —An error icon indicates that two different actions within the same tool have the same shortcut. For information on resolving these conflicts, see “Actions in the Same Tool Have the Same Shortcut — Evaluating Conflicts” on page 2-28.


Actions in Different Tools Have the Same Shortcut — Evaluating Conflicts

Typically, you want to resolve conflicts indicated by the informational icon  when all the following are true:

- You use both tools frequently.
- You perform both actions frequently.
- You have difficulty remembering the action that the shortcut performs in each tool.

For instance on Microsoft Windows platforms, by default, **Ctrl+Shift+U** undocks a tool from the MATLAB desktop. However if you select text in the Editor, and then press **Ctrl+Shift+U**, it changes the selected text to uppercase. If you frequently use both of these actions, you can specify a different keyboard shortcut for one or both actions.

Actions in the Same Tool Have the Same Shortcut — Evaluating Conflicts

Typically, you want to resolve conflicts indicated by the error icon .

It can be *unnecessary* to resolve these conflicts if one or more of the following are true:

- The situation is temporary.

For instance, you are performing a two-step procedure. In the first step, you assign the keyboard shortcut to an action that results in a conflict. Then, in the second step, you remove the shortcut from the original action.

- The two actions are associated with different modes of the same tool.

By default, when the MATLAB Editor is in cell mode, **Ctrl+Up** and **Ctrl+Down** move the cursor to the Next and Previous cell, respectively. When the Editor is not in cell mode, those keyboard shortcuts scroll up and scroll down, respectively. The shortcuts are in conflict, but the behavior probably is expected, for the given MATLAB Editor mode.

Although not evident from the preferences pane, **Ctrl+C** presents a similar situation on Windows systems. **Ctrl+C** is the keyboard shortcut for interrupting MATLAB execution. However, the default keyboard shortcut for the copy action is also **Ctrl+C**. Therefore, if you:

- Select an item, and then press **Ctrl+C**, it copies the selected item to the clipboard, — regardless of whether MATLAB is busy.

- Do not select an item and press **Ctrl+C**, it interrupts MATLAB execution.

If you change the default keyboard shortcut for the copy action from **Ctrl+C** to another keystroke, then **Ctrl+C** interrupts MATLAB execution, regardless of whether you have selected an item.

Resolve Keyboard Shortcut Conflicts



To resolve a conflict, change or delete shortcuts such that there is a one-to-one correspondence between a shortcut and a frequently used action. For examples, see “Changing a Keyboard Shortcut” on page 2-30 and “Deleting a Keyboard Shortcut” on page 2-31.

Examples of Creating, Modifying, and Deleting Keyboard Shortcuts

- “Creating a New Keyboard Shortcut” on page 2-29
- “Changing a Keyboard Shortcut” on page 2-30
- “Deleting a Keyboard Shortcut” on page 2-31

Creating a New Keyboard Shortcut

By default, no keyboard shortcut is available for adding a Help topic to the list of favorites. If you frequently mark topics as favorites, you can define a keyboard shortcut for this action, as follows:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
 - 2 In the filter field, type **Help**.
 - 3 Scroll through the **Action name** list, and select **Add to Favorites**.
 - 4 Click the plus button .
- MATLAB adds a row to the table above the plus button.
- 5 In the **Shortcut** field, click the down arrow, and then change **Limit** to 1 keystroke to **Limit** to 2 keystrokes.
 - 6 In the **Shortcut** field, press **Ctrl+S**, and then **Alt+V**.

Notice that the All possible conflicts table is empty, which indicates that no other desktop action is currently using this combination of keystrokes.

7 Click **Apply**.

Notice that:

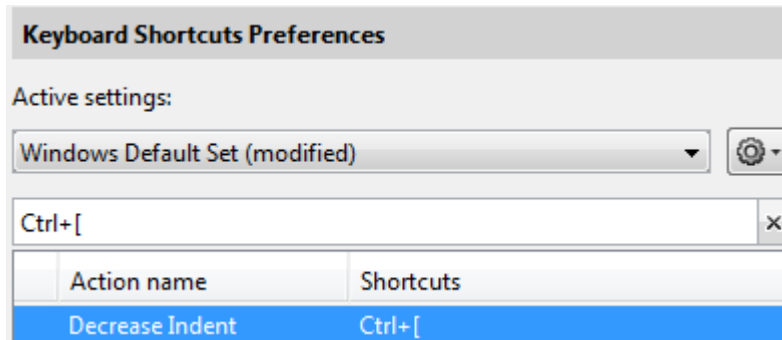
- The Add to Favorites dialog box opens when you press **Ctrl+S**, **Alt+V** in the Help browser.
- **Ctrl+S**, **Alt+V** appears next to **Add to Favorites** when you click the **Favorites** menu in the Help browser.

Changing a Keyboard Shortcut

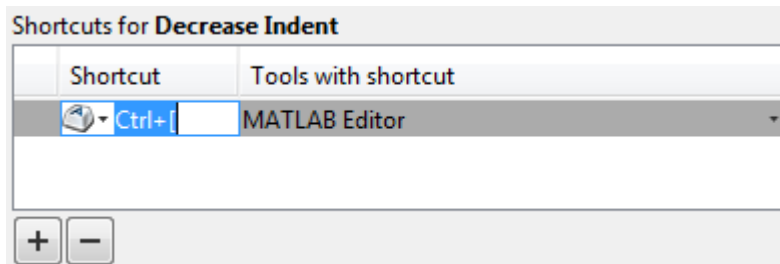
Suppose you frequently adjust indenting in the MATLAB Editor. However, you have difficulty remembering the default keyboard shortcut of **Ctrl+[** for decreasing the indent. So, you decide to change it to something that is easier to remember.

This example changes the keyboard shortcut for **Decrease Indent** in the MATLAB Editor from **Ctrl+[** to **Ctrl+Backspace**:

- 1 On the **Home** tab, in the **Environment** section, click **Preferences > Keyboard > Shortcuts**.
- 2 Under **Active settings**, choose **Windows Default Set**.
- 3 In the filter field, press **Ctrl+[**.
- 4 Under **Action name**, select **Decrease Indent**.



- 5 In the table labeled **Shortcuts for Decrease Indent**, under **Shortcut**, click **Ctrl+[**. MATLAB makes the field editable.



- 6 In the **Shortcut** field, press **Ctrl+Backspace** twice.


The first time you press the key combination, it deletes `Ctrl+[`. The second time you press it, `Ctrl+Backspace` appears in the field.

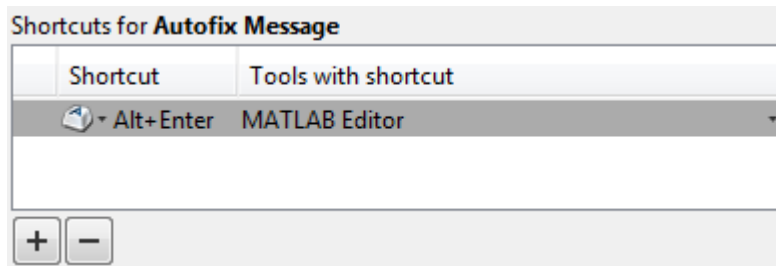
- 7 Click **Apply**.


MATLAB saves your changes to the `Windows Default Set (modified)` settings.

Deleting a Keyboard Shortcut

Suppose you find yourself frequently pressing the wrong keyboard shortcut. For example, on Windows, you press **Alt+Enter** (to apply a code analyzer autofix) instead of **Ctrl+Enter** (to evaluate the current cell in the MATLAB Editor). To avoid accidentally applying an autofix, delete the **Alt+Enter** shortcut by following these steps:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Keyboard > Shortcuts**.
- 2 Under **Active settings**, choose `Windows Default Set` or `Windows Default Set (modified)`.
- 3 In the filter field, press **Alt+Enter**.
- 4 Under **Action name**, select the row containing **Autofix Message**.
- 5 In the next table, under **Shortcuts for Autofix Message**, select the row containing **Alt+Enter**.




- 6 Click the remove button .
- 7 Click **Apply**.

If it does not exist, MATLAB creates a **Windows Default Set (modified)** keyboard shortcut set. This set consists of the **Windows Default Set** of keyboard shortcuts, less the shortcut for **Alt+Enter**. If the **Windows Default Set (modified)** settings file exists, then MATLAB deletes the **Alt+Enter** keyboard shortcut from that set of keyboard shortcuts.


See also “Delete a Set of Keyboard Shortcuts” on page 2-32.

Delete a Set of Keyboard Shortcuts

If you previously saved or copied a set of keyboard shortcuts to your system and you no longer want it, delete it as follows:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 2 Under **Active settings**, choose the set of keyboard shortcuts that you want to delete.


You cannot delete default keyboard shortcut sets, such as **Windows Default Set**.

- 3 Click the **Actions** button  and choose **Delete filename**, where *filename* is the name of a keyboard shortcut set you previously saved or copied to your system.

For information on deleting a single keyboard shortcut from a set that you want to keep, see “Deleting a Keyboard Shortcut” on page 2-31.

Use Keyboard Shortcuts Settings Files Created on Other Systems

If you find a keyboard shortcuts settings file that is useful to you, or if you want to use one you created on a different system, make it the active settings file as follows:

- 1 Copy the settings file to a folder on your system, such as:
`I:\my_matlab_files\active_settings_files\new_settings.xml`
- 2 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**.
- 3 In the **Active settings** field, click the down arrow, and then click **Browse**.
- 4 In the Open dialog box, navigate to the folder where you copied the settings file.
- 5 Select the settings file, and then click **Open**.
- 6 In the Keyboard Shortcuts preferences pane, click **Apply**. The settings file you specified is now the active settings file for MATLAB.

Keyboard Shortcut Restrictions

These sections describe the tools, portions of tools, and actions for which you cannot change keyboard shortcuts:

- “Tools for Which You Cannot Customize Keyboard Shortcuts” on page 2-33
- “Actions for Which You Cannot Customize Keyboard Shortcuts” on page 2-34

Tools for Which You Cannot Customize Keyboard Shortcuts

You cannot change the keyboard shortcuts associated with the following tools or portions of tools:

- Figure windows—For example, you cannot modify the keyboard shortcut, **Ctrl+S**, for saving a MATLAB `.fig` file.
- Toolboxes—For example, you cannot modify keyboard shortcuts in the SimBiology[®] desktop.
- Incremental search—You can modify the keyboard shortcuts for initiating a forward or backward incremental search. However, you cannot change the keyboard shortcuts that you use within incremental search mode, such as **Ctrl+Shift+S** to search forward.
- Dialog boxes—For example, you cannot create a keyboard shortcut for the **OK** button.

Actions for Which You Cannot Customize Keyboard Shortcuts

The following table describes some frequently used actions for which you cannot customize keyboard shortcuts.

Action	Keyboard Shortcut
Cancel the current action.	<p>Esc (escape)</p> <p>For example, if you select the Edit menu, the menu items display. Pressing Esc retracts the menu items.</p> <p>In the Function Browser, pressing Esc up to three times has the following effects:</p> <ol style="list-style-type: none"> 1 Dismisses the search history 2 Clears the search field 3 Closes the Function Browser
Interrupt MATLAB execution on all supported platforms.	Ctrl+C
Interrupt MATLAB execution on Windows and UNIX systems.	Ctrl+Cancel
Interrupt MATLAB execution on Macintosh systems.	Cmd+. (period)
Open context menu on Windows and UNIX systems.	Ctrl+Shift+F10
Close the desktop and consequently shut down the MATLAB program. Outside the desktop, close the active window (except on Macintosh platforms).	Alt+F4

Action	Keyboard Shortcut
Accessibility affordances	Tab for navigating through fields in dialog boxes, for example.
Make an open tool the active tool	<ul style="list-style-type: none">• Command Window: Ctrl+0• Command History: Ctrl+1• Current Folder: Ctrl+2• Workspace: Ctrl+3• Profiler: Ctrl+4• Figure Palette: Ctrl+6• Plot Browser: Ctrl+7• Property Editor: Ctrl+8• Editor: Ctrl+Shift+0• Figures: Ctrl+Shift+1• Web browser: Ctrl+Shift+2• Variables Editor: Ctrl+Shift+3• Comparison Tool: Ctrl+Shift+4• Help browser: Ctrl+Shift+5

Set Print Options

In this section...
“Page Setup Options” on page 2-36
“Layout Options for Page Setup” on page 2-36
“Header Options for Page Setup” on page 2-37
“Fonts Options for Page Setup” on page 2-37

Page Setup Options

MATLAB provides special page setup options for printing from the Command Window and Editor. Page setup is not supported in live scripts.

To specify page setup options for printing from the Command Window, right-click in the Command Window, and then select **Page Setup**.

To specify page setup options for printing from the Editor, first select the **Editor** tab. Then, in the **File** section, click **Print** ▾ and select **Page Setup**.

The Page Setup dialog box opens for that tool.

Then, perform these steps:

- 1 Click the **Layout**, **Header**, or **Fonts** tab in the dialog box and set those options for that tool, as detailed in subsequent sections. On Mac platforms, you must first select **MATLAB** in the **Settings** menu in order to see these tabs.
- 2 Click **OK**.
- 3 After specifying the options, select **Print** in the tool you want to print from, for example, the Command Window.

The contents from the tool print, using the options you specified in Page Setup.

Layout Options for Page Setup

You can specify the following layout options. A preview area shows you the effects of your selections.

- **Print header** — Print the header specified in the **Header** pane.

- **Print line numbers** — Print line numbers.
- **Wrap lines** — Wrap any lines that are longer than the printed page width.
- **Syntax highlighting** — For keywords and comments that are highlighted in the Command Window, specify how they are to appear in print. Options are black and white text (that is, no highlighting), colored text (for use with a color printer), or styled text. For styled text, keywords appear in bold, comments appear in italics, and all other text appears in the normal style. Only keywords and comments you input in the Command Window are highlighted; output is not highlighted.

Header Options for Page Setup

If you want to print a header, select the **Layout** tab and then select **Print header**. Next, select the **Header** tab and specify how the elements of the header are to appear. A preview area shows you the effects of your selections:

- **Page number** — Format for the page number, for example # of n
- **Border** — Border style for the header, for example, Shaded box
- **Layout** — Layout style for the header. For example, Standard one line includes the date, time, and page number all on one line

Fonts Options for Page Setup

Specify the font to use for the printed contents:

- 1 From **Choose font**, select the element, either **Body** or **Header**, where **Body** text is everything except the **Header**.
- 2 Select the font to use for the element.

For example, if you access this dialog box while using the Command Window, you can select **Use Command Window font** for **Body** text. The printed text matches the Command Window font.

- 3 Repeat for the other element.

If you did not select **Print header** on the **Layout** pane, you do not need to specify the **Header** font.

As an example, for **Header** text, select **Use custom font** and then specify the font characteristics—type, style, and size. After you specify a custom font, the **Sample** area shows how the font will look.

Tip You can change the font that a desktop tool uses. On the **Home** tab, in the **Environment** section, click **Preferences > Fonts > Custom**.

Web Browsers and MATLAB

In this section...

“About Web Browsers and MATLAB” on page 2-39

“Display Pages in Web Browsers” on page 2-41

“Specify Proxy Server Settings for Connecting to the Internet” on page 2-41

“Specify the System Browser for Linux Platforms” on page 2-42

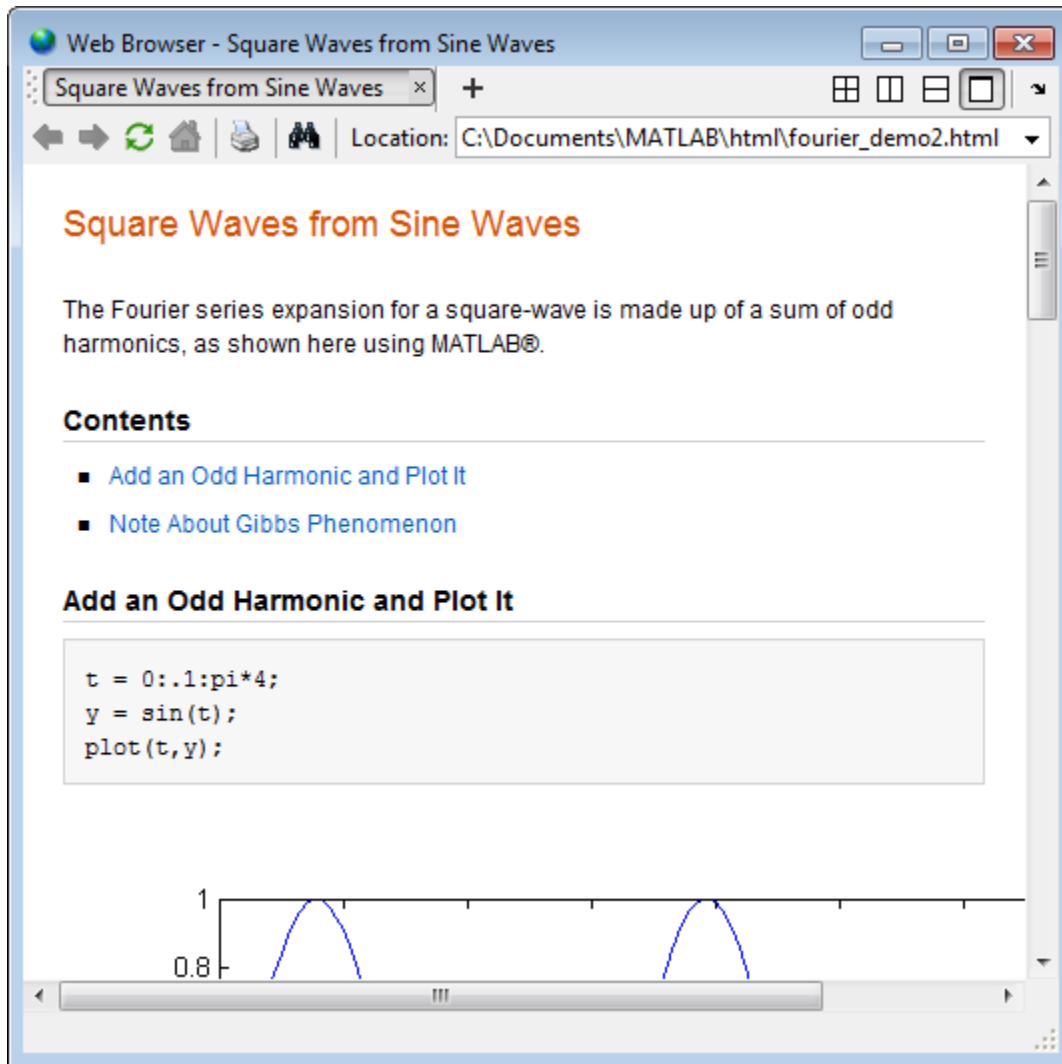
About Web Browsers and MATLAB

From MATLAB, Web sites and documents can display in any of the following browsers:

- MATLAB Web browser
- Help browser
- Your system Web browser, such as Mozilla® Firefox®

MATLAB uses the different browsers to display different types of information:

- Web sites display in your system browser.
- Documentation displays in the Help browser.
- Other HTML files display in the MATLAB Web browser. For example, after publishing a MATLAB program file to HTML, the HTML file displays in the MATLAB Web browser:



MATLAB Web and Help Browsers

The MATLAB Web and Help browsers may not support all the features that a particular Web site or HTML page uses. For example, the MATLAB Web browser does not display .bmp (bitmap) image files. Instead use .gif or .jpeg formats for image files in HTML pages.

System Browser

The system browser that MATLAB uses depends on your platform:

- On Microsoft Windows and Apple Macintosh platforms, MATLAB uses the default browser for your operating system.
- On UNIX platforms, MATLAB uses the Mozilla Firefox browser. You can specify a different system browser for MATLAB using Web preferences.

Display Pages in Web Browsers

To display an HTML document in the MATLAB Web browser, double-click the document name in the Current Folder browser.

To display a Web page or any file type in the MATLAB Web browser:


- 1 Open the browser using the `web` command.
- 2 Type a URL or full path to a filename in the **Location** field.

Specify Proxy Server Settings for Connecting to the Internet

If your network uses a firewall or another method of protection that restricts Internet access, provide information about your proxy server to MATLAB. Be aware that:

- MATLAB supports non-authenticated, basic, digest, and NTLM proxy authentication types.
- You cannot specify the proxy server settings using a script.
- There is no automated way to provide the proxy server settings your system browser uses to MATLAB.

To specify the proxy server settings:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Web**.
- 2 Select the **Use a proxy server to connect to the Internet** check box.
- 3 Specify values for **Proxy host** and **Proxy port**.

Examples of acceptable formats for the host are: `172.16.10.8` and `ourproxy`. For the port, enter an integer only, such as `22`. If you do not know the values for your proxy server, ask your system or network administrator for the information.

If your proxy server requires a user name and password, select the **Use a proxy with authentication** check box. Then enter your proxy user name and password.

Note: MATLAB stores the password without encryption in your `matlab.prf` file.


- 4 Ensure that your settings work by clicking the **Test connection** button.

MATLAB attempts to connect to `http://www.mathworks.com`:

- If MATLAB can access the Internet, **Success!** appears next to the button.
 - If MATLAB cannot access the Internet, **Failed!** appears next to the button. Correct the values you entered and try again. If you still cannot connect, try using the values you used when you authenticated your MATLAB license.
- 5 Click **OK** to accept the changes.

Specify the System Browser for Linux Platforms

To specify the system browser:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Web**.
- 2 Under **System Web browser**, in the **Command** field, specify the system command to open the browser, for example, `opera`, which opens the Opera Web browser.
- 3 Add options for opening your system browser in the **Options** field. For example, `geometry 1064x860` specifies the size of the window for Opera.
- 4 Click **OK**.

Note: The Mac platform does not have a **System Web browser** preference.

Manage Your Licenses

You can use the MATLAB licensing features to perform license management activities, such as activating licenses, deactivating licenses, or updating licenses. You also can visit the License Center at the MathWorks website to perform other license-related activities.

To access the licensing feature:

- 1 On the **Home** tab, in the **Resources** section, click **Help > Licensing**.
- 2 Select a Licensing option. The following table describes the Licensing options. Depending on your license type, your system might not include all these options.

Note: Some options require an internet connection. If your internet connection requires a proxy server, use MATLAB web preferences to specify the server host and port. See “Specify Proxy Server Settings for Connecting to the Internet” on page 2-41 for more information.

Option	Description
Update Current Licenses	Displays a list of all your MathWorks licenses on this computer, with their status. When you select a license and click Update Selected License , MATLAB contacts MathWorks to retrieve the most current version of the License File for the license. The update process overwrites the current License File on your system. You need to restart MATLAB.
Activate Software	Starts the activation application, which walks you through the activation process. Answer the questions on each dialog box, select the license you want to activate, and click Activate .
Deactivate Software	Displays a list of all your MathWorks licenses on this computer, with their status. When you select a license and click Deactivate , MATLAB deactivates all releases on this computer associated with the license, and updates the licensing information at the MathWorks website. You will not be able to use MathWorks software with that license on this computer. If you are not connected to the Internet, MATLAB deactivates the licences on your computer but cannot update the corresponding license information stored at the MathWorks website. In this

Option	Description
	scenario, MATLAB returns a <i>deactivation string</i> . To complete deactivation, save a copy of this string, go to a computer with an internet connection, and visit the License Center at the MathWorks website. There you can log in to your MathWorks Account and enter the deactivation string.
Manage Licenses	Starts a web browser, opening the My Licenses page associated with your MathWorks Account. You can use this page, called the License Center, to perform many licensing activities.

Check for Software Updates

To determine if more recent versions of your MathWorks products are available, and to view latest version numbers for all MathWorks products, follow these steps:

- 1 Make sure that you have an active internet connection.
- 2 On the **Home** tab, in the **Environment** section, select **Add-Ons > Check for Product Updates**. The Check for Updates dialog box displays.
- 3 From the **Select View** list, choose to view the latest version numbers for all MathWorks products installed on your system, or all MathWorks products.

The latest versions display.

- 4 Click any column heading to sort or reverse the sort order by that column.
- 5 To access the release notes for a product, use the What's New column.

Release notes document new features and changes, bug reports, and compatibility considerations.

- 6 Decide whether you want to upgrade to the most recent version.
 - If you do, click **Download Products at MathWorks.com**
 - If you do not, go to step 7.
- 7 Click **Close**.

Macintosh Platform Conventions

In this section...

“Mouse Instructions and Macintosh Platforms” on page 2-46

“Navigating Within the MATLAB Root Folder on Macintosh Platforms” on page 2-46

Mouse Instructions and Macintosh Platforms

The documentation typically presents conventions for Microsoft Windows platforms. Therefore, some conventions and operations differ on the Macintosh platform from those that appear in the rest of the documentation. The intended action for the Macintosh platform is typically obvious. Mouse operations follow Macintosh conventions.

Make the following replacements to adjust documented mouse instructions for Macintosh platforms if you are using a one-button mouse:


- Replace right-click with **Ctrl**+click
- Replace middle-click with **Command**+click

Navigating Within the MATLAB Root Folder on Macintosh Platforms

On Macintosh platforms, MATLAB is installed as an application bundle. The root folder, the string returned by the `matlabroot` command, has a `.app` extension.

To view the contents of the MATLAB root folder in the Mac Finder, right-click the MATLAB application bundle, and then select **Show Package Contents** from the context menu.

To view the content of the MATLAB root folder from within MATLAB:

- 1 On the **Home** tab, in the **File** section, click 
- 2 In the File Browser dialog box, press **Command+Shift+G** to open the Go To Folder dialog box.
- 3 Enter the full path to the MATLAB folder, for example, `/Applications/MATLAB_R2012a.app`.
- 4 Press **Go**.

To open a file with a MATLAB command, such as `edit`, specify the full path of the MATLAB root folder. For example:

```
edit(fullfile(matlabroot, '/toolbox/matlab/demos/lotka.m'))
```

Preferences

In this section...

“Set Preferences for MATLAB” on page 2-48

“Where MATLAB Stores Preferences” on page 2-49

“Preferences MATLAB Uses When Multiple Releases Are Installed” on page 2-49

“General Preferences” on page 2-51

“Confirmation Dialog Boxes Preferences” on page 2-52

“Source Control Preferences” on page 2-54

“Keyboard Shortcuts Preferences” on page 2-54

“Colors Preferences” on page 2-55


“Colors Programming Tools Preferences” on page 2-56

“Comparison Preferences” on page 2-57

“Toolbars Preferences” on page 2-58

Set Preferences for MATLAB

MATLAB provides various options called *preferences* for customizing MATLAB. To access and set preferences:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**.
- 2 From the left pane of the Preferences dialog box, select a tool, product, or an entry revealed when you click an arrow preceding a tool or product name.
- 3 Change settings in the right pane of the Preferences dialog box.
- 4 Click **Apply** or **OK**.

Preferences take effect immediately. They remain persistent across sessions of MATLAB.

Function Alternative

Open the Preferences dialog box using the `preferences` function.

Where MATLAB Stores Preferences

MATLAB and other MathWorks products store their preferences in the file `matlab.prf`. This file loads when you start MATLAB. The folder containing this file is called the preferences folder. The preference folder also contains other related files.

Path to and File Name for the Preferences Folder

To see the full path for the folder where `matlab.prf` and related files are located, type `prefdir` in the MATLAB Command Window. The name of the preferences folder matches the name of the release. For instance, for MATLAB R2016b, the name of the preferences folder is `R2016b`.

On Mac OS X and iOS, the folder might be in a hidden folder. If so, to access the hidden folder:

- 1 In the Apple Mac OS Finder tool, select **Go > Go to Folder**.
- 2 In the resulting dialog box, type the path returned by `prefdir`, and then press **Enter**.

Effects of Changing Preferences

When you change preferences using the MATLAB Desktop, it updates `matlab.prf`. When you close MATLAB, it saves those changes to `matlab.prf`.

Effects of Installation and Deinstallation on the Preferences Folder

Installing MATLAB has no effect on the preferences folder. That is, MATLAB creates, checks, copies, and writes to the preferences folder when you start up MATLAB, not when you install it. When you uninstall MATLAB, there is an option in the uninstaller to remove the preferences folder. However, this option is not selected by default.

Preferences MATLAB Uses When Multiple Releases Are Installed

The files in the preferences folder that MATLAB uses depends on the version of MATLAB you are starting up. How and if MATLAB migrates (reuses) preferences files from one version of MATLAB to the next also depends on the version.

Process for Creating and Migrating Preferences Folder and Files

When you start up, MATLAB looks for a preferences folder name that matches the release starting up, and then it does one of the following:

- If MATLAB finds a preferences folder name matching the release starting up, then it uses that folder and the files within it. This is usually the case after the first time you start MATLAB.

If a preference folder exists but is empty, then MATLAB recreates the default preference files for the release starting up.

- If MATLAB does not find a preferences folder name matching the release starting up, then it creates one. Next, MATLAB checks to see if a preference folder exists for any of the three releases of MATLAB that immediately precede the one you are starting.
 - If none of the preference folders from the previous three releases exist, then MATLAB creates the default preference files for the release starting up.

For example, if you start up R2016a and neither R2015b, R2015a, or R2014b are installed, then MATLAB creates the default files for the R2016a release. This is true even if a preference folder exists for the R2014a release or earlier.

- If one or more of the preference folders for the previous three releases exist, then MATLAB migrates the files from the preferences folder corresponding to the latest previous release to the preferences folder for the release starting up.

For example, if you start up R2016a and a preference folder exists for both the R2015b and R2014b releases, then MATLAB migrates the files from the R2015b preferences folder to the R2016a preferences folder.

Control Preferences Files MATLAB Uses


This table describes how to control which versions of preferences files MATLAB uses.

