MATLAB®

Desktop Tools and Development Environment
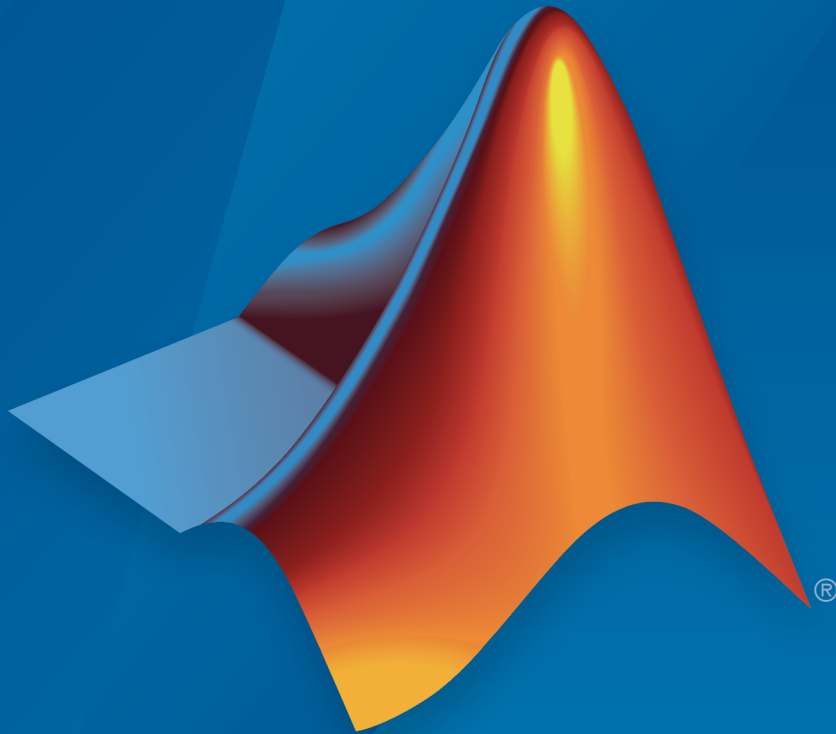
MATLAB®

MathWorks®

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

Contents

Startup and Shutdown

1

Desktop

2

<div align="right">Entering Commands</div>

3

Help and Product Information

4

Workspace Browser and Variable Editor

5

Managing Files in MATLAB

6

7

Editor Preferences

8

Internationalization

1

Startup and Shutdown

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, R2013b.

- On Windows 8 systems, go to the Windows Start screen or the desktop and click the **MATLAB *Release*** icon.
- 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***
- On Windows XP systems, select **Start** > **Programs** > **MATLAB** > ***Release*** > **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 "Associate Files with MATLAB on Windows Platforms" on page 1-4.

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 "Startup Options" on page 1-20 and "MATLAB Startup Folder" on page 1-15.

If you have trouble starting MATLAB, see Troubleshooting topics 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 "Post-Installation Tasks" in the Installation documentation.

Associate 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 a particular file type, identified by its file extension, Windows starts MATLAB and opens the file in the appropriate tool. For information about MathWorks file extensions, see "MathWorks File Extensions".

File associations for the Windows Explorer tool do not affect what happens when you open one of these file types from *within* MATLAB. MATLAB acts on the file using the MATLAB tool associated with that file type. For example, if your system associates `.mat` files with the Access™ application, when you open a MAT-file from within MATLAB, it opens the Import Wizard to load the data. To open a `.mat` file with the Access application, right-click `myfile.mat`, and from the context menu, select **Open With**, then the Access application.

To associate a different file type with MATLAB, use the Windows **Default Programs** control pane. On Windows 7 systems:

1 Click the Windows Start menu.

2 Select **Control Panel** and **View by: Category**.

3 Select **Programs**.

4 Select **Default Programs**.

5 Select **Associate a file type or protocol with a program**.

6 In the Set Associations window, find the file extension in the displayed list and either double-click it or click **Change Program**.

7 The **Open with** dialog box opens, which lists other programs that might be recommended for this file extension. If it is a file extension associated with MATLAB, such as `.mat`, the list includes all the versions of MATLAB you have installed.

8 Click **OK** in the Open With dialog box.

9 To close the Set Associations window, click **Close**.

For more information about using the **Default Programs** control pane, see your Windows documentation.

Sometimes double-clicking a MATLAB file in Windows Explorer opens a new instance of MATLAB. 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.

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 topics in the Installation Guide.

You can specify the initial working folder as well as other options—for more information, see "MATLAB Startup Folder" on page 1-15 and "Startup Options" on page 1-20.

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`.

See Also
```
matlab (Linux)
```

Start MATLAB on Mac Platforms

In this section...

From Applications Folder

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

Double-click the MATLAB icon in the `Applications` folder.

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/MATLAB_Release.app/bin`

where *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 1-15
- Troubleshooting

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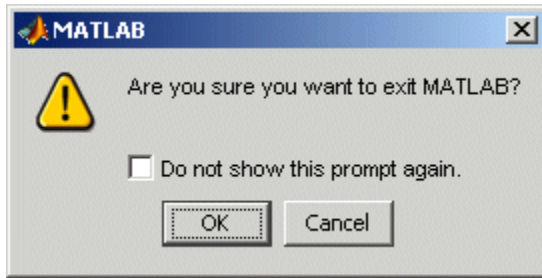
You can also display your own exit confirmation dialog box using a `finish.m` script, as described in the following section.

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:

- `finishsav.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 Dialogs 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, note that 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, Load, and Delete Workspace Variables" on page 5-12
- "Save Files"

More About
- "Commonly Used Startup Options" on page 1-18

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.

java.opts Files

If you use a `java.opts` file to set startup options, you can put the file in one of the following folders:

- MATLAB startup folder
- *matlabroot*/bin/*arch* folder. A `java.opts` file in this location applies to all users, but individual users might not have permissions to modify files there.

If there is a `java.opts` file in both locations with settings that conflict, the setting in the file in the MATLAB startup folder takes precedence. Check both locations for existing `java.opts` files and then decide which one to modify.

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, 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".

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`.

In this section...
"Default Folder on Windows Platforms" on page 1-15
"Default Folder on Mac Platforms" on page 1-16
"Default Folder on Linux Platforms" on page 1-16
"userpath as Initial Working Folder" on page 1-16
"Changing the Startup Folder" on page 1-17

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 `userpath` value, whose default value is `Documents\MATLAB`. The `userpath` folder is automatically added to the search path. If MATLAB does not find a valid `userpath` value, the startup folder is `C:\Windows\System32`.
Double-click a file type associated with MATLAB	The folder in which the file resides is the startup folder. The `userpath` folder is automatically added to the search path.
In a DOS window	The folder in which you ran the `matlab` command is the startup folder. The `userpath` folder is automatically added to the search path.

Default Folder on Mac Platforms

How You Start MATLAB	Startup Folder
Double-click the MATLAB application	The startup folder is the value returned when you enter `userpath`, which by default is *userhome*/`Documents/MATLAB`. MATLAB automatically adds the `userpath` folder to the top of its search path upon startup. To specify a different folder for `userpath`, and for other options, use the `userpath` function. You can specify that the `userpath` be the startup folder by setting the value of the environment variable `MATLAB_USE_USERWORK` to 1.
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
- "Assign userpath as Startup Folder (Macintosh or UNIX)" on page 6-54

More About
- "General Preferences"
- "Determine if Files and Folders Are on the Search Path" on page 6-45

Commonly Used Startup Options

The following table lists commonly used startup options.

Platform	Option	Description
All	`-c licensefile`	Set `LM_LICENSE_FILE` to `licensefile`. It can have the form `port@host`.
All	`-h or -help`	Display startup options (without starting MATLAB).
All	`-logfile "logfilename"`	Automatically write output from MATLAB to the specified log file.
Windows platforms	`-minimize`	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	`-nojvm`	Start MATLAB without loading the JVM™ software. This minimizes memory usage and improves initial startup speed, but restricts functionality. With `nojvm`, 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 `-nojvm` option. However, you can start MATLAB once *without* the `-nojvm` option, set the preference, and quit MATLAB. MATLAB remembers that preference when you start it again, even if you use the `-nojvm` option.
All	`-nosplash`	Start MATLAB without displaying its splash screen.
All	`-r "statement"`	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 (`"statement"`). To separate multiple statements, use semicolons or commas.
All	`-singleCompThread`	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)`

Startup Options

Startup Options from Command Line

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, you 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. For more information about the options, see "Commonly Used Startup Options" on page 1-18.

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 folder, `/home/`*username*`/mytools`, to the search path.

```
addpath /home/username/mytools
```

To change the current folder on startup to `mytools`, set the **Initial working folder** value, described in "General Preferences", to:

```
/home/username/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".

Toolbox Path Caching in MATLAB

In this section...
"About Toolbox Path Caching in MATLAB" on page 1-23
"Using the Cache File Upon Startup" on page 1-23
"Updating the Cache and Cache File" on page 1-23
"More Diagnostics with Toolbox Path Caching" on page 1-25

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**.

2

Desktop

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. Access this dialog 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 on 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 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 the table below.

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	Indicates the font currently being used in the selected desktop tool. Use one of these fonts to change it.
	
	• **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.
	
	  SansSerif / Plain / 10
	
	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.

Change Color Settings

In this section...

Changing Text, Background, and Hyperlink 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 **Sample** area in the dialog box updates to show you how it looks.

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

---

4   Under **Other colors**, select the color you want to use for hyperlinks.

5   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 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 an 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 provide access to frequently used operations. This toolbar is always visible, even when you navigate between different MATLAB Toolstrip tabs.
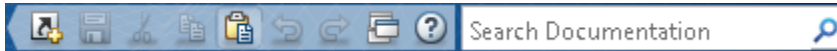
You can change the location of the quick access toolbar. On the **Home** tab, in the **Environment** section, click **Layout**, and then select an option for the **Quick Access Toolbar**.

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 order of 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 ⬅.
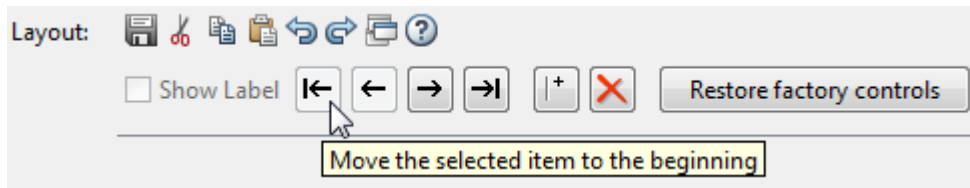
5 Click **Apply** or **OK**.

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 ![icon] or ![icon].
Open or hide a tool	On the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select or deselect 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 `edit`. To open the Profiler, use `profile` with the `viewer` option.
Maximize a tool	Do one of the following: • 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** ![icon]. 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 temporarily open the tool 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: • 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 deselect **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: • 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 • 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. **Note:** MATLAB stores the arrangements you save as XML files in the preferences folder for MATLAB. The layout last used in a session is `MATLABDesktop.xml`. The `MATLABDesktop.xml` 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 "Startup Options".

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	In the Editor and Variables editor, select the **View** tab. In the **Tiles** section, click a tile option. In a Figure panel, Help browser, or Web browser, select a tile option, ⊞, ⊓, ⊟, or ☐, on the right side of the toolbar.
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 quickly access desktop features, 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	• 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	• `Windows Default Set` (Default) • `Emacs Default Set`
UNIX	• `Emacs Default Set` (Default) • `Windows Default Set`
Macintosh	• `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` directory. 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 will 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:

