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.
1 Apple Hill Drive
Natick, MA 01760-2098

Revision History

June 2004	First printing	New for MATLAB 7.0 (Release 14). Formerly part of Using MATLAB.
October 2004	Online only	Revised for Version 7.0.1 (Release 14SP1)
March 2005	Online only	Revised for Version 7.0.4 (Release 14SP2)
March 2005	Second printing	Revised for Version 7.0.4 (Release 14SP2)
June 2005	Third printing	Minor revision for Version 7.0.4 (Release 14SP2)
September 2005	Online only	Revised for Version 7.1 (Release 14SP3)
March 2006	Online only	Revised for Version 7.2 (Release 2006a)
September 2006	Online only	Revised for Version 7.3 (Release 2006b)
March 2007	Online only	Revised for Version 7.4 (Release 2007a)
September 2007	Online only	Revised for Version 7.5 (Release 2007b)
March 2008	Online only	Revised for Version 7.6 (Release 2008a)
October 2008	Online only	Revised for Version 7.7 (Release 2008b)
March 2009	Online only	Revised for Version 7.8 (Release 2009a)
September 2009	Online only	Revised for Version 7.9 (Release 2009b)
March 2010	Online only	Revised for Version 7.10 (Release 2010a)
September 2010	Online only	Revised for Version 7.11 (Release 2010b)
April 2011	Online only	Revised for Version 7.12 (Release 2011a)
September 2011	Online only	Revised for Version 7.13 (Release 2011b)
March 2012	Online only	Revised for Version 7.14 (Release 2012a)
September 2012	Online only	Revised for Version 8.0 (Release 2012b)
March 2013	Online only	Revised for Version 8.1 (Release 2013a)
September 2013	Online only	Revised for Version 8.2 (Release 2013b)
March 2014	Online only	Revised for Version 8.3 (Release 2014a)
October 2014	Online only	Revised for Version 8.4 (Release 2014b)
March 2015	Online only	Revised for Version 8.5 (Release 2015a)
September 2015	Online only	Revised for Version 8.6 (Release 2015b)
October 2015	Online only	Rereleased for Version 8.5.1 (Release 2015aSP1)
March 2016	Online only	Revised for Version 9.0 (Release 2016a)
September 2016	Online only	Revised for Version 9.1 (Release 2016b)
March 2017	Online only	Revised for Version 9.2 (Release 2017a)
September 2017	Online only	Revised for Version 9.3 (Release 2017b)
March 2018	Online only	Revised for Version 9.4 (Release 2018a)
September 2018	Online only	Revised for Version 9.5 (Release 2018b)
March 2019	Online only	Revised for Version 9.6 (Release 2019a)
September 2019	Online only	Revised for Version 9.7 (Release 2019b)
March 2020	Online only	Revised for Version 9.8 (Release 2020a)
September 2020	Online Only	Revised for MATLAB 9.9 (Release 2020b)
March 2021	Online Only	Revised for MATLAB 9.10 (Release 2021a)
September 2021	Online Only	Revised for MATLAB 9.11 (Release 2021b)

Contents

Desktop

2

Entering Commands

3

Help and Product Information

4

Workspace Browser and Variable Editor

5

Managing Files in MATLAB

6

Editor Preferences

7

8

9

10

11 | Post-Installation Tasks

12 | MATLAB Cloud Computing

Startup and Shutdown

Start MATLAB on Windows Platforms

Choose one of these ways to start MATLAB.

- "Select MATLAB Icon" on page 1-2
- "Call matlab from Windows System Command Line" on page 1-2
- "Call matlab from MATLAB Command Prompt" on page 1-3
- "Open File Associated with MATLAB" on page 1-3
- "Select MATLAB Executable from Windows Explorer Tool" on page 1-3

When you start MATLAB, it automatically adds the `userpath` folder to the search path. MATLAB also reopens any desktop components that were open when you last shut down.