To Use:	Do This:
Default preference files for a given release of MATLAB	<ul style="list-style-type: none"> • If <i>none</i> of the preference folders for the three releases immediately preceding the given release exist, do nothing. • If one or more of the preference folders for the previous three releases exist, make sure that the preferences folder for the given release exists, but is empty before starting up.
All the preference files from a release of MATLAB up to three releases preceding the release you plan to start up	Ensure that the preferences folder exists for that previous release. Delete the entire preferences folder for each release after the

To Use:	Do This:
	release you want to migrate from, including the folder for the release of MATLAB you plan to start up.
The release-specific default for just a particular file in the preferences folder	Delete just that file from the preferences folder for the release of MATLAB you plan to start up. One file to consider keeping is <code>history.m</code> . For more information, see “Command History” on page 3-28.

General Preferences

You can set preferences for the initial working folder, deleting files, and toolbox path caching.


On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > General**. Then, adjust preference options as described in this table.

Preference	Usage
Initial working folder	<p>Select an option to specify the current folder in MATLAB when it starts.</p> <p>If you select Location based on MATLAB startup rules, MATLAB determines the initial working folder based on how you started MATLAB and on the <code>userpath</code>. This is the default option. For details, see “MATLAB Startup Folder” on page 1-16.</p> <p>Alternatively, select the last working folder from your previous MATLAB session, or specify the full path to a folder on your system.</p> <p>If you create a <code>startup.m</code> file, the commands in that file can override the preference option.</p>
Deleting files	Select an option to specify what MATLAB does with files you delete using the <code>delete</code> function.

Preference	Usage
	<p>Selecting Delete permanently makes the <code>delete</code> function run faster.</p> <p>On Linux systems, if you select Move to a temporary folder, MATLAB moves files to a subfolder with the prefix <code>MATLAB_Files_</code> in the system temporary folder, as returned by the <code>tempdir</code> function.</p>
Toolbox path caching	<p>Select Enable toolbox path caching to have MATLAB cache toolbox folder information across sessions for quicker startup performance.</p> <p>Select Enable toolbox path cache diagnostics to display information about startup time when you start MATLAB.</p> <p>Click Update Toolbox Path Cache to add files to the toolbox folders under the <code>matlabroot</code> folder. (Use after you use tools not provided with MATLAB to create MATLAB files.)</p> <p>For details, see “Toolbox Path Caching in MATLAB” on page 1-24.</p>
Desktop language (selected non-English systems only)	Select the language in which the MATLAB desktop appears. This option affects the text in dialog boxes, button names, menu items, and error and warning messages.

Confirmation Dialog Boxes Preferences

You can specify whether MATLAB displays specific confirmation dialog boxes.


On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > General > Confirmation Dialogs**. Then, adjust preference options as described in the table below.

This table summarizes the core MATLAB confirmation dialog boxes. There might be additional confirmation dialog boxes for other products you install.

Option	Confirmation Dialog Box Appears
Warn before deleting Command History items	When you delete entries from the Command History window. For details, see “Use Command History Commands” on page 3-29.
Warn before clearing the Command Window	When, on the Home tab, in the Code section, you click Clear Commands . Does not appear when you use the <code>clc</code> function.
Confirm when overwriting variables in MAT-files	When you save variables by dragging them from the Workspace browser onto a MAT-file in the Current Folder browser.
Confirm when overwriting workspace variables via drag-and-drop	When you load variables by dragging them from the Details Panel of the Current Folder browser to the Workspace browser or Command Window.
Prompt when editing files that do not exist	When you type <code>edit filename</code> and <code>filename</code> does not exist in the current folder or on the search path.
Prompt to exit debug mode when saving file	When you try to save a modified file while in debug mode. For details, see “End Debugging Session”.
Prompt to save on activate	When you have unsaved changes to a figure and program file and you activate the UI by clicking the Run button, for example. For details, see “GUIDE Preferences”.
Prompt to save on export	When you have unsaved changes to a figure and program file and you select File > Export . For details, see “GUIDE Preferences”.
Confirm changing default callback implementation	When you have modified a callback signature in GUIDE. For details, see “GUIDE Preferences”.
Confirm before exiting MATLAB	When you quit MATLAB.
Confirm when deleting variables	When you delete variables from the workspace using menu items. Does not appear with the <code>clear</code> function. For details, see “Save and Load Workspace Variables” on page 5-15.

Source Control Preferences


You can select which previously installed and configured source control system to use with MATLAB.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > General > Source Control**. Then, select an option from the list.


For more information, see “Select or Disable Source Control System”.


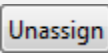
Keyboard Shortcuts Preferences

You can set keyboard shortcuts for actions you perform using MathWorks software. You can specify or import sets of predefined keyboard shortcuts, set individual shortcuts on an action-by-action basis, or use a combination of both approaches.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard > Shortcuts**. Then, adjust preference options as described in the table below.


For step-by-step instructions, see “Customize Keyboard Shortcuts” on page 2-22.

Preference	Usage
Active settings	<p>Select or import a set of predefined keyboard shortcuts.</p> <p>For details, see “Choose a Set of Keyboard Shortcuts” on page 2-15 and “Use Keyboard Shortcuts Settings Files Created on Other Systems” on page 2-33.</p>
	<p>Select any one of these options:</p> <ul style="list-style-type: none"> • Save As—Save active settings to a file. • Copy to clipboard— so you can import into Microsoft Excel, for example. <p>For details, see, “Display Keyboard Shortcuts” on page 2-19.</p>

Preference	Usage
	<ul style="list-style-type: none"> • Compare active settings to another set. For details, see, “Compare Sets of Keyboard Shortcuts” on page 2-18. • Undo Modifications to a default keyboard shortcut set. • Delete a set of keyboard shortcuts you previously saved or added. For details, see “Delete a Set of Keyboard Shortcuts” on page 2-32.
Search by action name or shortcut	Search the list of displayed actions.
Shortcuts for <action-name>	View the keyboard shortcut assigned to a selected action.
	<p>Add or delete a keyboard shortcut to a selected action.</p> <p>For details, see, “Examples of Creating, Modifying, and Deleting Keyboard Shortcuts” on page 2-29.</p>
All possible conflicts	<p>Display conflicts when two or more different actions have the same shortcut.</p> <p>For details, see “Evaluate and Resolve Keyboard Shortcut Conflicts” on page 2-27.</p>
	<p>Remove the keyboard shortcut from the selection in the All possible conflicts list.</p> <p>For details, see “Evaluate and Resolve Keyboard Shortcut Conflicts” on page 2-27.</p>

Colors Preferences


You can specify the text and background color for desktop tools, as well as colors for highlighting syntax elements of MATLAB code.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Colors**. Then, set options as described in the table below.

Preference	Usage
Desktop tool colors	<p>Specify that desktop tools use the same text and background colors that your platform uses for other applications by selecting Use system colors.</p> <p>Customize colors by clearing Use system colors, and then choose Text and Background colors from the drop-down menus.</p> <p>These colors do not apply to the Help display pane, nor to the web browser.</p> <p>For details, see “Changing Text and Background Colors in Desktop Tools” on page 2-6.</p>
MATLAB syntax highlighting colors	<p>Set colors to help you quickly identify elements of MATLAB syntax in the Editor, Command Window, Command History window, and the MATLAB shortcuts callback area.</p> <p>For details, see “Changing Syntax Highlighting Colors” on page 2-6.</p>
MATLAB Command Window colors	<p>Set colors to help you quickly identify errors, warnings, and hyperlinks in the Command Window.</p>

Colors Programming Tools Preferences

You can specify options used for editing and debugging code, including code analysis colors, variable and function colors, and cell display options.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Colors > Programming Tools**. Then, set options as described in the table below.


Preference	Usage
Code analyzer colors	<ul style="list-style-type: none"> • Warnings—Specifies the color Code Analyzer uses to identify code in the Editor for which there are warning messages.

Preference	Usage
	<ul style="list-style-type: none"> • Autofix highlight—Specifies the color Code Analyzer uses to identify code in the Editor for which there is an automatic fix. <p>For details, see “Automatically Check Code in the Editor — Code Analyzer”.</p>
Variable and function colors	<ul style="list-style-type: none"> • Automatically highlight—Specifies the color the Editor uses to highlight all occurrences of a specific variable or function. For details, see “Find and Replace Functions or Variables in the Current File”. • Variables with shared scope—Specifies the color of variables with shared scope. The text is colored, not shaded. For details, see “Check Variable Scope in Editor”
Section display options	<p>Highlight sections—Specifies the color the Editor uses to shade code sections.</p> <p>Show lines between sections—Specifies that code section divisions appear with a gray line between each section in the Editor. These lines do not appear in the published or printed file.</p> <p>See also “Run Code Sections”.</p>

Comparison Preferences

Colors

You can change and save your diff color preferences for the Comparison tool. You can apply your color preferences to all comparison types.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Comparison**. Then, set options as described in the table below.

Preference	Usage
Colors	<p>Set colors to help you quickly identify differences, modifications, and merges in comparison reports. Choose colors from the drop-down menus. View the colors in the Sample pane. To use your modified settings in comparisons, click Apply. Refresh any open comparison reports to use the new colors.</p>


Preference	Usage
Active Settings	<p>To save your modified color preferences for use in future MATLAB sessions, click Save As. Enter a name for your color settings profile and click OK.</p> <p>After saving settings, you can select them in the Active Settings list.</p> <p>For details, see “Change Color Preferences” on page 6-23.</p>

External Source Control Integration

Use the check box to control external source control interactions: **Allow external source control tools to use open MATLAB sessions for diffs and merges**. After you configure your source control tool to use MATLAB Comparison tool, the Comparison tool prompts you to set this preference. For details, see “Customize External Source Control to Use MATLAB for Diff and Merge”.

Toolbars Preferences

You can customize some toolbars in the MATLAB application.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Toolbars**. Then, set options as described in the table below.

For step-by-step instructions on setting these preferences, see “Access Frequently Used Features” on page 2-9.


Preference	Usage
Toolbar	Select the toolbar you want to customize.
Layout	Rearrange controls in the toolbar by dragging and dropping them to a new location in the Layout .
Controls	Select which buttons appear on the selected toolbar.

Web Preferences

Web preferences enable you to specify Internet connection information to MATLAB.

Limitations

- MATLAB supports nonauthenticated, basic, digest, and NTLM proxy authentication types.
 - You cannot specify proxy server settings using a script.
 - There is no automated way to provide MATLAB with the proxy server settings that your system browser uses.
-

You can set Web preferences on the **Home** tab, in the **Environment** section. Click  **Preferences**. Select **MATLAB > Web**, and then adjust preference options as described in the table below.

Preference	Usage
Use a proxy server to connect to the Internet	Provide information that MATLAB needs to access the internet when your network uses a firewall or another method of protection that restricts Internet access.
Proxy host	Specify a value for the Proxy host . For example, <code>172.16.10.8</code> or <code>ourproxy</code> . If you do not know the values for your proxy server, ask your system or network administrator for the information.
Proxy port	Specify an integer value for the Proxy port . For example, <code>22</code> . If you do not know the values for your proxy server, ask your system or network administrator for the information.
Use a proxy with authentication	Specifies that your proxy server requires a user name and password.
Proxy username	Specify the proxy server user name.
Proxy password	Specify the proxy server password. Note: MATLAB stores the password without encryption in your <code>matlab.prf</code> file.

Preference	Usage
Test connection	<p>Ensure that your settings work.</p> <p>If MATLAB cannot access the Internet, Failed! appears next to the button. Correct the values you entered and try again. If you still cannot connect, try using the values you used when you authenticated your MATLAB license.</p>
System Web browser UNIX platforms only — excluding Macintosh	<ul style="list-style-type: none">• Command—Specifies the system command to open the browser. For example, <code>opera</code>, opens the Opera Web browser.• Options—Specifies options for the system browser. For example, <code>geometry 1064x860</code> specifies the size of the window for Opera.

Entering Commands

- “Enter Statements in Command Window” on page 3-2
- “Find Functions to Use” on page 3-4
- “Format Output” on page 3-7
- “Stop Execution” on page 3-11
- “Find Text in Command Window or History” on page 3-12
- “Create Shortcuts to Rerun Commands” on page 3-15
- “Set Command Window Preferences” on page 3-17
- “Set Keyboard Preferences” on page 3-19
- “Check Syntax as You Type” on page 3-22
- “Command History” on page 3-28

Enter Statements in Command Window

As you work in MATLAB, you can enter individual statements in the Command Window. For example, create a variable named `a` by typing this statement at the command line:

```
a = 1
```

MATLAB immediately adds variable `a` to the workspace and displays the result in the Command Window.

```
a =  
    1
```

When you do not specify an output variable, MATLAB uses the variable `ans`, short for *answer*, to store the results of your calculation.

```
sin(a)  
ans =  
    0.8415
```

The value of `ans` changes with every command that returns an output value that is not assigned to a variable.

If you end a statement with a semicolon, MATLAB performs the computation, but suppresses the display of output in the Command Window.

```
b = 2;
```

To enter multiple statements on multiple lines before running any of the statements, use **Shift+Enter** between statements. This action is unnecessary when you enter a paired keyword statement on multiple lines, such as `for` and `end`.

You also can enter more than one statement on the same line by separating statements. To distinguish between commands, end each one with a comma or semicolon. Commands that end with a comma display their results, while commands that end with a semicolon do not. For example, enter the following three statements at the command line:

```
A = magic(5), B = ones(5) * 4.7; C = A./B  
A =  
    17    24     1     8    15
```

```
23     5     7     14     16
  4     6     13     20     22
 10     12    19     21     3
 11     18    25     2     9
```

```
C =
 3.6170    5.1064    0.2128    1.7021    3.1915
 4.8936    1.0638    1.4894    2.9787    3.4043
 0.8511    1.2766    2.7660    4.2553    4.6809
 2.1277    2.5532    4.0426    4.4681    0.6383
 2.3404    3.8298    5.3191    0.4255    1.9149
```

MATLAB displays only the values of **A** and **C** in the Command Window.

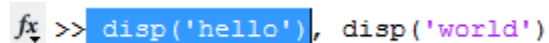
To recall previous lines in the Command Window, press the up- and down-arrow keys, \uparrow and \downarrow . Press the arrow keys either at an empty command line or after you type the first few characters of a command. For example, to recall the command `b = 2`, type `b`, and then press the up-arrow key.

To clear a command from the Command Window without executing it, press the Escape (**Esc**) key.

You can evaluate any statement already in the Command Window. Select the statement, right-click, and then select **Evaluate Selection**.

In the Command Window, you also can execute only a portion of the code currently at the command prompt. To evaluate a portion of the entered code, select the code, and then press **Enter**.


For example, select a portion of the following code:

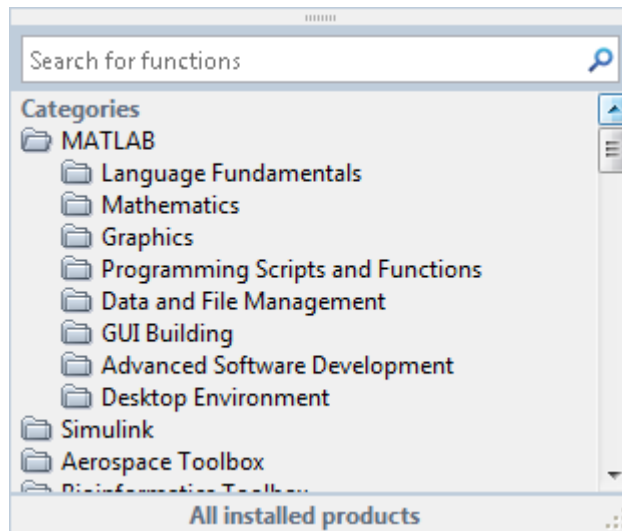
```
 fx >> disp('hello'), disp('world')
```

```
hello
```

Find Functions to Use

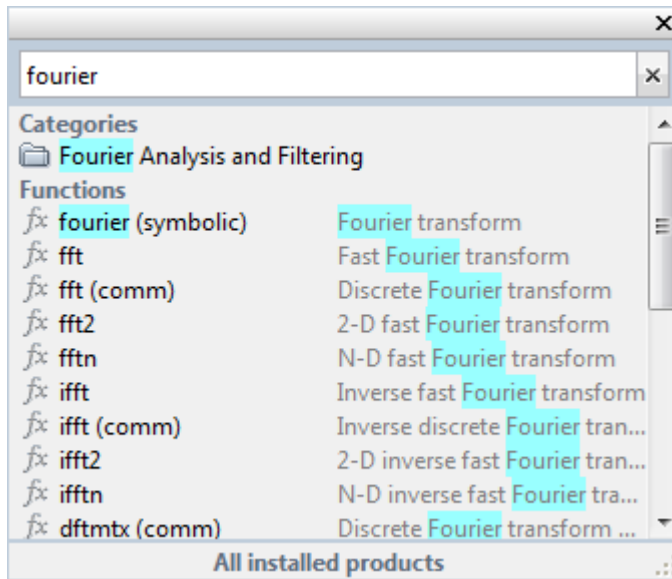
This example shows how to find the name and description of a MathWorks function from the Command Window or Editor using the Function browser. The Function browser is not supported in live scripts.

- 1 Click the Browse for functions button, . In the Command Window, this button is to the left of the prompt. In the Editor, the button is on the **Editor** tab, in the **Edit** section. The Function browser opens.



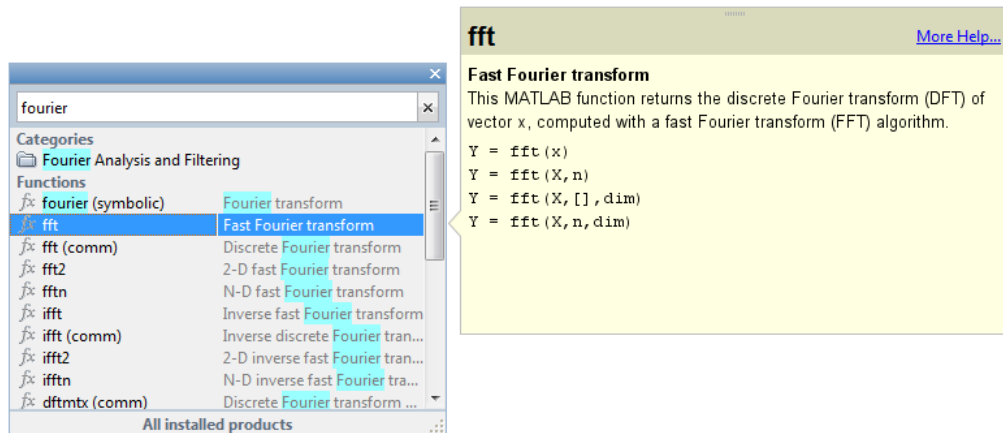
Tip The Function browser closes when you move the pointer outside of it. To keep the browser open, drag it by the top edge to a different location.

- 2 Optionally, select a subset of products to display in the list. Click the product area at the bottom of the browser (where the text **All installed products** appears by default), and then set the **Selected Products** preference and click **OK**. This preference also applies to the Help browser.
- 3 Find functions by browsing the list or by typing a search term. For example, search for the term *fourier*.



In the search results, a parenthetical term after a function name indicates either that the function is in a product folder other than MATLAB, or that there are multiple functions with the same name. For example, `fft (comm)` corresponds to the `fft` function in the Communications System Toolbox™ folder.

- 4 Select a function that you would like to use or learn more about, as follows.
 - Insert the function name into the current window by double-clicking the name. Alternatively, drag and drop the function name into any tool or application.
 - View syntax information for the function by single-clicking its name. A brief description for each of the syntax options displays in a yellow pop-up window.



Tip The pop-up window automatically closes when you move your pointer to a new item in the results list. To keep the pop-up window open, drag it by the top edge to a different location.

You can change the font that the Function browser uses by setting preferences. On the **Home** tab, in the **Environment** section, select **Preferences > Fonts**. By default, the Function browser uses the desktop text font and the pop-up window uses the Profiler font.

Format Output

MATLAB displays output in both the Command Window and the Live Editor. You can format the output display using several provided options.

In this section...

“Format Line Spacing in Output” on page 3-7

“Format Floating-Point Numbers” on page 3-8

“Wrap Lines of Code to Fit Window Width” on page 3-9

“Suppress Output” on page 3-9

“View Output by Page” on page 3-9

“Clear the Command Window” on page 3-10

Format Line Spacing in Output

By default, MATLAB displays blank lines in command window output.

You can select one of two numeric display options in MATLAB.

- **loose** — Keeps the display of blank lines (default).

```
>> x = [4/3 1.2345e-6]
```

```
x =
```

```
    1.3333    0.0000
```


- **compact** — Suppresses the display of blank lines.

```
>> x = [4/3 1.2345e-6]
```

```
x =
```

```
    1.3333    0.0000
```

To format the output display, do one of the following:

- On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command Window**, and then choose a **Numeric display** option.
- Use the `format` function at the command line, for example:

```
format loose
```

```
format compact
```

Note: Line spacing display options do not apply in the Live Editor.

Format Floating-Point Numbers


You can change the way numbers display in both the Command Window and the Live Editor. By default, MATLAB uses the short format (5-digit scaled, fixed-point values).

For example, suppose that you enter `x = [4/3 1.2345e-6]` in the Command Window. The MATLAB output display depends on the format you selected. This table shows some of the available numeric display formats, and their corresponding output.

Numeric Display Format	Example Output
short (default)	x = 1.3333 0.0000
short e	x = 1.3333e+00 1.2345e-06
long	x = 1.3333333333333333 0.000001234500000
+	x = ++

Note: The text display format affects only how numbers are shown, not how MATLAB computes, or saves them.

To format the way numbers display, do one of the following:


- On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command Window**, and then choose a **Numeric format** option.
- Use the `format` function, for example:

```
format short
format short e
format long
```

See the `format` reference page for a list and description of all supported numeric formats.

Wrap Lines of Code to Fit Window Width

A line of code or its output can exceed the width of the Command Window, requiring you to use the horizontal scroll bar to view the entire line. To break a single line of input or output into multiple lines to fit within the current width of the Command Window:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command Window**.
- 2 Select **Wrap Lines**.
- 3 Click **OK**.

Note: Line wrapping options do not apply in the Live Editor.

Suppress Output

To suppress code output, add a semicolon (;) to the end of a command. This is useful when code generates large matrices.

Running the following code creates A, but does not show the resulting matrix in the Command Window or the Live Editor:

```
A = magic(100);
```

View Output by Page

Output in the Command Window might exceed the visible portion of the window. You can view the output, one screen at a time:

- 1 In the Command Window, type `more on` to enable paged output.
- 2 Type the command that generates large output.
- 3 View the output:
 - Advance to the next line by pressing **Enter**.
 - Advance to the next page by pressing **Space Bar**.
 - Stop displaying the output by pressing **q**.

To disable paged output, type `more off`.

Note: Paged output options do not apply in the Live Editor.

Clear the Command Window

If the Command Window seems cluttered, you can clear all the text (without clearing the workspace) by doing one of the following:

- On the **Home** tab, in the **Code** section, select **Clear Commands > Command Window** to clear the Command Window scroll buffer.
- Use the `clc` function to clear the Command Window scroll buffer.
- Use the `home` function to clear your current view of the Command Window, without clearing the scroll buffer.

See Also

`clc` | `format` | `home` | `more`

Stop Execution

To stop execution of a MATLAB command, press **Ctrl+C** or **Ctrl+Break**.

On Apple Macintosh platforms, you also can use **Command+.** (the Command key and the period key).

Ctrl+C does not always stop execution for files that run a long time, or that call built-ins or MEX-files that run a long time. If you experience this problem, include a `drawnow`, `pause`, or `getframe` function in your file, for example, within a large loop.

Also, **Ctrl+C** might be less responsive if you start MATLAB with the `-nodesktop` option.

Note: For certain operations, stopping the program might generate errors in the Command Window.

See Also

`drawnow` | `getframe` | `pause`

Find Text in Command Window or History

In this section...

“Find Text in the Command Window” on page 3-12


“Find Text in the Command History Window” on page 3-14

Find Text in the Command Window

You can search text currently in the Command Window. This includes text that is visible on the screen, and text that is in the scroll buffer.


- “Search Using Find Dialog Box” on page 3-12
- “Search Using Keyboard Shortcuts” on page 3-12

Search Using Find Dialog Box

To search for specified text in the Command Window, on the Command Window title bar, click , and then select **Find**. The Find dialog box opens. The search begins at the current cursor position. MATLAB finds the text you specified and highlights it.

MATLAB beeps when a search for **Find Next** reaches the end of the Command Window, or when a search for **Find Previous** reaches the top of the Command Window. If you have **Wrap around** selected, MATLAB continues searching after beeping.

To search for the specified text in other MATLAB desktop tools, change the selection in the **Look in** field.

You can increase the amount of information available in the Command Window so that more text is available for searching. Doing so requires more memory. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command Window**, and then increase the setting for **Number of lines in the command window scroll buffer**.

Clearing the command window (for example, with the `clc` function), empties the scroll buffer. The cleared text is no longer available for searching. To clear your display in the Command Window without clearing the buffer, use the `home` function.

Search Using Keyboard Shortcuts

You can also perform an incremental search in the Command Window using keyboard shortcuts.

- 1 Begin an incremental search by using one of the defined keyboard shortcuts.

Action	Windows Default Shortcut	Macintosh or Emacs Default Shortcut
Initiate a forward incremental search.	Ctrl+Shift+S	Ctrl+S
Initiate a backward incremental search.	Ctrl+Shift+R	Ctrl+R

An incremental search field appears in the bottom right corner of the MATLAB Desktop window. For a forward search, the text **F Inc Search** appears. The **F** indicates a forward search.

- 2 Begin typing your search term.

When you enter lowercase letters in the incremental search field, MATLAB looks for both lowercase and uppercase instances of the letters. For example, if you enter **b**, MATLAB looks for **b** and **B**. However, if you enter uppercase letters, MATLAB only looks for instances that match the case you entered.

- 3 Perform incremental search actions using these keyboard shortcuts:

Action	Keyboard Shortcut
Complete a partially highlighted set of characters.	Ctrl+W
Find the next occurrence of a set of characters.	Ctrl+S
Remove characters from the incremental search field, back to the last successful search	Ctrl+G


If you search for a set of characters that does not appear in the Command Window text, **Failing** appears in the incremental search field.

- 4 End incremental searching by pressing **Esc** (escape), **Enter**, or any other key that is not a character or number.

The incremental search field disappears. The cursor remains at the position where the text was last found, with the search text highlighted.

Find Text in the Command History Window




You can search for text in the Command History Window. You can search for text either at the beginning of a command, or anywhere within a command.

- 1 In the Command History window, type in the Search field. To display the Search field if it is not visible, click , and then select **Find**.



- 2 Begin typing your search term.

The Command History window searches backward and selects the previous entry that contains the sequence of letters you typed.

- 3 Select from the different search options using the buttons to the right of the search field. Options include  (match case),  (match anywhere within command), and  (match at beginning of command).
- 4 Find the previous or next occurrence of the entry with the up and down arrow keys, respectively.
- 5 Press **Esc** to clear the search.

Create Shortcuts to Rerun Commands

This example shows how to create, run, edit, and organize MATLAB shortcuts. A MATLAB shortcut is an easy way to run a group of MATLAB language statements that you use regularly. For example, use a shortcut to set up your environment when you start working, or to set the same properties for figures you create.

- 1 On the **Home** tab, click **New**, and then select **Command Shortcut**.

If the **Shortcuts** tab is on the desktop, you can also click **New Shortcut** in the **Manage** section.

- 2 Complete the Shortcut Editor dialog box:

- 1 In the **Label** field, enter a name for the shortcut.

For this example, enter `my_Shortcut`.

- 2 In the **Callback** field, type statements you want the shortcut to run.

You also can drag and drop statements from the Command Window, Command History Window, or a file.

For this example, enter these statements:

```
format compact
clear
workspace
filebrowser
clc
```

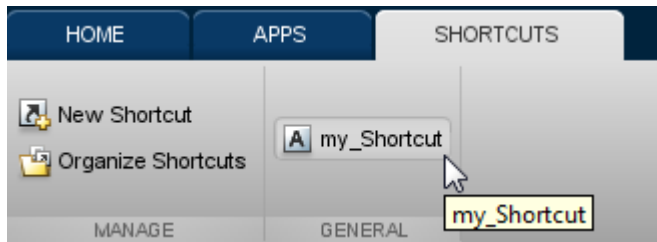
Tip If command prompts (`>>`) appear, MATLAB automatically removes them from the **Callback** field when you save the shortcut.

- 3 In the **Category** field, type the name of a new category or select an existing category from the drop-down list. If you leave this field blank, the shortcut appears in the **General** section of the toolbar.
- 4 In the **Icon** field, select an icon.
- 5 Select both the **Add to quick access toolbar** and **Show label on quick access toolbar** options.
- 6 Click **Save**.

The shortcut icon and label appear on the quick access toolbar. If the shortcut icon does not appear on the quick access toolbar, use the drop-down to see the full list.

To organize and edit shortcuts, on the **Shortcuts** tab, in the **Manage** section, click **Organize Shortcuts** to open the Shortcuts Organizer dialog box. If the **Shortcuts** tab is not visible, go to the **Home** tab, and in the **Environment** section, click **Layout**. Then, under **Show**, select **Shortcuts Tab**.


- 3 Run a shortcut by clicking its icon on the **Shortcuts** tab.



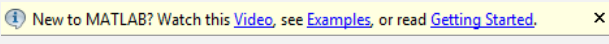

All the statements in the shortcut **Callback** field execute as if you ran those statements from the Command Window, although they do not appear in the Command History window.