```
51  Close    MATLAB Desktop  Ctrl+W  Closes the selected window
52  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   11
Autofix Message      MATLAB Editor   Shift+F9    Applies the suggested autofix   12
```

- 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 strings; 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.

Keyboard Shortcuts Preferences

Active settings:

Windows Default Set ▼ ⚙ ▾

Editor ✕

	Action name	Shortcuts
	Adjust To Bottom	
	Adjust To Top	
	Autofix Message	Alt+Enter
	Back	Alt+Left
ⓘ	Break Line	Enter
ⓘ	Break Line Without I...	Shift+Enter
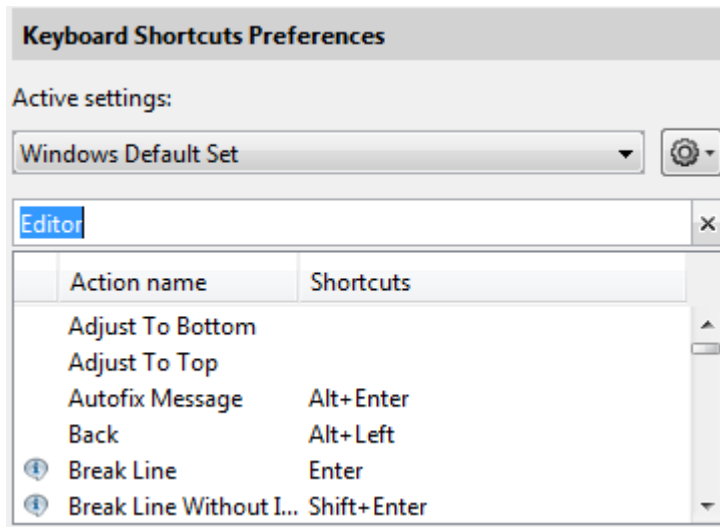
3 Narrow the list of **Action names** that the preferences pane displays by adding a string describing the action. For example, add `clear`, if you want to find the keyboard shortcut for clearing selected text in the Editor. Type a short string to increase the likelihood of the filter returning the action you seek.

Keyboard Shortcuts Preferences

Active settings:

Windows Default Set ▼ ⚙ ▾

Editor clear ✕

	Action name	Shortcuts
	Clear Breakpoints in ...	
ⓘ	Clear Selection	Escape; Ctrl+\
	Set/Clear Bookmark	Ctrl+F2
	Set/Clear Breakpoint	F12
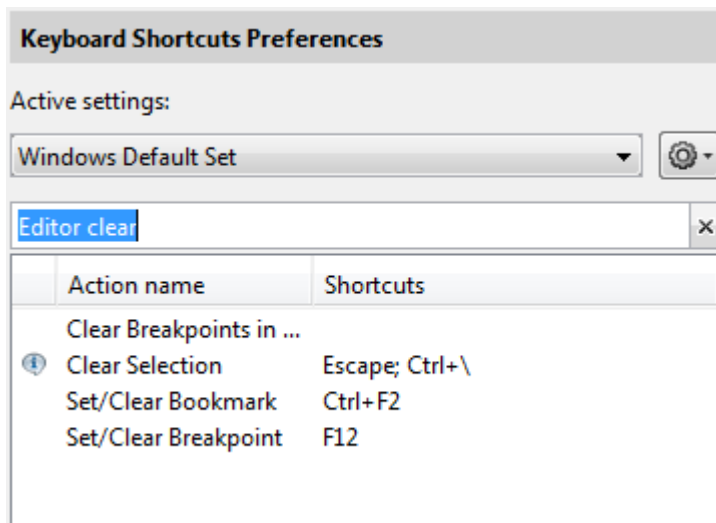
4   Select the action name of interest. In this example, select `Clear Selection`.

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.

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.

Shortcuts for **Clear Command History**

Shortcut	Tools with shortcut
🔍▾	All Tools ▾

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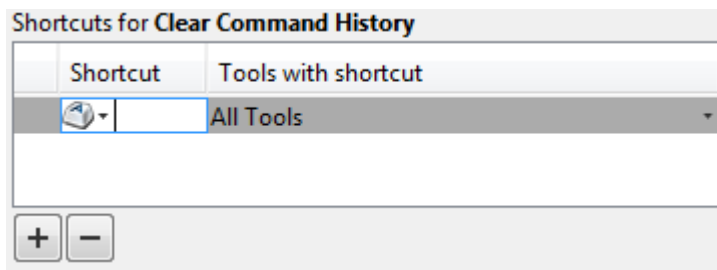
5 Type the shortcut that you want to use for the action you selected in Step 3. Alternatively, you can choose a shortcut from the drop-down menu.

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:

  a 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.

  b Select a check box to assign the shortcut to a tool. Clear a check box to remove it.

Tools with shortcut
☐ Command History
☑ MATLAB Desktop
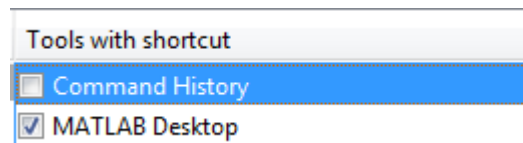
7 Evaluate and resolve any conflicts, indicated by the informational ⓘ and error ⊗ icons.

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**.
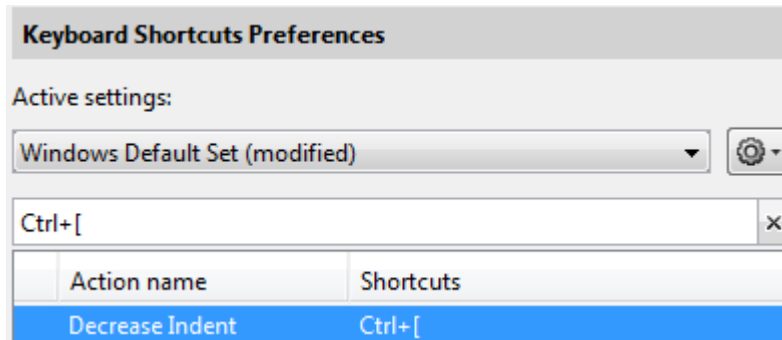
5 In the table labeled **Shortcuts for Decrease Indent**, under **Shortcut**, click `Ctrl +[`. MATLAB makes the field editable.
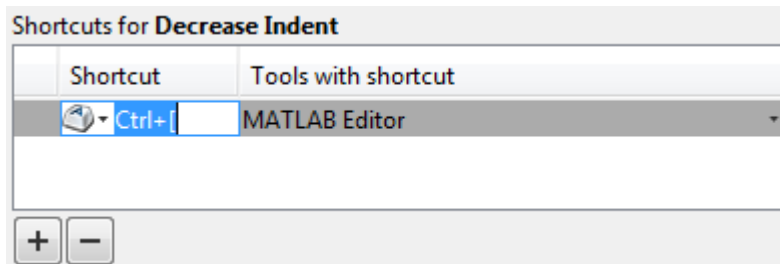
6 In the **Shortcut** field, press **Ctrl+Backspace** twice.

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**.

Shortcuts for **Autofix Message**

	Shortcut	Tools with shortcut	
	☁ ▾ Alt+Enter	MATLAB Editor	▾

[+] [−]
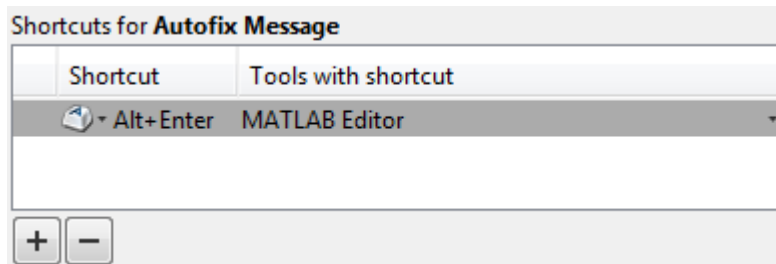
6 Click the remove button [+].

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.	**Esc** (escape) For example, if you select the **Edit** menu, the menu items display. Pressing **Esc** retracts the menu items. In the Function Browser, pressing **Esc** up to three times has the following effects: **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	• 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.

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:
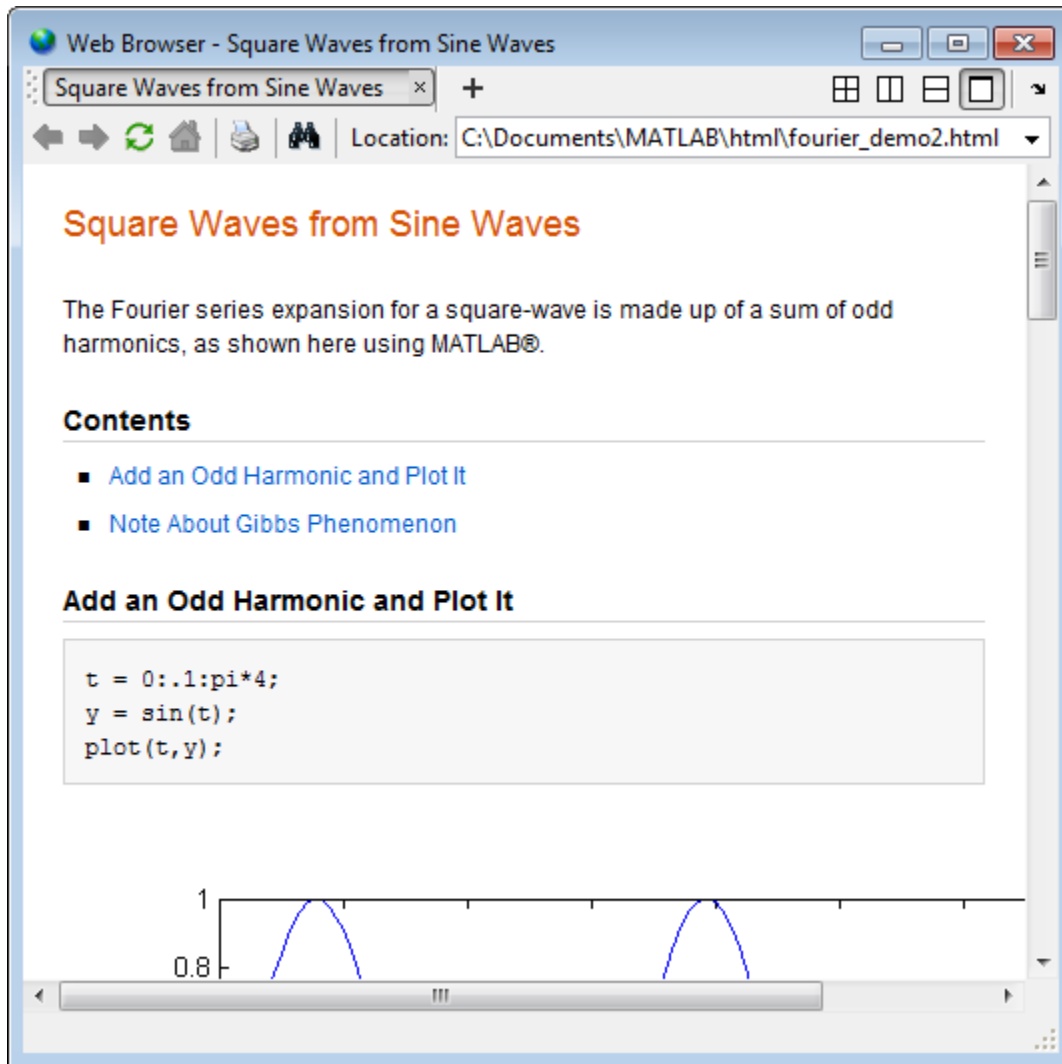
MATLAB Web and Help Browsers

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.

License Management and Software Updates

In this section...
"Manage Your Licenses" on page 2-43
"Check for Software Updates" on page 2-44

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 Web site 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 of 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 current 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 will 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 current status. When you select a license and click **Deactivate Selected License**, MATLAB deactivates all releases on

Option	Description
	this computer associated with the license, and updates the licensing information at the MathWorks Web site. 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 Web site. In this scenario, MATLAB returns a *deactivation string*. 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 Web site. 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 you have an active Internet connection.
2 On the **Home** tab, in the **Resources** 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 Use the What's New column to access the release notes for a product.

  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

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

MATLAB provides a variety of 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.

The 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.

On Apple Macintosh platforms, the folder can be in a hidden folder, for example, `myname/.matlab/R2009b`. 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**.

The name of the preferences folder, matches the name of the release. For instance, for MATLAB R2010b, the name of the preferences folder is `R2010b`.

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 Folder and Files MATLAB Uses When Multiple MATLAB 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 to the next also depends on the version.

Process MATLAB Uses to Create and Migrate the Preferences Folder and its Files

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

- If MATLAB finds a preferences folder name matching the release starting up, it uses that folder and the files within it.

  If that folder is empty, MATLAB recreates the default files for the release starting up.

- If MATLAB does not find a preferences folder name matching the release starting up, it creates one. Then, MATLAB checks to see if the release of MATLAB that immediately precedes the one you are starting up is installed.

  - If that previous release is not installed, MATLAB recreates the folder and default files for the version starting up.

    For example, if you start up R2010b and R2010a is not installed, then MATLAB recreates the default files for the R2010b preferences folder. This is true even if R2009b or earlier is installed.

  - If that previous release is installed, MATLAB migrates the files from the preferences folder corresponding to that previous release to the preferences folder for the release starting up.

    For example, if you start up R2010b and R2010a is installed, then MATLAB migrates the files from R2010a preferences folder to the R2010b preferences folder.

Control the 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	Make sure the preferences folder for that release exists, but is empty before starting up that MATLAB version.
All the preference files from the release of MATLAB immediately preceding the release you plan to start up	Ensure that the preferences folder exists for that preceding release. If so, delete the entire preferences folder for the release of MATLAB you plan to start up.

To Use:	Do This:
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 `history.m`. For more information, see "Command History".

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	Select an option to specify the current folder in MATLAB when it starts. If you select **Location based on MATLAB startup rules**, MATLAB determines the initial working folder based on how you started MATLAB and on the userpath. This is the default option. For details, see "MATLAB Startup Folder". Alternatively, select the last working folder from your previous MATLAB session, or specify the full path to a folder on your system. If you create a `startup.m` file, the commands in that file can override the preference option.
Deleting files	Select an option to specify what MATLAB does with files you delete using the `delete` function. Selecting **Delete permanently** makes the `delete` function run faster.

Preference	Usage
	On Linux systems, if you select **Move to a temporary folder**, MATLAB moves files to a subfolder with the prefix MATLAB_Files_ in the system temporary folder, as returned by the tempdir function.
Toolbox path caching	Select **Enable toolbox path caching** to have MATLAB cache toolbox folder information across sessions for quicker startup performance.
	Select **Enable toolbox path cache diagnostics** to display information about startup time when you start MATLAB.
	Click **Update Toolbox Path Cache** to add files to the toolbox folders under the matlabroot folder. (Use after you use tools not provided with MATLAB to create MATLAB files.) For details, see "Toolbox Path Caching in MATLAB".
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 Dialogs Preferences

You can specify whether or not 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".
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 `clc` 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 `edit filename` and `filename` 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".
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 `clear` function. For details, see "Save, Load, and Delete Workspace Variables".

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".
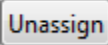
Keyboard Shortcuts Preferences

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	Select or import a set of predefined keyboard shortcuts.
	For details, see "Choose a Set of Keyboard Shortcuts" and "Use Keyboard Shortcuts Settings Files Created on Other Systems".
⚙▾	Select any one of these options:
	• **Save As**—Save active settings to a file.
	• **Copy to clipboard**— so you can import into Microsoft Excel, for example.
	For details, see, "Display Keyboard Shortcuts".
	• **Compare** active settings to another set.

Preference	Usage
	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.
⊞ ⊟	Add or delete a keyboard shortcut to a selected action. For details, see,"Examples of Creating, Modifying, and Deleting Keyboard Shortcuts" on page 2-29.
All possible conflicts	Display conflicts when two or more different actions have the same shortcut. For details, see "Evaluate and Resolve Keyboard Shortcut Conflicts" on page 2-27.
Unassign	Remove the keyboard shortcut from the selection in the **All possible conflicts** list. For details, see "Evaluate and Resolve Keyboard Shortcut Conflicts" on page 2-27.

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	Specify that desktop tools use the same text and background colors that your platform uses for other applications by selecting **Use system colors**. Customize colors by clearing **Use system colors**, and then choose **Text** and **Background** colors from the drop-down menus. These colors do not apply to the Help display pane, nor to the Web browser. For details, see "Changing Text, Background, and Hyperlink Colors in Desktop Tools" on page 2-6.
MATLAB syntax highlighting colors	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. For details, see "Changing Syntax Highlighting Colors" on page 2-6.
MATLAB Command Window colors	Set colors to help you quickly identify errors, warnings, and hyperlinks in the Command Window.

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	• **Warnings**—Specifies the color Code Analyzer uses to identify code in the Editor for which there are warning messages.

Preference	Usage
	• **Autofix highlight**—Specifies the color Code Analyzer uses to identify code in the Editor for which there is an automatic fix. For details, see "Automatically Check Code in the Editor — Code Analyzer".
Variable and function colors	• **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	**Highlight sections**—Specifies the color the Editor uses to shade code sections. **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. See also "Run Code Sections".

Comparison Colors Preferences

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	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.

Preference	Usage
Active Settings	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. For details, see "Change Color Preferences" on page 6-23.

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".

Preference	Usage
Toolbar	Select the toolbar you want to customize.
Layout	Rearrange the order of 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, `172.16.10.8` or `ourproxy`. 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, `22`. 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 `matlab.prf` file.

Preference	Usage
Test connection	Ensure that your settings work.
	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.
Automatically start MATLAB Connector when MATLAB is launched	Specifies that MATLAB Connector start automatically when you start MATLAB, thereby improving startup performance. If you use MATLAB Mobile™, you still must start the Connector as described in the MATLAB Mobile documentation.
System Web browser UNIX platforms only — excluding Macintosh	• **Command**—Specifies the system command to open the browser.
	For example, `opera`, opens the Opera Web browser.
	• **Options**—Specifies options for the system browser.
	For example, `geometry 1064x860` specifies the size of the window for Opera.

3

Entering Commands

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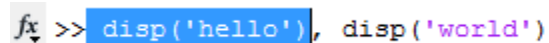
To recall previous lines in the Command Window, press the up- and down-arrow keys, ↑ and ↓. Press the arrow keys either at an empty command line or after you type the first few characters of a command. For example, to recall the command b = 2, type b, and then press the up-arrow key.

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:


```
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.