To customize startup, see "Specify Startup Options" on page 1-18.

The MATLAB startup folder is the folder you are in when you get the MATLAB prompt. To change the default startup folder, see "MATLAB Startup Folder" on page 1-14.

If you have trouble starting MATLAB, then see MATLAB Installation, Activation & Startup Help.

Select MATLAB Icon

On Microsoft Windows platforms, double-click the MATLAB icon. The term *Release* refers to your MATLAB release number; for example, `R2017a`.

- Installer shortcut on your Windows desktop — Double-click the MATLAB icon 🎢.
- Windows 10 systems — **Start** > **All apps** > **MATLAB *Release***
- Windows 8 systems — On the Start screen or the desktop, click **MATLAB *Release***.
- Windows 7 systems — If you chose to have the installer put a shortcut to the MATLAB program on the Windows Start menu, then select **Start** > **MATLAB *Release***

With this option, the startup folder is the last working folder from the previous MATLAB session or the custom path specified in the **Initial working folder** preference. The **Initial working folder** preference is located in the **General Preferences** page of the **Preferences Window**.

You also can specify a custom startup folder in the MATLAB shortcut icon using the **Start in** field. However, this technique is the same as setting a custom path in the **Initial working folder** preference. If you enter an invalid path in the **Start in** field, then the startup folder is unpredictable. To set a custom path in the **Start in** field:

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 **Start in** field, type the full path to a folder on your system, and then click **OK**.

Call matlab from Windows System Command Line

From the Windows system prompt, type `matlab`.

The startup folder is the folder in which you run the `matlab` command. To use the folder specified by the **Initial working folder** preference, type:

```
matlab -useStartupFolderPref
```

Call matlab from MATLAB Command Prompt

If you start another MATLAB session from within MATLAB, then the startup folder is folder where you call the command:

```
!matlab
```

To display the startup folder, type `pwd` before calling `matlab`.

Open File Associated with MATLAB

The MATLAB installer sets up associations between certain file types and MathWorks® products. MATLAB starts when you open one of those files. The startup folder is the folder containing the file.

For example, using the Windows Explorer tool, double-click a file with a `.m` extension. MATLAB starts and opens the file in the MATLAB Editor.

Select MATLAB Executable from Windows Explorer Tool

When you double-click the `matlab.exe` file from the Windows Explorer, the startup folder is one of the following:

- The last working folder from the previous MATLAB session.
- The custom path specified in the **Initial working folder** preference.

See Also
`matlab (Windows)` | `userpath` | **Preferences Window**

More About
- "Specify Startup Options" on page 1-18
- "MATLAB Startup Folder" on page 1-14
- MATLAB Installation, Activation & Startup Help

Associate .mat Files with MATLAB

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

1 Open the Windows **Control Panel**. The **Control Panel** is typically available from the Windows Start menu, or refer to your Windows documentation.

2 From **View by: Category**, select **Programs**.

3 Select **Default Programs**.

4 Select **Set your default programs**. The **Programs** pane shows installed versions of MATLAB, R2010b and later.

5 Select a MATLAB release, then **Choose defaults for this program**.

6 Check the box next to the `.mat` entry.

7 Select **Save**.

8 Close all dialog boxes and menus, and exit the **Control Panel**.

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

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

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

See Also

Start MATLAB on Linux Platforms

To view supported Linux distributions, select the **Linux** tab on System Requirements for MATLAB.

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, then type *matlabroot*`/bin/matlab`. *matlabroot* is the name of the folder in which you installed MATLAB. To see the folder, type `matlabroot`.

After starting MATLAB, the desktop opens. Desktop components that were open when you last shut down MATLAB reopen. If the `DISPLAY` environment variable is not set or is invalid, then the desktop does not display.

MATLAB automatically adds the `userpath` folder to the search path.

To customize startup, see "Specify Startup Options" on page 1-18.