Set Command Window Preferences

You can customize the visual display of the Command Window and command output within it.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command Window**, and then adjust preference options as described in the table below.


Preference	Usage
Text display	Select a Numeric format option to specify the output format of numeric values in the Command Window. For details, see “Format Floating-Point Numbers” on page 3-8.
	Select a Numeric display option to specify whether blank lines appear in Command Window output. To suppress blank lines, select compact . To display blank lines, select loose .
Datetime format	Select a Locale option to specify the default input locale of the Datetime object. You also can enter a custom locale. For more information, including a list of common values, see datetime .
	Select a Default date and time format option to specify the default format of the Datetime object. You also can enter a custom format. For more information, see datetime Properties .
	Select a Default date-only format option to specify the default date format of the Datetime object. You also can enter a custom format. For more information, see datetime Properties .
Display	Select Wrap lines to make each line of input or output in the Command Window break into multiple lines to fit within the current width of the Command Window. For details, see “Wrap Lines of Code to Fit Window Width” on page 3-9.
	Select Set matrix display width to eighty columns to limit the width of matrix output.

Preference	Usage
	<p>Note: If you also select Wrap lines, and the width of the Command Window is fewer than 80 characters, each row of 80 characters of matrix output wraps to fit within the width of the Command Window.</p> <hr/> <p>Select Show getting started message bar to display the Command Window message bar that provides links to introductory information.</p>  <p>Select Show function browser button to display the Function Browser button  to the left of the prompt in the Command Window. You can use the Function Browser to search for MATLAB functions.</p> <p>Select Suggest corrections for mistyped functions and variables to display suggestions in the Command Window. If you enter an undefined function, variable name, or MATLAB operator, MATLAB displays:</p> <p>Did you mean:</p> <p>followed by a suggested command at the command line. You can press Enter to execute that command, or Esc to delete the suggestion.</p> <p>Number of lines in command window scroll buffer specifies the maximum number of lines displayed in the Command Window. A larger scroll buffer provides a larger base for search features, but requires more memory. By default, the scroll buffer is set to 5,000 lines.</p> <p>The scroll buffer size does not impact the number of lines you can recall. By default, you can use the up arrow key ↑ to recall all lines shown in the Command History window, regardless of how many lines you can see in the Command Window.</p>
<p>Tab key</p>	<p>Tab size specifies the number of spaces assigned to the tab key.</p> <hr/> <p>Note: This setting does not apply if you have enabled tab completion. To change tab completion settings, on the Home tab, select Preferences > Keyboard.</p>

Set Keyboard Preferences

Keyboard preferences enable you to set tab completion, function hints, and delimiter matching in the Command Window and Editor.

Note: Keyboard preference changes do not apply in live scripts.

To set Keyboard Preferences, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard**, and then adjust preference options as described in this table.

Preference	Usage
Tab completion	Select the tool or tools in which you want the Tab key to complete names known to MATLAB after you type the first few letters of the name.
	For details, see “Tab Completion” on page 3-23. Select Tab key narrow completions to have MATLAB continue to reduce the list of possible names for completion as you type each additional character and press the Tab key.
Function hints	Specify the selected tool or tools that you want to display syntax function hints. When enabled, if you type a function name with an opening parenthesis, and then pause, a tooltip opens showing the basic syntax for the function. For example:

Preference	Usage
	<pre>x = edit(edit('fun.m') edit('file.ext') edit('fun1','fun2','fun3',...) edit('classname/fun') edit('private/fun') edit('classname/private/fun') edit('+packagename/classname/fun') edit('my file.m')</pre> <p style="text-align: right;">More Help...</p> <p>For details, see “Function Syntax Hints” on page 3-26.</p>
<p>Delimiter Matching</p>	<p>Specify when and if MATLAB alerts you to matched and mismatched delimiters. Delimiters include parentheses, brackets, braces, and, in the Editor only, paired keywords.</p> <p>If you select Match while typing, MATLAB alerts you to matched and mismatched delimiters as you type.</p> <p>If you select Match on arrow key, MATLAB alerts you to matched and mismatched delimiters when you move the cursor over a delimiter using an arrow key.</p> <p>For details, see “Delimiter Matching” on page 3-23.</p> <p>Select one of these Show match with options to specify how MATLAB indicates matching delimiters:</p> <ul style="list-style-type: none"> • Balance — The corresponding delimiter highlights briefly (default). • Underline — Both delimiters in the pair display underlines briefly. • Highlight — Both delimiters in the pair highlight briefly.

Preference	Usage
	<p>Select one of these Show mismatch with options to specify how MATLAB indicates mismatched delimiters</p> <ul style="list-style-type: none"><li data-bbox="501 390 842 418">• Beep — MATLAB beeps.<li data-bbox="501 430 1317 493">• Strikethrough — The delimiter you type appears crossed out briefly (default).<li data-bbox="501 505 851 532">• None — There is no alert.

Check Syntax as You Type

In this section...

“Syntax Highlighting” on page 3-22

“Delimiter Matching” on page 3-23

“Tab Completion” on page 3-23

“Function Syntax Hints” on page 3-26

Syntax Highlighting


To help you identify MATLAB elements, some entries appear in different colors in the Command Window and the Editor. This is known as *syntax highlighting*. By default:

- Keywords are blue.
- Character vectors are purple.
- Unterminated character vectors are maroon.
- Comments are green.

```
if A > B
'greater'
elseif A < B
'less'
end
```

Except for errors, output in the Command Window does *not* appear with syntax highlighting.

When you paste or drag a selection from the Editor to another application, such as Microsoft Word, the pasted text maintains the syntax highlighting colors and font characteristics from the Editor. MATLAB software pastes the selection to the Clipboard in RTF format, which many Microsoft Windows and Macintosh applications support.

You can change syntax highlighting preferences. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Editor/Debugger > Languages**. Preference changes do not apply in live scripts.

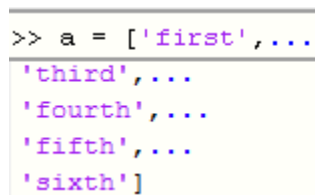
Delimiter Matching

MATLAB indicates matched and mismatched delimiters, such as parentheses, brackets, and braces, to help you avoid syntax errors. MATLAB also indicates paired language keywords, such as `for`, `if`, `while`, `else`, and `end` statements.

By default, MATLAB indicates matched and mismatched delimiters and paired language keywords as follows:

- Type a closing delimiter—MATLAB briefly highlights or underlines the corresponding opening delimiter.
- Type more closing delimiters than opening delimiters—MATLAB puts a strike line through or underlines the unmatched delimiter.
- Use the arrow keys to move the cursor over one delimiter—MATLAB briefly underlines both delimiters in a pair. If no corresponding delimiter exists, MATLAB puts a strike line through the unmatched delimiter.

If a matching delimiter exists, but it is not visible on the screen, a pop-up window appears and shows the line containing the matching delimiter. Click in the pop-up window to go to that line.



The screenshot shows a MATLAB command window with the following code:



```
>> a = ['first', ...
        'third', ...
        'fourth', ...
        'fifth', ...
        'sixth']
```

 A blue-bordered pop-up window is overlaid on the first line of the code, containing the text:


```
>> a = ['first', ...
```

 This indicates that MATLAB has found a matching opening delimiter for the closing quote on the first line.

The pop-up window for delimiter matching is not supported in live scripts.

You can change delimiter matching indicators, and when and if they appear. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard**. Preference changes do not apply in live scripts.

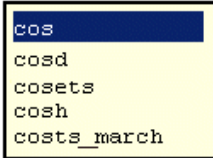
Tab Completion

MATLAB can help you avoid typographical errors by completing the names of functions, models, MATLAB objects, files, folders, variables, structures, graphics properties, parameters, and options.

To complete names in the Command Window, type the first few characters of the name you want to complete, and then press the **Tab** key.

If MATLAB presents a list of possible matches, use the arrow keys to select the name you want, and then press the **Tab** key.

```
>> cos|  
cos  
cosd  
cosets  
cosh  
costs_march
```

A screenshot of the MATLAB Command Window. The prompt '>>' is followed by the text 'cos|'. A yellow dropdown menu is open, listing five options: 'cos', 'cosd', 'cosets', 'cosh', and 'costs_march'. The first option, 'cos', is highlighted with a blue background.

In addition, you can:

- Clear the list without selecting anything, by pressing the **Esc** (escape) key.
- Search a long list before making a selection, by adding additional characters to your original term.
- Complete parts of a name that use dot notation by adding a dot, and then pressing the **Tab** key.
- Complete the names and values of graphics properties. Begin typing the first part of a property, and then press the **Tab** key. Type a comma after each property.
- Complete parameter names and options for certain functions. When entering a command, at the location of a parameter or option, type ' ', and then press the **Tab** key. Completion of parameters and options is not available for all functions.


For MATLAB to complete a file or folder name, it must be on the search path or in the current folder. Variables and properties must be in the current workspace.

In the Editor, MATLAB completes:

- Nested functions only when they are available at the current location of the cursor. Not supported in live scripts.
- Names of variables defined in the active document. The variable must be valid at the current location of the cursor (that is, already defined).
- Names of class properties and methods in class definition files. Not supported in live scripts.

In the Editor, MATLAB does not complete field names of structure arrays defined only within the active file.

Note: To add spaces within statements using the **Tab** key in the Editor, first add a space, and then press **Tab**. Otherwise, when tab completion is enabled, MATLAB attempts to complete a name.

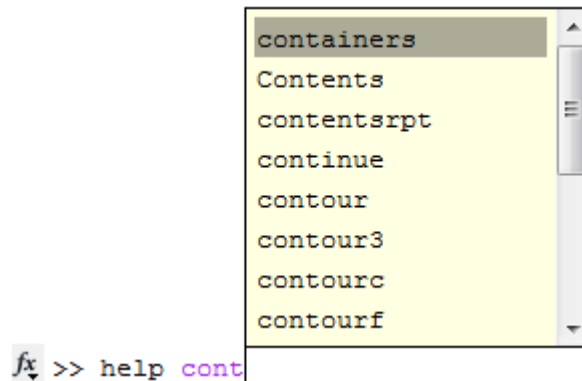
Tab completion is enabled by default. To change this setting, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard**. Preference changes do not apply in live scripts.

Example of Name Completion

This example shows how to complete the name for the `containers.Map.keys` method.

- 1 In the Command Window, type `help cont`, and then press **Tab**.

MATLAB displays a list of selections.



- 2 Select `containers`, and then press **Tab**.

The Command Window displays `help containers`.

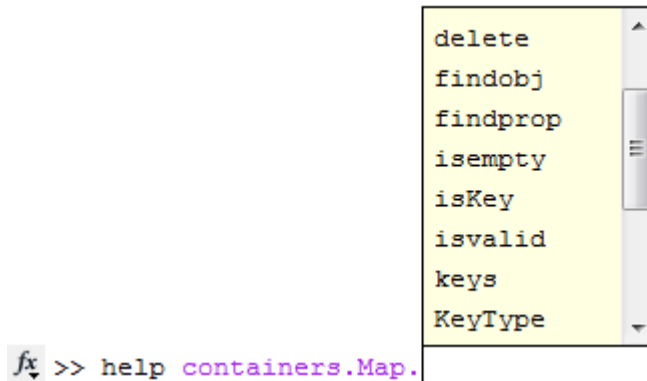
- 3 At the command prompt, add a dot after `containers`, and then press **Tab**.

The Command Window displays:

```
help containers.Map
```

- 4 At the command prompt, add a dot after `Map`, and then press **Tab**.

MATLAB displays a new list.



- 5 Scroll down the list, select `keys`, and then press the **Tab** key.

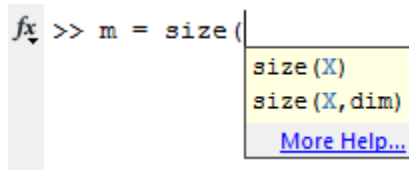
The Command Window displays `help containers.Map.keys`.

Function Syntax Hints

As you enter a function in the Command Window or Editor, syntax hints open in a pop-up window to display allowable input arguments for a function.

Function hints appear for both MATLAB installed functions and functions you create. The syntax hints for MATLAB functions comes from the documentation. The syntax for functions you create comes from the function definition statement (first executable line) in the MATLAB program file. That file must be on the search path or in the current folder.

To use function syntax hints, type a function name with an opening parenthesis, and then pause. A tooltip opens showing the basic syntax for the function.




You can type a variable for any argument that appears in blue. Enter your variable names, and not the argument names shown in the window.

The displayed syntax options change, based on the argument you just entered.

Some function names are overloaded. That is, there are methods with the same name as a function that support different types of inputs. Overloaded methods require that you pass an object as the first input. When you specify the object name, the syntax hints update to reflect the associated method, as shown.

```
>> m = matfile('topography.mat');  
fx >> size (m,  
size(matlab.io.MatFile object,variable)  
size(matlab.io.MatFile object,variable,dim)  
More Help...
```

Function syntax hints are suggestions only. Some allowable arguments might not appear, or could be in black text when they should be blue.

Function hints are enabled by default. To change this setting, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Keyboard**, and then set the options for **Function hints**. Preference changes do not apply in live scripts.

Command History

In this section...

“What Is the Command History?” on page 3-28

“Use Command History Commands” on page 3-29


“Change the Command History Date Format” on page 3-30

“Command History Preferences” on page 3-30

What Is the Command History?

The Command History window displays a log of statements that you ran in the current and previous MATLAB sessions. The Command History lists the time and date of each session in your operating system's short date format, followed by the statements for that session. Brackets in the left margin indicate commands that are processed as a group. A colored mark precedes each command that generates an error.


To view the command history, press the up-arrow key, ↑, in the Command Window. To retrieve a command using a partial match, type any part of the command at the prompt, and then press the up-arrow key.

To dock or detach the Command History window, click , and then select an option. To view the Command History if it is closed: on the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, click **Command History** and select either **Docked** or **Popup**.

MATLAB saves statements that run in the Command Window to the history file, `History.xml`. These statements include those you run using the **Evaluate Selection** item on context menus in tools such as the Editor, Command History, and Help browser. By default, MATLAB automatically saves the command history file after each command. The history file does not include every action taken in MATLAB. For example, modifications of values in the Variables editor are not included in the Command History. All entries remain until you delete them, or until the number of commands in the history file exceeds the number of commands to save, as specified in the Command History preferences. When the specified limit is reached, MATLAB automatically deletes the oldest entries. By default, the Command History saves 25,000 commands.

Use Command History Commands

You can select entries in the Command History window, and then perform the following actions for the selected entries.

Action	How to Perform the Action
Create a script from a statement or statements.	Select an entry or entries, and then right-click and select Create Script or Create Live Script from the context menu. The Editor opens a new file that contains the commands you selected from the Command History window.
Rerun previous commands.	Do one of the following: <ul style="list-style-type: none"> • Press the up arrow key (↑) until the command you want appears at the prompt, and then press Enter • Double-click an entry or entries in the Command History window, or select an entry and press Enter. <p>To extend the selection to include multiple commands, press Shift+↑.</p>
Copy statements to another window.	Do one of the following: <ul style="list-style-type: none"> • Select an entry or entries, and then select Copy from the context menu. Paste the selection into an open file in the Editor or any application. • Drag the selection from the Command History window to an open file or another application.
Create a shortcut from a statement or statements.	Do one of the following: <ul style="list-style-type: none"> • Select an entry or entries, and then right-click and select Create Shortcut from the context menu. • Drag the selection to the desktop Toolstrip. The Add Shortcut dialog box opens and the selected commands appear in the Callback field.
Delete Entries	Select the entries to delete, and then right-click and select Delete , or press the Delete key. <p>To delete all entries, click , and then select Clear Command History from the context menu. You cannot recall these entries.</p>

Change the Command History Date Format

MATLAB uses your operating system's short date format to display dates in the Command History window. To change the date format, for instance from MM/DD/YYYY to DD/MM/YYYY:

- 1 Change the short date format for your operating system as described in its documentation.
- 2 Restart MATLAB.


Note: Clearing the command history deletes all entries from the Command History window. You can no longer recall those entries in the Command Window.

Command History Preferences

You can exclude statements from the command history and specify how many commands to save to the command history file, `History.xml`. MATLAB uses the command history file for both the Command History window and statement recall in the Command Window.

Note: When you exclude statements from the command history file, you cannot recall them in the Command Window, nor can you view them in the Command History window.

You can also change the way you search for previously executed statements in the command history. Select from different search text matching options and change the way results are displayed in the Command History window.

To set Command History preferences, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Command History**, and then adjust the preference options as described in this table:

Option	Usage
Save	Select Save exit/quit commands to save <code>exit</code> and <code>quit</code> commands in the command history.
	Select Save consecutive duplicate commands to save consecutive executions of the same statement in the command history.

Option	Usage
	<ul style="list-style-type: none"> • With this option selected, if you run <code>magic(5)</code> two times in a row, the entries for <code>magic(5)</code> appear on separate lines in the command history. • With this option cleared, the command history retains only one entry for <code>magic(5)</code> and displays a tally of consecutive executions to the left. <p>Select Don't save history file to prevent saving the command history across sessions. This option is useful when multiple users share the same machine. For example, the option prevents each user from viewing statements others have run.</p> <p>Any entries predating the current session remain unless you first delete entries from the Command History window.</p> <p>Save last <i>n</i> commands specifies the number of commands to save.</p>
Match	<p>Select Match anywhere to retrieve statements that contain the search text in any location.</p> <p>Select Match beginning to retrieve statements that begin with the search text. This option is selected by default.</p> <p>Select Match case to retrieve statements that match the case of the search text.</p> <p>Select Filter matches to display only statements that match the search text. Clear Filter matches to display all previously executed statements and highlight the statements that match the search text.</p>
Show	<p>Select Show match toolbar to display a search toolbar at the top of the Command History window. Search for previously executed statements using the search field and change Match preferences using the provided buttons.</p> <p>Select Show match locations to display yellow markers to the right of the scroll bar in the Command History window that indicate the location of matches throughout the command history.</p>



Option	Usage
	Select Show execution time to display an approximate execution time to the right of each statement. Times display for any statements that takes longer than 0.1 seconds to execute.

Help and Product Information

- “Ways to Get Function Help” on page 4-2
- “MATLAB Code Examples” on page 4-3
- “Search Syntax and Tips” on page 4-7
- “Bookmark and Share Page Locations” on page 4-10
- “Contact Technical Support” on page 4-12
- “Help Preferences” on page 4-14
- “Japanese Documentation” on page 4-16
- “Korean and Chinese Documentation” on page 4-17
- “Information About Your Installation” on page 4-18

Ways to Get Function Help

Each MATLAB function has supporting documentation that includes examples and describes the function inputs, outputs, and calling syntax. This table describes ways to access that documentation.

Type of Help	How to Access	Example or Icon
Reference page in Help browser	Use the <code>doc</code> command. — <i>or</i> — Select a function name in the Editor, Command Window, or Help browser; right-click; and then select Help on Selection .	<code>doc mean</code>
Function syntax hints in Command Window	After you type an open parenthesis for function inputs, pause or press Ctrl + F1 .	<code>mean (</code>
Abbreviated help text in Command Window	Use the <code>help</code> command.	<code>help mean</code>
Function browser in Command Window	Click the function icon to the left of the command prompt.	
Complete documentation in Help browser	Click the Help button on the quick access toolbar or on the Home tab. — <i>or</i> — Enter search terms in the Search Documentation box.	

See Also

`doc`

More About

- “MATLAB Code Examples” on page 4-3

MATLAB Code Examples


In this section...

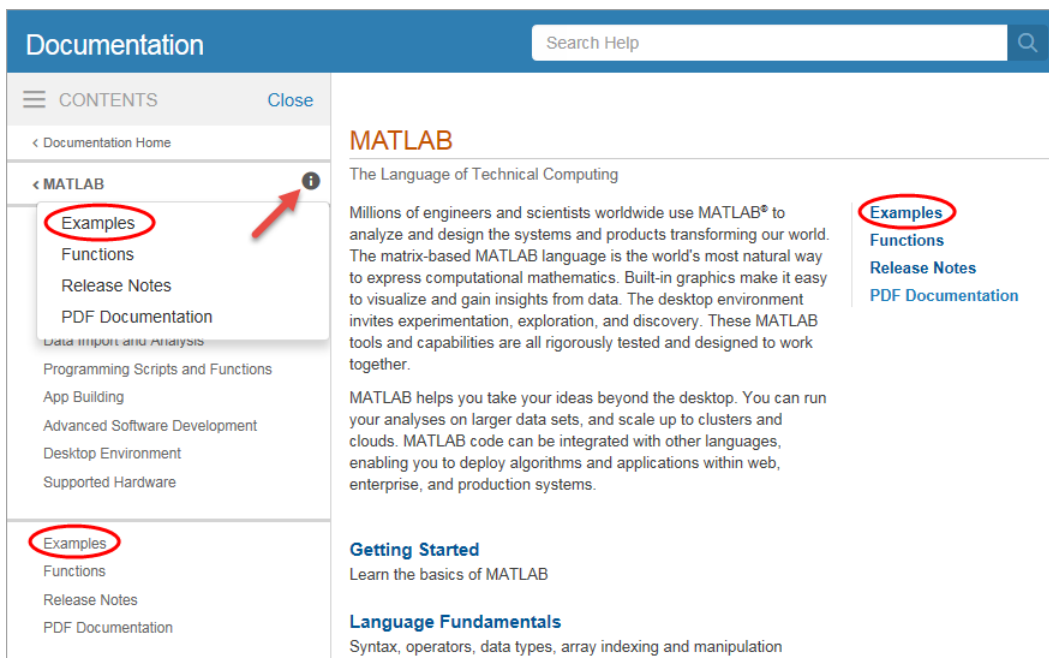
“Standalone Examples” on page 4-3

“Inline Examples” on page 4-5

Standalone Examples

A standalone example is a readable version of a MATLAB script that shows how to accomplish a particular task. MATLAB and all MATLAB toolboxes include examples as part of the installed documentation. (Before release R2012b, these examples were called *demos*.)

Access examples by clicking **Examples** on the right or bottom left of the main documentation page for a particular product. You can also access examples by clicking the  icon to the right of the product name.

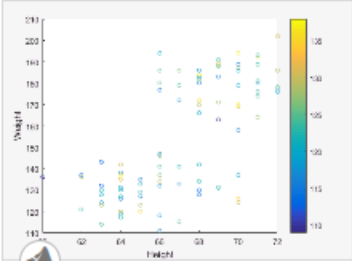


The screenshot shows the MATLAB Documentation website. The top navigation bar includes a search box and a 'Documentation' title. The left sidebar contains a 'CONTENTS' menu with a 'Close' button. Under the 'MATLAB' section, the 'Examples' link is circled in red. A red arrow points to the 'i' icon next to the 'MATLAB' header. The main content area displays the 'MATLAB' title and a description of the language. On the right side of the main content area, a sidebar contains links for 'Examples', 'Functions', 'Release Notes', and 'PDF Documentation', with the 'Examples' link circled in red.

MATLAB include various examples that demonstrate various functionality. For instance, click on **Graphics** to view examples demonstrating plotting functionality in MATLAB.

Graphics

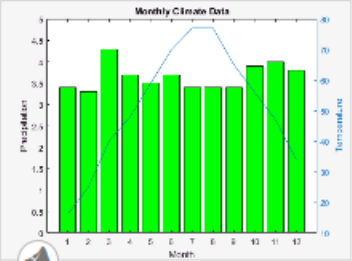
2-D Plots



Creating 2-D Plots

Create a variety of 2-D plots in MATLAB®.

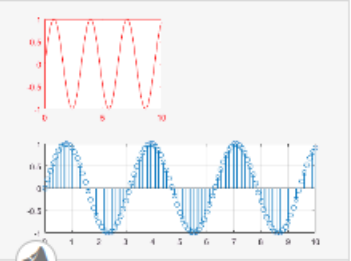
[Read More](#) [Open Live Script](#)



Creating Plots with Two y-Axes

Create and customize plots with two y-axes.

[Read More](#) [Open Live Script](#)



Displaying Multiple Plots in a Single Figure

Display multiple plots in a single figure in MATLAB® by using subplot.

[Read More](#) [Open Live Script](#)




Each example combines comments, code, and output together in a formatted document. You can open the corresponding script in the Editor by clicking **Open Script** or **Open Live Script** to the right of the example name, or by clicking **Open Script** or **Open Live Script** at the top of the example page in the Help browser.

The screenshot shows the MATLAB Help browser interface. At the top, there is a blue header with the word 'Examples' on the left and a search bar labeled 'Search Help' on the right. Below the header, there is a 'CONTENTS' sidebar on the left with a 'Close' button. The sidebar lists navigation options: '< Examples Home', '< MATLAB', '< Graphics', '< 2-D Plots', and 'Creating 2-D Plots'. Under 'ON THIS PAGE', it lists 'Line Plots', 'Bar Plots', and 'Stairstep Plots'. The main content area is titled 'Creating 2-D Plots' and contains the text: 'This example shows how to create a variety of 2-D plots in MATLAB®.' Below this is a section titled 'Line Plots' with the text: 'The plot function creates simple line plots of x and y values.' A code block contains the following MATLAB code:

```
x = 0:0.05:5;
y = sin(x.^2);
figure
plot(x,y)
```

To the right of the code block is a blue button labeled 'Open Live Script' with a mouse cursor pointing to it.

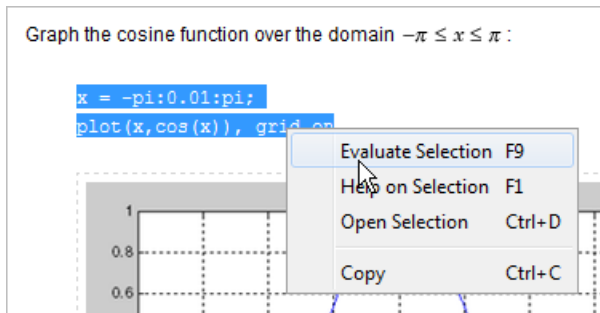
In the Editor, there are two ways to run the script:

- Run one section at a time and view the incremental results. Select the first section, and then step through the script by clicking **Run and Advance**, .
- Run the entire script by clicking **Run**  or **Run All** .

Additional examples, created by members of the MATLAB community, are available at the File Exchange.

Inline Examples

The product documentation also includes inline code excerpts, such as examples on function pages like `cos` or `plot`. You can run inline code from the Help browser by selecting the code, right-clicking, and then selecting **Evaluate Selection**, as shown. (On Macintosh systems, press **Shift+F7**, which copies code to the Command Window for evaluation.)



See Also

demo | echodemo

Related Examples

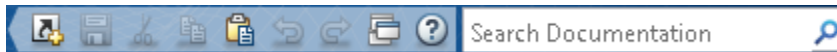
- “Run Code Sections”

External Websites

- File Exchange

Search Syntax and Tips

Find keywords in the documentation by entering text in the Search box on the Desktop or in the Help browser.



When you view pages linked from the search results, search terms appear with highlights. To clear the highlights, press the **Esc** key.

The search engine ignores common, insignificant words such as *a*, *an*, *the*, and *of*, unless they are part of an exact phrase in quotation marks. It also ignores capitalization, punctuation, and special characters such as **+**. To find a symbol or special character:

- Search for the word instead of the symbol or character, such as **plus** instead of **+**.
- View the documentation on Operators and the Symbol Reference.
- Search the PDF documentation, available from the documentation home page.

Searches can include the following operators:

" " Exact phrase

Example: "plot tools" finds pages that contain *plot tools*, in that sequence, with no words between them.

* Wildcard

Requires at least two nonwildcard characters, and cannot appear at the start of a keyword or in an exact phrase.

Example: plot* finds *plot*, *plot3*, and *plotting*.

OR Boolean OR

Example: plot OR graph finds pages with either *plot* or *graph*.

NOT Boolean NOT

Example: "plot tools" NOT "time series" finds pages with *plot tools* but excludes pages with *time series*.

AND Boolean AND

Implied when no operator is present between keywords.

Example: `plot AND tools` is equivalent to `"plot" "tools"`.

The Help browser search evaluates NOT operators first, OR operators second, and AND operators last. For example,

`"plotting tool" OR "plot tools" NOT "time series" AND workspace`

finds pages that contain either *plotting tool* or *plot tools* and contain *workspace*, but do not contain *time series*.

You can filter search results using facets that appear on the left side of the page. For example, view MATLAB topics by selecting **MATLAB** and **Help Topics**.

The screenshot shows the MATLAB Help browser search interface. At the top, there is a search bar with the text "fft" and a magnifying glass icon. Below the search bar, there is a "FILTER" section with a "Close" button. Under "FILTER", there is a "Refine by Type" section with "Help Topics" selected and circled in red. Below that is a "Refine by Category" section with "MATLAB" selected and circled in red. The main content area shows "YOUR SELECTIONS" with "MATLAB" and "Help Topics" buttons, and a "Remove All" button. Below this, it says "Searched for fft" and "Results 1 to 10 of 12". The search results are listed as follows:

- Polynomial Interpolation Using FFT**
This example shows how to use **FFT** to estimate coefficients of a trigonometric polynomial that interpolates a set of regularly-spaced data
[Documentation > MATLAB > Mathematics > Fourier Analysis and Filtering](#)
- Two-Dimensional FFT**
This section discusses generalizations of the DFT in one dimension.
[Documentation > MATLAB > Mathematics > Fourier Analysis and Filtering](#)
- Fourier Analysis and Filtering - Fourier transforms, convolution, digital filtering**
Fourier transforms, convolution, digital filtering
[Documentation > MATLAB > Mathematics](#)

The search engine searches the following text in the documentation:

- Documentation — Text and code shown in the Help browser

- User interface examples — Help comments in the program file
- Videos — Title

Bookmark and Share Page Locations

In this section...