1. Click the Browse for functions button, $f_{\ast}$. 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.
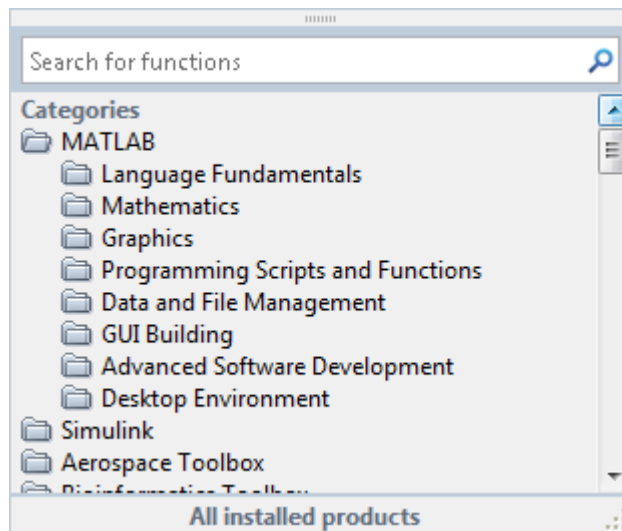
> **Tip** The Function browser closes when you move the pointer outside of it. To keep the browser open, drag it by the top edge to a different location.

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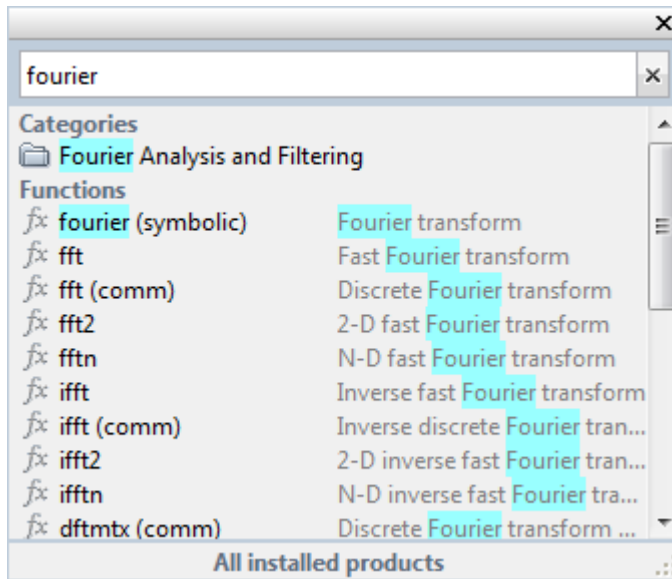
In the search results, a parenthetical term after a function name indicates either that the function is in a product folder other than MATLAB, or that there are multiple functions with the same name. For example, `fft (comm)` corresponds to the `fft` function in the Communications System Toolbox™ folder.

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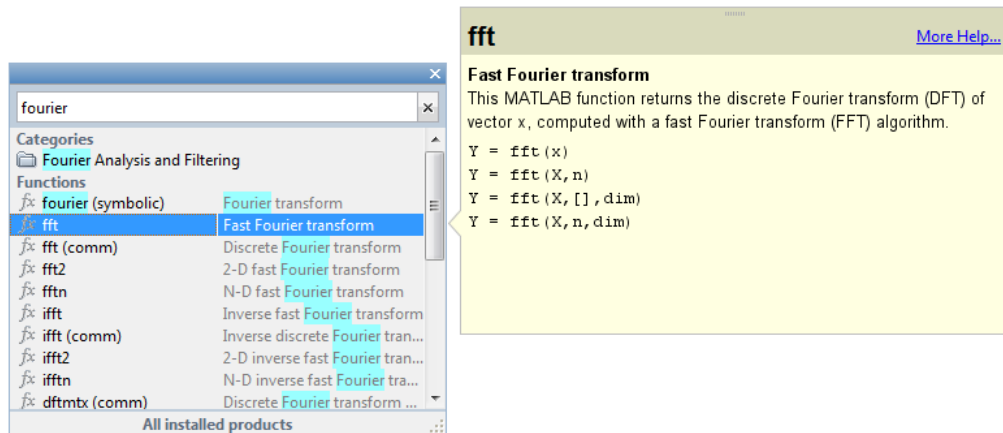
Tip The pop-up window automatically closes when you move your pointer to a new item in the results list. To keep the pop-up window open, drag it by the top edge to a different location.

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 in Command Window

Format Line Spacing in Output

By default, MATLAB displays blanks lines in command 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 format** option.

- Use the format function at the command line, for example:

  ```
  format loose
  format compact
  ```

Format Floating-Point Numbers

You can change the way numbers display. By default, MATLAB uses the short format (5-digit scaled, fixed-point values).

For example, suppose you enter x = [4/3 1.2345e-6] in the Command Window. The MATLAB output display depends on the format you selected.

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

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

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**.

Suppress Output

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

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

```
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`.

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 3-11
"Find Text in the Command History Window" on page 3-13

Find Text in the Command Window

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

- "Search Using Find Dialog" on page 3-11
- "Incremental Search Using Keyboard Shortcuts" on page 3-11

Search Using Find Dialog

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. Be aware that 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.

Incremental Search Using Keyboard Shortcuts

This topic shows how to perform an incremental search in the Command Window.

With the incremental search feature, you do not have to leave the Command Window while you perform your search.

1 Begin an incremental search using one of the following keyboard shortcuts, depending on your operating system's active settings file.

Action	Windows Default Active Settings	Macintosh or Emacs Default Active Settings
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 at the bottom of the Command Window. For a forward search, the text F incSearch appears. The F indicates a forward search.

2 Begin typing your search term.

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

3 Perform incremental search actions using the following keyboard shortcuts.

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

If you search for a string 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 **Inc 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 find text in the Command History Window that matches the case. You also can search for text at the beginning of or anywhere within a command.

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

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

2 Find the previous or next occurrence of the entry with the up and down arrow keys, respectively.

3 Press **Esc** to clear the search.

3-13

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 currently 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 Window 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   Click **Save**.

The shortcut icon and label appear on the toolbar. If you have more shortcuts on the toolbar than the desktop can display concurrently, use the drop-down list to access them all.

To organize and edit shortcuts, on the **Shortcuts** tab, in the **Manage** section, click **Organize Shortcuts** to open the Shortcuts Organizer dialog box.

3 Run a shortcut by clicking its icon on the **Shortcuts** tab.
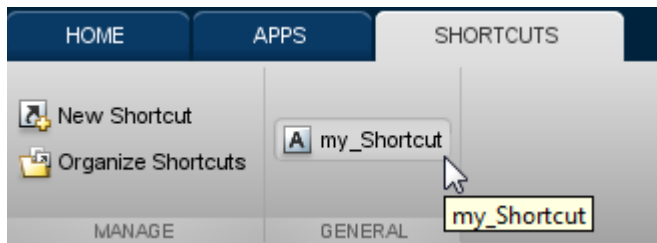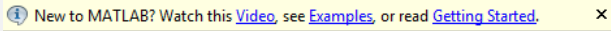
All the statements in the shortcut **Callback** field execute as if you ran those statements from the Command Window, although they do not appear in the Command History window.

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`.
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 3-8.
	Select **Set matrix display width to eighty columns** to limit the width of matrix output.
	Note: If you also select **Wrap lines**, and the width of the Command Window is less than 80 characters, each row of 80 characters of matrix output wraps to fit within the width of the Command Window.
	Select **Show getting started message bar** to display the Command Window message bar that provides links to introductory information.
	ⓘ New to MATLAB? Watch this Video, see Examples, or read Getting Started. ✕
	Select **Show function browser button** to display the Function Browser button 𝑓𝑥 to the left of the prompt in the Command Window. You can use the Function Browser to search for MATLAB functions.

Preference	Usage
	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: `Did you mean:` followed by a suggested command at the command line. You can press **Enter** to execute that command, or **Esc** to delete the suggestion.
	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. 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.
Tab key	**Tab size** specifies the number of spaces assigned to the tab key. **Note:** This setting does not apply if you have enabled tab completion. To change tab completion settings, on the **Home** tab, select **Preferences > Keyboard**.

Set Keyboard Preferences

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

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 3-21.
	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: `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') More Help... For details, see "Function Syntax Hints" on page 3-24.

Preference	Usage
Delimiter Matching	Specify when and if MATLAB alerts you to matched and mismatched delimiters. Delimiters include parentheses, brackets, braces, and, in the Editor only, paired keywords. If you select **Match while typing**, MATLAB alerts you to matched and mismatched delimiters as you type. 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. For details, see "Delimiter Matching" on page 3-21.
	Select one of these **Show match with** options to specify how MATLAB indicates matching delimiters: • `Balance` — The corresponding delimiter highlights briefly (default). • `Underline` — Both delimiters in the pair display underlines briefly. • `Highlight` — Both delimiters in the pair highlight briefly.
	Select one of these **Show mismatch with** options to specify how MATLAB indicates mismatched delimiters • `Beep` — MATLAB beeps (default). • `Strikethrough` — The delimiter you type appears crossed out briefly. • `None` — There is no alert.

Check Syntax As You Type

In this section...
"Syntax Highlighting" on page 3-20
"Delimiter Matching" on page 3-21
"Tab Completion" on page 3-21
"Function Syntax Hints" on page 3-24

Syntax Highlighting

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

- Keywords are blue.
- Strings are purple.
- Unterminated strings 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**.
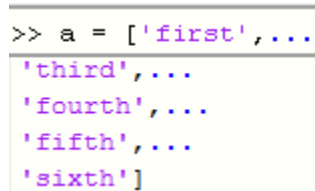
Delimiter Matching

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 the corresponding opening delimiter.
- Type more closing delimiters than opening delimiters—MATLAB beeps.
- 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.

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

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**.
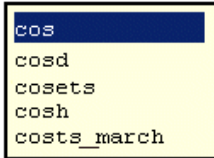
Tab Completion

MATLAB can help you avoid typographical errors by completing the names of functions, models, MATLAB objects, files, folders, variables, structures, and Handle Graphics® property names.

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.

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 Handle Graphics properties. Begin typing the first part of a property, and then press the **Tab** key. Type a comma after each property.

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.
- 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.

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**.

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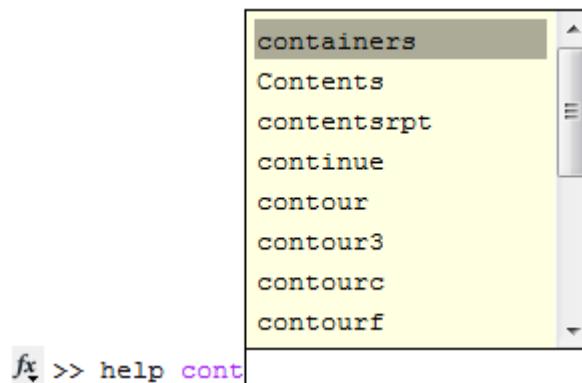
2   Select `containers`, and then press **Tab**.

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.