The MATLAB startup folder is the folder you are in when you get the MATLAB prompt. To change the default startup folder, see "MATLAB Startup Folder" on page 1-14.

If you have trouble starting MATLAB, then see MATLAB Installation, Activation & Startup Help.

Startup Folder

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`. To specify a different folder for `userpath`, and for other options, call the MATLAB `userpath` function.

Execute MATLAB Script from Remote ssh Login

To execute a MATLAB script `hello.m` via a remote ssh login, in a terminal window type:

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

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

See Also
`matlab (Linux)` | `userpath` | `matlabroot` | **Preferences Window**

More About
- "Specify Startup Options" on page 1-18
- "MATLAB Startup Folder" on page 1-14
- MATLAB Installation, Activation & Startup Help

Start MATLAB on macOS Platforms

Choose one of these ways to start MATLAB.

- "Start from Applications Folder" on page 1-6
- "Start from Terminal Window" on page 1-6

When you start MATLAB, it automatically adds the `userpath` folder to the search path. MATLAB also reopens any desktop components that were open when you last shut down.

To customize startup, see "Specify Startup Options" on page 1-18.

The MATLAB startup folder is the folder you are in when you get the MATLAB prompt. To change the default startup folder, see "MATLAB Startup Folder" on page 1-14.

If MATLAB fails to start due to a problem with required system components such as Java® software, then diagnostics run automatically. The software advises you of the problem and offers suggestions to correct it. For information about other startup problems, see MATLAB Installation, Activation & Startup Help.

Start from Applications Folder

- Double-click the MATLAB icon in the *matlabroot* folder. Replace *matlabroot* with the name of the folder in which you installed MATLAB.

The startup folder is the last working folder from the previous MATLAB session or the custom path specified in the **Initial working folder** preference. The **Initial working folder** preference is located in the **General Preferences** page of the **Preferences Window**.

Start from Terminal Window

- To start from the Terminal window, you need to know the value of *matlabroot*, the full path to the folder where MATLAB is installed. If you do not know the path, then open MATLAB from the Applications folder, type `matlabroot`, and note the value. Exit MATLAB.
- Open a Terminal window.
- Navigate to the following location from your terminal window:

  matlabroot/bin
- Start MATLAB.

  ./matlab

The startup folder is the MATLAB installation folder.

See Also
`matlab (macOS)` | `userpath` | `matlabroot` | **Preferences Window**

More About
- "Specify Startup Options" on page 1-18

- "MATLAB Startup Folder" on page 1-14
- MATLAB Installation, Activation & Startup Help

Exit MATLAB

In this section...
"Ways to Quit or Exit" on page 1-8
"Confirm Exiting" on page 1-8
"Run Script When Exiting" on page 1-9

Ways to Quit or Exit

At any time, do one of the following:

- Click the close button ⊠ on 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 on page 1-8, if that preference is specified.
- Prompting you to save any unsaved files
- Running a script when exiting on page 1-9, 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.
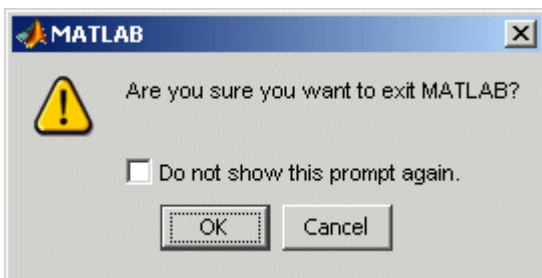
You can also display your own exit confirmation dialog box using a `finish.m` script, as described in the following section.

Run Script When Exiting

When MATLAB exits, it runs the script `finish.m`, if the file exists in the current folder or anywhere on the search path. You create the script containing statements such as saving the workspace or displaying a confirmation dialog box. For more information, see `finish`.