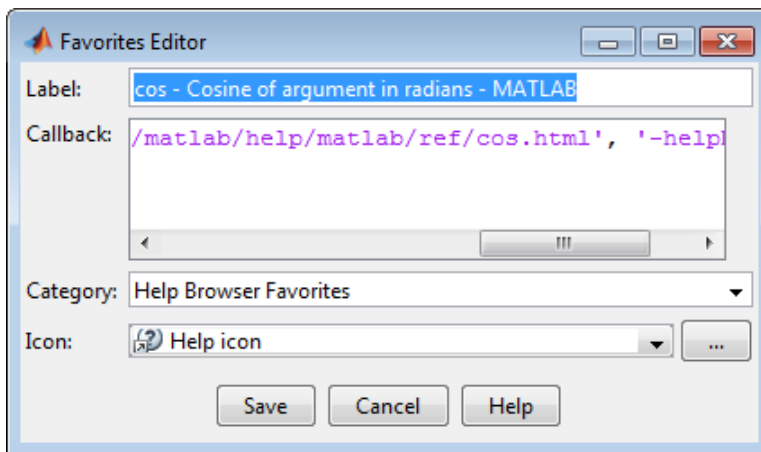
“Bookmark Favorite Pages” on page 4-10

“View Page Locations” on page 4-11

Bookmark Favorite Pages

In MATLAB, bookmarks are called *favorites*. Add, find, and organize favorites by clicking the Favorites button in the Help browser, .

When you add a favorite, do *not* change the **Callback**. MATLAB requires special values to create a shortcut that opens the page in the Help browser. In addition, if you want the bookmark to appear in your list of favorites, keep the **Category** set to **Help Browser Favorites**, as shown.



Note: You cannot migrate favorites that you save in one MATLAB release to a new release.

View Page Locations

To identify the location of a page in the Help browser to share with someone else, right-click within the page, and then select **Get Page Address**.


The Help Page Location dialog box provides two ways to access the page:

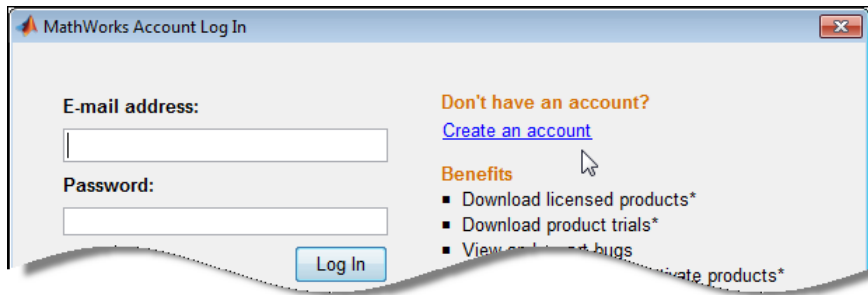
- A **web** command to run from the command line that opens the page from the installed documentation. This command is subject to change between releases, so it is not always accurate for someone running a different version of MATLAB.
- A URL for the page corresponding to your product version at the MathWorks website. This documentation is available to anyone, even if they do not have MathWorks products. However, to access archived documentation from previous releases, as well as Korean and Chinese documentation, you must log in with a MathWorks Account.

Note: If you are running a prerelease version, the URL is invalid because the documentation does not yet exist on the website.

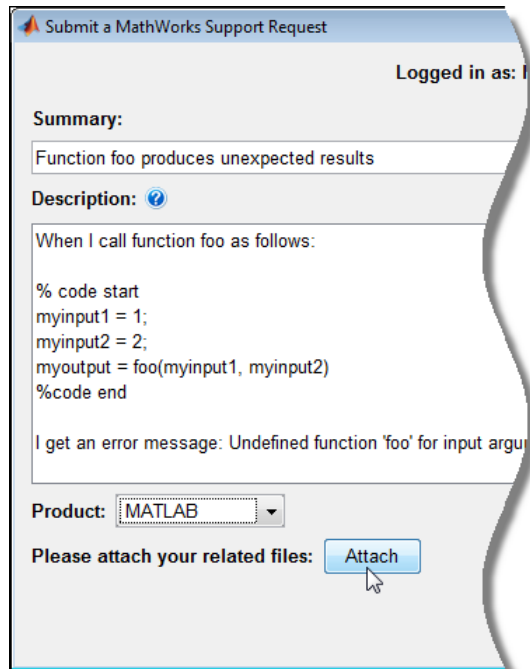
Contact Technical Support

This example shows how to contact MathWorks Technical Support to report a bug or request help. This procedure requires internet access.

- 1 On the **Home** tab, in the **Resources** section, click  **Request Support**.
- 2 When requested, log in using your MathWorks Account email address and password. If you do not have a MathWorks Account, create one.



- 3 Provide information to help technical support reproduce your issue, such as a description of the steps you followed or a code excerpt. Optionally, you can attach up to five files to your request, where each file is no larger than 3 MB. To submit files larger than 3 MB, upload them to the MathWorks FTP site.



The screenshot shows a web form titled "Submit a MathWorks Support Request". The user is logged in. The form has the following sections:

- Summary:** A text box containing "Function foo produces unexpected results".
- Description:** A text area containing:

```
When I call function foo as follows:  
  
% code start  
myinput1 = 1;  
myinput2 = 2;  
myoutput = foo(myinput1, myinput2)  
%code end  
  
I get an error message: Undefined function 'foo' for input argu
```
- Product:** A dropdown menu with "MATLAB" selected.
- Please attach your related files:** A button labeled "Attach" with a mouse cursor over it.


- 4 Specify the product that is related to the issue.
- 5 Submit the request.

External Websites

- How do I access the MathWorks FTP site?
- MathWorks Support Page

Help Preferences

To set Help preferences:

- 1 On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Help**.
- 2 Adjust the preference options as described in the table.

Preference	Usage
Documentation Location	<p>Specify whether to view the documentation provided with your installed products or the documentation on the web at http://www.mathworks.com/help. Viewing the web documentation requires an internet connection and a MathWorks Account.</p> <p>If your preference is set to view web documentation, but your internet connection becomes unavailable, MATLAB changes the preference to view the installed documentation. You can reset the preference after your connection is restored.</p> <p>Changes to this preference apply only to new Help browser tabs.</p>
Selected Products	<p>Select the products to include for viewing and searching documentation in the Help browser or Function browser.</p> <p>If your Documentation Location is set to view documentation on the web, then you can select Show products that are not installed to select and access documentation for all MathWorks products.</p> <p>When the Help browser is already open, changes to this preference apply only to new Help browser tabs.</p>
Quick Help Display	<p>Specify whether help links display content in the Help browser or in a small window. This preference applies to reference pages or program help that you access using:</p> <ul style="list-style-type: none"> • Help on Selection in context menus or F1 • Function hints or the Function Browser • Links in error messages

Preference	Usage
	Links to reference pages from the Current Folder browser always open in the Help browser.
Language (selected non-English systems only)	Specify whether you want documentation in the Help browser and context-sensitive help to appear in English. Installed non-English documentation is not always current. This option is only available if your Documentation Location is set to view documentation on the web, and for selected non-English systems only.


To adjust the font size in the Help browser or MATLAB Web browser, use the **Ctrl + Mouse Scroll** keyboard shortcut.

More About

- “Japanese Documentation” on page 4-16
- “Korean and Chinese Documentation” on page 4-17

Japanese Documentation

Many MathWorks products provide versions of the documentation translated from English to Japanese.

The new version of most products installs the translated documentation from the *previous* version and the English documentation for the *current* version. To view the English documentation, access the **Environment** section on the **Home** tab, and click  **Preferences**. Select **MATLAB > Help**, and set the Help Language preference to English.

The **Language** preference is available when the system locale is Japanese and the translated documentation is installed. The preference changes the language only in the Help browser and context-sensitive help. If the documentation for a product is not translated, the Help browser displays the English documentation.

When the translated documentation is available, you can view it by setting your Help **Documentation Location** preference to view documentation on the web. Alternatively, download it from the MathWorks website at <http://www.mathworks.co.jp/help>.

For information about documentation in other languages, contact your MathWorks sales and service office.

Related Examples

- “Set Locale on Windows Platforms” on page 9-5
- “Set Locale on Linux Platforms” on page 9-8
- “Set Locale on Mac Platforms” on page 9-7


Korean and Chinese Documentation

A *subset* of MATLAB documentation in Korean and simplified Chinese is available on the web to licensed MATLAB users.

Access this documentation from MATLAB using these steps:

- 1 From the **Home** tab, in the **Environment** section, click the **Preferences** button.
- 2 From the left pane, select **Help**.
- 3 For **Documentation Location**, select **Web, on mathworks.com (Internet connection required)**. Then for **Language**, select either your default language (**Korean** or **Chinese**) or **English**, and click **OK**.
- 4 Open the documentation from the **Home** tab **Resources** section by clicking the **Help** button.

You also can view the Help directly on the web:

- 1 Open the MathWorks help website, <http://www.mathworks.com/help>.
- 2 Click the button with the globe icon  and the country name at the bottom left of the web page, and select a country based on the language in which you want to view the documentation.

If the documentation does not display in the language you want, the subset of documentation you are viewing might not be translated.

Information About Your Installation

MATLAB software can tell you what products are installed, their versions, and other information about your license and platform. This information is important to have in the event you contact technical support.

Type of Information You Want	To Get the Information
Version and license for Installed product	From the product, select Help > About . Or use functions: <ul style="list-style-type: none"> • <code>license</code> — for the license number • <code>ver</code> — for version numbers for MATLAB and libraries • <code>version</code> — for version numbers for MathWorks products
MATLAB platform	In MATLAB, select Help > About MATLAB . The About MATLAB dialog box shows 32-bit or 64-bit.
<code>arch</code> value used to locate library files for the <code>mex</code> function and standalone applications	In MATLAB, select Help > About MATLAB . The About MATLAB dialog box shows the <code>arch</code> value, for example <code>win64</code> . Or use the <code>computer</code> function.
Passcodes and licenses	From any desktop tool, select Help > Licensing > Manage Licenses .

Workspace Browser and Variable Editor

- “Create and Edit Variables” on page 5-2
- “Save and Load Workspace Variables” on page 5-15
- “Workspace and Variable Preferences” on page 5-20

Create and Edit Variables


In MATLAB, you can create, edit, and copy variables in the MATLAB workspace. The MATLAB workspace consists of the variables you create and store in memory during a MATLAB session.

View Workspace Contents

To view a list of variables in your workspace, use the Workspace browser. To open the Workspace browser if it is not currently visible, do either of the following:

- On the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select **Workspace**.
- Type `workspace` in the Command Window.

By default, the Workspace browser displays the base workspace. You also can view function workspaces if MATLAB is in debug mode. For more information, see “Base and Function Workspaces”.

To display additional columns such as size and range, on the Workspace browser title bar, click , and then click **Choose Columns**.

To select which columns to display, right-click the Workspace browser title bar and select or clear the desired column names.

You can also use the `who` command in the Command Window to view a list of variables. To list information about size and class, use the `whos` command.

Create Variables

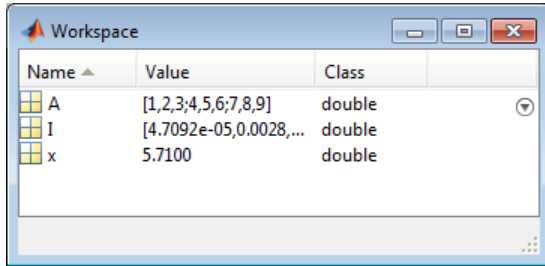
You can create new variables in the workspace by running MATLAB code or using existing variables.

To create a new variable, enter the variable name in the Command Window, followed by an equal sign (=) and the value you want to assign to the variable. For example, if you run these statements, MATLAB adds the three variables `x`, `A`, and `I` to the workspace:

```
x = 5.71;
```



```
A = [1 2 3; 4 5 6; 7 8 9];
I = besseli(x,A);
```



You do not have to declare variables before assigning values to them.

If you do not end the assignment statement with a semicolon (;), MATLAB displays the result in the Command Window. For example,

```
x = 5.71
x =
    5.7100
```

If you do not explicitly assign the output of a statement to a variable, MATLAB generally assigns the result to the reserved word `ans`. The value of `ans` changes with every statement that returns an output value that is not assigned to a variable. For example,

```
sin(1)
ans =
    0.8415
```

To create a new workspace variable from an existing variable, in the Variables editor, select an element, data range, row, or column in an array, and then in the **Variable** tab, select **New from Selection**.



View Variable Contents

You can view the contents of a variable in several ways:

- Command Window — Type the variable name at the command prompt.

- Variables editor — In the Workspace browser, double-click a variable name. You can also use the `openvar` function. For example, to open the variable `A`, type `openvar('A')`.

Some variables open a viewer or other tool appropriate for the type of value they have, such as `timeseries`. For details, see the documentation for that data or object type.

A  or  icon next to a variable property in the Variables editor indicates that the property is protected or private.

Note: The maximum number of elements in a variable that you can open in the Variables editor depends on your operating system and the amount of physical memory installed on your system.

To change how the Variables editor displays variables, go to the **View** tab, and in the **Format** section, select a number display format. The display format does not affect how values are displayed in the Command Window or Workspace browser, or how the variables are saved.

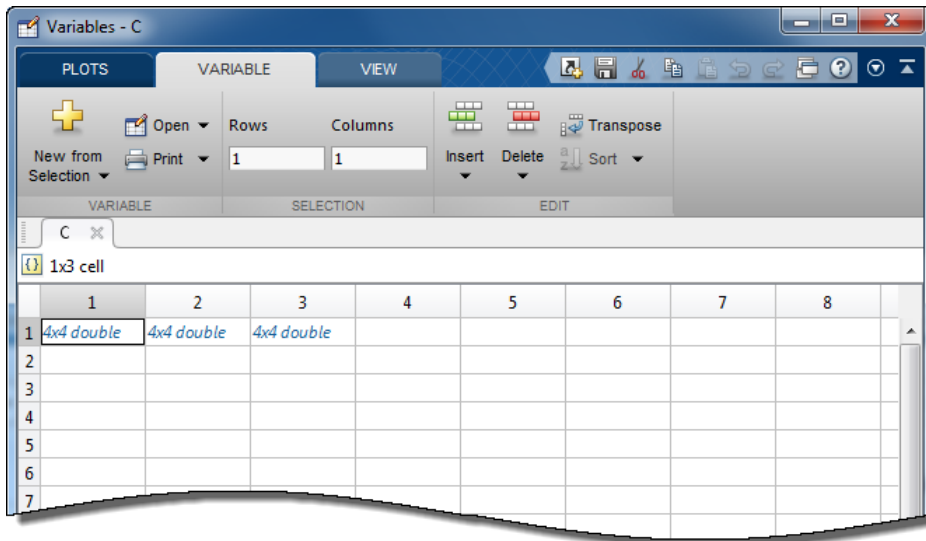
Edit Variable Contents

You can edit the contents of scalar (1-by-1) variables in the Workspace browser. Right-click the variable and select **Edit Value**.

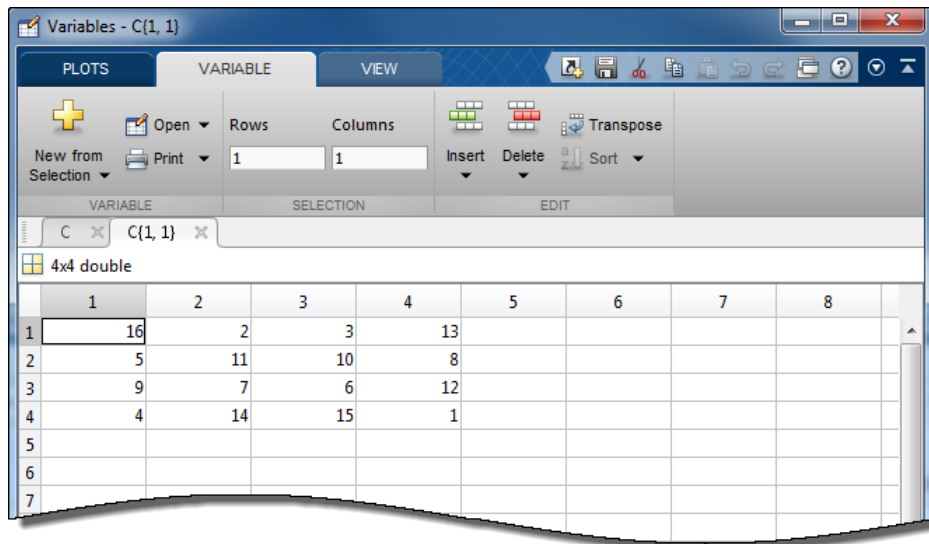
To edit other variables, open them in the Variables editor. For example, suppose that you create a cell array, `C`, by running these commands in the Command Window:

```
A = magic(4);  
C = {A A A};
```

In the Workspace browser, double-click the variable name `C` to open it in the Variables editor.



To edit an element of a variable, double-click the element. The element opens in a new document within the Variables editor. For example, if you double-click element $C\{1, 1\}$ in the Variables editor, the contents of that cell open in a new tab. You can edit the value of a variable element by clicking the element and typing a new value. Press **Enter** or click another element to save the change.



To return to the parent cell array or structure of an element, go to the **View** tab and click the **Go Up** button.

Changes you make in the Variables editor are automatically saved in the workspace. Changes you make to variables via the Command Window or other operations automatically update the information for those variables in the Workspace browser and Variables editor.

Note:

- You cannot edit elements or subsets of multidimensional arrays in the Variables editor.
 - You cannot edit tall arrays in the Variables editor.
 - When editing strings in the Workspace browser or as part of a structure in the Variables editor, you must use double quotes to surround the string value.
-

To modify the size, shape and order of variable elements in the Variable editor, use the following procedures:

Action	Procedure
Delete row, column, or variable elements	Right-click the desired row header, column header, or selected elements and select Delete Row or Delete Column .
Insert new row or column	Right-click the desired row header, column header, or element and select Insert Row Above , Insert Row Below , Insert Column to the Left , or Insert Column to the Right . You can also add rows or columns simply by entering a value in an empty row or column. For example, to add a row and column to the array in <code>C{1,1}</code> , enter a value in element (5,5).
Cut variable elements	Right-click the desired row header, column header, or selected elements and select Cut . The cut values move to the clipboard and are replaced by the default value for empty elements. For more information, see “Empty Elements” on page 5-7.
Copy variable elements	Right-click the desired row header, column header, or selected elements and select Copy .
Paste variable elements	Right-click the row header, column header, or element where you want the insertion to begin and select Paste .
Paste cells from Microsoft Excel spreadsheet	Right-click the element where you want the insertion to begin, and then select Paste Excel Data .

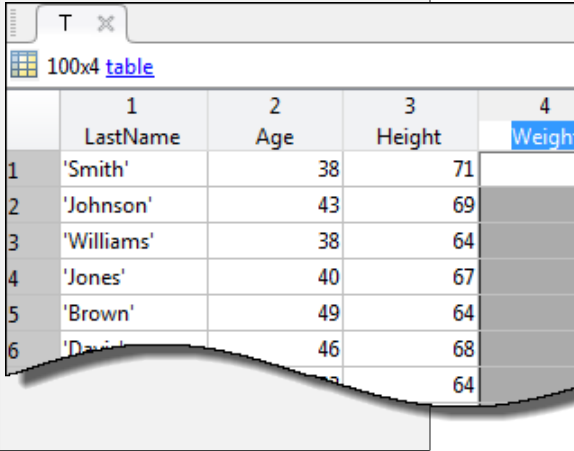
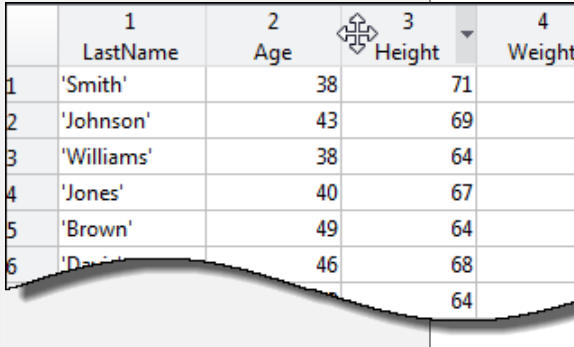
Empty Elements

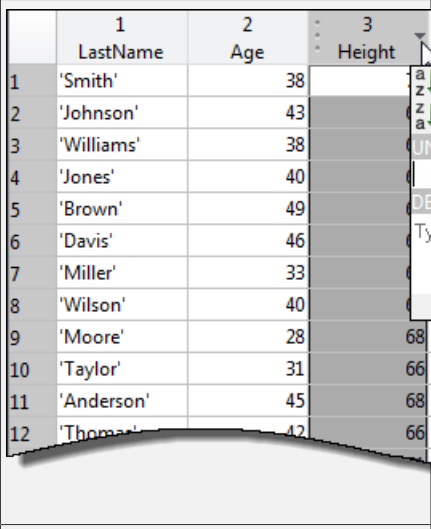
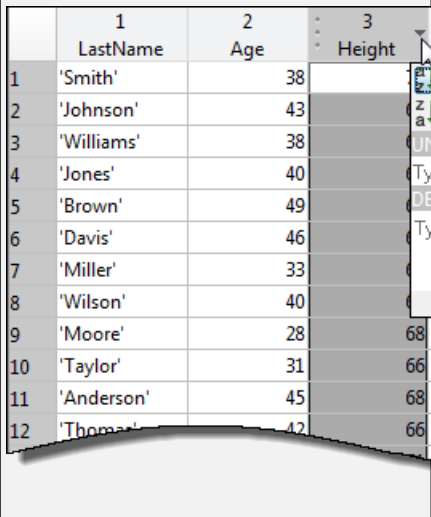
Empty elements in variables are assigned default values. Default assignments are:

- 0 for numeric arrays
- [] for cell arrays and structure arrays
- <undefined> for categorical variables

Edit Table and Structure Array Variables

Tables (including timetables) and structure arrays support additional editing actions.

Action	Procedure	Example																																								
Modify column or row name	Double-click the name and enter the new text.	 <p>100x4 table</p> <table border="1"> <thead> <tr> <th></th> <th>1 LastName</th> <th>2 Age</th> <th>3 Height</th> <th>4 Weight</th> </tr> </thead> <tbody> <tr><td>1</td><td>'Smith'</td><td>38</td><td>71</td><td></td></tr> <tr><td>2</td><td>'Johnson'</td><td>43</td><td>69</td><td></td></tr> <tr><td>3</td><td>'Williams'</td><td>38</td><td>64</td><td></td></tr> <tr><td>4</td><td>'Jones'</td><td>40</td><td>67</td><td></td></tr> <tr><td>5</td><td>'Brown'</td><td>49</td><td>64</td><td></td></tr> <tr><td>6</td><td>'David'</td><td>46</td><td>68</td><td></td></tr> <tr><td></td><td></td><td></td><td>64</td><td></td></tr> </tbody> </table>		1 LastName	2 Age	3 Height	4 Weight	1	'Smith'	38	71		2	'Johnson'	43	69		3	'Williams'	38	64		4	'Jones'	40	67		5	'Brown'	49	64		6	'David'	46	68					64	
	1 LastName	2 Age	3 Height	4 Weight																																						
1	'Smith'	38	71																																							
2	'Johnson'	43	69																																							
3	'Williams'	38	64																																							
4	'Jones'	40	67																																							
5	'Brown'	49	64																																							
6	'David'	46	68																																							
			64																																							
Reorder variables	Hover over the left side of a variable until a four-headed arrow appears. Then, click and drag the column to a new location.	 <table border="1"> <thead> <tr> <th></th> <th>1 LastName</th> <th>2 Age</th> <th>3 Height</th> <th>4 Weight</th> </tr> </thead> <tbody> <tr><td>1</td><td>'Smith'</td><td>38</td><td>71</td><td></td></tr> <tr><td>2</td><td>'Johnson'</td><td>43</td><td>69</td><td></td></tr> <tr><td>3</td><td>'Williams'</td><td>38</td><td>64</td><td></td></tr> <tr><td>4</td><td>'Jones'</td><td>40</td><td>67</td><td></td></tr> <tr><td>5</td><td>'Brown'</td><td>49</td><td>64</td><td></td></tr> <tr><td>6</td><td>'David'</td><td>46</td><td>68</td><td></td></tr> <tr><td></td><td></td><td></td><td>64</td><td></td></tr> </tbody> </table>		1 LastName	2 Age	3 Height	4 Weight	1	'Smith'	38	71		2	'Johnson'	43	69		3	'Williams'	38	64		4	'Jones'	40	67		5	'Brown'	49	64		6	'David'	46	68					64	
	1 LastName	2 Age	3 Height	4 Weight																																						
1	'Smith'	38	71																																							
2	'Johnson'	43	69																																							
3	'Williams'	38	64																																							
4	'Jones'	40	67																																							
5	'Brown'	49	64																																							
6	'David'	46	68																																							
			64																																							

Action	Procedure	Example																																																																						
<p>Modify units and description of variables</p>	<p>Click the arrow that appears to the right of a variable name. Then, enter new text in the Units and Description fields.</p>	 <table border="1" data-bbox="911 296 1340 826"> <thead> <tr> <th></th> <th>1 LastName</th> <th>2 Age</th> <th>3 Height</th> <th>4 Weight</th> </tr> </thead> <tbody> <tr><td>1</td><td>'Smith'</td><td>38</td><td></td><td></td></tr> <tr><td>2</td><td>'Johnson'</td><td>43</td><td></td><td></td></tr> <tr><td>3</td><td>'Williams'</td><td>38</td><td></td><td></td></tr> <tr><td>4</td><td>'Jones'</td><td>40</td><td></td><td></td></tr> <tr><td>5</td><td>'Brown'</td><td>49</td><td></td><td></td></tr> <tr><td>6</td><td>'Davis'</td><td>46</td><td></td><td></td></tr> <tr><td>7</td><td>'Miller'</td><td>33</td><td></td><td></td></tr> <tr><td>8</td><td>'Wilson'</td><td>40</td><td></td><td></td></tr> <tr><td>9</td><td>'Moore'</td><td>28</td><td>68</td><td>183</td></tr> <tr><td>10</td><td>'Taylor'</td><td>31</td><td>66</td><td>132</td></tr> <tr><td>11</td><td>'Anderson'</td><td>45</td><td>68</td><td>128</td></tr> <tr><td>12</td><td>'Thomas'</td><td>42</td><td>66</td><td>137</td></tr> <tr><td></td><td></td><td></td><td></td><td>174</td></tr> </tbody> </table>		1 LastName	2 Age	3 Height	4 Weight	1	'Smith'	38			2	'Johnson'	43			3	'Williams'	38			4	'Jones'	40			5	'Brown'	49			6	'Davis'	46			7	'Miller'	33			8	'Wilson'	40			9	'Moore'	28	68	183	10	'Taylor'	31	66	132	11	'Anderson'	45	68	128	12	'Thomas'	42	66	137					174
	1 LastName	2 Age	3 Height	4 Weight																																																																				
1	'Smith'	38																																																																						
2	'Johnson'	43																																																																						
3	'Williams'	38																																																																						
4	'Jones'	40																																																																						
5	'Brown'	49																																																																						
6	'Davis'	46																																																																						
7	'Miller'	33																																																																						
8	'Wilson'	40																																																																						
9	'Moore'	28	68	183																																																																				
10	'Taylor'	31	66	132																																																																				
11	'Anderson'	45	68	128																																																																				
12	'Thomas'	42	66	137																																																																				
				174																																																																				
<p>Sort variable data</p>	<p>Click the arrow that appears to the right of a variable name and select Ascending or Descending.</p>	 <table border="1" data-bbox="911 826 1340 1345"> <thead> <tr> <th></th> <th>1 LastName</th> <th>2 Age</th> <th>3 Height</th> <th>4 Weight</th> </tr> </thead> <tbody> <tr><td>1</td><td>'Smith'</td><td>38</td><td></td><td></td></tr> <tr><td>2</td><td>'Johnson'</td><td>43</td><td></td><td></td></tr> <tr><td>3</td><td>'Williams'</td><td>38</td><td></td><td></td></tr> <tr><td>4</td><td>'Jones'</td><td>40</td><td></td><td></td></tr> <tr><td>5</td><td>'Brown'</td><td>49</td><td></td><td></td></tr> <tr><td>6</td><td>'Davis'</td><td>46</td><td></td><td></td></tr> <tr><td>7</td><td>'Miller'</td><td>33</td><td></td><td></td></tr> <tr><td>8</td><td>'Wilson'</td><td>40</td><td></td><td></td></tr> <tr><td>9</td><td>'Moore'</td><td>28</td><td>68</td><td>183</td></tr> <tr><td>10</td><td>'Taylor'</td><td>31</td><td>66</td><td>132</td></tr> <tr><td>11</td><td>'Anderson'</td><td>45</td><td>68</td><td>128</td></tr> <tr><td>12</td><td>'Thomas'</td><td>42</td><td>66</td><td>137</td></tr> <tr><td></td><td></td><td></td><td></td><td>174</td></tr> </tbody> </table>		1 LastName	2 Age	3 Height	4 Weight	1	'Smith'	38			2	'Johnson'	43			3	'Williams'	38			4	'Jones'	40			5	'Brown'	49			6	'Davis'	46			7	'Miller'	33			8	'Wilson'	40			9	'Moore'	28	68	183	10	'Taylor'	31	66	132	11	'Anderson'	45	68	128	12	'Thomas'	42	66	137					174
	1 LastName	2 Age	3 Height	4 Weight																																																																				
1	'Smith'	38																																																																						
2	'Johnson'	43																																																																						
3	'Williams'	38																																																																						
4	'Jones'	40																																																																						
5	'Brown'	49																																																																						
6	'Davis'	46																																																																						
7	'Miller'	33																																																																						
8	'Wilson'	40																																																																						
9	'Moore'	28	68	183																																																																				
10	'Taylor'	31	66	132																																																																				
11	'Anderson'	45	68	128																																																																				
12	'Thomas'	42	66	137																																																																				
				174																																																																				

Action	Procedure	Example
Modify column or row name	Double-click the name and enter the new text.	