```
fx >> help containers.Map.
```
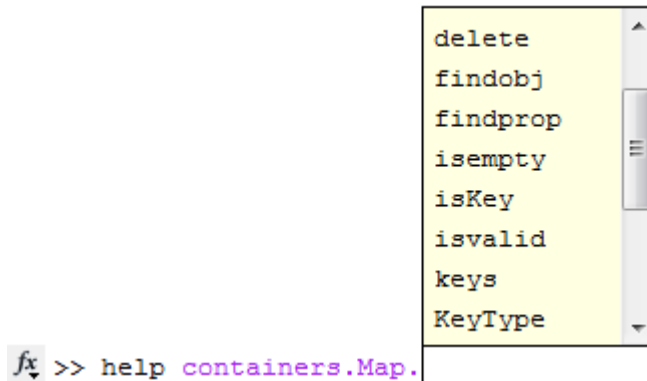
5   Scroll down the list, select keys, and then press the **Tab** key.

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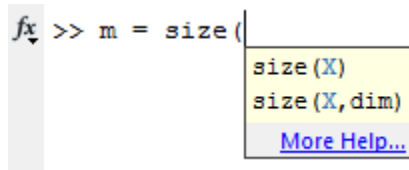
You can type a variable for any argument that appears in blue. Enter your variable names, and not the argument names shown in the window.

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**.

Command History

In this section...
"What Is the Command History?" on page 3-26
"Use Command History Commands" on page 3-27
"Change the Command History Date Format" on page 3-28
"Command History Preferences" on page 3-28

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** 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: • 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**. To extend the selection to include multiple commands, press **Shift**+↑.
Copy statements to another window.	Do one of the following: • 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: • 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. To delete all entries, click ⊙, and then select **Clear Command History** from the context menu. You cannot recall these entries.

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.

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 below.

Option	Usage
Save exit/quit commands	Save `exit` and `quit` commands in the command history.
Save consecutive duplicate commands	Save consecutive executions of the same statement in the command history. • With this option selected, if you run `magic(5)` two times in a row, the entries for `magic(5)` appear on separate lines in the command history.

Option	Usage
	• With this option cleared, the command history retains only one entry for `magic(5)` and displays a tally of consecutive executions to the left.
Don't save history file	This option is useful when multiple users share the same machine. For example, the option prevents each user from viewing statements others have run. Any entries predating the current session remain unless you first delete entries from the Command History window.
Save last *n* commands	Specify the number of commands to save.

4

Help and Product Information

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 `doc` command. — *or* — Select a function name in the Editor, Command Window, or Help browser; right-click; and then select **Help on Selection**.	`doc mean`
Function syntax hints in Command Window	After you type an open parenthesis for function inputs, pause or press **Ctrl + F1**.	`mean(`
Abbreviated help text in Command Window	Use the `help` command.	`help mean`
Function browser in Command Window	Click the function icon to the left of the command prompt.	$f_{\textbf{x}}$
Complete documentation in Help browser	Click the Help button on the quick access toolbar or on the **Home** tab. — *or* — 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

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** at the top of the main documentation page for a particular product.

For instance, MATLAB includes various examples that demonstrate mathematics functionality.
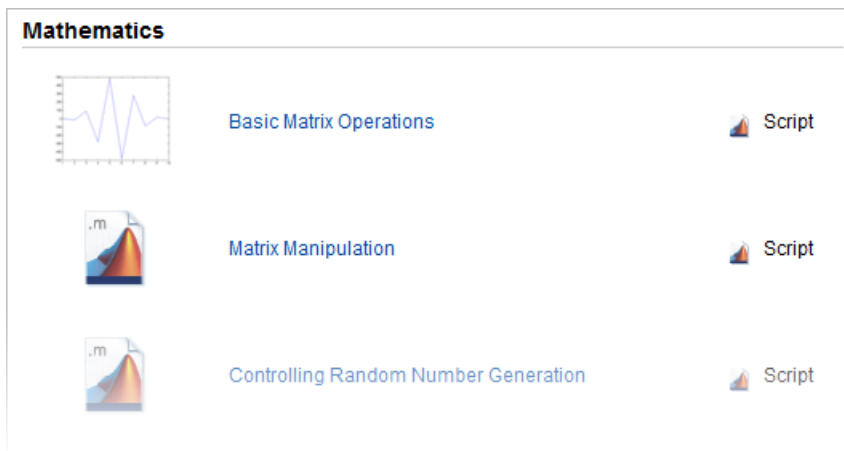
Each example combines comments, code, and output together in a formatted document. You can open the corresponding script in the Editor by clicking **Open this Example** at the top of the page in the Help browser.
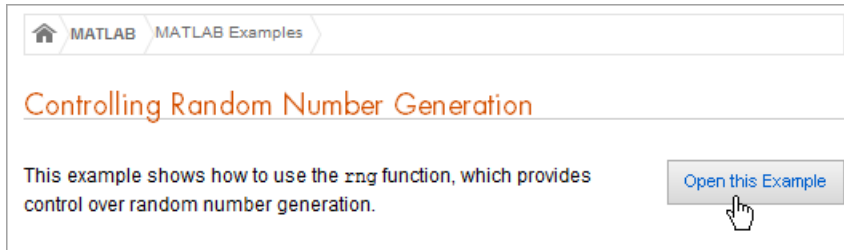
In the Editor, there are two ways to run the script:

• 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**, .

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.)
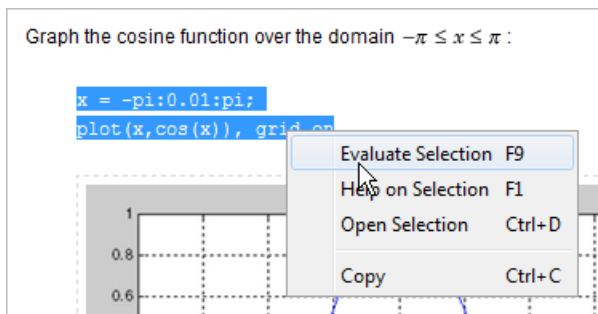
See Also

`demo` | `echodemo`

Related Examples

- "Run Code Sections"
- "Document and Share Code Using Examples"

External Web Sites

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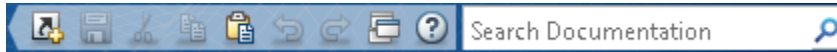
When you view pages linked from the search results, search terms appear with highlights. To clear the highlights, press the **Esc** key.

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 examples by selecting **MATLAB** and **Examples and How To**.

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 4-8
"View Page Locations" on page 4-9

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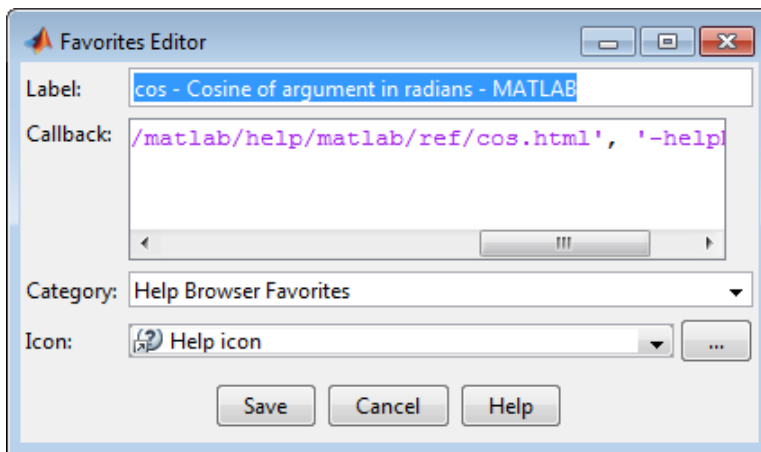
Note: You cannot migrate favorites that you save in one MATLAB release to a new release.

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**.

Note: This feature is not available on Macintosh systems.

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, 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.
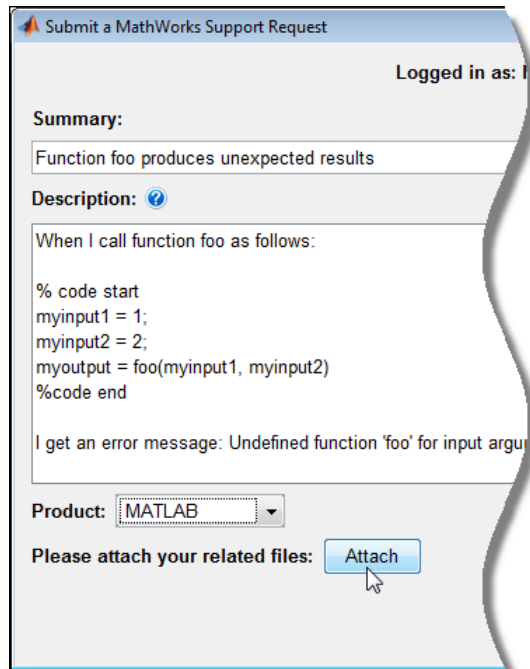
Contact Technical Support

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

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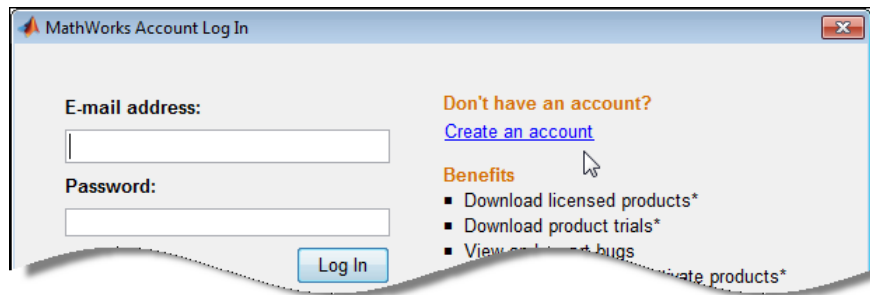
3 Provide information to help technical support reproduce your issue, such as a description of the steps you followed or a code excerpt. Optionally, you can attach up to five files to your request, where each file is no larger than 3 MB. To submit files larger than 3 MB, upload them to the MathWorks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

4   Specify the product that is related to the issue.

5   Submit the request.

External Web Sites

- 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	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.
	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.
	Changes to this preference apply only to new Help browser tabs.
Selected Products	Select the products to include for viewing and searching documentation in the Help browser or Function browser.
	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.
	When the Help browser is already open, changes to this preference apply only to new Help browser tabs.
Quick Help Display	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:
	• **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.

To adjust the font size in the Help browser or MATLAB Web browser, right-click the page and select **Zoom In** or **Zoom Out**.

More About

- "Japanese Documentation" on page 4-14

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

- "Setting Locale on Windows Platforms" on page 8-4
- "Setting Locale on Linux Platforms" on page 8-8
- "Setting Locale on Mac Platforms" on page 8-7

Korean and Chinese Documentation

A *subset* of MATLAB documentation is available on the web in Korean and simplified Chinese to licensed MATLAB users. To access that documentation, follow these steps:

1 On the MATLAB Home tab, in the **Environment** section, click **Preferences**.
2 In the left pane, click **Help**.
3 For **Documentation Location**, select **Web on mathworks.com,** and then click **OK**.
4 On the MATLAB Home tab, in the **Resources** section, click **Help**.

If the documentation does not display in the language you want, follow these additional steps and view the help directly on the web.

1 Open the MathWorks website, `http://www.mathworks.com`.
2 At the top of the web page, click the globe icon , and then select the language in which you want to view the documentation.

If the documentation still does not display in the language you want, the subset of documentation you are viewing is not 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: • license — for the license number • ver — for version numbers for MATLAB and libraries • version — 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.
arch value used to locate library files for the mex function and standalone applications	In MATLAB, select **Help** > **About MATLAB**. The About MATLAB dialog box shows the arch value, for example win64. Or use the computer function.
Passcodes and licenses	From any desktop tool, select **Help** > **Web Resources** > **MathWorks Account**.

5

Workspace Browser and Variable Editor

What Is the MATLAB Workspace?

The MATLAB workspace consists of the variables you create and store in memory during a MATLAB session. You add variables to the workspace by using functions, running MATLAB code, and loading saved workspaces. For example, if you run these statements:

```
A = magic(4);
R = randn(3,4,5);
```
the workspace includes two variables, A and R.
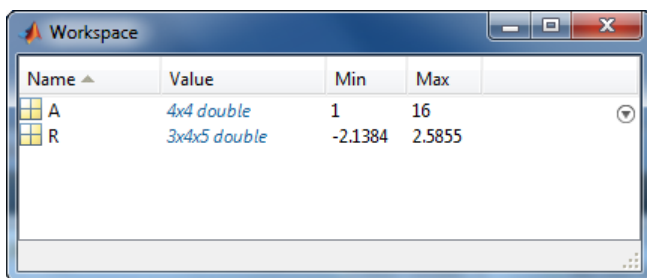
The Workspace browser displays the variables in your workspace. From the Workspace browser, you can select variables to view, modify, or plot.

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` at the Command Window prompt.

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 "Debugging Process and Features" and the `dbstack` and `evalin` functions.

You can display additional columns, such as size (dimensions) and size in bytes in the Workspace browser. On the Workspace browser title bar, click ⊙, and then click **Choose Columns**.

More About
- "Statistical Calculations in the Workspace Browser" on page 5-15
- "Base and Function Workspaces"

View, Edit, and Copy Variables

In this section...
"View and Edit Variables" on page 5-3
"Copy, Paste, and Rename Workspace Variables" on page 5-9

View and Edit Variables

- "Display Variables and Values" on page 5-3
- "View and Edit Variables Interactively" on page 5-4
- "Edit Table and Structure Array Data Interactively" on page 5-7

Display Variables and Values

The following shows how to view workspace variables and their values.

Action	Procedure
List current workspace variables	Use the `who` function. To also list information about size and class, use the `whos` function.
Display variable contents in the Command Window	Type the variable name at the Command prompt.

Open a variable in the Variables editor	Do one of the following:
	
	- Use the `openvar` function. For example, to open the variable A, type
	
	  `openvar('A')`
	
	- In the Workspace browser, double-click a variable name.
	
	  Some toolboxes allow you to double-click an object in the Workspace browser to open a viewer or other tool appropriate for that object. For details, see the toolbox documentation for that object type.

Action	Procedure
	There are special attributes for `timeseries` objects; for more information, see "Viewing Time Series Objects".
	Note: MATLAB software does not limit the maximum number of elements in a variable that you can open in the Variables editor. The limit is based on your operating system or the amount of physical memory installed on your system.

View and Edit Variables Interactively

This example shows how to open a variable, and then navigate and edit its contents graphically. Note that you cannot edit elements or subsets of multidimensional arrays in the Variables editor.

1   Create a cell array, `C`.

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

2   In the Workspace browser, open variable `C` to view its contents, by doing one of the following:

   • Use the `openvar` function.

     ```
     openvar('C')
     ```

   • In the Workspace browser, double-click the variable name `C`.

   The variable `C` opens in the Variables editor.

   Lock icons, which can appear during debugging, denote protected and private

    properties of an object, indicating you do not have `get` access to those values outside class methods.
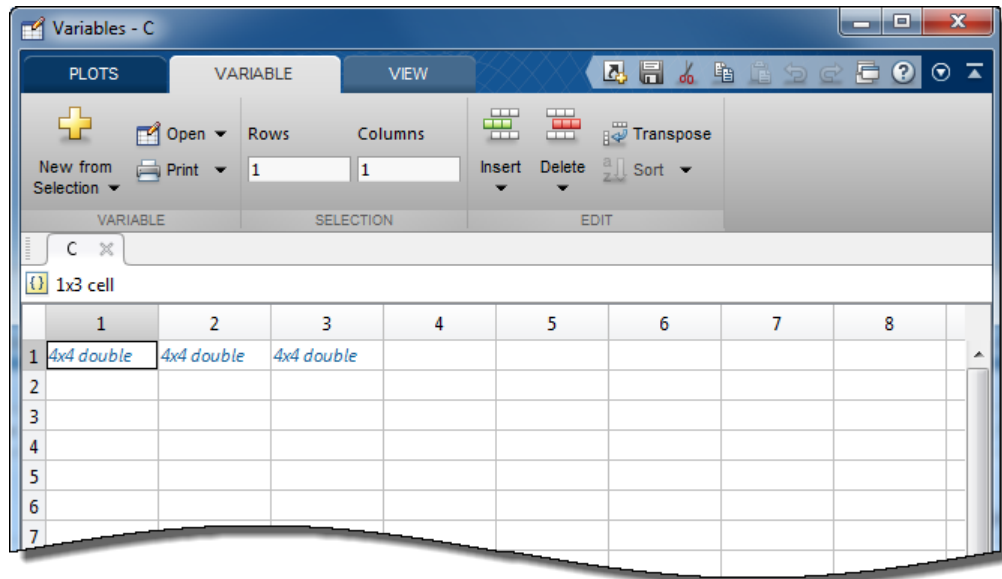
3 Double-click element C{1,1} to view the contents of that cell. The element opens in a new document within the Variables editor.
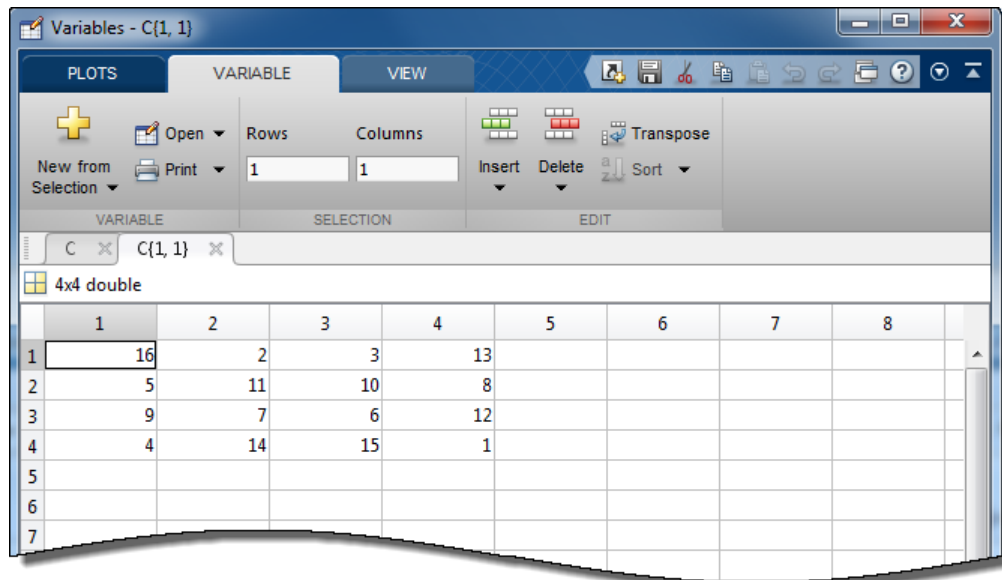
4 Edit the value of an array element by clicking the element, and then typing a new value. Press **Enter**, or click another element.

Increase the size of the array in C{1,1} by entering a value in element (5,5).

Empty elements are assigned default values. Default assignments are:

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

5 Remove a row by clicking in its row header. Right-click, and then select **Delete Row**.

6 Cut the elements of the first column by selecting column header. Right-click, and then select **Cut**. The value of each element you cut becomes 0 if numeric, or [ ] if a cell array. The cut values move to the clipboard.

7 Paste the cut elements into another column by selecting the element where you want the insertion to begin. Right-click, and then select **Paste**.

8 Change how numbers display when editing variables. On the **View** tab, in the **Format** section, select a number display format.

9   On the **View** tab, use the **Go Up** button to return to the cell array or structure.

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 Variables editor.
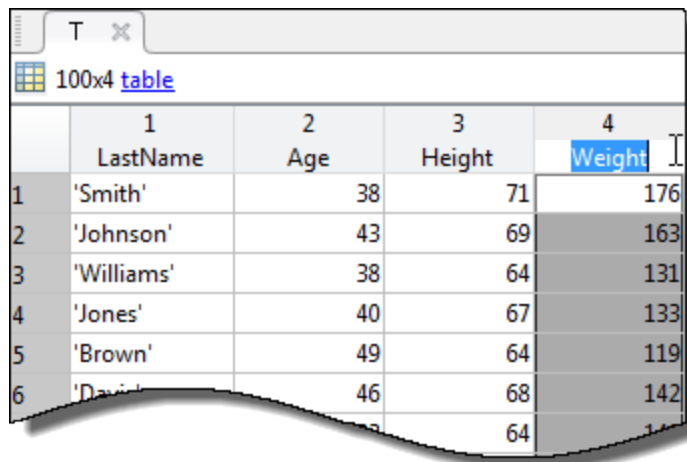
Edit Table and Structure Array Data Interactively

This example shows how to edit table data in the Variables editor. Editing structure array data is similar.

Create a sample table and open it in the Variables editor.