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` | `quit` | `finish` | **Preferences Window**

More About
- "Recovering Data After Abnormal Termination" on page 1-10
- "Crash Reporting" on page 1-11
- MATLAB Installation, Activation & Startup Help

Recovering Data After Abnormal Termination

If MATLAB terminates unexpectedly, then 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, then 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, then you might be able to recover output.

- If you saved the workspace to a MAT-file during the session, then 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, then you should be able to recover changes you made to files you had not saved. To recover, open the backup version *filename*`.asv` in the Editor. Then save it as *filename*`.m` to use the last good version of *filename*.

- If you were in a Simulink® session when a segmentation violation occurred, and you have the Simulink **Autosave Options** preference selected, then 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` | `save` | `load`

Related Examples
- "Save and Load Workspace Variables" on page 5-13
- "Save and Back Up Code"

More About
- "Commonly Used Startup Options" on page 1-17

Crash Reporting

If MATLAB crashes, then the MathWorks Crash Reporter dialog box prompts you to send the crash report to MathWorks. Doing so helps us improve our products. The crash report contains one or more crash logs containing information on the stack trace and the MATLAB configuration. You can see the entire contents of the crash report by clicking the **Show Report** button in the dialog box. This data, along with information that you provide as reproduction steps, is the only information sent to MathWorks.

Reproduction steps are most helpful if you know what appears to cause the crash, but any information on what you were doing at the time helps our developers better understand the cause.

- If you do know the cause of the crash, then look for information about it in the MathWorks Bug Reports database.
- If the problem is reproducible, then submit a Service Request at https://www.mathworks.com/support/contact_us.html.

Locating Crash Logs

In some situations, The Crash Reporter does not open. For example, the Crash Reporter is unavailable when you start MATLAB with the `-batch` option or run it in deployed mode. If you experience abnormal termination, but do not see the Crash Reporter, then you can email the crash logs to MathWorks instead at https://www.mathworks.com/support/contact_us.html. The crash logs are files with the prefix `matlab_crash_dump`, `java.log`, or `hs_err`. To locate the logs, type one of these commands in the MATLAB Command Window, replacing *prefix* with `matlab_crash_dump`, `java.log`, and `hs_err`.

- On Windows platforms

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

  Alternatively, open Windows Explorer and navigate to the `%USERPROFILE%\AppData\Local\` folder.

- On Linux and macOS platforms

  ```
  dir(fullfile('~','prefix.*'))
  ```

  Alternatively on macOS, open a Finder window. Press **Command+Shift+G** to open the Go To Folder dialog box and enter ~.

See Also
`tempdir`

More About
- "When MATLAB Terminates Unexpectedly" on page 1-12

When MATLAB Terminates Unexpectedly

If MATLAB detects an internal error, then MATLAB displays one of two dialog boxes: the "MathWorks Crash Reporter" on page 1-12 or the "MathWorks Crash Analyzer" on page 1-12. In both cases, the internal state of MATLAB is unreliable and not suitable for further use.

MathWorks Crash Reporter

The MathWorks Crash Reporter provides an easy mechanism to send any relevant crash logs to MathWorks Technical Support. If you click **Send Report** and provide a valid email address, then you should receive an automated message within 5 minutes. If you want help, reply to the automated message and a technical support agent will contact you.

If you do not receive an email, then you can directly contact Technical Support at https://www.mathworks.com/support/contact_us.html.

MathWorks Crash Analyzer

When MATLAB terminates unexpectedly, MATLAB locally analyzes the crash report on your machine to determine a potential cause. If there is a potential solution, MATLAB displays the **MathWorks Crash Analyzer** dialog box. Click **View Solution** to open a web page in your browser that points to the relevant MATLAB Answers™ page or MathWorks Bug Reports database. Follow the steps in the solution. To apply a fix, you must restart MATLAB.

After viewing the solution, if you feel that it is not relevant, then click **Send Report** and enter reproduction steps in the window.

If you apply the fix and still experience a crash, then click **Send Report** or create a Service Request directly at https://www.mathworks.com/support/contact_us.html.