Note: The contents of a table are only visible and modifiable when the number of variables is fewer than 5000. When the number of variables equals or exceeds 5000, you can only view the table properties.

Changes made to certain variable types in the Variables editor also appear in the Command Window. For example, suppose you have a table `T` that contains three columns, `A`, `B`, and `C`. If you delete column `A` in the Variables editor, the line `T(:, 'A') = [];` displays in the Command Window. To suppress code display in the Command Window, on the **View** tab, clear the **Show MATLAB Code** check box.

Navigate Variable Contents

When editing variables in the Variables editor, some variables can contain large amounts of data, making it difficult to navigate between elements. Use these keyboard shortcuts to move easily between variable elements in the Variables editor. You cannot modify these keyboard shortcuts.

Action	Keyboard Shortcut
Commit changes to element and move to next element.	Enter

Action	Keyboard Shortcut
“Variables Preferences” on page 5-22 enable you to specify what the next element is. The default is to move down.	
Move right. Within a selection, Tab also moves from the last column in one row to the first column in the next row.	Tab
Move in opposite direction of Enter or Tab .	Shift+Enter or Shift+Tab
Move up <i>m</i> rows, where <i>m</i> is the number of visible rows.	Page Up
Move down <i>m</i> rows, where <i>m</i> is the number of visible rows.	Page Down
Move to column 1.	Home
Move to row 1, column 1.	Ctrl+Home
Edit current element, positioning cursor at the end of the element.	F2 (Ctrl+U on Apple Macintosh platforms)

Action	Keyboard Shortcut
Commit changes to element and move to next element.	Enter
Move right. Within a selection, Tab also moves from the last column in one row to the first column in the next row.	Tab
Move in opposite direction of Enter or Tab .	Shift+Enter or Shift+Tab

Copy, Rename, and Delete Variables

You can copy and paste, duplicate, rename, and delete variables within the Workspace browser.

Action	Procedure
Copy variable to and from clipboard	Select the variables, right-click, and then select Copy . Then, you can paste the names, for example, into the Command Window or an external application. Multiple variables are comma-separated.

Action	Procedure
Duplicate variable	Select the variables, right-click, and then select Duplicate . MATLAB creates a copy of the selected variables.
Rename a workspace variable	Right-click the variable name, and then select Rename . Type the new variable name and press Enter .
Delete all variables in workspace	On the Home tab, in the Variable section, click Clear Workspace . You can also use the <code>clear</code> function in the Command Window.
Delete selected variables from workspace	Select the variables in the Workspace browser, right-click, and then select Delete . You can also use the <code>clear</code> function in the Command Window. For example, to clear variables <code>A</code> and <code>B</code> , use the command <code>clear A B</code> . To preserve specified variables, but delete others, use the <code>clearvars</code> function with the <code>-except</code> option. For example, to clear all variables except variable <code>A</code> , use the command <code>clearvars -except A</code> .

You can change the character that delimits decimals in the data when you cut and paste values from the Variables editor into text files or other applications. You might do this, for instance, if you provide data to a locale that uses a character other than the period (.). To change the delimiter character, specify a **Decimal separator for exporting numeric data via system clipboard** in the “Variables Preferences” on page 5-22.

Delete Variables


You can delete variables within the Workspace browser.

Action	Procedure
Delete all variables in workspace	On the Home tab, in the Variable section, click Clear Workspace .

Action	Procedure
	You can also use the <code>clear</code> function in the Command Window.
Delete selected variables from workspace	<p>Use the <code>clear</code> function in the Command Window. For example, to clear variables <code>A</code> and <code>B</code>, use the command <code>clear A B</code>.</p> <p>To preserve specified variables, but delete others, use the <code>clearvars</code> function with the <code>-except</code> option. For example, to clear all variables except variable <code>A</code>, use the command <code>clearvars -except A</code>.</p>

Display Statistics in the Workspace Browser

For each variable or object, the Workspace browser can display statistics such as the **Min**, **Max**, and **Mean**, when relevant. MATLAB performs these calculations using the `min`, `max`, and `mean` functions, and updates the results automatically.

To display statistics, on the Workspace browser title bar, click , and then select **Choose Columns**. Select the statistics you want MATLAB to calculate.


Improve Workspace Browser Performance During Statistical Calculations

If you show statistical columns in the Workspace browser, and you work with very large arrays, you might experience performance issues when the data changes as MATLAB updates the statistical results. To improve performance, consider one or both of the following:

- Show only the statistics of interest to you.


On the Workspace browser title bar, click , and then select **Choose Columns**. Clear the statistics you do not want MATLAB to calculate.

- Exclude large arrays from statistical calculations.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Workspace**, and then use the arrow buttons under **Statistical calculations** to change the maximum array size for which the Workspace browser performs statistical calculations. Any variable exceeding the maximum array size

reports <Too many elements> in Workspace browser statistics columns instead of statistical results.

Include or Exclude NaN Values in Statistical Calculations

If your data includes NaNs (Not-a-Number values), you can specify that the Workspace browser statistical calculations consider or ignore the NaNs. On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Workspace**, and then select one of the following:

- **Use NaNs when calculating statistics**

If a variable includes a NaN, and you select this option, the values for **Min**, **Max**, **Var**, and several other statistics appear as NaN. However, **Mode** and several other statistics show a numeric result.

- **Ignore NaNs whenever possible**

If a variable includes a NaN, and you select this option, numeric results appear for most statistics including **Min**, **Max**, and **Mode**. However, **Var** still appears as NaN.

More About

- “Workspace and Variable Preferences” on page 5-20
- “Save and Load Workspace Variables” on page 5-15

Save and Load Workspace Variables

The workspace is not maintained across sessions of MATLAB. When you quit MATLAB, the workspace clears. However, you can save any or all the variables in the current workspace to a MAT-file (.mat). You can load MAT-files later during the current MATLAB session, or during another session, if you want to reuse the workspace variables.

You can save any or all the variables in the current workspace to a MAT-file (.mat). You can load MAT-files later during the current MATLAB session, or during another session, if you want to reuse the workspace variables.

This table describes how to save and load workspace variables.

Action	Procedure
Save all workspace variables to a MAT-file	<p>On the Home tab, in the Variable section, click Save Workspace.</p> <p>You can also use the <code>save</code> function. For example, save all current workspace variables to the file <code>june10.mat</code></p> <pre>save('june10')</pre>
Save selected variables to a MAT-file	<p>Do one of the following:</p> <ul style="list-style-type: none"> • Select the variables in the Workspace browser, right-click, and then select Save As. • Drag the variables from the Workspace browser to the Current Folder browser. <p>You can also use the <code>save</code> function. For example, save only variables <code>A</code> and <code>B</code> to the file <code>june10.mat</code></p> <pre>save('june10', 'A', 'B')</pre>
Save part of a variable	Use the <code>matfile</code> function. For an example, see “Save Parts of Variables to MAT-Files”.
Load a MAT-file	Select the MAT-file in the Current Folder browser, right-click, and then select Load .

Action	Procedure
	<p>You can also use the <code>load</code> function. For example, load all variables from the file <code>durer.mat</code></p> <pre>load('durer')</pre>
Load selected variables from a MAT-file	<p>Do one of the following:</p> <ul style="list-style-type: none"> • On the Home tab, in the Variable section, click Import Data. Select the MAT-file you want to load and click Open. • In the Current Folder browser, select the MAT-file that contains the variables. Drag variables from the Details panel of the Current Folder browser to the Workspace browser. <p>You can also use the <code>load</code> function. For example, load variables <code>X</code> and <code>map</code> from the file <code>durer.mat</code></p> <pre>load('durer','X','map')</pre>
Load part of a variable	<p>Use the <code>matfile</code> function. For an example, see “Load Parts of Variables from MAT-Files”.</p>
Action	Procedure
Save all workspace variables to a MAT-file	<p>Use the <code>save</code> function. For example, save all current workspace variables to the file <code>june10.mat</code></p> <pre>save('june10')</pre>
Save selected variables to a MAT-file	<p>Use the <code>save</code> function. For example, save only variables <code>A</code> and <code>B</code> to the file <code>june10.mat</code></p> <pre>save('june10','A','B')</pre>
Save part of a variable	<p>Use the <code>matfile</code> function. For an example, see “Save Parts of Variables to MAT-Files”.</p>
Load a MAT-file	<p>Double-click the MAT-file in the Current Folder browser</p>

Action	Procedure
	<p>You can also use the <code>load</code> function. For example, load all variables from the file <code>durer.mat</code></p> <pre>load('durer')</pre>
Load selected variables from a MAT-file	<p>On the Home tab, in the Variable section, click Import Data. Select the MAT-file you want to load and click Open.</p> <p>You can also use the <code>load</code> function. For example, load variables <code>X</code> and <code>map</code> from the file <code>durer.mat</code></p> <pre>load('durer','X','map')</pre>
Load part of a variable	Use the <code>matfile</code> function. For an example, see “Load Parts of Variables from MAT-Files”.

Caution When you load data into the MATLAB workspace, the new variables you create overwrite any existing variables in the workspace that have the same name.

View Contents of MAT-File

To see the variables in a MAT-file before loading the file into your workspace, click the file name in the Current Folder browser. Information about the variables appears in the **Details** pane. Alternatively, use the command `whos -file filename`. This function returns the name, dimensions, size, and class of all variables in the specified MAT-file.

To see the variables in a MAT-file before loading the file into your workspace, use the command `whos -file filename`. This function returns the name, dimensions, size, and class of all variables in the specified MAT-file.

For example, you can view the contents of the example file `durer.mat`.

```
whos -file durer.mat
```

Name	Size	Bytes	Class	Attributes
X	648x509	2638656	double	

caption	2x28	112	char
map	128x3	3072	double

The byte counts represent the number of bytes that the data occupies in memory when loaded into the MATLAB workspace. Because of compression, data encoding, and metadata, the space occupied in the file by a variable may be different from the in-memory size. MATLAB compresses data in Version 7 or higher MAT-files. For more information, see “MAT-File Versions”.

Save Variables to MATLAB Script

You also can save workspace variables to a MATLAB script:

- To save all workspace variables, on the **Home** tab, click **Save Workspace**.
- To save selected workspace variables, select the variables in the Workspace browser, right-click, and then select **Save As**.

Then, in the **Save As** window, specify a file name. In the **Save as type** menu, select **MATLAB Script**.

Variables that cannot be saved to a script are saved to a MAT-file with the same name as that of the script.

To load the saved variables into the workspace, simply run the script.

Save Structure Fields as Separate Variables

If any of the variables in your current workspace are structure arrays, the default behavior for saving is to store the entire structure. To store fields of a scalar structure as individual variables, use the `save` function in the Command Window with the `-struct` option.

For example, consider the structure `S`.

```
S.a = 12.7; S.b = {'abc', [4 5; 6 7]}; S.c = 'Hello!';
```

Save the entire structure to `newstruct.mat`. The file contains the variable `S`.

```
save newstruct.mat S
whos -file newstruct.mat
```

Name	Size	Bytes	Class	Attributes
------	------	-------	-------	------------


```
S          1x1                810  struct
```

Save the fields individually. The file contains variables **a**, **b**, and **c**, but not **S**.

```
save newstruct.mat -struct S
whos -file newstruct.mat
```

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	
b	1x2	262	cell	
c	1x6	12	char	

Save only selected fields, such as **a** and **c**. The file contains variables **a** and **c**, but not **b** or **S**.

```
save newstruct.mat -struct S a c
whos -file newstruct.mat
```

Name	Size	Bytes	Class	Attributes
a	1x1	8	double	
c	1x6	12	char	

More About

- “Load Parts of Variables from MAT-Files”
- “Save Parts of Variables to MAT-Files”
- “MAT-File Versions”

Workspace and Variable Preferences


In this section...

“Workspace Preferences” on page 5-20

“Variables Preferences” on page 5-22

Workspace Preferences

Workspace preferences enable you to configure options for saving workspace variables to MATLAB scripts. They also enable you to restrict the size of arrays on which MATLAB performs calculations, and to specify if you want those calculations to include or ignore NaNs.

To open Workspace preferences, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Workspace**.


Preference	Usage
Threshold for saving variables to MATLAB script	<p>Specify Maximum array size to limit the number of elements of arrays saved to a MATLAB script.</p> <p>Specify Maximum struct/object nesting levels to limit the nesting level of structures, arrays, or objects saved to a MATLAB script.</p>
Multidimensional array formatting	<p>Specify how multidimensional arrays are formatted when saved to a MATLAB script.</p> <p>To create a compact script, select Row vector with reshape (compactness and efficiency).</p> <p>To retain array representation, select As 2-D pages (readability). Specify the dimensions of a 2-D slice as positive integers less than or equal to the dimensions of the n-D array. The second integer must be greater than the first.</p>

Preference	Usage
File formatting	Set the character width at which text in the MATLAB script is wrapped by specifying Maximum characters per line .
<i>n</i> element and smaller arrays show statistics	Limit the size of arrays for which the Workspace browser displays statistics to improve performance when MATLAB updates the statistical results in the Workspace browser. For more information, see “Improve Workspace Browser Performance During Statistical Calculations” on page 5-13.
Handling NaN values in calculations	Specify whether NaN values are included or excluded from calculations for the statistics displayed in the Workspace browser. You can select to either Use NaNs when calculating statistics or Ignore NaNs when calculating statistics .

Preference	Usage
MATLAB array size limit	<p>By default, MATLAB can use up to 100% of the size of RAM (not including virtual memory) of your computer to allocate memory for each MATLAB array. To change this limit to a smaller percentage, select the Limit the maximum array size to a percentage of RAM check box. Then move the slider to adjust the percentage of RAM.</p> <p>To allow MATLAB to use both RAM and virtual memory when creating an array, clear the Limit the maximum array size to a percentage of RAM check box. If MATLAB attempts to allocate memory that exceeds the resources available on the computer, your system might become nonresponsive.</p> <p>This limit applies to the size of each array, not the total size of all MATLAB arrays.</p>

Variables Preferences

When working in the Variables editor, Variables preferences enable you to specify array formatting, cursor movement, and the decimal separator for exporting data using the system clipboard.

To open Variables preferences, on the **Home** tab, in the **Environment** section, click  **Preferences**. Select **MATLAB > Variables**.

Preference	Usage
Format	Select an option from the Default array format to specify the default array output format of numeric values displayed in the Variables editor. This format preference affects only how numbers display, not how MATLAB computes or saves them. For information on formatting options, see the reference page for the <code>format</code> function.

Preference	Usage
Editing	<p>Specify where the cursor moves to after you type an element and press Enter.</p> <p>To keep the cursor in the element where you typed, clear the Move selection after Enter check box.</p> <p>To move the cursor to another element, select the Move selection after Enter check box. In the Direction field, specify how you want the cursor to move.</p>
International number handling	<p>In the Decimal separator for exporting numeric data via system clipboard field, specify the decimal separator for numbers you cut or copy from the Variables editor and then paste into text files or other applications.</p> <p>This preference has no effect on numeric data copied from and pasted into MATLAB. Within MATLAB, decimal separators are always periods.</p>

More About

- “Create and Edit Variables” on page 5-2

Managing Files in MATLAB

- “Find Files and Folders” on page 6-2
- “Comparing Files and Folders” on page 6-6
- “Manage Files and Folders” on page 6-25
- “MathWorks File Extensions” on page 6-28
- “Files and Folders that MATLAB Accesses” on page 6-29
- “Current Folder Browser Preferences” on page 6-32
- “Specify File Names” on page 6-34
- “Create and Extract from Zip Archives” on page 6-38
- “What Is the MATLAB Search Path?” on page 6-41
- “Change Folders on the Search Path” on page 6-45
- “Use Search Path with Different MATLAB Installations” on page 6-48
- “Add Folders to the MATLAB Search Path at Startup” on page 6-49
- “Assign userpath as Startup Folder (Macintosh or UNIX)” on page 6-51
- “Path Unsuccessfully Set at Startup” on page 6-52
- “Errors When Updating Folders on Search Path” on page 6-54

Find Files and Folders

In this section...


“Simple Search for File and Folder Names” on page 6-2

“Advanced Search for Files” on page 6-2


Simple Search for File and Folder Names

If you know the name of the file you want to find, begin typing the first characters of the name in the Current Folder browser. As you type, the Current Folder browser searches downward from the top of the window, looking through all expanded folders.

Alternatively, to find a series of characters anywhere in the name of a file:

- 1 Click the search button  in the current folder toolbar. The address bar becomes a search field.
- 2 Type a portion of a file name. The asterisk character (*) is a wildcard. For example, to show only file names that begin with `coll` and have a `.m` extension, type `coll*.m`.
- 3 Press **Enter**.

MATLAB displays all files within the current folder (including its subfolders) that match that file name. If you typed the full path to a folder, that folder becomes the current folder.
- 4 Clear the results and show all items in the current folder by pressing the **Esc** key.

Tip To open the Current Folder browser if it is not open, do the following: on the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select **Current Folder**. Show and hide columns, or sort and group columns by clicking , and then selecting an option.

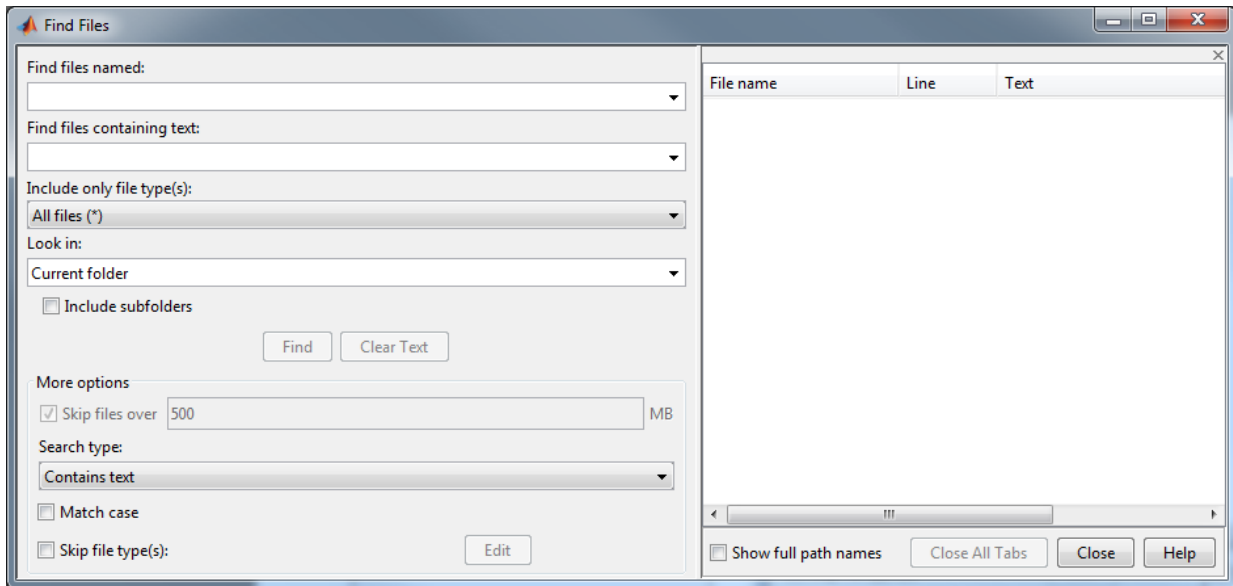
Advanced Search for Files

Use the Find Files tool to:

- Search for specific text in file names and within files
- Include or exclude specified file types from your search

- Search in specified folders
- Exclude large files from your search

To conduct an advanced search for files, open the Find Files tool. On the **Home** tab, in the **File** section, click **Find Files**. Enter your search criteria in the dialog box that opens.



Use the **Look in** menu to specify the folders you want to search. Select **Entire MATLAB Path** to search all folders on the MATLAB search path. Alternatively, you can enter the full path for one or more folders. Separate each path with a semicolon (;).

Click **Find** to begin the search. Search results appear in the right pane of the dialog box, with a summary at the bottom. For text searches, results include the line number and line of code. To see file locations, select **Show full path names**.

Open one or more files by right-clicking the files and selecting one of the **Open** options.

Include or Exclude File Types

You can search for files with only a specified extension, by selecting an option in the **Include only file type(s)** menu. For example, select ***.m**, ***.mlx** to limit the search to MATLAB program files.

To exclude some file types from the search:

- 1 In the **Include only file type(s)** menu, select **All files (*)**.
- 2 Under **More options**, select the **Skip file type(s)** box and click **Edit**. The Edit Skipped File Extensions dialog box opens.
- 3 Select the **State** check box for the file types to exclude from your search.
- 4 Click **OK** to accept your changes.

You can remove any file extension from the list by selecting the extension to highlight it. Then, click **Remove**.

Search Within File

Under **More options**, you can choose to search file contents for a partial word. From the **Search type** menu, select **Contains text**. To find an exact full-word match, select **Matches whole word**.

Searching within large files can be time consuming. To speed up your search, specify a file size in the **Skip files over** field. The Find Files tool ignores files larger than the size you specify.

Troubleshooting

If the Find Files tool does not find the file you want, try the following:

- When searching for file names, use the asterisk character (*) as a wildcard character to expand your search. For example, to show file names that begin with `coll` and have a `.m` extension, type `coll*.m`.
- Select the **Include Subfolders** check box if the file might be in a subfolder.
- If you select the **Skip file type(s)** check box, ensure that you do not exclude relevant file types from your search. Click **Edit** and review the Edit Skipped File Extensions dialog box. Ensure that relevant file types do not have the **State** check box selected.

See Also

`dir` | `exist` | `what` | `which`

More About

- “Find Functions to Use” on page 3-4

- “What Is the MATLAB Search Path?” on page 6-41

Comparing Files and Folders

In this section...

- “Comparing Files and Folders” on page 6-6
- “Comparing Folders and Zip Files” on page 6-8
- “Comparing Text and Live Scripts” on page 6-11
- “Comparing Files with Autosave Version or Version on Disk” on page 6-16
- “Comparing MAT-Files” on page 6-17
- “Comparing Variables” on page 6-20
- “Comparing Binary Files” on page 6-20
- “Using Comparison Tool Features” on page 6-21
- “Function Alternative for Comparing Files and Folders” on page 6-24

Comparing Files and Folders

You can use the Comparison tool to determine and display the differences between selected pairs of files or folders. The comparison process involves three steps:

- 1** “Select the Files or Folders to Compare” on page 6-6
- 2** “Choose a Comparison Type” on page 6-7
- 3** “Explore the Comparison Report” on page 6-7

Select the Files or Folders to Compare

You can compare files and folders using any of these methods:

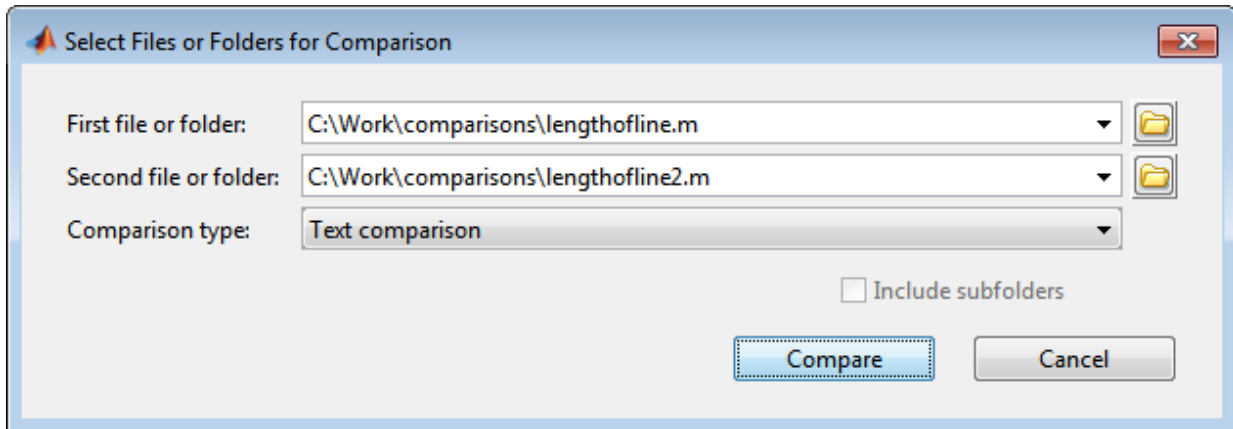
- From the Current Folder browser:
 - Select a file or folder, right-click and select **Compare Against**, and browse to select a second item to compare.
 - For two files or subfolders in the same folder, select the files or folders, right-click and select **Compare Selected Files/Folders**.
- If you have a file open in the Editor, on the **Editor** or **Live Editor** tab, in the **File** section,
 - Click **Compare** to browse to a second file for comparison.

- Alternatively, under **Compare**, select **Compare with Version on Disk** or **Save and Compare with Autosave**. See “Comparing Files with Autosave Version or Version on Disk” on page 6-16. These options are not available for live scripts.
- From the MATLAB desktop, on the **Home** tab, in the **File** section, click **Compare**. Select the files or folders to compare.
- From the command line, use the `visdiff` function.

Choose a Comparison Type

If you specify two files or folders to compare using either the Current Folder browser or the `visdiff` function, then the Comparison tool automatically performs the default comparison type.

If there are multiple comparison types available for your selections, you can change what type of comparison to run. For example, text, binary, file list, or XML comparison. To change the comparison type, create a new comparison using the Comparison tool. You can change comparison type in the Select Files or Folders for Comparison dialog box.



For example, from the Current Folder browser, if you select two MAT-files to compare, you get the default comparison type showing information about the variables. To change the comparison type to binary, create a new comparison using the Comparison tool. See “Select Files or Folders to Compare from the Comparison Tool” on page 6-21.

Explore the Comparison Report

Comparison report features depend on your comparison type. You can use the tool to:

- Compare and merge lines in two text files (some other applications refer to this as a *file diff* operation). See “Comparing Text and Live Scripts” on page 6-11.
- Compare and merge variables in two MAT-files. See “Comparing MAT-Files” on page 6-17.
- Determine whether the contents of two binary files match. See “Comparing Binary Files” on page 6-20.
- Compare any combination of folders, zip files, or Simulink manifests to determine:
 - Which file and folder names are unique to each list
 - If files and folders with the same name in each list have the same content

See “Comparing Folders and Zip Files” on page 6-8.

- Compare XML files:
 - If you select XML files to compare and you have MATLAB Report Generator™ software, the Comparison tool runs a hierarchical matching algorithm. You then see a report showing a hierarchical view of the portions of the two XML files that differ.
 - If you have Simulink Report Generator software, you can select a pair of Simulink models to compare XML files generated from them. For information, see “Model Comparison”.

Comparing Folders and Zip Files

- “Folder Comparison Report” on page 6-8
- “Highlighting of Differences” on page 6-9
- “Next Steps Using the Report” on page 6-11

Folder Comparison Report

To select items to compare, see “Select the Files or Folders to Compare” on page 6-6. You can perform *file list comparisons* for any combinations of folders, zip files, and Simulink Manifests.

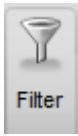
When you use the Comparison tool to compare two folders (sometimes referred to as *directories*) or any file list comparison (for example, folder versus zip file), a window opens and presents the contents side by side. The tool enables you to:

- Determine the files that the comparison lists have in common.

- Determine if files with identical names that are common to both comparison lists also have identical content.
- Open a new comparison of two files or folders that are common to both comparison lists, but have different content.
- Open a file for viewing in the Editor.
- Specify filters to ignore certain files or folders

For list comparisons, if you want to expand the list to see all files in subfolders in one report, select the **Include subfolders** check box when selecting items to compare. If you do not include subfolders, you can click **compare** links in the report to open a new comparison of two folders with changed content.

To define filters to exclude unimportant differences, on the **View** tab, click the **Filter**



button, then select **Add/Remove Filter**.

The File and Folder Filters dialog box opens. Specify filters to ignore certain files and folders, such as backup files or files created by a revision control system. Filters can save time when reviewing differences, especially when comparing many subfolders. Double-click to edit existing filters.

For example, to ignore all files and folders in a folder named **CVS**, open the File and Folder Filter dialog box and enter:

```
CVS/
```

To ignore all files in a folder named **CVS**, but not ignore subfolders, enter:

```
CVS/*
```

Highlighting of Differences

The Comparison tool displays the contents of the lists side by side and highlights files and subfolders that do not match. The following table describes how the tool highlights each type of change. The status message (such as **identical** or **contents changed**) appears in the **Difference Summary** column.

Difference Summary	Highlighting for Files and Folders	Notes
Contents changed	Pink	The contents of the files or folders differ. Click the compare link to investigate.
Added or Removed	Green	File or folder only exists in the left or right list.
Identical	None	

The following image shows an example of the Comparison tool when two folders are compared. The results are sorted by **Type**.