```
load patients.mat;
T = table(LastName,Age,Height,Weight);
openvar('T')
```
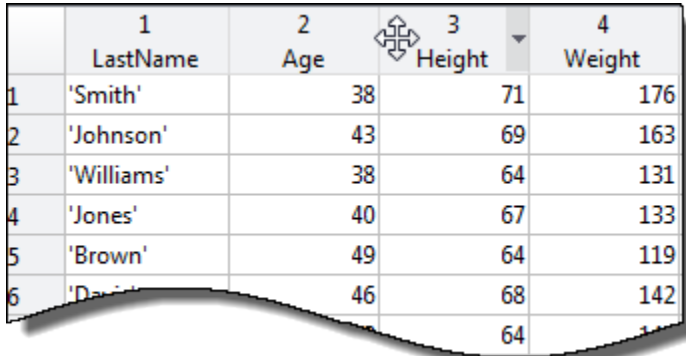
Modify a column name or row name of the table by double-clicking the name, and then typing new text. In this example, double-click the variable name, `Weight`, to rename it. For structure arrays, you can modify a field name in this manner.

T ✕			
100x4 table			
1 LastName	2 Age	3 Height	4 Weight
'Smith'	38	71	176
'Johnson'	43	69	163
'Williams'	38	64	131
'Jones'	40	67	133
'Brown'	49	64	119
'David'	46	68	142
		64	

By default, changes made in the Variables editor appear at the command line. You can suppress code display at the command line. On the **View** tab, clear the **Show MATLAB Code** check box.

Reorder the variables in the table by hovering over the left side of a variable until a four-headed arrow appears. Then, click and drag the column to a new location. You can reorder the fields in a structure array in the same manner.

	1 LastName	2 Age	3 Height	4 Weight
1	'Smith'	38	71	176
2	'Johnson'	43	69	163
3	'Williams'	38	64	131
4	'Jones'	40	67	133
5	'Brown'	49	64	119
6	'David'	46	68	142
			64	

Modify the units and description properties of the table variable, `Height`, by clicking the arrow that appears to the right of the variable name. Then, enter the new text in the **Units** and **Description** fields.

Copy, Paste, and Rename Workspace Variables

Action	Procedure
Copy variable names to clipboard	In the Workspace browser, 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.
Paste cells from Microsoft Excel spreadsheet	In a variable open in the Variables editor, right-click, and then select **Paste Excel Data**.
Create a new workspace variable from an existing variable.	For a variable open in the Variables editor, select an element, data range, row, or column in an array, and then select **New Variable from Selection**.
Rename a workspace variable	In the Workspace browser, do either of the following: • Right-click the variable name, and then select **Rename**.

Action	Procedure
	• Type the new variable name over the existing name, and then press **Enter**.

Tip If you cut and paste values from the Variables editor into text files or other applications, you can change the character that delimits decimals in the data that is exported. 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 5-19.

Keyboard Shortcuts for Navigating Variable Elements

Use the following keyboard shortcuts to move among variable elements in the Variables editor. You cannot modify these keyboard shortcuts.

Action	Keyboard Shortcut
Commit changes to an element and move to next element. "Variables Preferences" on page 5-19 enable you to specify what the next element is (the default is down).	**Enter**
Move right. Within a selection, also moves from the last column to the first column in the next row.	**Tab**
Move in opposite direction of **Enter** or **Tab**.	**Shift+Enter** or **Shift+Tab**
Move up m rows, where m is the number of visible rows.	**Page Up**
Move down m rows, where m 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)

Save, Load, and Delete 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 of the variables in the current workspace to a MAT-file (`.mat`). You can load MAT-files at a later time during the current MATLAB session, or during another session, if you want to reuse the workspace variables.

The following table describes how to save, load, and delete workspace variables.

Action	Desktop Workflow	Programmatic Workflow
Save all workspace variables to a MAT-file	On the **Home** tab, in the **Variable** section, click **Save Workspace**.	Use the `save` function. For example, save all current workspace variables to the file `june10.mat`: `save('june10')`
Save selected variables to a MAT-file	Do one of the following: • Select the variables in the Workspace browser, right-click, and then select **Save As**. • Drag variables from the Workspace browser to the Current Folder browser.	Use the `save` function. For example, save only variables A and B to the file `june10.mat`: `save('june10','A','B')`
Save part of a variable	(None)	Use the `matfile` 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**.	Use the `load` function. For example, load all variables from the file `durer.mat`: `load('durer')`
Load selected variables from a MAT-file	Do one of the following: • On the **Home** tab, in the **Variable** section, click **Import Data**.	Use the `load` function. For example, load variables X and map from the file `durer.mat`: `load('durer','X','map')`

Action	Desktop Workflow	Programmatic Workflow
	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.	
Load part of a variable	(None)	Use the `matfile` function. For an example, see "Load Parts of Variables from MAT-Files".
Delete all variables in the workspace	On the **Home** tab, in the **Variable** section, click **Clear Workspace**.	Use the `clear` function. `clear`

Delete selected variables from the workspace	Select the variables in the Workspace browser, right-click, and then select **Delete**.	Do one of the following:
		
		• Delete specified variables using the `clear` function. For example, clear variables A and B:
		
		`clear A B`
		
		• Preserve specified variables, but delete others, using the `clearvars` function with the `-except` option.
		
		`clearvars -except A`

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.

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.

Related Examples
- "View the Contents of a MAT-File"

Statistical Calculations in the Workspace Browser

In this section...
"Improve Workspace Browser Performance during Statistical Calculations" on page 5-15
"Include or Exclude NaN Values in Statistical Calculations" on page 5-15

Improve Workspace Browser Performance during Statistical Calculations

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

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 to change the value of the maximum array size for which you want the Workspace browser to perform 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, 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 some other statistics will appear as NaN. However, **Mode**, for example, shows 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** and **Max**. **Var**, however, is still appears as NaN.

Workspace and Variable Preferences

In this section...
"Workspace Browser Preferences" on page 5-17
"Variables Preferences" on page 5-19

Workspace Browser Preferences

Workspace browser 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 you perform calculations and to specify if you want those calculations to include or ignore NaNs.

To open Workspace browser preferences, on the **Home** tab, in the **Environment** section, click **Preferences**. Select **MATLAB** > **Workspace**.

Preference	Usage
Threshold for writing variables to MATLAB script	Specify **Maximum array size** to limit the number of elements of arrays saved to a MATLAB script.
	Specify **Maximum struct/object nesting levels** to limit the nesting level of structures, arrays, or objects saved to a MATLAB script.
Multidimensional array formatting	Specify how multidimensional arrays are formatted when saved to a MATLAB script.
	To create a compact script, select **Row vector with reshape (compactness and efficiency)**.
	To retain array representation, select **As 2-D pages (readability)**. The dimensions of a 2-D slice are integers that meet the following requirements:
	• They are positive.

Preference	Usage
	• They are less than or equal to the dimensions of the n-D array. • The second integer is greater than the first.
File formatting	Set the character width at which text in the MATLAB script is wrapped by specifying **Maximum characters per line**
n **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 "Statistical Calculations in the Workspace Browser" on page 5-15.
Handling NaN values in calculations	Specify whether NaN values are included or excluded from calculations for the statistics displayed in the Workspace browser: • **Use NaNs when calculating statistics** • **Ignore NaNs when calculating statistics**

Preference	Usage
MATLAB array size limit	By default, MATLAB uses 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.
	To allow MATLAB to use both RAM and virtual memory when creating an array, clear the check box. If MATLAB attempts to allocate memory that exceeds the resources available on the computer, your system might become nonresponsive.
	This limit applies to each array, not the limit for all MATLAB arrays.

Variables Preferences

When working in the Variables editor, Variables preferences enable you to specify the 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 `format` function.
Editing	Specify where the cursor moves to after you type an element, and then press **Enter**:

Preference	Usage
	• To keep the cursor in the element where you typed, clear the **Move selection after Enter** check box.
	• 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.
International number handling	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 when you paste them into text files or other applications. This preference has no effect on numeric data copied from and pasted into MATLAB. Within MATLAB, decimal separators are always periods.

6

Managing Files in MATLAB

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 a specified string 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.
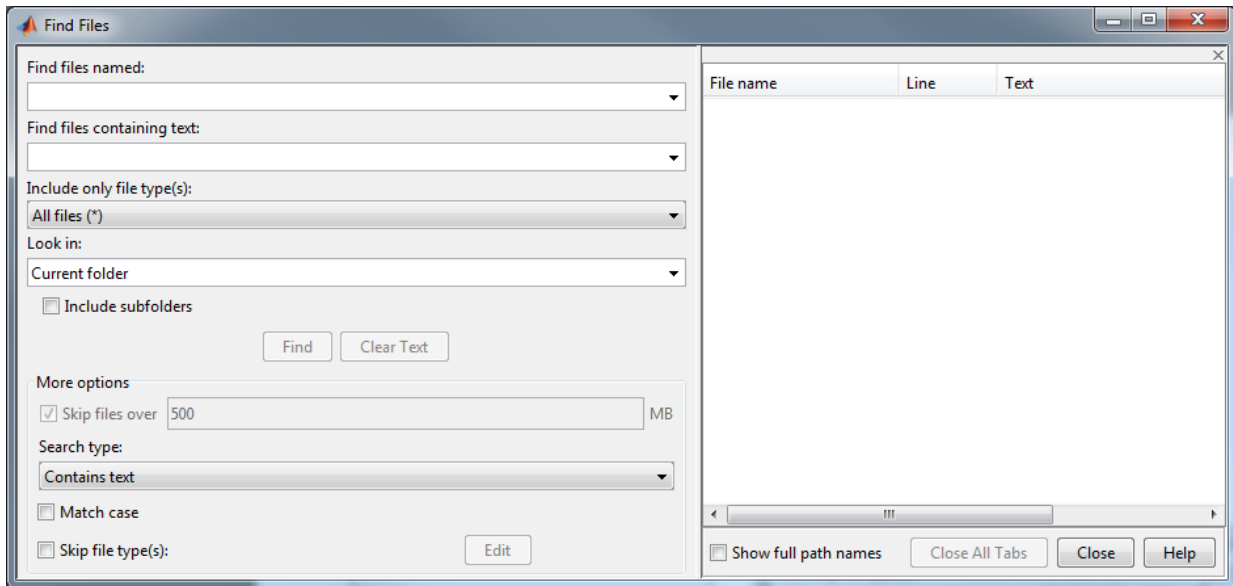
Use the **Look in** menu to specify the folders you want to search. Select **Entire MATLAB Path** to search all folders on the MATLAB search path. Alternatively, you can enter the full path for one or more folders. Separate each path with a semicolon (**;**).

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` 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-string 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"

• "What Is the MATLAB Search Path?"

Comparing Files and Folders

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** 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 **Compare with Autosave**. See "Comparing Files with Autosave Version or Version on Disk" on page 6-16.
- 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.
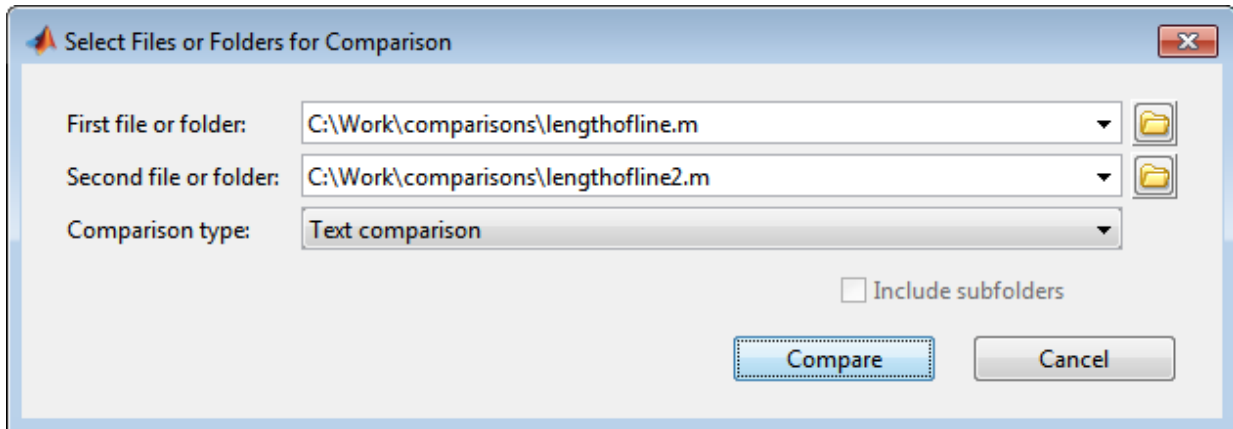
For example, from the Current Folder browser, if you select two MAT-files to compare, you get the default comparison type showing information about the variables. To change the comparison type to binary, create a new comparison using the Comparison tool. See "Select Files or Folders to Compare from the 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 Files" 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**
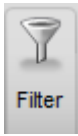
button, , then select **Add/Remove Filter**.

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**.

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 Files

- "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.

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.
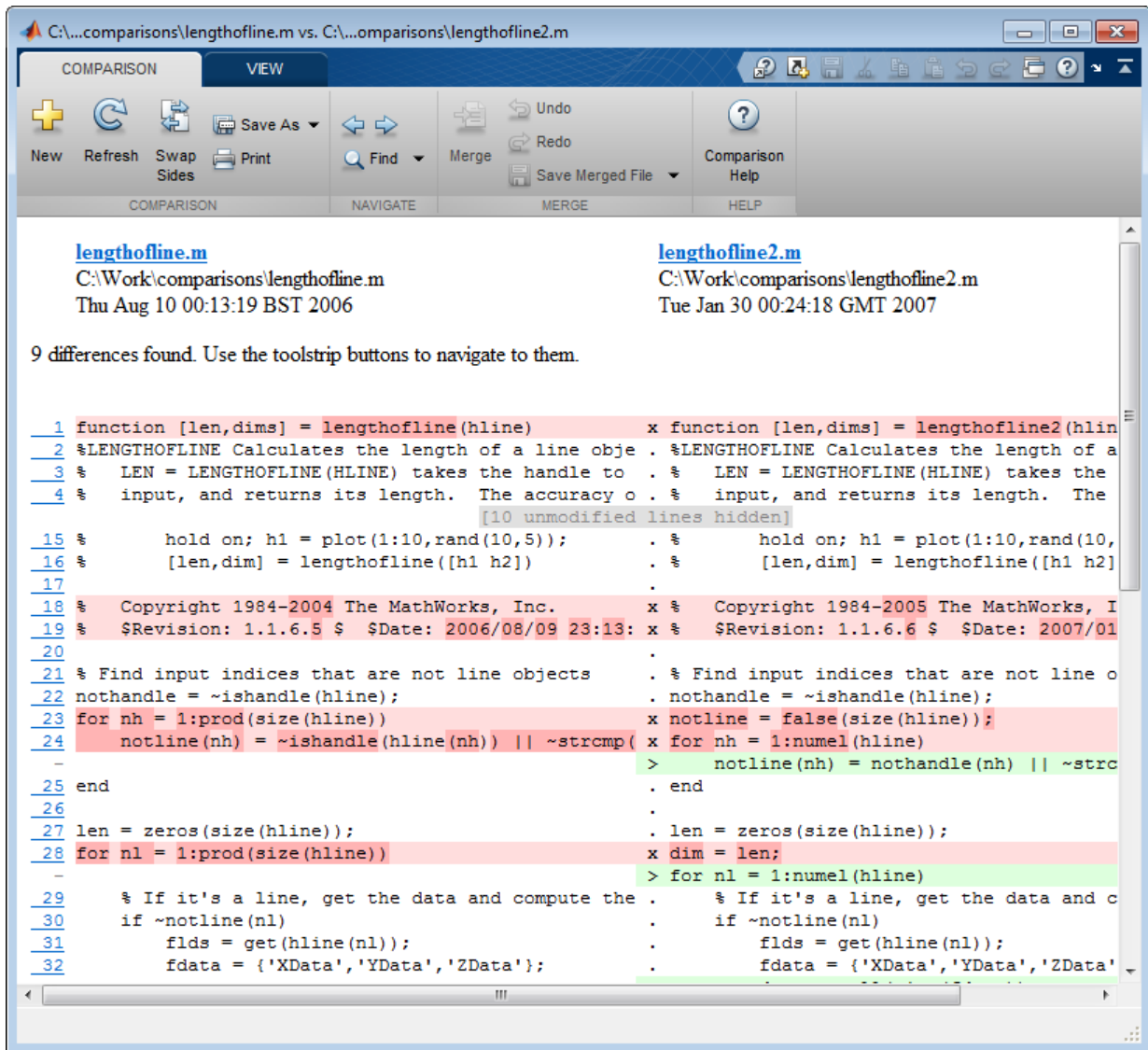
The Comparison tool attempts to match lines and detects local text that is added, deleted, or changed. It does not do a simple line-by-line comparison. In the previous image, for example, the tool determines that `lengthofline.m` has a line of code that does not exist in `lengthofline2.m` and highlights 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.

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.
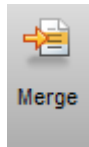
Merged differences display gray row highlighting, and a green merge arrow.

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** 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 toolstrip.

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 View differences button 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`.

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	*(not in this file)*			*Removed*	✖ 🗗
(not in this file)			b	1x5	char	*Added*	🗗 ✖
p	6x1	uint32	p	6x1	uint32	*Equivalent*	
q	6x1	uint32	q	6x1	uint32	*Equivalent*	
x	1x1	double	x	1x1	double	*Equivalent*	
y	1x2	double	y	1x3	double	Modified (compare)	🗗 🗗
z	2x2x2	double	z	2x2x2	int32	Class changed (compare)	🗗 🗗

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.
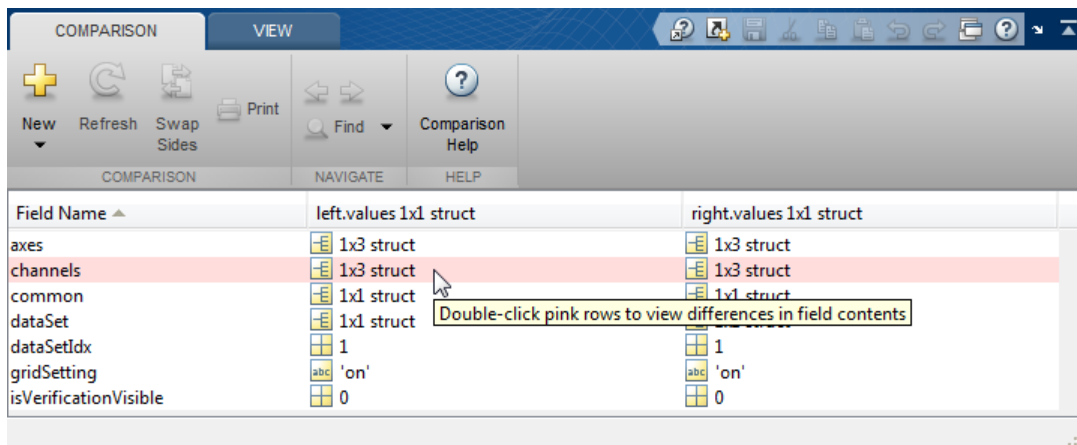
Comparing Binary Files

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
- "View Previous 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.
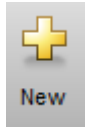
The dialog box Select Files or Folders for Comparison appears.

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 left and right 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.
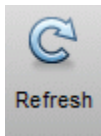
Find Text

To find a phrase in the current display, on the **Comparison** tab, in the **Navigate** section, click the **Find** button.
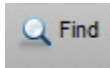
The resulting Find dialog box is the same as the one you use in the Command Window. For more information, see "Find Text in Command Window or History" on page 3-11.

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 tabs in the Comparison tool. You can drag tabs out of the tool if you want a separate window.

View Previous Comparisons

You can see the results of previous comparisons in the current session by selecting that comparison's tab entry on the document bar at the bottom of the window. If you close the Comparison tool, the current and previous comparisons are lost.

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')
```

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	In the Current Folder browser, right-click in white space, and then select **New Folder**. MATLAB creates and selects a folder named New Folder in the current folder.	Use the `mkdir` function. For example, create a subfolder named `newdir` in a parent folder named `parentFolder`: `mkdir('parentFolder','newdir');`
Move a file or folder	In the Current Folder browser, click and drag the item. You cannot move a folder while it is on the search path.	Use the `movefile` function. For example, move the file named `myfile.m` in the current folder to the folder, `d:/work`: `movefile('myfile.m','d:/work');`
Rename a file or folder	In the Current Folder browser, right-click the item and select **Rename**. File names must start with a letter, and can contain letters, digits, or underscores.	Use the `movefile` function. For example, in the current folder, rename `myfile.m` to `oldfile.m`: `movefile('myfile.m','oldfile.m');`
Open a file in MATLAB	On the **Home** tab, in the **File** section, click **Open** 📁, and then select a file to open. 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**.	Use the `open` function. The file opens in MATLAB or in an external application, depending on the file extension.
Open a file in another program	In the Current Folder browser, right-click the file and select	

Action	Tools Workflow	Function Alternative
	Open Outside MATLAB. The 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.	none
Delete a file or folder	In the Current Folder browser, select the file or folder and press **Delete**. 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** 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**.	To delete a file, use the `delete` function. For example, delete a file named `myfile.m` in the current folder: `delete('myfile.m');` By default, files are permanently removed. To move deleted files to a temporary folder instead, use the `recycle` function or set the **Deleting files** preference. To delete a folder, use the `rmdir` function. --- **Note:** You cannot recover folders deleted using `rmdir`.

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, Load, and Delete Workspace Variables"
- "Run Functions in the Editor"
- "Errors When Updating Folders on Search Path" on page 6-57

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.
`.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.

`.mlappinstall`	MATLAB App Installer File — Archive file for sharing an app created using MATLAB.
`.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.

More About

- "Associate Files with MATLAB on Windows Platforms"

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:
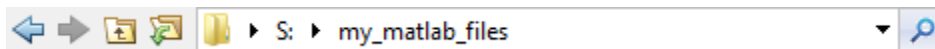
- Add the folder that contains the files to the search path. Changes you make to the search path apply to the current MATLAB session. To reuse the modified search path in future MATLAB sessions, save your changes.
- Store individual files in the *userpath* MATLAB folder, which is on the search path. 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 "Class and Path Folders".

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?"

- "MATLAB Startup Folder"
- "Toolbox Path Caching in MATLAB"

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	Specify how frequently the Current Folder browser updates to reflect changes to files made from programs and tools other than MATLAB. 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. To manually refresh the view at any time, right-click in the Current Folder browser and select **Refresh**.
Path indication	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. 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. 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.

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	Specify whether the Current Folder browser displays files and folders that the operating system hides from system file browsers and file-listing commands.

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**. |

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

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 string 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 `'\\'` string
- 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?"