See Also

More About

Specifying Java Startup Options

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

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

- MATLAB startup folder, if starting MATLAB from an operating system prompt. For more information, see "MATLAB Startup Folder" on page 1-14.
- If there is no `java.opts` file in the startup folder, then MATLAB checks the *`matlabroot`*`/bin/`*`arch`* folder. *`matlabroot`* is the output of the `matlabroot` function. *`arch`* is the output of the MATLAB `computer('arch')` function, for example `glnxa64`.

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

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

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

  To modify the library path, create a `javalibrarypath.txt` file. For information, see "Locate Native Method Libraries".
- To override built-in options that MATLAB specifies to Java at startup. Options in `java.opts` are appended to the end of the built-in list. Whether these additional options override built-in options is JVM-dependent and can change between Java versions. To see what options MATLAB provides to Java, run this command:

  `java.lang.management.ManagementFactory.getRuntimeMXBean.getInputArguments`

See Also
`matlab (Linux)` | `matlab (macOS)`

Related Examples
- "Java Heap Memory Preferences"
- "Static Path of Java Class Path"
- "Locate Native Method Libraries"

More About
- "MATLAB Startup Folder" on page 1-14

MATLAB Startup Folder

Default Startup Folder

By default, MATLAB sets the startup folder based on the way you 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 **General Preferences** options in the Preferences Window. For convenience, make this folder a folder that you frequently use.

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

Default Folder on Windows Platforms

How You Start MATLAB	Startup Folder
Double-click the MATLAB icon on your Windows desktop or in the Start menu	Last working folder from the previous MATLAB session or the custom path specified in the **Initial working folder** preference. The **Initial working folder** preference is located in the **General Preferences** page of the **Preferences Window**.
From a Windows system prompt	Folder in which you run the `matlab` command (To use the folder specified by the **Initial working folder** preference, use the `-useStartupFolderPref` startup option.)
From the MATLAB command prompt	Folder in which you run the `!matlab` command
Double-click a file type associated with MATLAB	Folder containing the file
Double-click the `matlab.exe` executable file from Windows Explorer Tool	Last working folder from the previous MATLAB session or the custom path specified in the **Initial working folder** preference

For more information, see "Start MATLAB on Windows Platforms" on page 1-2.

If you start MATLAB by double-clicking the MATLAB shortcut icon and the startup folder is set to an unexpected location, follow these steps to ensure that the **Start in** field in the MATLAB shortcut icon is empty:

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 If the **Start in** field contains text, clear the text.

3 Click **OK** and try starting MATLAB again.

Default Folder on macOS Platforms

How You Start MATLAB	Startup Folder
Double-click the MATLAB application	Last working folder from the previous MATLAB session or the custom path specified in the **Initial working folder** preference, located in the **General Preferences** page of the **Preferences Window**
Start MATLAB from a Terminal window	Folder in which you run the `matlab` command

For more information, see "Start MATLAB on macOS Platforms" on page 1-6.

Default Folder on Linux Platforms

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

For more information, see "Start MATLAB on Linux Platforms" on page 1-5.

Change Startup Folder

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

Alternatively on Windows platforms, specify the initial working folder in the MATLAB shortcut icon. For more information, see "Select MATLAB Icon" on page 1-2.

userpath as Initial Working Folder

Using `userpath` as the **Initial working folder** value offers these benefits.

- You can store the MATLAB files you work with in one folder, such as `Documents/MATLAB`.
- 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, the software automatically creates the `userpath` folder if it does not exist.
- When you upgrade to a newer version of MATLAB, the software automatically continues to use the same startup folder and your existing files.
- The default `userpath` uses the benefits provided by the standard location in the Windows and macOS environments for storing personal files. Files in the `Documents/MATLAB` folder are available to you when you use other machines. Each user has an individual `Documents/MATLAB` folder. Other users, even those using your machine, cannot access files in your `Documents/ MATLAB` folder.