Comparing folder **curvefitting** vs. folder **curvefitting2**

Left file list Contents of folder C:\Work\comparisons\curvefitting
 Right file list Contents of folder C:\Work\comparisons\curvefitting2

Click on a column header to sort the table

Type	File Name	In left list (folder curvefitting)		In right list (folder curvefitting2)		Difference Summary
		Size (bytes)	Last Modified Date	Size (bytes)	Last Modified Date	
folder	cftoolgui/	-	2012-03-07 14:24:57	-	2012-03-07 14:25:04	contents changed (compare)
MATLAB Code File	csapidem.m (open)	(not in this list)		15038	2009-01-08 13:56:28	added
folder	curvefit/	-	2012-03-07 14:25:00	-	2012-03-07 14:25:08	contents changed (compare)
folder	demosearch/	-	2012-03-07 14:25:01	-	2012-03-07 14:25:09	identical
folder	html/	-	-	(not in this list)		removed
Text Document	kcdm.txt (open)	(not in this list)		1367	2008-10-03 11:13:37	added
Editor Autosave	lengthofline.asv (open)	(not in this list)		2403	2009-11-16 15:24:45	added
MATLAB Code File	lengthofline.m (open left right)	2405	2009-11-12 16:56:52	2408	2009-11-16 15:25:14	contents changed (compare)
Firefox HTML Document	manifest_report.html (open)	2410	2008-11-27 14:24:50	(not in this list)		removed
XML File	report.xml (open)	(not in this list)		4868	2008-09-11 17:53:32	added
folder	sftoolgui/	(not in this list)		-	-	added
MAT-file	splinetool.mat (open)	1720	2001-08-20 18:14:12	(not in this list)		removed

Next Steps Using the Report

To explore the report you can use the following tools:

- You can sort the results by name, type, size, or last modified timestamp by clicking the column headers. For example, click the **Type** column header to sort by folder and file type, as shown in the preceding figure.
- To open a new comparison of two files or folders with changed contents, click the **compare** link next to file or folder names highlighted in pink.
- To open a file in the Editor, click the **open** link next to a file name.

If the file is present in both folders, you can click links to open the **left** or **right** file.

- If subfolders are very large and contain many files, analysis continues in the background. The tool displays the number of items still to be compared at the top of the report, as shown in the next figure. You can click the links to **Skip Current** item or **Cancel All** to stop further analysis.



- For details on other comparison tool features, see “Using Comparison Tool Features” on page 6-21.

Comparing Text and Live Scripts

- “Select Text Files to Compare” on page 6-11
- “Highlighting of Differences” on page 6-12
- “Step Through Differences” on page 6-14
- “Merge Differences” on page 6-14
- “View a Summary of Differences” on page 6-15
- “Ignore White-Space Differences in Text Comparisons” on page 6-16
- “Show Differences Only” on page 6-16
- “Change the Display Width of a Text Comparison” on page 6-16
- “Save HTML Report” on page 6-16

Select Text Files to Compare

To select files to compare, see “Select the Files or Folders to Compare” on page 6-6.

To view an example text comparison, enter:

```
visdiff(fullfile(matlabroot,'help','techdoc','matlab_env',...  
'examples','lengthofline.m'), fullfile(matlabroot,'help',...  
'techdoc','matlab_env','examples','lengthofline2.m'))
```

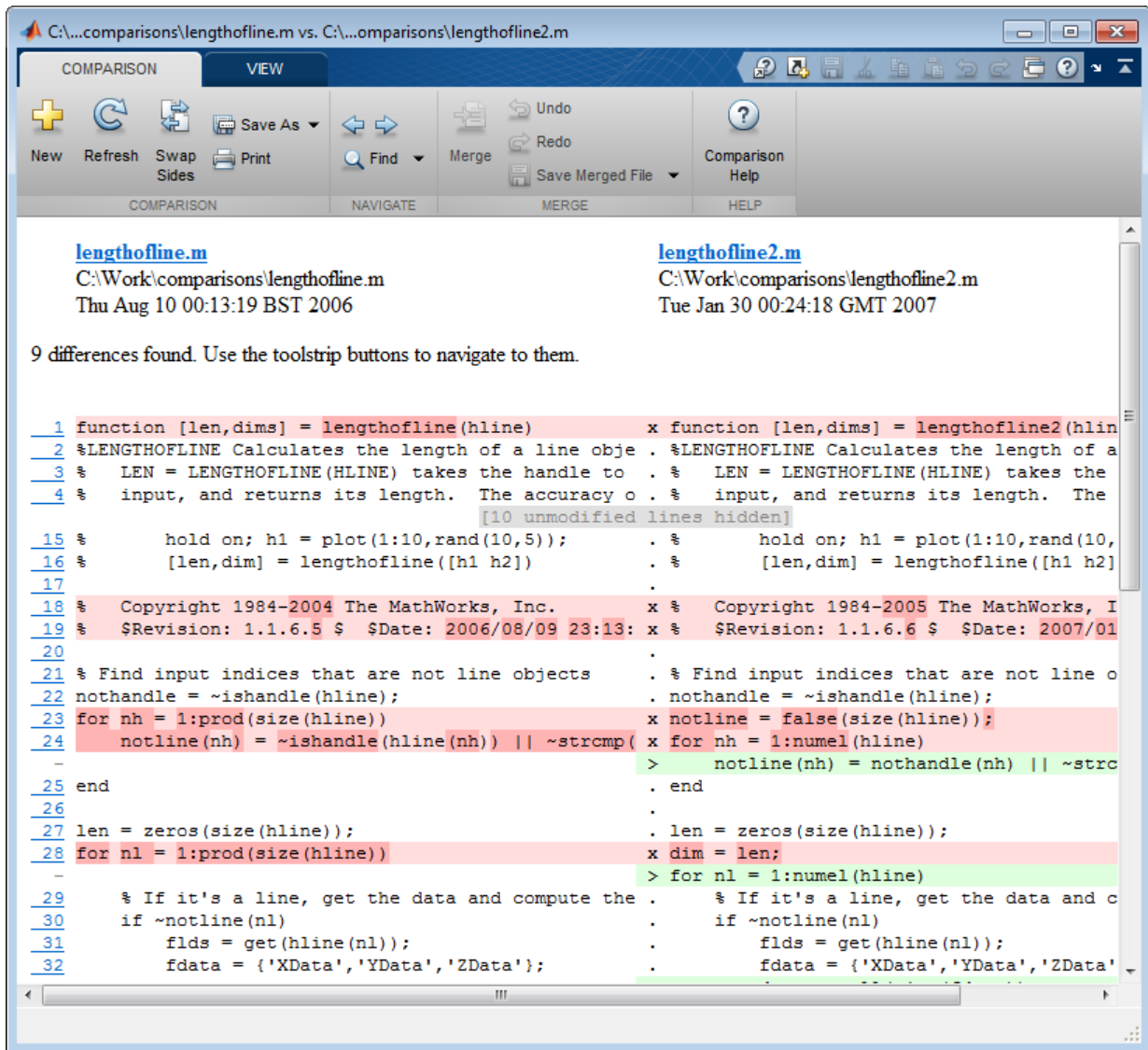
Highlighting of Differences

When you use the Comparison tool to compare two text files, a window opens and presents the two files side by side. Symbols indicate how you can adjust the files to make them match. This feature can be useful when you want to compare the latest version of a text file to an autosave version.

Note: When you compare live scripts, the Comparison tool converts all formatted content to publish markup. This generates the same code as when you save a live script as a MATLAB Code file (.m). The files are then compared as text. The tool does not compare stored output.

The Comparison tool report displays the files side by side and highlights lines that do not match, as follows:

- Dark pink highlighting indicates changed characters within lines.
- Pink highlighting and an **x** between the two files indicate that the content of the lines differs between the two files.
- Green highlighting and a right (>) or left (<) angle bracket between the two files indicate a line that exists on one side only.



The Comparison tool attempts to match lines and detects local text that is added, deleted, or changed. It does not do a simple line-by-line comparison. In the previous image, for example, the tool determines that `lengthofline.m` has a line of code that does not exist in `lengthofline2.m` and highlights it (line 24) in green. Also, notice that the tool

takes the additional line into account and determines that the line containing the `end` statement in each file matches, even though the `end` statement does not occur on the same line number.

If the files you are comparing are extremely long, the tool could run out of memory while attempting to perform the file comparison. In which case, the message,

```
Maximum file length exceeded.  
Defaulting to line-by-line comparison.  
appears. In a line-by-line comparison, the tool highlights the lines containing the end statement because in performing this operation, it finds that the last line in one file does not match the last line in the other file.
```

Step Through Differences

Because text files can be lengthy, the Comparison tool provides toolstrip buttons to help you step through the results from one difference to the next.



To navigate through comparison results:

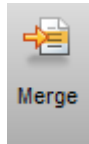
- Click the right arrow button to go to the next set of lines that differ.
If no additional sets of lines differ, the right arrow takes you to the end of the file.
- Click the left arrow button to go to a previous set of lines that differ.
If no previous set of lines differ, the left arrow takes you to the beginning of the file.

Merge Differences

When comparing text files you can merge changes from one file to the other. Merging changes can be useful when resolving conflicts between different versions of files. Merging is not available when comparing live scripts.

Tip You can only merge from left to right. If you want to merge into the other file, use **Swap Sides** before you start merging. **Swap Sides** reverts any merges already made and creates a new comparison report for the original files.

- 1 Select a difference in the report and click the **Merge** button to copy the selected difference from the left file to the right file.



Merged differences display gray row highlighting, and a green merge arrow.

```
1 function [len,dims] = lengthofline(hline)      function [len,dims] = lengthofline(hline)      1 →
```

The merged file name at the top of the report displays the dirty flag (*filename.m**) to show you that the file contains unsaved changes.

- 2 To revert the last merge operation, click **Undo** in the Merge section. You can click **Undo** repeatedly, or **Redo** to reapply a merge.

Tip You can click **Swap Sides** to start again and revert all merges.

- 3 To save your changes, click **Save Merged File**. To save to a different name, select **Save Merged File > Save Merged File As**.
- 4 If you want to inspect the files in the Editor, click the line number links in the report.

Tip Save your merge changes from the comparison report before making any changes in the Editor, otherwise the comparison report can become incorrect. The report does not update to reflect changes you make in the Editor.

View a Summary of Differences

To see a summary of differences between two text files, scroll to the bottom of the Comparison tool and view the list, which contains information such as:

- Number of matching lines: 51
- Number of unmatched lines in left-hand file: 13
- Number of unmatched lines in right-hand file: 16

Ignore White-Space Differences in Text Comparisons

You may want to hide white-space differences to help you distinguish between functional changes and changes to indentation.

On the **View** tab, click the **Filter** button, and check or uncheck the **Ignore White Space** item to toggle the display of differences only involving white-space characters.

Show Differences Only

You can specify whether to show only differences or entire files. It can be useful to hide unmodified lines in large text comparison reports. When you are showing differences only and sections are hidden, the report displays messages like the following: 10 unmodified lines hidden.

On the **View** tab, click the **Filter** button, and check or uncheck the **Show Differences Only** item to toggle the display of sections of the report that do not contain any differences.

Change the Display Width of a Text Comparison

You can increase or decrease the line lengths of the text files in the comparison display. On the **View** tab, in the **Display** section, edit the number in the **Column Width** edit box. Resize the window, if necessary.

For details on other comparison tool features, see “Using Comparison Tool Features” on page 6-21.

Save HTML Report

On the **Comparison** tab, in the **Comparison** section, click **Save As > Save as HTML** to save a copy of the comparison report as an HTML file.

Comparing Files with Autosave Version or Version on Disk

From the Editor you can compare one open text file with another, or you can choose to compare the latest version of a file in the Editor to an autosave version or a saved version. For an example, follow these steps:

- 1 Open one of the text files you want to compare in the Editor.

To open the example file provided, `lengthofline.m`, run the following command in the Command Window:

```
open(fullfile(matlabroot,'help','techdoc','matlab_env',...  
'examples','lengthofline.m'))
```

- 2 On the **Editor** or **Live Editor** tab, in the **File** section, click **Compare**. If your file is modified, the Editor saves the file before comparing. Alternatively, under **Compare**, select **Save and Compare with**.

Navigate to the file you want to compare against, select the file, and click **Open**. For example, select the example file `lengthofline2.m` from the folder where you found `lengthofline.m`.

Other options available are:

- To compare the open file to the Editor's automatic copy (*filename.asv*), under **Compare**, select **Save and Compare with Autosave**. If your file is modified, the Editor saves the file before comparing. For more information, see “Backing Up Files”.
- To compare an open file that has been changed, but not saved, to the saved version, under **Compare**, select **Compare with Version on Disk**.

Comparing MAT-Files

Note: To select files to compare, see “Select the Files or Folders to Compare” on page 6-6.

You can use the Comparison tool to compare two MAT-files. The tool presents the variables in the two files side by side, which enables you to:


- View and sort by the name, size, class, and change summary of all variables.
- View details of differences between variables, to see which fields of a structure are different, and view differences in individual elements of an array.
- Merge changes between files by copying modified variables from one file to the other (*Caution:* No undo).
- See which variables are common to each file and which are unique.
- Load the contents of the variables into the Variable Editor by clicking the name of that variable.
- Load the MAT-files into the workspace by clicking a **Load** link.

- Save a copy of the report as an HTML file. Click **Save As > Save as HTML** on the toolbar.

The Comparison tool report highlights changes in variables as follows.

Difference Summary	Highlighting	Notes
Modified	Pink	Values of the variable differ between the two files. Click the compare link to investigate. A new variable comparison report opens to display differences in individual array elements or differing fields of a structure. Double-click pink rows or cells to investigate further layers of differences.
Added or Removed	Green	Variable only exists in left or right file.
Equivalent	None	The variables in both files are equivalent. The tool ignores differences in NaN patterns, field ordering in structs, and negative zero or positive zero.
Class changed	Pink (only in Class columns)	Variable data class changed. Click the compare link to investigate.

The report displays a message if the variables in both files are equivalent, but the files are not identical. Possible causes of the differences between the files include: file formats, file timestamps, the order in which the variables are stored, or variables contain ignored differences.

In the **Merge** column, click the Merge button  to copy modified variables from one file to the other.

The following image shows the results when you compare two files, `data1.mat` and `data2.mat`.

C:\Work\comparisons\data1.mat vs. C:\Work\comparisons\data2.mat









COMPARISON VIEW

New Refresh Swap Sides Save As Print Find Comparison Help

File Comparison - data1.mat vs. data2.mat

Left file	C:\Work\comparisons\data1.mat
Right file	C:\Work\comparisons\data2.mat

Click on a column header to sort the table

Variables in data1.mat			Variables in data2.mat			Difference Summary	Merge (no undo)
Name	Size	Class	Name	Size	Class		
a	1x2	cell	<i>(not in this file)</i>			<i>Removed</i>	 
<i>(not in this file)</i>			b	1x5	char	<i>Added</i>	 
p	6x1	uint32	p	6x1	uint32	<i>Equivalent</i>	
q	6x1	uint32	q	6x1	uint32	<i>Equivalent</i>	
x	1x1	double	x	1x1	double	<i>Equivalent</i>	
y	1x2	double	y	1x3	double	<i>Modified (compare)</i>	 
z	2x2x2	double	z	2x2x2	int32	<i>Class changed (compare)</i>	 

[Load C:\Work\comparisons\data1.mat](#)
[Load C:\Work\comparisons\data2.mat](#)

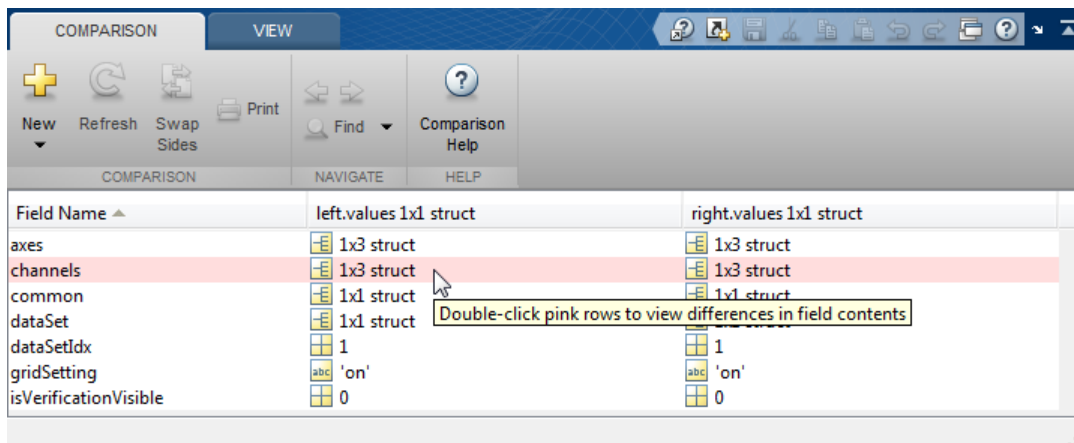
If values of the variable differ between the two files, you can click the **compare** link to investigate. A new variable comparison report opens. See “Comparing Variables” on page 6-20.

To view an example MAT-file comparison, enter:

```
visdiff(fullfile(matlabroot,'toolbox','matlab','demos','gatlin.mat'), ...
fullfile(matlabroot,'toolbox','matlab','demos','gatlin2.mat'))
```

Comparing Variables

The variable comparison report displays differences in individual array elements or differing fields of a structure. Double-click pink rows or cells to investigate further layers of differences, as shown in the following example.



Comparing Binary Files

To select files to compare, see “Select the Files or Folders to Compare” on page 6-6.

Note: If you are comparing SLX files, and do not have Simulink Report Generator, you see a binary comparison. For information on SLX comparison reports instead, see “Model Comparison”.

You can use the Comparison tool to compare two binary files such as DLL files or MEX-files. Also, you can select the **Binary comparison** type for any pair of files with a choice of comparison types.

- If the files are the same, the tool displays the message: `The files are identical.`
- If the files differ, the tool displays the message: `The files are different.`

If the files differ, you can click the **Show Details** link to view the binary files and the byte offset of the first difference.

To view an example binary comparison, compare two example text files and specify comparison type as binary:

```
visdiff(fullfile(matlabroot,'help','techdoc','matlab_env',...  
'examples','lengthofline.m'), fullfile(matlabroot,'help',...  
'techdoc','matlab_env','examples','lengthofline2.m'), 'binary')
```

Using Comparison Tool Features

You can use the Comparison tool for the following tasks:

- “Select Files or Folders to Compare from the Comparison Tool” on page 6-21
- “Exchange the Left and Right Sides of the Report” on page 6-22
- “Refresh the Report to Show Updated Files” on page 6-22
- “Find Text” on page 6-23
- “Create New Comparisons” on page 6-23
- “Change Color Preferences” on page 6-23

Select Files or Folders to Compare from the Comparison Tool


To compare two files or folders from the Comparison tool, follow these steps:

- 1 From the MATLAB desktop, on the **Home** tab, in the **File** section, click **Compare**. Select the files or folders to compare.

If the Comparison tool is already open, compare files or folders by clicking the **New** button.



The dialog box **Select Files or Folders for Comparison** appears.

- 2 In the dialog box, select two files or folders to compare. Use the drop-down lists to select recent comparison items, or the **Browse** buttons  to locate and select the items that you want to compare.

You also can drag and drop a file or folder from Windows Explorer to the first and second file and folder fields.

- 3 Optionally, choose the comparison type you want to use. Either use the default **Comparison type** value, or if multiple comparison types are available, select a different one from the list. For example, for text files you could select text or binary comparison types.
- 4 Click **Compare**.

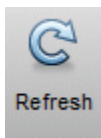
Exchange the Left and Right Sides of the Report

To move the file or folder on the left side to the right side and vice versa, on the **Comparison** tab, in the **Comparison** section, click the **Swap Sides** button.



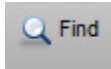
Refresh the Report to Show Updated Files

After making changes to and saving the files in the Editor, to update the results in the Comparison tool, on the **Comparison** tab, in the **Comparison** section, click the **Refresh** button.



Find Text

To find a phrase in the current display, on the **Comparison** tab, in the **Navigate** section, click the **Find** button.




The resulting Find dialog box is the same as the one you use in the Command Window. For more information, see “Find Text in Command Window or History” on page 3-12.

Create New Comparisons

To perform another file or folder comparison, on the **Comparison** tab, in the **Comparison** section, click the **New** button.



The dialog box Select Files or Folders for Comparison appears, with the last comparison files preselected in the first and second file fields. Use the drop-down lists to select recent comparison items, or the **Browse** buttons  to locate and select the items that you want to compare.

New comparisons open additional comparison reports.

Change Color Preferences

You can change and save your diff color preferences for the Comparison tool. You can apply your color preferences to all comparison types.

- 1 On the MATLAB Home tab, click **Preferences**.
- 2 In the Preferences dialog box, under **MATLAB**, click **Comparison**.
- 3 Edit color settings as desired for differences, modified lines, modified contents, and merged lines. View the colors in the **Sample** pane.

The **Active Settings** list displays **Default (modified)**.

- 4 To use your modified settings in the comparison, click **Apply** and refresh the comparison report.

- 5 To return to the default settings, in the Preferences dialog box, click **Reset** and click **Apply**. Refresh the comparison report.
- 6 If you want to save your modified color preferences for use in future MATLAB sessions, click **Save As**. Enter a name for your color settings profile and click **OK**.

After saving settings, you can select them in the **Active Settings** list.

Function Alternative for Comparing Files and Folders

Use the `visdiff` function to open the Comparison tool from the Command Window.

```
visdiff(fileorfoldername1, fileorfoldername2)
```

For example, type:

```
visdiff('lengthofline.m', 'lengthofline2.m')
```

See Also


`visdiff`


Related Examples

- “Model Comparison”

Manage Files and Folders

This table shows how to create, open, move, and rename files and folders.

Action	Tools Workflow	Function Alternative
Create a new folder	<p>In the Current Folder browser, right-click in white space, and then select New Folder.</p> <p>MATLAB creates and selects a folder named New Folder in the current folder.</p>	<p>Use the <code>mkdir</code> function. For example, create a subfolder named <code>newdir</code> in a parent folder named <code>parentFolder</code>:</p> <pre>mkdir('parentFolder', 'newdir');</pre>
Move a file or folder	<p>In the Current Folder browser, click and drag the item.</p> <p>You cannot move a folder while it is on the search path.</p>	<p>Use the <code>movefile</code> function. For example, move the file named <code>myfile.m</code> in the current folder to the folder, <code>d:/work</code>:</p> <pre>movefile('myfile.m', 'd:/work');</pre>
Rename a file or folder	<p>In the Current Folder browser, right-click the item and select Rename.</p> <p>File names must start with a letter, and can contain letters, digits, or underscores.</p>	<p>Use the <code>movefile</code> function. For example, in the current folder, rename <code>myfile.m</code> to <code>oldfile.m</code>:</p> <pre>movefile('myfile.m', 'oldfile.m');</pre>
Open a file in MATLAB	<p>On the Home tab, in the File section, click Open , and then select a file to open.</p> <p>To open a file in the Editor as a text file, even if the file type is associated with another application or tool right-click the file in the Current Folder browser and select Open as Text.</p>	<p>Use the <code>open</code> function. The file opens in MATLAB or in an external application, depending on the file extension.</p>
Open a file in another program	<p>In the Current Folder browser, right-click the file and select Open Outside MATLAB. The</p>	

Action	Tools Workflow	Function Alternative
	file opens in the application or tool that the operating system associates with the file type.	
Preview file contents without opening the file	In the Current Folder browser, right-click the file and select Show Details . The Details Panel expands. This feature is not available in live scripts.	none
Delete a file or folder	<p>In the Current Folder browser, select the file or folder and press Delete.</p> <p>By default, MATLAB deletes or recycles files and folders according to your operating system preferences. To permanently remove the selection when the system preference is set to recycle, press Shift+Delete</p> <p>On Linux systems, you can request that MATLAB move deleted files to a temporary folder by setting the Deleting files preference. Access this preference on the Home tab, in the Environment section, by clicking  Preferences. Select MATLAB > General.</p>	<p>To delete a file, use the <code>delete</code> function. For example, delete a file named <code>myfile.m</code> in the current folder:</p> <pre>delete('myfile.m');</pre> <p>By default, files are permanently removed. To move deleted files to a temporary folder instead, use the <code>recycle</code> function or set the Deleting files preference.</p> <p>To delete a folder, use the <code>rmdir</code> function.</p> <hr/> <p>Note: You cannot recover folders deleted using <code>rmdir</code>.</p>

To open the Current Folder browser if it is not visible, do the following: on the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select **Current Folder**. Double-clicking a subfolder displays its contents, and makes that folder the current folder.

See Also

`delete` | `edit` | `mkdir` | `movefile` | `open` | `recycle` | `rmdir`

More About

- “Save and Load Workspace Variables” on page 5-15
- “Run Functions in the Editor”
- “Errors When Updating Folders on Search Path” on page 6-54

MathWorks File Extensions

This table lists some common MathWorks file extensions.

File Extension	Description
.fig	MATLAB Figure.
.m	MATLAB Code — A MATLAB script, function, or class.
.mlx	MATLAB Live Script — A MATLAB live script.
.mat	MATLAB Data — Binary file that stores MATLAB variables.
.mdl .slx	Simulink Model.
.mdlp .slxp	Simulink Protected Model.
.mexa64 .mexmaci64 .mexw32 .mexw64	MATLAB MEX — Shared library files that can be dynamically loaded into MATLAB. The MEX-file extensions are platform-dependent.
.mlapp	MATLAB App File — App created using MATLAB App Designer.
.mlappinstall	MATLAB App Installer File — Opens Installer for MATLAB app.
.mlpkginstall	Support Package Installer File — Opens Installer for a support package.
.mltbx	MATLAB Toolbox File — Opens Installer for a toolbox.
.mn	MuPAD [®] Notebook
.mu	MuPAD Code
.p	MATLAB P-Code — Protected function file.
.sldd	Simulink Data Dictionary.

More About

- “Associating Files with MATLAB on Windows Platforms” on page 1-4

Files and Folders that MATLAB Accesses

In this section...

“Where Does MATLAB Look for Files?” on page 6-29

“Files and Folders You Should Add to the Search Path” on page 6-29

“When Multiple Files Have the Same Name” on page 6-30

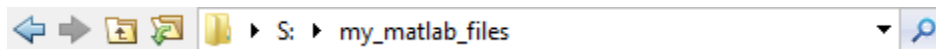
“Locations of MathWorks Products” on page 6-30

Where Does MATLAB Look for Files?

When you do not specify a path to a file, MATLAB looks for the file in the current folder or on the search path. Functions in the current folder take precedence over functions with the same file name that reside anywhere on the search path. To identify the current folder, type `pwd` in the Command Window.

To make files accessible to MATLAB, do one of the following:

- Change the current folder to the folder that contains the files. Use the `cd` function or browse to a different folder in the current folder toolbar:



- Add the folder that contains the files to the search path. Changes you make to the search path apply to the current MATLAB session. To reuse the modified search path in future MATLAB sessions, save your changes.
- Store individual files in the *userpath* MATLAB folder, which is on the search path. To determine the location of this folder, run the `userpath` function.

Files and Folders You Should Add to the Search Path

The MATLAB search path should include:

- Folders containing files that you run.
- Folders containing files that are *called by* files you run.
- Subfolders containing files that you run. Making a folder accessible does not make its subfolders accessible.

For files in @ (class) and + (package) folders, make the parent folder accessible. For details, see “Folders Containing Class Definitions”.

If files call other files that are in multiple folders, determine the location of all the called files by creating a Dependency Report. See “Dependencies Within a Folder”

When Multiple Files Have the Same Name

Name conflicts arise when MATLAB has access to multiple files with the same name, and when a file has the same name as a variable in the base workspace or a built-in function for a MathWorks product.

When there are name conflicts, MATLAB follows these precedence rules:

- “Function Precedence Order”
- “Class Precedence and MATLAB Path”

The file that MATLAB does *not* use is called a *shadowed* file. In some cases, MATLAB warns you that a shadowed file exists.

Locations of MathWorks Products

Files and folders for products provided by MathWorks are in *matlabroot*/toolbox. The files and folders under *matlabroot* are important to your installation. In particular:

- Do *not* store your personal files and folders in *matlabroot*/toolbox.
- Do *not* change files, folders, and subfolders in *matlabroot*/toolbox. The exception is the `pathdef.m` file, which you can update and save in its default location, *matlabroot*/toolbox/local.

To see a list of all toolbox folder names supplied with MathWorks products, run:

```
dir(fullfile(matlabroot,'toolbox'))
```

See Also

`cd` | `pwd` | `userpath`


More About

- “What Is the MATLAB Search Path?” on page 6-41

- “MATLAB Startup Folder” on page 1-16
- “Toolbox Path Caching in MATLAB” on page 1-24

Current Folder Browser Preferences

You can specify the number of files that display in the Current Folder browser, and customize their appearance.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Then, select **MATLAB > Current Folder**.