Run External Commands, Scripts, and Programs

The exclamation point character (!) sometimes called bang, is a *shell escape* and indicates that the rest of the input line is a command to the operating system. Use it to invoke utilities or call other executable programs without quitting MATLAB. On UNIX platforms, for example, the following code invokes the vi editor for a file named `yearlystats.m`:

```
!vi yearlystats.m
```

After the external program completes or you quit the program, the operating system returns control to MATLAB. Add `&` to the end of the line, such as

```
!dir &
```

on Windows platforms to display the output in a separate window or to run the application in background mode. For example

```
!excel.exe &
```

opens Microsoft Excel software and returns control to the Command Window so you can continue running MATLAB language statements.

Restrictions maintained within the operating system determine the maximum length of the argument list you can provide as input to the bang (!) command. If you are running the Microsoft Windows XP operating system, for example, the length of the argument list input to the bang command cannot exceed 8189 characters.

See the reference pages for the `unix`, `dos`, and `system` functions for details about running external programs that return results and status.

Note To execute operating system commands with specific environment variables, include all commands to the operating system within the `system` call. Separate the commands using `&` (ampersand) for DOS, and `;` (semicolon) for UNIX platforms. This applies to the MATLAB `!` (bang), `dos`, `unix`, and `system` functions. Another approach is to set environment variables before starting MATLAB.

On Macintosh platforms, you cannot run AppleScript (from Apple) directly from MATLAB. However, you can run the Apple Mac OS X `osascript` function from the MATLAB `unix` or `!` (bang) function to run AppleScript from MATLAB.

Running UNIX Programs That Are Off the System Path

You can run a UNIX program from MATLAB when the folder containing that file is not on the UNIX system path that is visible to MATLAB. To determine the system path that is visible to MATLAB, type the following in the Command Window:

```
getenv('PATH')
```

You can make modifications to the system path that persist for the current MATLAB session or across subsequent MATLAB sessions, as described in the sections that follow.

Modify the System Path for the Current MATLAB Session

Do one of the following:

- Change the current folder in MATLAB to the folder that contains the program you want to run.

- Issue these commands using the Command Window:

```
path1 = getenv('PATH')
path1 = [path1 ':/usr/local/bin']
setenv('PATH', path1)
!echo $PATH
```

If you restart MATLAB, the folder is no longer on the system path visible to MATLAB.

Modify the System Path Across MATLAB Sessions Within the Current Shell Session

To add a folder to the system path from the shell:

1 Stop MATLAB.
2 Depending on the shell you are using, type one of the following at the system command prompt, where *myfolder* is the folder that contains the program you want to run:

  - Type this if you are using bash or a related shell:

    ```
    export PATH="$PATH:myfolder"
    ```
  - Type this if you are using tcsh or a related shell:

    ```
    setenv PATH "${PATH}:myfolder"
    ```
3 Start MATLAB.
4 In the MATLAB Command Window, type:

```
!echo $PATH
```

If you restart MATLAB within the current shell session, the folder remains on the system path visible to MATLAB. However, if you restart the shell session, and then restart MATLAB, the folder is no longer on the system path visible to MATLAB.

Modify the System Path Across All MATLAB Sessions

To make adjustments that persist across shell and MATLAB sessions, add the following commands to the MATLAB startup file as described in "Startup Options in MATLAB Startup File":

```
path1 = getenv('PATH')
path1 = [path1 ':/usr/local/bin']
setenv('PATH', path1)
!echo $PATH
```

Create and Extract from Zip Archives

In this section...
"Create a Zip Archive" on page 6-41
"Add Files to a Zip Archive" on page 6-42
"Extract Files from a Zip Archive" on page 6-42
"Compare Zip Archive to Unzipped Files" on page 6-43

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 Untitled*n*.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.
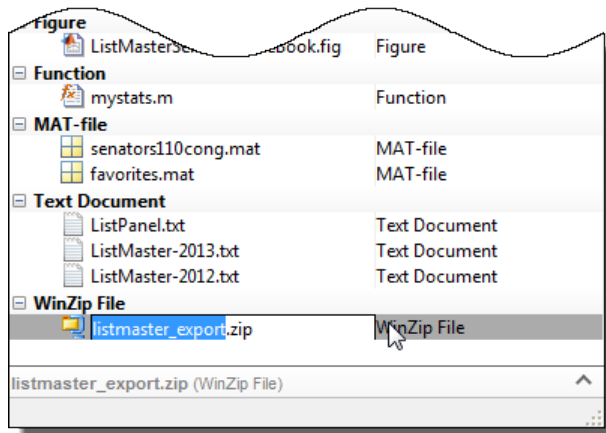
You also can create zip archives programmatically using the `zip` function. For example, to zip all `.m` and `.mat` files in the current folder to a zip file archive named `backup.zip`, call:

```
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?

In this section...
"Search Path Basics" on page 6-44
"userpath Folder on the Search Path" on page 6-44
"Determine if Files and Folders Are on the Search Path" on page 6-45
"The Search Path Is Not the System Path" on page 6-47
"How MATLAB Stores the Search Path" on page 6-47

Search Path Basics

The search path, or *path* is a subset of all the folders in the file system. MATLAB software uses the search path to efficiently locate files used with MathWorks products. MATLAB can access all files in the folders on the search path.

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

- Folders provided with MATLAB and other MathWorks products

  These folders are under *matlabroot*/toolbox, where *matlabroot* is the folder displayed when you type matlabroot in the Command Window.
- The MATLAB *userpath*

  The *userpath* folder is a location for storing files that MATLAB adds to the search path at startup.

You can explicitly add folders to the search path for the files you run.

Class, package, and private folders are *not* on the search path.

userpath Folder on the Search Path

The *userpath* folder is first on the search path, above the folders supplied by MathWorks. By default, MATLAB adds the *userpath* folder to the search path at

startup. Therefore, the *userpath* is convenient for storing files where MATLAB can access them.

The *userpath* consists of a primary path, and on Macintosh and UNIX platforms, it also contains a secondary path. The primary path is only one folder, but the secondary path can contain multiple folders.

The default primary *userpath* folder is platform-specific.

- On all platforms except Windows platforms released before Windows Vista™, the default primary *userpath* is `Documents/MATLAB`.
- On Windows platforms released before Windows Vista, it is `My Documents/MATLAB`.
- On Mac platforms, it is `$home/Documents/MATLAB`.
- On UNIX platforms, it is`$home/Documents/MATLAB` if `$home/Documents` exists.

To determine the current *userpath*, call `userpath`.

By default, the *userpath* folder is the startup folder when you start MATLAB by double-clicking either the MATLAB shortcut on Windows systems, or the MATLAB application on Macintosh systems.

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.

- "View Files and Folders on Search Path" on page 6-45
- "View Entire Search Path" on page 6-46

View Files and Folders on Search Path

For a file, run `which` *filename*. If the file is on the search path, MATLAB returns the full path to the file.

Use the Current Folder browser to determine if files or folders in the current folder are on the search path:

1 In the Current Folder browser, right-click any file or folder, and ensure 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.

View Entire Search Path

Run the `path` command to view 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.
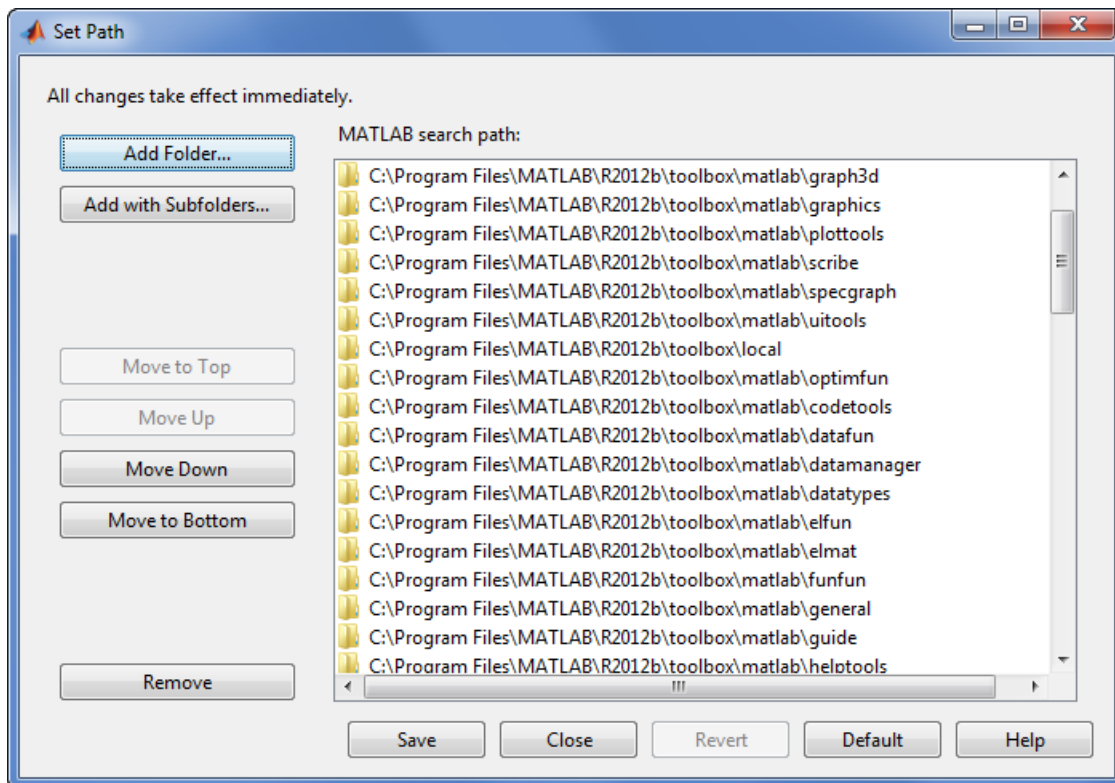
The Search Path Is Not the System Path

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 the search path information in the `pathdef.m` file. The `pathdef.m` file is a series of full path names, one for each folder on the search path, separated by a semicolon (`;`).

By default, `pathdef.m` is in *matlabroot*`/toolbox/local`.

When you change the search path, MATLAB uses it in the current session. 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 the `pathdef.m` file.

See Also
`userpath`

Change Folders on the Search Path

In this section...
"For Current and Future Sessions" on page 6-48
"For the Current Session Only" on page 6-50

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.
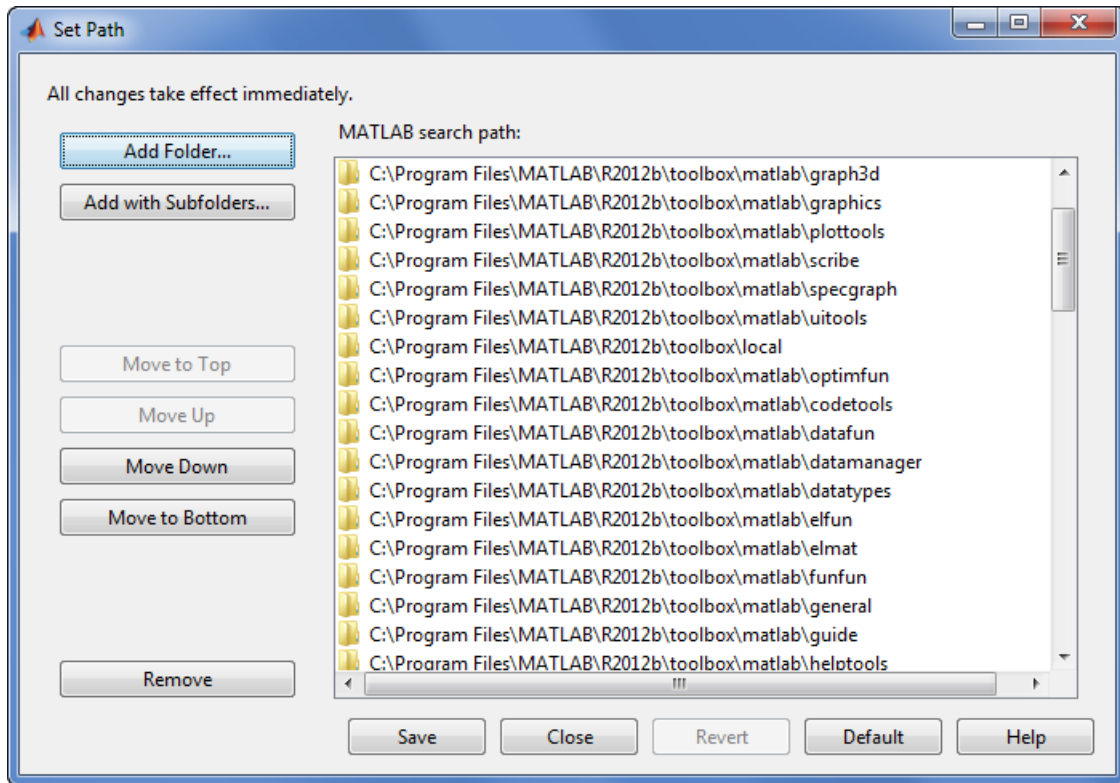
2   Use the Set Path dialog box to modify the search path.

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 6-44

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"

Add Folders to Search Path Upon Startup

In this section...

Use a `startup.m` File on Any Platform

The `startup.m` file is for specifying startup options. You can add folders to the search path by including `addpath` statements in `startup.m`. For example, to add the specified folders, `/home/`*username*`/mytools` to the search path, include this statement:

```
addpath /home/username/mytools
```

For more information, see "Startup Options".

Set MATLABPATH Environment Variable on UNIX or Macintosh

On UNIX and Macintosh platforms, you can define a secondary *userpath* by setting the `MATLABPATH` environment variable. By default, MATLAB adds *userpath* to the search path upon startup.

This example shows how to add two folders, `/home/j/Documents/MATLAB/mine` and `/home/j/Documents/MATLAB/mine/research`, to the search path upon startup on a UNIX platform. The procedure is similar for Macintosh platforms.

Assume `userpath` is set to the default value on a UNIX platform with a csh shell, where `j` is your home folder. That is, the primary *userpath* is `/home/j/Documents/MATLAB`. In a terminal, set the `MATLABPATH` environment variable. Separate multiple folders using a colon (`:`):

```
setenv MATLABPATH '/home/j/Documents/MATLAB/mine':'/home/j/Documents/MATLAB/mine/research'
```

MATLAB displays

```
MATLABPATH

home/j/Documents/MATLAB
home/j/Documents/MATLAB/mine
home/j/Documents/MATLAB/mine/research
```