See Also
`startup` | **Preferences Window**

Related Examples

- "Start MATLAB on Windows Platforms" on page 1-2
- "Start MATLAB on macOS Platforms" on page 1-6
- "Start MATLAB on Linux Platforms" on page 1-5

Commonly Used Startup Options

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

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.
macOS 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	`-batch "statement"`	Run the specified statement non-interactively.
All	`-r "statement"`	Run the specified statement interactively.
All	`-singleCompThread`	Limit MATLAB to a single computational thread. This option is for numerical computations only. On Windows systems, this option uses the multithreading capabilities of the computer on which it is running by default.

For a complete list of options, refer to the input arguments for `matlab (Windows)`, `matlab (macOS)`, or `matlab (Linux)`.

See Also

`matlab (Windows)` | `matlab (macOS)` | `matlab (Linux)`

Specify Startup Options

In this section...
"Startup Options from Operating System Prompt" on page 1-18
"Startup Options in Shortcut on Windows Systems" on page 1-18
"Startup Options in MATLAB Startup File" on page 1-19
"Passing Perl Variables on Startup" on page 1-19
"Startup and Calling Java Software from MATLAB" on page 1-19

Startup Options from Operating System Prompt

Startup options instruct MATLAB to perform certain operations when you start the program. Startup options also are called command flags or command-line switches. When you start at the operating system prompt, specify the options as arguments to the `matlab` command. 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 to the target path for your shortcut on the Windows platform for MATLAB.

To use startup options for the MATLAB shortcut icon, 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\R2016b\bin\matlab.exe" -r "results"
```

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

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

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

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

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

Startup Options in MATLAB Startup File

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

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

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

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

At startup, MATLAB automatically executes the `matlabrc.m` file and `startup.m`, if it exists on the MATLAB search path. 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, then 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 (macOS)`, 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 "Locate Native Method Libraries".

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

See Also

`matlab (Windows)` | `matlab (macOS)` | `matlab (Linux)` | **Preferences Window**

More About

- "Commonly Used Startup Options" on page 1-17
- "What Is the MATLAB Search Path?" on page 6-45

Toolbox Path Caching in MATLAB

In this section...

About Toolbox Path Caching in MATLAB

For performance reasons, MATLAB disables folder change detection for the $matlabroot$ folder and caches the folder information across sessions. The disabled folder change detection and 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 path cache, then 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 $matlabroot$ folder cache. Because of the cache file, startup is faster, especially if you run MATLAB from a network server or if you have many folders within the $matlabroot$ folder. 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** page of the Preferences Window. 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 the $matlabroot$ folder. This folder is 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 $matlabroot$ folder tree. If you keep your files in the $matlabroot$ folder, then they might be overwritten when you install a new version of MATLAB.

When to Update the Cache

When you add files to the $matlabroot$ folder, 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 the $matlabroot$ folder.

When you add or remove files in the $matlabroot$ folder by some other means, MATLAB might not recognize those changes. For example, when you:

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

and then attempt to use one of the new files, MATLAB displays this message:

```
Undefined function or variable
```

Update the cache so MATLAB recognizes the changes you made in the *matlabroot* folder.

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

Function Alternative

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

Remove canberra-gtk-module and pk-gtk-module Messages

On some Linux distributions, one or both of these messages appear in the terminal window when starting MATLAB. These messages are informational and have no effect on MATLAB.

```
Gtk-Message: <timestamp>: Failed to load module "canberra-gtk-module"
Gtk-Message: <timestamp>: Failed to load module "pk-gtk-module"
```

To suppress these messages, open a terminal with root privileges and execute the relevant commands to install the `libcanberra` Linux package.

Red Hat Distributions

On Red Hat distributions (including CentOS and Fedora™):

- Install the package.