Preference	Usage
History	Specify the number of recently used folders maintained in the Current Folder Toolbar drop-down list.
Refresh	<p>Specify how frequently the Current Folder browser updates to reflect changes to files made from programs and tools other than MATLAB.</p> <p>When you access files on a network, frequent refreshing of the Current Folder browser can slow performance in MATLAB. If this seems to be a problem, try increasing the value for Number of seconds between auto-refresh. Alternatively, clear the Auto-refresh view from file system selection to disable automatic refresh.</p> <p>To manually refresh the view at any time, right-click in the Current Folder browser and select Refresh.</p>
Path indication	<p>Control the appearance of folders and files that are inaccessible to MATLAB, and whether to display tooltips describing their status. MATLAB cannot access files if they are not on the search path or, in some cases, if they are in a private folder.</p> <p>Select the Indicate inaccessible files check box to dim the display of files and folders inaccessible to MATLAB. Move the Text and icon transparency slider to adjust the level of dimming.</p> <p>Select the Show tooltip explaining why files are inaccessible check box to display a tooltip that provides information on why a dimmed file is inaccessible, when you hover over it.</p>

Preference	Usage
	If you do not select the Indicate inaccessible files check box, then the Current Folder browser displays all files and folders as undimmed and provides no tooltips.
Toolbar	Access the Toolbars preferences to adjust the toolbar layout and controls for Desktop tools, including the Current Folder browser.
Initial working folder	Access the General preferences to specify the current folder in MATLAB when it starts.
Hidden Files	<p>Specify whether the Current Folder browser displays files and folders that the operating system hides from system file browsers and file-listing commands.</p> <p>This preference does not apply to Microsoft Windows platforms. On Windows platforms, the Current Folder browser follows the Windows preference for showing hidden files. To set or change the Windows preference, access the Folder Options, and then select an option for viewing Hidden files and folders.</p>

To change how dates display in the Current Folder browser, change the short date format for your operating system. Then, refresh the date display: right-click in the Current Folder browser and select **Refresh**. MATLAB uses your operating system's short date format to display dates in both the Current Folder browser and the Command History.

Specify File Names

In this section...
“Construct Valid Path and File Names” on page 6-34
“Case Sensitivity of File Names” on page 6-36

Construct Valid Path and File Names

Specify Path Names

A path name specifies file locations, for example, `C:\work\my_data` (on Microsoft Windows platforms) or `/usr/work/my_data` (on Linux or Mac platforms). If you do not specify a path name when accessing a file, MATLAB first searches in the current folder. To indicate a file in a particular location, specify a path name.

Path name specifications differ, depending on the platform on which you are running MATLAB. Use the `fullfile` function to construct path names in statements that work on any platform. This function is particularly useful when you provide code to someone using it on a platform other than your own.

`fullfile` inserts platform-specific file separators where necessary. The file separator character is the symbol that distinguishes one folder level from another in a path name. A forward slash (`/`) is a valid separator on any platform. A backward slash (`\`) is valid only on Microsoft Windows platforms. In the full path to a folder, the final slash is optional. Type `filesep` in the Command Window to determine the correct file separator character to use on your platform.

To identify the platform on which MATLAB is currently running, use the `ismac`, `ispc`, and `isunix` functions.

Characters Within File and Folder Names

File names must start with a letter, and can contain letters, digits, or underscores.

Avoid using accent characters such as umlauts or circumflexes in path names. MATLAB might not recognize the path. In addition, attempts to save a file to such a path might result in unpredictable behavior.

If a path or file name contains spaces, enclose the input in single quotes. For example:


```
load 'filename with space.mat'  
or  
load('filename with space.mat')
```

Absolute and Relative Path Names

MATLAB always accepts *absolute* path names (also called *full* path names), such as `I:/Documents/My_Files`. An absolute path name can start with any of the following:

- UNC path '\\ '.
- Drive letter, on Microsoft Windows platforms, such as `C:\`.
- `/` character on Linux platforms.

Some MATLAB functions also support relative path names. Unless otherwise noted, the path name is relative to the current folder. For example:

- `myfile.m` refers to the `myfile.m` file in the current folder.
- `/myfolder` refers to the `myfolder` folder in the current folder.
- `../myfolder/myfile.m` refers to the `myfile.m` file in the `myfolder` folder, where `myfolder` is at same level as the current folder. Each repetition of `../` at the beginning of the path moves up an additional folder level.

Tip If multiple documents are open and docked in the Editor, you can copy the absolute path of any of these documents to the clipboard. This practice is useful if you need to specify the absolute path in another MATLAB tool or an external application. Right-click the document tab, and then select **Copy Full Path to Clipboard**

Partial Path Names in MATLAB

A partial path name is the last portion of a full path name for a location on the MATLAB search path. Some functions accept partial path names.

Examples of partial path names are: `matfun/trace`, `private/cancel`, and `demos/clown.mat`.

Use a partial path name to:

- Specify a location independent of where MATLAB is installed.

- Locate a function in a specific toolbox when multiple toolboxes contain functions with that name. For example, to open the file for the `set` function in the Database Toolbox™ product, type:

```
open database/set
```

- Locate method files. For example, to check if a `plot` method exists for the time series object, type:

```
exist timeseries/plot
```

Specifying the at sign character (@) in method folder names is optional.

- Locate private and method files, which sometimes are hidden.

Be sure to specify enough of the path name to make the partial path name unique.

Maximum Length of Path Names in MATLAB

The maximum length allowed for a path name depends on your platform.

For example, on Microsoft Windows platforms:

- The maximum length is known as `MAX_PATH`.
- You cannot use an absolute path name that exceeds 260 characters.
- For a relative path name, you might need to use fewer than 260 characters. When the Windows operating system processes a relative path name, it can produce a longer absolute path name, possibly exceeding the maximum length.

If you get unexpected results when working with long path names, use absolute instead of relative path names. Alternatively, use shorter names for folders and files.

Case Sensitivity of File Names

In general, it is best to specify path and case precisely when specifying a file name.

Case Sensitivity When Calling Functions

You call function files by specifying the file name without the file extension. MATLAB returns an error if it cannot find a case-sensitive match on the search path. By default, MATLAB suggests a function with the correct case.

When multiple files have the same name, MATLAB follows precedence rules to determine which to call. For more information, see “Function Precedence Order”.

Case Sensitivity When Loading and Saving Files

Linux platforms — File names are case sensitive.

- When loading or reading from a file, specify the file name using the correct case.
- When saving or writing to a file, MATLAB saves the file in the case you specify. Two files with the same name, but different cases can exist in the same folder.

Windows platforms — File names are case insensitive. The Windows operating system considers two files with the same name to be the same file, regardless of case. Therefore, you cannot have two file names that differ only by case in the same folder.

- When loading or reading from a file, MATLAB accesses the file with the specified name that is higher on the search path, regardless of case. For example, if you attempt to load MYFILE and `myfile.mat` is higher on the search path than MYFILE.MAT, then MATLAB loads `myfile.mat` without warning that there is a case mismatch.
- When saving or writing to a file, if you specify a file name that already exists in the folder, MATLAB accesses the existing file without warning. For example, if you save data to a file named `myfile` using the `save` function, and MYFILE.mat already exists in the folder, the data replaces the contents of MYFILE.mat. However, the file name remains MYFILE.mat.

See Also

`filesep` | `fullfile` | `ismac` | `ispc` | `isunix` | `which`

More About

- “What Is the MATLAB Search Path?” on page 6-41

Create and Extract from Zip Archives

In this section...
“Create a Zip Archive” on page 6-38
“Add Files to a Zip Archive” on page 6-39
“Extract Files from a Zip Archive” on page 6-39
“Compare Zip Archive to Unzipped Files” on page 6-40

Create a Zip Archive

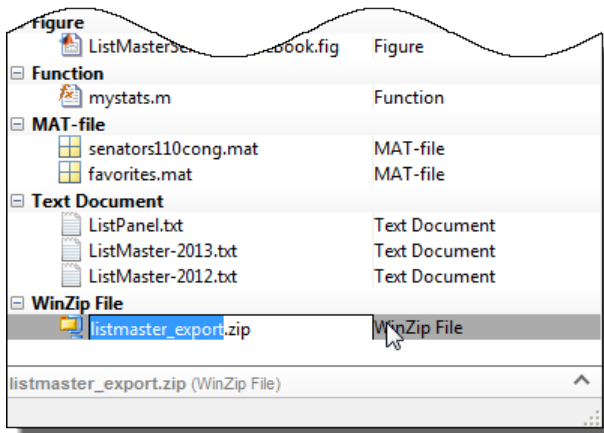
Create archives using zip files to back up files, conserve file storage space, or to share collections of files with others. You can either create an empty archive, or select files and folders to create an initial archive. In either case, you can add more files later.

Create zip archives interactively using the Current Folder browser:

- To create an empty zip file, right-click white space, and then select **New File > Zip File**.
- To create a populated zip file from selected files and folders, select the folders and files you want to archive, right-click, and then select **Create Zip File**.

MATLAB creates an archive with a default name of `Untitledn.zip`, where *n* is an integer.

Type over the default file name to specify a descriptive name, for example `listmaster_export.zip`, as shown here.



You also can create zip archives programmatically using the `zip` function. For example, to zip all files with a `.m` and `.mat` extension in the current folder to a zip file archive named `backup.zip`, call:

```
zip('backup', {'*.m', '*.mat'});
```

Add Files to a Zip Archive

To add files and folders to a zip file archive in the Current Folder browser, do one of the following:

- Select, and then drag the file that you want to add onto the archive.
- Copy the file that you want to add to the archive. Then, select the archive to which you want to add the file and paste the file into the archive.

Extract Files from a Zip Archive

To extract a single file from within a zip file archive in the Current Folder browser:

- 1 Expand the zip file archive to view the archive contents, by clicking the associated + (expand) button. By default, files within a zip file archive appear dimmed to indicate that they are not on the MATLAB path.
- 2 Drag the file into a folder in the Current Folder browser.

MATLAB extracts the file and saves it to the folder where you dragged or pasted it.

To extract all the contents of a zip file, double-click the zip file in the Current Folder browser. MATLAB creates a folder with the same name as the zip file, and extracts the entire contents of the zip file into this folder.

To extract the contents of a zip file programmatically, use the `unzip` function. `unzip` also allows you to specify a target folder. For example, to unzip the file, `examples.zip`, to a folder named `myfolder`, call:

```
unzip('examples.zip', 'myfolder')
```

Note: Archives created outside of MATLAB can be encrypted or password-protected. You cannot add files to, or extract files from, protected archives from within MATLAB.

Compare Zip Archive to Unzipped Files

To determine differences between archived and unarchived files, use the Comparison Tool from within the Current Folder browser as you would for any other files and folders.

- Right-click a zip archive, and then from the context menu select **Compare Against** and specify the folder to which you want to compare the contents of the zip archive.
- Expand a zip archive, right-click a file within it, and then from the context menu select **Compare Against**. Specify the file to which you want to compare the archived file.

See Also

`unzip` | `zip`

More About

- “Comparing Files and Folders” on page 6-6

What Is the MATLAB Search Path?

The MATLAB search path, is a subset of all the folders in the file system. MATLAB uses the search path to locate files used with MathWorks products efficiently.

The order of folders on the search path is important. When files with the same name appear in multiple folders on the search path, MATLAB uses the one found in the folder nearest to the top of the search path.

By default, the search path includes

- The MATLAB *userpath* folder, which is added to the search path at startup, and is the default location for storing user files
- The folders defined as part of the `MATLABPATH` environment variable
- The folders provided with MATLAB and other MathWorks products, which are under *matlabroot*/toolbox, where *matlabroot* is the folder displayed when you run `matlabroot` in the Command Window

Class, package, and `private` folders should *not* be specified explicitly as part of the search path.

You can explicitly add folders to the search path for the files you run. For more information about adding files to the search path, see “Change Folders on the Search Path” on page 6-45.

userpath Folder on the Search Path

The *userpath* folder is first on the search path. By default, MATLAB adds the *userpath* folder to the search path at startup. This folder is a convenient place for storing files that you use with MATLAB.

The default *userpath* folder is platform-specific.

- Windows platforms — `%USERPROFILE%/Documents/MATLAB`.
- Mac platforms — `$home/Documents/MATLAB`.
- Linux platforms — `$home/Documents/MATLAB` if `$home/Documents` exists.

Alternatively, to determine or change the current *userpath* folder, call `userpath`.

By default, the *userpath* folder is also the startup folder when you start MATLAB by double-clicking either the MATLAB shortcut on Windows systems or the MATLAB application on Mac systems.

MATLABPATH Environment Variable

The MATLABPATH environment variable can contain a list of additional folders to be added to the MATLAB search path at startup. These folders are placed after the *userpath* folder, but ahead of the folders supplied by MathWorks. By default, the MATLABPATH environment variable is not set. For more information on how to set the MATLABPATH environment variable, see “Set the MATLABPATH Environment Variable” on page 6-49.

Determine If Files and Folders Are on the Search Path


There are several ways to determine if files and folders are on the search path. You can either check whether an individual file or folder is on the search path, or you can view the entire search path.

Check Whether File or Folder on Search Path

To determine whether a file is on the search path, run `which filename`. If the file is on the search path, MATLAB returns the full path to the file.

To determine whether a file or folder is on the search path, use the Current Folder browser:

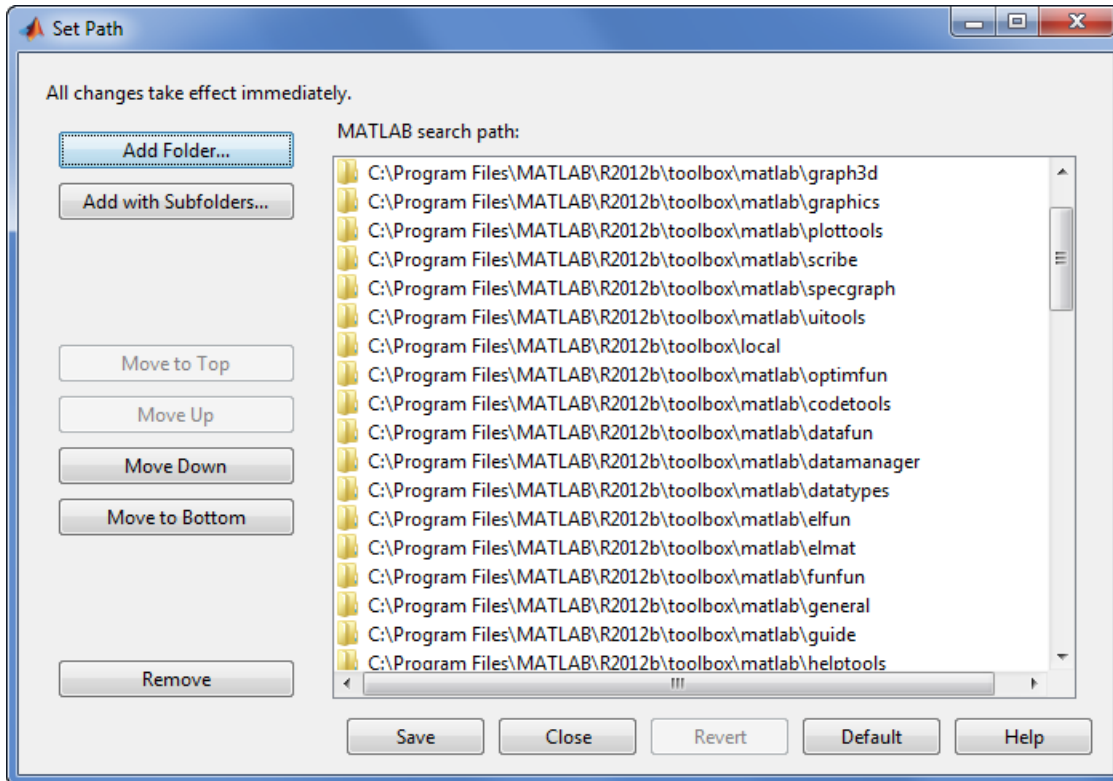
- 1 In the Current Folder browser, right-click any file or folder, and ensure that there is a check mark next to **Indicate Files Not on Path**.
- 2 Hover the pointer over any dimmed file or folder in the Current Folder browser to find out why it is dimmed.

A tooltip opens with an explanation. Frequently, the tooltip indicates that the file or folder is not on the MATLAB path. If a tooltip does not appear, it may be disabled. To enable it, go the **Home** tab and in the **Environment** section, click  **Preferences**. Then, select **MATLAB > Current Folder**. Select **Show tooltip explaining why files are inaccessible** to display the tooltip.

View Entire Search Path

Run the `path` command to view all the folders on the MATLAB search path.

Alternatively, use the Set Path dialog box to view the entire MATLAB search path. On the **Home** tab, in the **Environment** section, click **Set Path**. The Set Path dialog box opens, listing all folders on the search path. For more information on using the Set Path dialog box, see “Change Folders on the Search Path” on page 6-45.



The Search Path Is Not the System Path

The search path is *not* the same as the system path. Furthermore, there is no explicit relationship between the MATLAB search path and the system path. However, both paths help in locating files, as follows:

- MATLAB uses the search path to locate MATLAB files efficiently.
- The operating system uses a system path to locate operating system files efficiently.

How MATLAB Stores the Search Path

MATLAB saves search path information in the `pathdef.m` file. This file contains a series of full path names, one for each folder on the search path.

By default, `pathdef.m` is in `matlabroot/toolbox/local`.

When you change the search path, MATLAB uses it in the current session, but does not update `pathdef.m`. To use the modified search path in the current and future sessions, save the changes using `savepath` or the **Save** button in the Set Path dialog box. This updates `pathdef.m`.

See Also

`userpath`

More About

- “Add Folders to the MATLAB Search Path at Startup” on page 6-49
- “Change Folders on the Search Path” on page 6-45

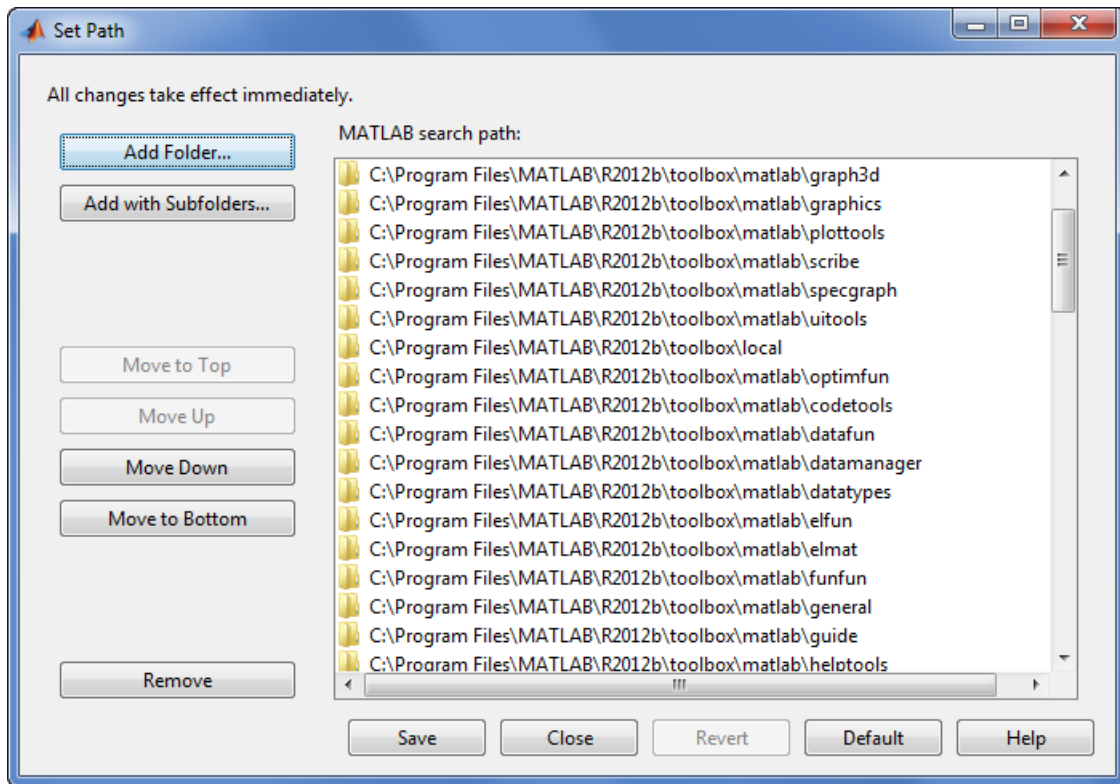
Change Folders on the Search Path

In this section...
“For Current and Future Sessions” on page 6-45
“For the Current Session Only” on page 6-47

For Current and Future Sessions

You can interactively add and remove folders, and change the order of folders on the search path, for the current MATLAB session and for future MATLAB sessions. When files with the same name appear in multiple folders on the search path, MATLAB uses the one found in the folder nearest to the top of the search path.

- 1 On the **Home** tab, in the **Environment** section, click **Set Path**. The Set Path dialog box appears.



- 2 Use the Set Path dialog box to modify the search path.
- 3 Apply or cancel the search path changes:
 - To use the newly modified search path only in the current session, click **Close**.
 - To reuse the newly modified search path in the current session and future sessions, click **Save**, and then **Close**.
 - To undo your changes, click **Revert**, and then **Close**.
 - To restore the default search path, click **Default**, and then **Close**. The default search path contains only folders provided by MathWorks.

Note: The MATLAB (*userpath*) folder automatically moves to the top of the search path the next time you start MATLAB. For more information about the *userpath* folder, see “*userpath* Folder on the Search Path” on page 6-41

For the Current Session Only

There are three ways to change the folders on the search path for the current MATLAB session only:

- Use the Set Path dialog box to make changes to the search path, and do not save the changes.
 - 1 On the **Home** tab, in the **Environment** section, click **Set Path**.
 - 2 After making the changes, click **Close**.
- Use the Current Folder browser to add or remove folders from the search path.
 - 1 From the Current Folder browser, select, and then right-click the folder or folders to add or remove.
 - 2 From the context menu, select **Add to Path** or **Remove from Path**, and then select an option:
 - **Selected Folders**
 - **Selected Folders and Subfolders**
- In the Editor, you also can add or remove the folder that contains an Editor document from the search path. Right-click the document tab, and then select an option to Add or Remove the folder from the Search Path.

See Also

addpath | rmpath | savepath

Use Search Path with Different MATLAB Installations

The default search path changes for each MATLAB version because the default folders that come with the products change. Different MATLAB versions cannot use the same `pathdef.m` file.

To use your files with a new MATLAB version or with multiple versions, do one of the following:

- For each version, add the folders containing your files to the search path. Save the search path (that is, save the `pathdef.m` file) where that version of MATLAB can access it.
- Include `addpath` statements in the `startup.m` file. Use the same `startup.m` file with the multiple versions of MATLAB.

Including `addpath` statements in the `startup.m` file also allows you to use your files with MATLAB on different platforms.

See Also

`addpath`

More About

- “Startup Options in MATLAB Startup File” on page 1-22

Add Folders to the MATLAB Search Path at Startup

There are two ways to add folders to the MATLAB search path at startup. You can either use a `startup.m` file or you can set the `MATLABPATH` environment variable.

Use a `startup.m` File

The `startup.m` file is for specifying startup options. You can add folders to the search path by including `addpath` statements in a `startup.m` file. For example, to add the specified folder, `/home/username/mytools` to the search path at startup, include this statement in a `startup.m` file:

```
addpath /home/username/mytools
```

For more information on creating a `startup.m` file with `addpath` statements, see “Startup Options in MATLAB Startup File” on page 1-22.

Set the `MATLABPATH` Environment Variable

You can also add folders to the search path at startup by setting the `MATLABPATH` environment variable. :

Windows

To set the `MATLABPATH` environment variable in Windows, from the Windows **Control Panel**, go to **System** and select **Advanced system settings**. Click the **Environment Variables...** button. Click **New...** or **Edit...** to create or edit the `MATLABPATH` environment variable. In the dialog box that appears, set the variable name to `MATLABPATH` and the variable value to a semicolon-separated list of folders you want to add to the search path. For example, to add two folders, `c:\matlab_files\myfolder1` and `c:\matlab_files\myfolder2`, to the `MATLABPATH` environment variable, enter `c:\matlab_files\myfolder1;c:\matlab_files\myfolder2` as the variable value. Click **OK** to set the variable and exit the dialog box. Restart MATLAB for the new settings to take effect.

To set the environment variable from a command window, run the command `set MATLABPATH=folders`, where `folders` is a semicolon-separated list of folders. For example, suppose that you want to add two folders, `c:\matlab_files\myfolder1` and `c:\matlab_files\myfolder2`, to the `MATLABPATH` environment variable. Run the command

```
set MATLABPATH=c:\matlab_files\myfolder1;c:\matlab_files\myfolder2
```

Once the environment variable is set, you must start MATLAB from the same command window for the settings to take effect. The environment variable persists only as long as the command window is open.

UNIX and Mac

To set the MATLABPATH environment variable in UNIX and Mac, in a terminal, run the command `export MATLABPATH=folders`, where *folders* is a colon-separated list of folders.

For example, suppose that you want to add two folders, `/home/j/Documents/MATLAB/mine` and `/home/j/Documents/MATLAB/research`, to the MATLABPATH environment variable on a UNIX platform. Run the command

```
export MATLABPATH=/home/j/Documents/MATLAB/mine:/home/j/Documents/MATLAB/research
```

Once the environment variable is set, you must start MATLAB from the same shell for the settings to take effect. The environment variable persists only as long as the shell remains open.

Note: If you are using a C shell (`csh` or `tcsh`), the command for setting the MATLABPATH environment variable is `setenv MATLABPATH folders`.

To add the folders for all future MATLAB sessions, set the MATLABPATH environment variable as part of your shell configuration script.

More About

- “What Is the MATLAB Search Path?” on page 6-41
- “Change Folders on the Search Path” on page 6-45

Assign userpath as Startup Folder (Macintosh or UNIX)

This example shows how to assign the *userpath* folder as the startup folder on a Macintosh platform. The procedure is similar for UNIX platforms. Assume that *userpath* is set to the default value on a Macintosh platform where `smith` is the home folder.

Using a bash shell, set the `MATLAB_USE_USERWORK` environment variable so that *userpath* will be used as the startup folder.

```
export MATLAB_USE_USERWORK=1
```

From that shell, start MATLAB. Next, verify the current folder in MATLAB.

```
pwd
```

```
/Users/smith/Documents/MATLAB
```

Confirm that this is the same as the folder defined for *userpath*.

```
userpath
```

```
/Users/smith/Documents/MATLAB;
```

Confirm that the *userpath* is at the top of the search path.

```
path
```

```
/Users/smith/Documents/MATLAB  
/Users/smith/Applications/MATLAB/R2009a/toolbox/matlab/general  
/Users/smith/Applications/MATLAB/R2009a/toolbox/matlab/ops
```

```
...
```

Path Unsuccessfully Set at Startup

When there is a problem with the search path, you cannot use MATLAB successfully.

Search path problems occur when:

- You save the search path on a Windows platform, and then try to use the same `pathdef.m` file on a Linux platform.
- The `pathdef.m` file becomes corrupt, invalid, renamed, or deleted.
- MATLAB cannot locate the `pathdef.m` file.

When MATLAB starts, if there is a problem with the search path, a message such as the following appears:

```
Warning: MATLAB did not appear to successfully set the search path...
```

For problems with the search path, try these recovery steps. Proceed from one step to the next only as necessary.

- 1** Ensure MATLAB is using the `pathdef.m` file you expect:
 - a** Run
which pathdef
 - b** If you want MATLAB to use the `pathdef.m` file at another location, make corrections. For example, delete the incorrect `pathdef.m` file and ensure the correct `pathdef.m` file is in a location that MATLAB can access.
- 2** Look for and correct problems with the `pathdef.m` and `startup.m` files:
 - a** Open `pathdef.m` and `startup.m` in a text editor. Depending on the problem, you might not be able to open the `pathdef.m` file.
 - b** Look for obvious problems, such as invalid characters or path names.
 - c** Make corrections and save the files.
 - d** Restart MATLAB to ensure that the problem does not recur.
- 3** Try to correct the problem using the Set Path dialog box:
 - a** Restore the default search path and save it. See “Change Folders on the Search Path” on page 6-45. Depending on the problem, you might not be able to open the dialog box.
 - b** Restart MATLAB to ensure that the problem does not recur.
- 4** Restore the default search path using functions:

- a Run `restoredefaultpath`, which sets the search path to the default and stores it in `matlabroot/toolbox/local`.
- b If `restoredefaultpath` seems to correct the problem, run `savepath`.
- c Restart MATLAB to ensure that the problem does not recur.

Depending on the problem, a message such as the following could appear:

The path may be bad. Please save your work (if desired), and quit.

5 Correct the search path problems encountered during startup:

- a Run

```
restoredefaultpath; matlabrc
```

Wait a few minutes until it completes.

- b If there is a `pathdef.m` file in the startup folder, it caused the problem. Either remove the bad `pathdef.m` file or replace it with a good `pathdef.m` file. For example, run:

```
savepath('path_to_your_startup_folder/pathdef.m')
```

See “MATLAB Startup Folder” on page 1-16.

- c Restart MATLAB to ensure that the problem does not recur.

After correcting problems with the search path, make any changes to run your files. For example, add the `userpath` folder or other folders to the search path.

Errors When Updating Folders on Search Path

You can encounter errors or unexpected behavior when you try to delete, rename, or move folders that:

- Are on the search path
- Contain subfolders that are on the search path

The behavior varies by platform because it depends on the behavior of similar features in the operating system.

If your task fails and the error message indicates it is because the folder is on the search path, then do the following:

- 1** Remove the folder from the search path.
- 2** Delete, rename, or move the folder.
- 3** Add the folder to the search path.

Editor Preferences

- “Editor/Debugger Preferences” on page 7-2
- “Code Analyzer Preferences” on page 7-12

Editor/Debugger Preferences

In this section...

“General Preferences for the Editor/Debugger” on page 7-2

“Editor/Debugger Display Preferences” on page 7-3

“Editor/Debugger Tab Preferences” on page 7-4

“Editor/Debugger Language Preferences” on page 7-5

“Editor/Debugger Code Folding Preferences” on page 7-8

“Editor/Debugger Backup Files Preferences” on page 7-9


“Editor/Debugger Autoformatting Preferences” on page 7-10

You can customize the visual display and functionality of the Editor and Debugger using Editor/Debugger preferences.