```
home/j/Applications/MATLAB/R2009a/toolbox/matlab/general
home/j/Applications/MATLAB/R2009a/toolbox/matlab/ops
...
```

The two folders, `/home/j/Documents/MATLAB/mine` and `/home/j/Documents/MATLAB/mine/research`, are set as the secondary *userpath*. They are both added to the search path upon startup, as long as the shell runs. To set the secondary *userpath* for all future MATLAB sessions, set the `MATLABPATH` environment variable in your startup script.

More About

- "userpath Folder on the Search Path" on page 6-44

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 6-48. 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 1-15.

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

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

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	Select which editor you want the MATLAB desktop to use when you edit a file: · **MATLAB Editor** · **Text editor** If you select **Text editor**, specify the full path for the editor application you want to use, such as Emacs or vi. For example, `c:/Applications/Emacs.exe`.
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	Select **On restart open files from previous MATLAB sessions** if you want the Editor and

Preference	Usage
	the files it contained during your last MATLAB session to reopen when you restart MATLAB.
	Select **Automatically open files when MATLAB reaches a breakpoint** to open a running program file when MATLAB encounters a breakpoint in that file.
Automatic file changes	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.
	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.
	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.

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.
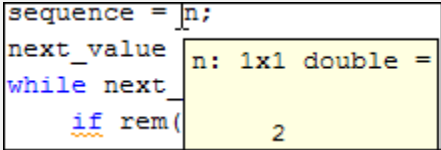
Preference	Usage
General display option	Select **Highlight Current Line** and select a color to highlight the row with the cursor (also called the caret).
	Select **Show line numbers** to display line numbers along the left edge of the Editor window.
	Select **Enable data tips in edit mode** to display data tips when you are editing a MATLAB code file. (Data tips are always enabled in debug mode.)
	For details, see "View Values as Data Tips in the Editor".
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	Specify the amount of space inserted when you press the **Tab** key.
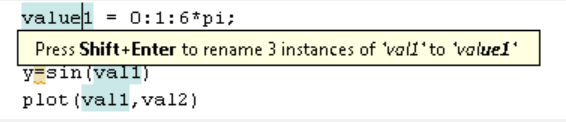
Option	Usage
	When you change the **Tab size**, it changes the tab size for existing lines in that file, unless you also select **Tab key inserts spaces**.
Indent size	Specify the indent size for *smart indenting*. Smart indenting is one of the "Editor/Debugger Language Preferences" on page 7-5.
Tab key inserts spaces	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**.
Emacs-style Tab key smart indenting	Specifies an indenting style similar to the style that the Emacs editor uses. 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. Smart indenting is one of the "Editor/Debugger Language Preferences" on page 7-5. If you select this preference, you cannot insert tabs within a line.

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	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. Access color options for the MATLAB language by clicking **Set syntax colors**. For all other languages, color options appear under **Enable syntax highlighting**. For details, see "Syntax Highlighting".

Variable and function renaming
 	
MATLAB Language only
	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.
	
	`value1 = 0:1:6*pi;`
	Press **Shift+Enter** to rename 3 instances of 'val1' to 'value1'
	`y=sin(val1)`
	`plot(val1,val2)`
	
	For details on when MATLAB prompts you, see Automatically Renaming All Functions or Variables in a File.

Preference	Usage
Comment formatting MATLAB Language only	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. Consider selecting: · **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.
	Select **Wrap comments automatically while typing** to automatically wrap comments at the **Maximum column width** value when you type comments in an Editor document. If you clear this option, you can still wrap comments manually, as described in "Add Comments to Programs".
Indenting	Select **Apply smart indenting while typing** to automatically: · 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. This is called *smart indenting*. You also can manually apply smart indenting after you type the code. For more information, see "Indenting Code".

Preference	Usage
	Select an option from **Function Indenting Format** (MATLAB Language only) to specify how functions indent in the Editor, as follows: • **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. For more information and examples of each indenting format, see "Indenting Code".
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	**Save every _n_ minutes** specifies how often you want MATLAB to save a copy of the file you are editing.
	Save untitled files saves a copy of new, untitled, files to `Untitled.asv`. When there is more than one untitled file, each additional file is saved to `Untitledn.asv` (where _n_ is an integer value). For details, see "Backing Up Files".
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	Select the naming convention you want MATLAB to use for autosave files. For example: • If you specify **Replace with extension: asv**, the backup file for `filename.m` is `filename.asv` • If you specify **Append file name with ~**, the backup file for `filename.m` is `filename.m~`
Location	**Source file directories** specifies that you want backup files stored in the same folder as the files being edited. **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.

Code Analyzer Preferences

In this section...
"Code Analyzer Preferences" on page 7-11
"Searching Messages in the Code Analyzer Preferences Dialog Box" on page 7-12
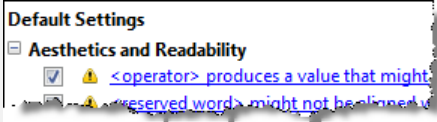
Code Analyzer Preferences

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.

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. 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.

Option	Usage
Actions button	Click to open a menu that enables you to select: • **Save as** — Saves the current Code Analyzer message settings to a file. The default location for settings is the MATLAB preferences folder (the folder returned when you run `prefdir`). • **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 7-12.
Code Analyzer message settings	Select or clear messages to enable or suppress their appearance in your Editor documents. To suppress a message on a line-by-line or file-by-file basis, see "Adjust Code Analyzer Message Indicators and Messages ".

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
Containing a given string in the:	Type the string in the search field.	You recall seeing a message containing a certain string that

To See a List of Messages ...	Perform this action...	Example Scenario
• Short message • Extended message • Message category • Message ID		you want to review, but you cannot remember the exact message text. For example, type `com` in the search field to display those messages that contain that string in the short message, extended message, or message ID.
Corresponding to a given message ID	Type `msgid:` followed by the message ID in the search field.	You are reviewing the code that someone else wrote and you want to see the message that corresponds to a suppressed one using the `%#ok<AGROW>` directive. Type `msgid:agrow` in the search field. Messages IDs containing `AGROW` display as links. Click each link for more information about the message. Not all Code Analyzer messages have additional information. These messages do not appear as links.
That you can set using Code Analyzer preferences	Click the down arrow to the right of the search field, and then click **Show All**.	You want to see the complete list of messages after you have searched the messages for a given string or search menu option.
Different from the default setting (of enabled or disabled)	Click the down arrow to the right of the search field, and then click **Show Messages Modified from Default**. A gray dot precedes a message with a setting different from the default. For example: ● ☐ ⚠ DATENUM(NOW)	A coworker gave you a settings file and you want to review each message that the coworker changed from its default setting.

To See a List of Messages ...	Perform this action...	Example Scenario
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.	You want to review messages that describe coding practices that make it difficult for others to use your code. Click the down arrow to the right of the search field, select **Show Messages in Category**, and then select **Aesthetics and Readability**. Click the messages that appear as links for more information. Not all messages appear as links.
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.
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, ⊗.	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 `parfor`. Click the down arrow to the right of the search field, and then select **Show All Errors**. Then, type a space and `parfor` in the search field. The Code Analyzer preference pane displays only error messages that contain the word `parfor`.

To See a List of Messages ...	Perform this action...	Example Scenario
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 string `variable` and are disabled:

1   Click the down arrow in the search field, and then select **Show All Errors**.

   The search field contains the string `severity:error`.

2   At the end of the string `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 the string, `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 ⌧.

8

Internationalization

How the MATLAB Process Uses Locale Settings

A *locale* is part of the user environment definition. It defines language, territory, and *codeset*, which is a coded character set. The MATLAB process uses the user-specified locale name on all platforms. MATLAB also reads the user-specified *UI language name*, and uses it to select localized resources in the specified language. By using this feature, you can select localized resources in US-English. The user-specified UI language setting also controls language and country settings of the Java Virtual Machine (JVM) software.

The term *I18n* is an abbreviation for `internationalization`, where `18` stands for the number of letters between the `i` and the `n`.

To set or view the current settings, see:

- "Setting Locale on Windows Platforms" on page 8-4
- "Setting Locale on Mac Platforms" on page 8-7
- "Setting Locale on Linux Platforms" on page 8-8

Consider the following when choosing your locale settings.

- **`Default Locale Setting`** — MATLAB does not support every locale setting. If the user-specified locale is not supported, MATLAB uses the default locale `en_US.US-ASCII`.
- **`UI Language Setting`** — Set the UI language to either the same language as the user-specified locale or to `US-English`. Otherwise, non-7-bit ASCII characters might not display properly.
- **`Supported Character Set`** — MATLAB supports the character set specified by the user locale setting. However, MATLAB might not properly handle character codes greater than 2 bytes.
- **`Script Compatibility`** — Non-7-bit ASCII characters in MATLAB scripts created with one locale setting might not be compatible with a different locale setting.

  For example, if you create a script with the `ja_JP.UTF-8` locale setting, the script might not be compatible when executed on a platform with the `ja_JP.eucJP` locale setting.
- **`Numeric Format Uses C Locale`** — MATLAB reads the user locale for all categories except for the `LC_NUMERIC` category. This category controls numeric data formatting and parsing. MATLAB always sets `LC_NUMERIC` to the `C` locale. For more information, see "Numbers Display Period for Decimal Point" on page 8-12.

- **Platform-Specific Localized Formats** — 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.

Windows Platform-Specific Behavior

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. For information on controlling these settings, see "Setting Locale on Windows Platforms" on page 8-4.

Mac Platform-Specific Behavior

On the Apple Mac OS X platform, MATLAB reads the user locale setting and the user UI language setting. For information on controlling these settings, see "Setting Locale on Mac Platforms" on page 8-7. If you customize the locale setting on Mac OS X, MATLAB ignores the customized portion. MATLAB ignores the LANG environment variable and the Terminal application locale setting.

MATLAB automatically chooses a codeset for each combination of language and territory on the Mac OS X platform.

Setting Locale on Windows Platforms

MATLAB uses the *system locale* and *user locale* on Windows platforms. The user locale and system locale must be the same value. If your user-specified locale is not supported, MATLAB uses the default locale en_US.US-ASCII.

When you change the system locale, you must reboot your system; otherwise, you might see unexpected locale-setting behaviors.

In this section...
"Locale on Windows 8 Platforms" on page 8-4
"Locale on Windows 7 Platforms" on page 8-5
"Locale on Windows Vista Platforms" on page 8-5
"Locale on Windows XP Platforms" on page 8-6

Locale on Windows 8 Platforms

Use Microsoft Windows 8 documentation to open the **Control Panel**.

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   Reboot 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 Reboot the system.

Locale on Windows Vista Platforms

User Locale

1 Select **Start > Control Panel > Regional and Language Options**.

2 Open **Formats** tab.

3 Select an item from the drop-down list.

System Locale

1 Select **Start > Control Panel > Regional and Language Options**.

2 Open **Administrative** tab.

3 Click **Change system locale...** button.

4 Select an item from the drop-down list.

5 Reboot the system.

Locale on Windows XP Platforms

User Locale

1   Select **Start** > **Control Panel** > **Regional and Language Options**.

2   Open **Regional Options** tab.

3   Select an item from the drop-down list.

System Locale

1   Select **Start** > **Control Panel** > **Regional and Language Options**.

2   Open **Advanced** tab.

3   Select an item from the drop-down list.

4   Reboot the system.

Setting Locale on Mac Platforms

To set the user locale (region) on Mac OS X 10.9 and 10.10:

1 Select **System Preferences** >**Language & Region**.

2 Select an item from the **Region** menu.

To set the UI (preferred) language:

1 From the **Language & Region** menu, either drag an existing item to the top of the **Preferred languages** list, or select the **+** button to add a language.

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 From the **Language & Region** menu, 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.

Note: If your user-specified locale is not supported, MATLAB uses the default locale `en_US.US-ASCII`.

Setting Locale on Linux Platforms

Linux platforms manage locale settings with six *locale categories*. These are the same categories used by C standard library functions.

The following locale categories are available:

- LC_CTYPE controls character data manipulations.
- LC_COLLATE controls character collation/sorting operations.
- LC_TIME controls date/time data formatting or parsing.
- LC_NUMERIC controls numeric data formatting or parsing.
- LC_MONETARY controls monetary data formatting or parsing.
- LC_MESSAGES controls the user UI language.

Setting User Locale and User UI Language

Use the LANG environment variable to specify a single locale for all locale categories. The locale specified with this variable might be partially or entirely over-written by other environment variables.

Use the environment variables LC_CTYPE, LC_COLLATE, LC_TIME, LC_NUMERIC, and LC_MONETARY to specify a locale for a particular category.

Use the LC_ALL environment variable to over-write all locales specified with other environment variables. If a single locale has to be set to all locale categories, use LANG instead of LC_ALL.

Note: If your user-specified locale is not supported, MATLAB uses the default locale en_US.US-ASCII.

Asian Characters Incorrectly Displayed on Linux Systems

On some Linux systems, to properly display Asian characters in the MATLAB Desktop, you must configure the font with the Java Runtime Environment (JRE™). If you previously configured fonts for your system, you must also make the configuration changes for the JRE distributed with MATLAB.

To configure, make a symbolic link between your font and the MATLAB font fallback directory. For example, to use the Kochi font, at the Linux system prompt type:

```
mkdir matlabroot/sys/java/jre/glnxa64/jre/lib/fonts/fallback
ln -s /usr/share/fonts/truetype/ttf-japanese-gothic.ttf
matlabroot/sys/java/jre/glnxa64/jre/lib/fonts/fallback
```

where *matlabroot* is the folder where you installed MATLAB.

Alternatively, edit the `fontconfig.properties` file. See your Java documentation for information about this file.

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. For information on controlling these settings, see "Setting Locale on Windows Platforms" on page 8-4.

datenum Might Not Return Correct Value

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;
```

Numbers Display Period for Decimal Point

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 "Comma — ," topic of the Programming Fundamentals Symbol Reference.

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. See **Supported Character Set** in "How the MATLAB Process Uses Locale Settings" on page 8-2.

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.

Non-ASCII Characters Incorrectly Displayed on Different Platforms

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, the script might not be compatible when executed on a platform with the `ja_JP.eucJP` locale setting.