  ```
  yum install libcanberra-gtk2
  ```
- Set the GTK_PATH.

  - `sh` or `bash` commands:

    ```
    $ export GTK_PATH=/usr/lib64/gtk-2.0
    ```
  - `csh/tcsh` commands:

    ```
    % setenv GTK_PATH /usr/lib64/gtk-2.0
    ```

Debian-Based Distributions

On Debian®-based distributions (including Ubuntu®):

- Install the package.

  ```
  apt-get install libcanberra-gtk-module
  ```
- Set the GTK_PATH.

  - `sh` or `bash` commands:

    ```
    $ export GTK_PATH=/usr/lib/x86_64-linux-gnu/gtk-2.0
    ```
  - `csh/tcsh` commands:

    ```
    % setenv GTK_PATH /usr/lib/x86_64-linux-gnu/gtk-2.0
    ```

See Also

2

Desktop

Change Desktop Fonts

You can change the font size, name, and style used by tools in MATLAB. You also can install new fonts to use with MATLAB.

Zoom

You can change the zoom level in the Editor, Live Editor, and Help browser. To zoom in and out, hold the **Ctrl** key and move the scroll wheel. On macOS systems, use the **Command** key instead.

Alternatively, you can zoom in and out using the available keyboard shortcuts.

Action	Keyboard Shortcut
Zoom in	**Ctrl+Plus** On macOS systems, use **Command+Shift+Plus**.
Zoom out	**Ctrl+Minus** On macOS systems, use **Command+Shift+Minus**.
Reset Zoom (Not supported in Help browser)	**Ctrl+Alt+0** On macOS systems, use **Command+Alt+0**.

Alternatively, you can change the zoom level in the Editor and Live Editor by going to the **View** tab and selecting **Zoom In**, **Zoom Out**, or **Reset Zoom**.

In MATLAB Online™, to change the zoom level, adjust the zoom settings for your web browser.

Change Font Size

You can change the font size in MATLAB using preferences or settings. This table describes how to change the font size for each tool.

Tool	Procedure
Live Editor	Specify the font size for code and text programmatically using settings. For more information, see matlab.fonts Settings.
Code tools • Editor • Command Window • Command History	Specify the font size using font preferences. On the **Home** tab, in the **Environment** section, click **Preferences**. Select **MATLAB > Fonts** and, in the **Desktop code font** section, select a font size.

Tool	Procedure
Text-based tools • Current Folder browser • Workspace browser • Variables editor	Specify the font size using font preferences. On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Fonts** and in the **Desktop text font** section, clear the **Use system font** check box. Then, select a font size.
Charts	See "Add Title and Axis Labels to Chart".

In MATLAB Online, you only can change the font size for the Editor (using font preferences) and the Live Editor (using settings).

Change Font Name and Style

You can change the font name and style in MATLAB using several different methods, depending on the tool. This table describes how to change the font name and style for each tool in MATLAB.

Tool	Procedure

Live Editor	Specify the font name and style for code and text programmatically using settings. For example, this code changes the font name and style of normal text in the Live Editor.
	
	``
	s = settings;
	s.matlab.fonts.editor.normal.Name.PersonalValue = 'Ca
	s.matlab.fonts.editor.normal.Style.PersonalValue = {'
	``
	
	For more information, see matlab.fonts Settings.

Code tools • Editor • Command Window • Command History	Specify the font name and style using font preferences. On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Fonts** and, in the **Desktop code font** section, select a font name and style. By default, the code font is set to monospaced to preserve vertical alignment. When the font is set to monospaced, the actual displayed font in the Editor is different than the font displayed in the Command Window and Command History. This is because monospaced is a logical font, not a physical font. To avoid this discrepancy, specify the font name as a physical font, for example, Courier New.

Tool	Procedure
Text-based tools • Current Folder browser • Workspace browser • Variables editor	Specify the font name and style using font preferences. On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Fonts** and, in the **Desktop text font** section, clear the **Use system font** check box. Then, select a font name and style. By default, the text font is set to your system font.
Help Browser MATLAB web browser	Changing the font name and style is not supported.

In MATLAB Online, you only can change the font name and font style for the Editor (using font preferences) and the Live Editor (using settings).