Note: Most preference changes do not apply in live scripts.

General Preferences for the Editor/Debugger

You can specify which editor MATLAB uses, as well as how the MATLAB Editor behaves under various circumstances.


On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger**, and then adjust preference options as described in the table below.

Preference	Usage
Editor	<p>Select which editor you want the MATLAB desktop to use when you edit a file:</p> <ul style="list-style-type: none"> • MATLAB Editor • Text editor <p>If you select Text editor, specify the full path for the editor application you want to</p>

Preference	Usage
	use, such as Emacs or vi. For example, <code>c:/Applications/Emacs.exe</code> .
Most recently used file list	In the Number of entries field, type the number of files that you want to appear in the list of recently used files at the bottom of the File menu.
Opening files in editor	<p>Select On restart reopen files from previous MATLAB sessions if you want the Editor and the files it contained during your last MATLAB session to reopen when you restart MATLAB.</p> <p>Select Automatically open files when MATLAB reaches a breakpoint to open a running program file when MATLAB encounters a breakpoint in that file.</p>
Automatic file changes	<p>Select Save changes upon clicking away from a file if you want the Editor to automatically save changes to a file in the Editor when you click away from the Editor. For the changes to be automatically saved upon clicking away from the Editor, you must have already saved the file at least once.</p> <p>Select Reload unedited files that have been externally modified if you want the Editor to automatically reload the version of a file that you opened and edited outside of MATLAB when the file currently open in the Editor has no unsaved changes.</p> <p>Select Add line termination at end of file to have MATLAB add a new empty line (sometimes referred to as a <CR>) to the end of a file automatically if the last line in the file is not empty.</p>

Editor/Debugger Display Preferences


You can change the appearance of the Editor.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger** > **Display**, and then adjust preference options as described in the table below.

Preference	Usage
General display option	Select Highlight Current Line and select a color to highlight the row with the cursor (also called the caret).
	Select Show line numbers to display line numbers along the left edge of the Editor window.
	Select Enable data tips in edit mode to display data tips when you are editing a MATLAB code file. (Data tips are always enabled in debug mode.) <div data-bbox="730 829 1169 979" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <pre>sequence = n; next_value n: 1x1 double = while next if rem(2</pre> </div> For details, see “View Variable Value”.
Right-hand text limit	Select Show line to display a vertical line with the specified Width and Color at the specified column (Placement) in the Editor. For details, see “Right-Side Text Limit Indicator”.

Editor/Debugger Tab Preferences


You can specify the size of tabs and indents and details about how tabs behave in the Editor.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger** > **Tab**, and then adjust preference options as described in the table below.

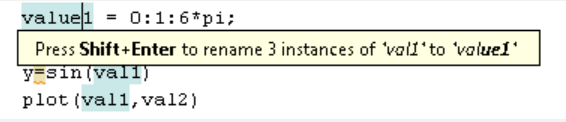
Option	Usage
Tab size	<p>Specify the amount of space inserted when you press the Tab key.</p> <p>When you change the Tab size, it changes the tab size for existing lines in that file, unless you also select Tab key inserts spaces.</p>
Indent size	<p>Specify the indent size for <i>smart indenting</i>. Smart indenting is one of the “Editor/Debugger Language Preferences” on page 7-5.</p>
Tab key inserts spaces	<p>Select to insert a series of spaces when you press the Tab key. Otherwise, a tab acts as one space whose length is equal to the Tab size.</p>
Emacs-style Tab key smart indenting	<p>Specifies an indenting style similar to the style that the Emacs editor uses.</p> <p>Lines indent according to smart indenting preferences when you position the cursor in a line or select a group of lines, and then press the Tab key.</p> <p>Smart indenting is one of the “Editor/Debugger Language Preferences” on page 7-5.</p> <p>If you select this preference, you cannot insert tabs within a line.</p>

Editor/Debugger Language Preferences

You can specify how various languages appear in the Editor. MATLAB applies language preferences based on the file extension of the file open in the Editor.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger** > **Language**, and then adjust preference options as described in the table below.

Note: Not all preferences are available for all languages.


Preference	Usage
Language	Select the language for which you want to set preferences.
Syntax highlighting	<p>Select Enable syntax highlighting to have the Editor use different colors for different language constructs. Then, adjust the colors you want to use for each language element.</p> <p>Access color options for the MATLAB language by clicking Set syntax colors.</p> <p>For all other languages, color options appear under Enable syntax highlighting.</p> <p>For details, see “Syntax Highlighting” on page 3-22.</p>
Variable and function renaming MATLAB Language only	<p>Select Enable automatic variable and function renaming to have MATLAB prompt you to rename all instances of a function or variable in a file when you rename a function or variable.</p>  <p>For details on when MATLAB prompts you, see “Automatically Rename All Functions or Variables in a File”.</p>

Preference	Usage
<p>Comment formatting</p> <p>MATLAB Language only</p>	<p>In the Maximum column width field, enter the maximum number of characters you want to allow in a line of comments, and then select where you want counting to begin.</p> <p>Consider selecting:</p> <ul style="list-style-type: none"> • Start from beginning of line when the absolute width of the comments is important. For example, set 75 columns from the start of the line to match the width that fits on a printed page when you use the default font for the Editor. • Start from beginning of comment when comments are indented, and you want each block of comments to have a consistent indent and width. <p>Select Wrap comments automatically while typing to automatically wrap comments at the Maximum column width value when you type comments in an Editor document.</p> <p>If you clear this option, you can still wrap comments manually, as described in “Add Comments to Programs”.</p>
<p>Indenting</p>	<p>Select Apply smart indenting while typing to automatically:</p> <ul style="list-style-type: none"> • Indent the body of loops within the start and end of the loop statement. • Align subsequent lines with lines you indent using tabs or spaces. • Indent functions as specified with the Function indenting format option. <p>This is called <i>smart indenting</i>. You also can manually apply smart indenting after you type the code.</p> <p>For more information, see “Indenting Code”.</p>

Preference	Usage
	<p>Select an option from Function Indenting Format (MATLAB Language only) to specify how functions indent in the Editor, as follows:</p> <ul style="list-style-type: none"> • Classic — The Editor aligns the function code with the function declaration. • Indent nested functions — The Editor indents the function code within a nested function. • Indent all functions — The Editor indents the function code for both main and nested functions. <p>For more information and examples of each indenting format, see “Indenting Code”.</p>
File extensions	Add one or more file extensions to associate with the Language . The preferences you set for that language apply to all files with the listed extensions.

Editor/Debugger Code Folding Preferences

Code folding enables you to expand and collapse blocks of MATLAB code that you want to hide when you are not currently working on them.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger > Code Folding**, and then adjust preference options as described in the table below.


For examples and detailed information about code folding, see “Code Folding — Expand and Collapse Code Constructs”.

Option	Usage
Enable Code Folding	Specifies whether you want code folding enabled for the programming constructs that have their corresponding Enable check box selected.
Enable	Specifies whether you want code folding enabled for the corresponding Programming Construct .

Option	Usage
	If you select this option for any construct, but clear the Enable Code Folding option, the construct will not have code folding enabled.
Fold Initially	Specifies whether the corresponding Programming Construct displays collapsed (folded) the first time that you open a MATLAB file.

Editor/Debugger Backup Files Preferences

You can specify if, when, and how you want MATLAB to automatically back up files that are open in the Editor.


On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger > Backup Files**, and then adjust preference options as described in the table below.

Preference	Usage
Automatically create backup files while working in the MATLAB Editor	Select to have MATLAB automatically save a copy of the files you are currently editing.
Save options	<p>Save the backup every: <i>n</i> minutes specifies how often you want MATLAB to save a copy of the file you are editing.</p> <p>Save untitled files saves a copy of new, untitled, files to <code>Untitled.asv</code>.</p> <p>When there is more than one untitled file, each additional file is saved to <code>Untitledn.asv</code> (where <i>n</i> is an integer value).</p> <p>For details, see “Backing Up Files”.</p>
Close options	Automatically delete backup files when the Editor closes directs MATLAB to delete the backup file when you close the source file in the Editor.

Preference	Usage
File name	<p>Select the naming convention you want MATLAB to use for autosave files. For example:</p> <ul style="list-style-type: none"> • If you specify Replace extension with: asv, the backup file for <code>filename.m</code> is <code>filename.asv</code> • If you specify Append file name with: ~, the backup file for <code>filename.m</code> is <code>filename.m~</code>
Location	<p>Source file directories specifies that you want backup files stored in the same folder as the files being edited.</p> <p>Single directory specifies that you want autosave files stored in a single folder. Specify the full path to that folder and be sure you have write permissions for it.</p>

Editor/Debugger Autoformatting Preferences

You can quickly format live scripts using autoformatting markup in the Live Editor. You can enable and disable autoformatting as a whole, or as individual options.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Editor/Debugger > Autoformatting**, and then adjust preference options as described in the table below.

For more information about autoformatting in live scripts, see “Autoformatting”.

Preference	Usage
Enable autoformatting in Live Editor	Select to enable autoformatting options in the Live Editor. Once enabled, options can then be enabled or disabled individually.
Text format	Select <i>*italic*</i> , **bold** to enable bold and italic formatting using asterisks (*).
	Select <u>_italic_</u> , <u>__bold__</u> to enable bold and italic formatting using underscores (_).

Preference	Usage
	Select <code>`monospace`</code> , <code> monospace </code> to enable monospace autoformatting.
Section break	Select <code>%%</code> , <code>***</code> , or <code>---</code> for section break to enable inserting section breaks by entering <code>%%</code> , <code>***</code> , or <code>---</code> and then Enter .
	Select <code>%% text</code> for section break and heading to enable inserting section breaks with headings by entering <code>%%text</code> and the Enter .
Insert Options	Select <code>\$LaTeX\$</code> for LaTeX equation to enable converting LaTeX expressions into equations using the format <code>\$LaTeX\$</code> .
	Select URL for hyperlink to enable converting internet paths automatically to hyperlinks.
	Select <code><URL></code> for hyperlink to enable converting internet paths to hyperlinks using the format <code><URL></code> .
	Select <code>[Label](URL)</code> for labeled hyperlink to enable converting internet paths to labeled hyperlinks using the format <code>[Label] (URL)</code> .
Text Style	Select #text for title to enable inserting titles using the format <code>#text</code> .
	Select ##text for heading to enable inserting headings using the format <code>##text</code> .
	Select Automatic bulleted lists (*, +, or -) to enable creating bulleted lists by entering <code>*</code> , <code>+</code> , or <code>-</code> followed by a space.
	Select Automatic numbered lists (1., 2., etc.) to enable creating numbered lists by entering <code>1.</code> , <code>2.</code> , and so on, followed by a space.

Code Analyzer Preferences

In this section...


“Code Analyzer Preferences” on page 7-12

“Searching Messages in the Code Analyzer Preferences Dialog Box” on page 7-13



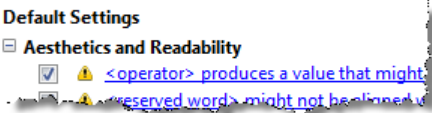
Code Analyzer Preferences

You can change how Code Analyzer messages appear in the Editor. With a few exceptions, these preferences apply to messages in the Editor, the MATLAB Function Block Editor (if your products use that tool), and the Code Analyzer Report.

Note: Code Analyzer preference changes do not apply in live scripts.

On the **Home** tab, in the **Environment** section, click  **Preferences**. Select **Code Analyzer**, and then adjust preference options as described in the table below.

Option	Usage
Enabled Integrated Warning and Error Messages	Specify whether you want to display Code Analyzer message indicators, such as the underlining of code and the message indicator bar, for documents open in the Editor. For more information, see “Automatically Check Code in the Editor — Code Analyzer”.
Underlining	Specify the type of coding issues that you want to have underlined. Regardless of the underlining menu option you choose, the Editor marks errors and warnings in the message indicator bar.
Autofix	Provides a link to a preference panel that enables you to adjust the color highlighting errors and warnings that MATLAB can autofix.



Option	Usage
	You trigger autofix by clicking the Fix button in a Code Analyzer message.
Active Settings	Select the set of message settings to use. Click the down arrow to select or browse to a previously saved settings file.
Actions button 	Click to open a menu that enables you to select: <ul style="list-style-type: none"> • Save as — Saves the current Code Analyzer message settings to a file. <p>The default location for settings is the MATLAB preferences folder (the folder returned when you run <code>prefdir</code>).</p> <ul style="list-style-type: none"> • Restore Defaults — Restores default Code Analyzer message settings.
Search field 	Searches the list of Code Analyzer messages that display below the search field. For details, see “Searching Messages in the Code Analyzer Preferences Dialog Box” on page 7-13.
Code Analyzer message settings 	Select or clear messages to enable or suppress their appearance in your Editor documents. <p>To suppress a message on a line-by-line or file-by-file basis, see “Adjust Code Analyzer Message Indicators and Messages”.</p>


Searching Messages in the Code Analyzer Preferences Dialog Box

You can search the list of Code Analyzer messages in the Preferences dialog box to display only those messages that are currently of interest to you. Use any combination of the methods that the following table presents.

Note: If you do not have the MATLAB Compiler™ installed, the Code Analyzer preferences pane does not display the **MATLAB Compiler (deployment) messages** category.

To See a List of Messages ...	Perform this action...	Example Scenario
<p>Containing specified text in the:</p> <ul style="list-style-type: none"> • Short message • Extended message • Message category • Message ID 	<p>Type the text in the search field.</p>	<p>You recall seeing a message containing some text that you want to review, but you cannot remember the exact message text.</p> <p>For example, type <code>com</code> in the search field to display those messages that contain that text in the short message, extended message, or message ID.</p>
<p>Corresponding to a given message ID</p>	<p>Type <code>msgid:</code> followed by the message ID in the search field.</p>	<p>You are reviewing the code that someone else wrote and you want to see the message that corresponds to a suppressed one using the <code>%#ok<AGROW></code> directive.</p> <p>Type <code>msgid:agrow</code> in the search field. Messages IDs containing <code>AGROW</code> display as links. Click each link for more information about the message.</p> <p>Not all Code Analyzer messages have additional information. These messages do not appear as links.</p>
<p>That you can set using Code Analyzer preferences</p>	<p>Click the down arrow to the right of the search field, and then click Show All.</p>	<p>You want to see the complete list of messages after you have searched the messages for some text or a given search menu option.</p>

To See a List of Messages ...	Perform this action...	Example Scenario
Different from the default setting (of enabled or disabled)	<p>Click the down arrow to the right of the search field, and then click Show Messages Modified from Default.</p> <p>A gray dot precedes a message with a setting different from the default. For example:</p> <p><input checked="" type="checkbox"/> <input type="checkbox"/>  DATENUM(NOW)</p>	A coworker gave you a settings file and you want to review each message that the coworker changed from its default setting.
In a given category	Click the down arrow to the right of the search field, click Show Messages in Category , and then click the category you want.	<p>You want to review messages that describe coding practices that make it difficult for others to use your code.</p> <p>Click the down arrow to the right of the search field, select Show Messages in Category, and then select Aesthetics and Readability.</p> <p>Click the messages that appear as links for more information. Not all messages appear as links.</p>
That are warnings	Click the down arrow to the right of the search field, and then select Show All Warnings . An exclamation point in a yellow triangle  indicates a warning message.	You recall previous warnings that your code generated, but you cannot remember enough details to use the search field to find it. You want to skim all the warning messages to find a particular one of interest.


To See a List of Messages ...	Perform this action...	Example Scenario
Are errors	Click the down arrow to the right of the search field, and then select Show All Errors . By default, an X in a red dot indicates an error message,  .	<p>You want to find a message elicited by a script you worked on previously. All you can recall is that it was an error and it involved <code>parfor</code>.</p> <p>Click the down arrow to the right of the search field, and then select Show All Errors. Then, type a space and <code>parfor</code> in the search field.</p> <p>The Code Analyzer preference pane displays only error messages that contain the word <code>parfor</code>.</p>
Are disabled	Click the down arrow to the right of the search field, and then select Show Disabled Messages .	You want to see the messages that are disabled by default or you have previously disabled.

Example of Searching Messages

To display Code Analyzer error messages that contain the word `variable` and are disabled:

- 1 Click the down arrow in the search field, and then select **Show All Errors**.
The search field contains `severity:error`.
- 2 At the end of the text `severity:error`, press the **Space** key, and then type `variable`.
- 3 Click the down arrow in the search field and select **Show Disabled Messages**.


The search field now contains `severity:error variable enabled:false`. Only the messages that fulfill those requirements appear in the Preferences pane.

To restore the list of all messages, click the clear search button .

Add-Ons

- “Get Add-Ons” on page 8-2
- “Manage Your Add-Ons” on page 8-3

Get Add-Ons

To extend the capabilities of MATLAB and gain additional functionality for specific tasks and application, use add-ons. You can find and install add-ons using the Add-On Explorer. To open the Add-On Explorer, go to the **Home** tab and in the **Environment** section, click the **Add-Ons**  icon.

Find add-ons by browsing through available categories located on the left side of the Add-On Explorer window, or by using the search bar.

Click an add-on to open its detailed information page. From this page you can:

- View additional information about the add-on, such as included files and available documentation.
- Install the add-on.

After you install an add-on, MATLAB manages the MATLAB path for you. Therefore, you can start using an add-on without making adjustments to your desktop environment.

Note: When installing a MathWorks product add-on, additional required products are installed automatically. For all other add-ons, you must install additional required products manually.

Install an Add-On Manually

You can install some add-ons manually. This is helpful if the add-on is not available for installation through the Add-On Explorer. For example, you could create a custom add-on yourself or receive one from someone else.

To install the add-on manually, double click the add-on installation file in the MATLAB Current Folder browser. An installer launches to guide you through the installation process.

Valid installation files include `.mltbx` files (toolboxes), `.mlappinstall` files (apps), and `.mlpkginstall` files (hardware support packages).

Related Examples

- “Manage Your Add-Ons” on page 8-3

Manage Your Add-Ons

You can view and manage installed add-ons using the Add-On Manager. To open the Add-On Manager, go to the **Home** tab, and select **Add-Ons > Manage Add-Ons**.

From the Add-On Manager you can:

- Open the installed location for add-ons installed in the add-ons installation folder.
- Open documentation for MathWorks products.
- View additional details such as a list of included files for apps and toolboxes.
- Uninstall add-ons.

The Add-On Manager displays all add-ons that are properly installed in the add-ons installation folder, as well as MATLAB products and hardware support packages. If you have an app or toolbox installed prior to R2015b, import it for use with R2015b and later by opening the Add-On Manager and clicking **Import**.

To run an app after you install it, go to the **Apps** tab and expand the apps gallery by clicking the down arrow ▼ to the far right. Then, navigate to your installed app and click the icon. Custom apps that you write or receive from someone else appear in the **My Apps** section. You can run multiple custom apps concurrently, including multiple instances of the same app.

Change Add-Ons Installation Folder

By default, MATLAB installs

- MATLAB product add-ons in `matlabroot`.
- Hardware support package add-ons and some additional package add-ons in their own custom location. For more information, see “Support Package Installation”.
- All other add-ons in the `userpath\Add-Ons` folder. The `userpath` is the path returned by the `userpath` command.

To change the add-ons installation folder:

- 1 On the **Home** tab, in the **Environment** section, click **Preferences > MATLAB > Add-Ons**.
- 2 In the **Installation Folder** field, specify a folder name to which you have write access.

Note: If you change your installation folder, add-ons installed in the previously selected folder are no longer accessible from within MATLAB.

Related Examples

- “Get Add-Ons” on page 8-2

Internationalization

- “Locale Settings for MATLAB Process” on page 9-2
- “Limitations to International Character Support” on page 9-4
- “Set Locale on Windows Platforms” on page 9-5
- “Set Locale on Mac Platforms” on page 9-7
- “Set Locale on Linux Platforms” on page 9-8
- “Unexpected Behavior on Mac Platforms” on page 9-9
- “Characters Incorrectly Displayed on Windows Systems” on page 9-10
- “datenum Might Not Return Correct Value” on page 9-11
- “Numbers Display Period for Decimal Point” on page 9-12
- “File or Folder Names Incorrectly Displayed” on page 9-13
- “Script Compatibility” on page 9-14
- “MATLAB Desktop Language Preference” on page 9-15

Locale Settings for MATLAB Process

In this section...

“Default Locale Setting” on page 9-3

“Supported Character Set” on page 9-3

“Platform-Specific Localized Formats for Current Folder Browser” on page 9-3

The locale setting defines the language of your user interface and the display formats for information like time, date, and currency. MATLAB uses the user-specified locale on all platforms.

If MATLAB does not correctly display characters in your language, you might have a locale setting problem. Locale is composed of individual settings which you can control. Each platform uses different parameters to specify the locale setting. The following terms are relevant to understanding locale settings.

- *locale* — Format: `language_territory.codeset`

For example, for the U.S. English locale setting `en_US.US-ASCII`, `en` means that the display language is English. `US` indicates that time and date displays use U.S. conventions. `US-ASCII` is the coded character set (codeset) used to display text.

- *character set* — Set of characters that make up a language used by a region. The MATLAB supported character set is the character set specified by the user locale setting.
- *codeset* — Abbreviation for *coded character set*. A set of characters with a unique numerical value assigned to each character.
- *encoding* — Scheme for assigning numerical values to a character set to create a codeset.
- *7-bit ASCII* — Either the codeset or the characters contained in that codeset. There are 128 characters, which include letters, digits, symbols, control characters, and graphics characters. The term *ASCII* in MathWorks documentation is the same as *7-bit ASCII*.
- *Unicode* — Character codeset. Excerpt from the unicode.org website: “Unicode® provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language.”
- *user locale* setting — Setting on your computer that specifies the locale that you want to use when running MATLAB. If your user-specified locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`.

- *system locale* setting — Setting on Microsoft Windows platforms. The user locale and system locale must be the same value. If these values are not the same, you might see garbled text or incorrectly displayed characters.
- *i18n* — Short for the word **internationalization**, where **18** stands for the number of letters between the letters **i** and **n**.

Default Locale Setting

MATLAB does not support every locale setting. If the user-specified locale is unsupported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

Supported Character Set

MATLAB supports the character set specified by the user locale setting.

Platform-Specific Localized Formats for Current Folder Browser

In the Current Folder Browser, MATLAB usually uses platform-neutral localized formats and rules. You can, however, use the operating system short date format to control the format for displaying file date and time data.

Related Examples

- “Set Locale on Windows Platforms” on page 9-5
- “Set Locale on Mac Platforms” on page 9-7
- “Set Locale on Linux Platforms” on page 9-8

Limitations to International Character Support

MATLAB does not support non 7-bit ASCII characters in the following features:

- MATLAB C/C++ and Fortran engine library functions and the `loadlibrary` function cannot find files located in folder names that contain non 7-bit ASCII characters.
- MATLAB C/C++ and Fortran engine library functions and the `calllib` function used to call C library functions cannot convert Unicode-compatible input arguments of type `char*` to MATLAB character arrays.
- The `matlab.wsd1.createWSDLClient` function cannot read non 7-bit ASCII characters in Web Services Description Language (WSDL) documents.

When you have a file containing text that has characters in a different encoding than that of your platform, when you save or publish your file, MATLAB displays those characters as garbled text.

Set Locale on Windows Platforms

MATLAB reads the user locale and system locale on Windows platforms. The user locale and system locale must be the same value. If these values are not the same, you might see garbled text or incorrectly displayed characters.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

When you change the system locale, restart your system; otherwise, you might see unexpected behaviors.

Locale on Windows 10 Platforms

To open the **Control Panel**, use Microsoft Windows 10 documentation.

- 1 From **Clock, Language, and Region**, select **Change date, time or number formats**.
- 2 On the **Formats** tab, select a target locale from the **Format** drop-down list and then click **Apply**. This action sets the user locale.
- 3 On the **Administrative** tab, click the **Change system locale...** button.
- 4 Select a target locale from the **Current system locale** drop-down list. This action sets the system locale.
- 5 Exit each dialog box by clicking the **Ok** buttons.
- 6 Restart the system.

Locale on Windows 8 Platforms

To open the **Control Panel**, use Microsoft Windows 8 documentation.

User Locale

- 1 From the **Control Panel**, select **Clock, Language, and Region > Region**.
- 2 Open **Formats** tab.
- 3 Select a target locale from the **Format** drop-down list.

System Locale

- 1 From the **Control Panel**, select **Clock, Language, and Region > Region**.

- 2 Open **Administrative** tab.
- 3 Look in the **Language for non-Unicode programs** section.
- 4 Click **Change system locale...** button.
- 5 Select a target locale from the **Current system locale** drop-down list.
- 6 Restart the system.

Locale on Windows 7 Platforms

User Locale

- 1 Select **Start > Control Panel > Clock, Language, and Region > Region and Language**.
- 2 Open **Formats** tab.
- 3 Select a target locale from the **Format** drop-down list.

System Locale

- 1 Select **Start > Control Panel > Clock, Language, and Region > Region and Language**.
- 2 Open **Administrative** tab.
- 3 Look in the **Language for non-Unicode programs** section.
- 4 Click **Change system locale...** button.
- 5 Select a target locale from the **Current system locale** drop-down list.
- 6 Restart the system.

Set Locale on Mac Platforms

On the Apple Mac OS X platform, MATLAB reads the user locale setting. MATLAB automatically chooses a codeset for each combination of language and territory.

If you customize the locale setting, MATLAB ignores the customized portion. MATLAB ignores the `LANG` environment variable and the Terminal application locale setting.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

To view or set the region on Mac OS X 10.10:

- 1 Select **System Preferences >Language & Region**.
- 2 Select a region and country from **Region**.

To set the Primary language:

- 1 In the **Language & Region** dialog box, either drag an existing item to the top of the **Preferred languages** list, or select the **+** button to add a language and then drag the language to the top.
- 2 MATLAB detects the latest setting; you do not need to restart your system.

Changing the preferred language might change the locale to a value that MATLAB does not support. When this happens, you might see garbled text or incorrect characters. To fix this problem, change the **Format language** value:

- 1 In the **Language & Region** dialog box, change the **Preferred languages** value to the original value.
- 2 Click the **Advanced...** button, and make note of the **Format language** value.
- 3 Click **OK**.
- 4 Change the **Preferred languages** value back to your preferred language.
- 5 Click the **Advanced...** button again. If the **Format language** value changed, reselect the original language value.

Set Locale on Linux Platforms

Use the `LANG` environment variable to specify the locale to be used by MATLAB.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

More About

- “Numbers Display Period for Decimal Point” on page 9-12

Unexpected Behavior on Mac Platforms

If you customize the locale setting, MATLAB ignores the customized portion.

MATLAB ignores the LANG environment variable and the Terminal application locale setting.

Related Examples

- “Set Locale on Mac Platforms” on page 9-7

Characters Incorrectly Displayed on Windows Systems

The user locale and system locale must be the same value on the Microsoft Windows platform. If these values are not the same, you might see garbled text or incorrect characters.

Related Examples

- “Set Locale on Windows Platforms” on page 9-5

datenum Might Not Return Correct Value

The results of the `datenum` function vary depending on the locale. To ensure the correct calculation of functions using date values associated with files and folders, replace `datenum` function calls with the use of the `dir` function `datenum` field.

For example, look at the modification date of your MATLAB `license_agreement.txt` file:

```
cd(matlabroot)
f = dir('license_agreement.txt')
```

MATLAB displays information similar to:

```
f =
    name: 'license_agreement.txt'
    date: '10-May-2015 17:48:22'
    bytes: 5124
    isdir: 0
    datenum: 7.3317e+005
```

If your code uses the `date` field of the `dir` command, similar to:

```
n = datenum(f.date);
```

replace it with the `datenum` field:

```
n = f.datenum;
```

See Also

`dir`

Numbers Display Period for Decimal Point

MATLAB reads the user locale for all categories except for the `numeric` category (which is equivalent with `LC_NUMERIC`). This category controls numeric data formatting and parsing. MATLAB always sets `LC_NUMERIC` to the `C` locale.

MATLAB uses a period for a decimal point, regardless of the format specified by the user locale. For example, the value of `pi` can be displayed as `3,1416` or `3.1416`, depending on the format used by a locale. MATLAB always displays `3.1416`.

The MATLAB language reserves the use of commas to the cases described in the Symbol Reference topic.

More About

- Symbol Reference: Comma

File or Folder Names Incorrectly Displayed

On Windows and Linux platforms, characters used in file or folder names must be in the supported character set.

On Mac platforms, for files and folders used by MATLAB, characters in the file or folder name must be in the 7-bit ASCII character set.

More About

- “Locale Settings for MATLAB Process” on page 9-2

Script Compatibility

Non-7-bit ASCII characters in plain text files, such as MATLAB scripts or functions, created with one locale setting might not be compatible with a different locale setting. This can happen when a script written on a Windows platform is run on a Linux platform, because the platforms use different default locale settings.

For example, if you create a script with the `ja_JP.UTF-8` locale setting on a Linux system, the script might not be compatible when executed on a Windows platform with the `Japanese_Japan.932` locale setting.

MATLAB Desktop Language Preference

MathWorks provides localized versions of MATLAB for selected non-English platforms. For these products only, you can choose the language in which the MATLAB desktop appears. Desktop items (such as dialog boxes, button names, and menu items) and error and warning messages appear in the language that you select. For information about this preference, see the **Desktop language** (selected non-English systems only) option in the General Preferences panel. If you change the preference, the change applies only to the MATLAB desktop.

Most desktop elements and Apps use the language selected in the **Desktop language** preference. However, system dialog messages, such as file selectors or color pickers, use the operating system display language.

More About

- “General Preferences” on page 2-51