To apply antialiasing and create a smoother desktop appearance on Linux systems, select **Use antialiasing to smooth desktop fonts**. You must restart MATLAB for the preference to take effect. On Microsoft Windows and macOS platforms, MATLAB uses system setting for antialiasing.

Advanced Customization

You can further customize fonts in MATLAB by changing the font preferences for an individual tool. To change the font preferences, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Fonts > Custom** and choose a tool from the list of **Desktop tools**. Then, select the font to use:

• **Desktop code** — Select this option to use the desktop code font for this tool.
• **Desktop text** — Select this option to use the desktop text font for this tool.
• **Custom** — Select this option to specify a custom font for this tool. Then, specify a font name, style, and size. Some options are not supported for all tools.

For example, to change the Editor font to a custom font, in the list of **Desktop tools**, select **Editor**. Then, in the **Font to use** section, select **Custom** and specify the desired font name, style, and size.

Setting custom font preferences is not supported for the Live Editor. To change the font name and style of code and text in the Live Editor, using settings. For more information, see matlab.fonts Settings.

Default Font Preferences

This table shows the default font preferences for the tools in MATLAB. You can use this table to restore fonts to their original state.

Tool	Font to Use	Default Font
Command History	**Desktop code**	Monospaced, plain, 10 point When the font is set to monospaced, the actual displayed font in the Editor is different than the font
Command Window		

Tool	Font to Use	Default Font
Editor (and Shortcuts Editor)		displayed in the Command Window and Command History. This is because monospaced is a logical font, not a physical font. To avoid this discrepancy, specify the font name as a physical font, for example, `Courier New`.
Current Folder browser (and Path browser)	**Desktop text**	Your system's current font
Workspace browser		
Variables editor		
Function Browser		
Profiler	**Custom**	Sans serif, plain, 10 point

Add New Font

You can install new fonts to use with MATLAB. 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 documentation for your system.

Choose a new font that is compatible with MATLAB. MATLAB only includes compatible fonts in the font preferences.

- For desktop components (such as the Command Window), figure windows, and uicontrols, fonts must be compatible with TrueType and Microsoft OpenType® fonts
- For graphics objects, such as `xlabel`, `ylabel`, `title`, and `text`, fonts must be TrueType and Microsoft OpenType fonts

Install your new font in a location where MATLAB looks for fonts. MATLAB looks for fonts in these 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

If you install a font with MATLAB running, restart MATLAB to include the font in the font preferences.

Then, to use the new font, follow the instructions in "Change Font Name and Style" on page 2-3.

See Also
Preferences Window

Related Examples
- "Add Title and Axis Labels to Chart"
- "Change Desktop Colors" on page 2-6
- "Set Print Options for Command Window and Editor" on page 2-38

Change Desktop Colors

In this section...

You can change the colors used by tools in MATLAB, including the text and background colors, syntax highlighting colors, and output colors.

Change Text and Background Colors

By default, the tools in the MATLAB desktop environment use the same text and background colors that your system uses in other applications. You can change the text and background colors for certain desktop tools, including the Editor, Command Window, Current Folder browser, Workspace browser, and Import Wizard. In MATLAB Online, changing the colors of the Current Folder browser and the Workspace browser is not supported.

To change the text and background colors:

1 On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**.

2 Select **MATLAB > Colors**.

3 In the **Desktop tool colors** section, clear the **Use system colors** check box.

In MATLAB Online, the **Use system colors** check box is not available and this step can be skipped.

4 Use the **Text** and **Background** fields to change the colors. For example, select white for the text color and black for the background color.

The **Syntax Highlighting sample** and **Command Window sample** areas show a preview of the selected colors.
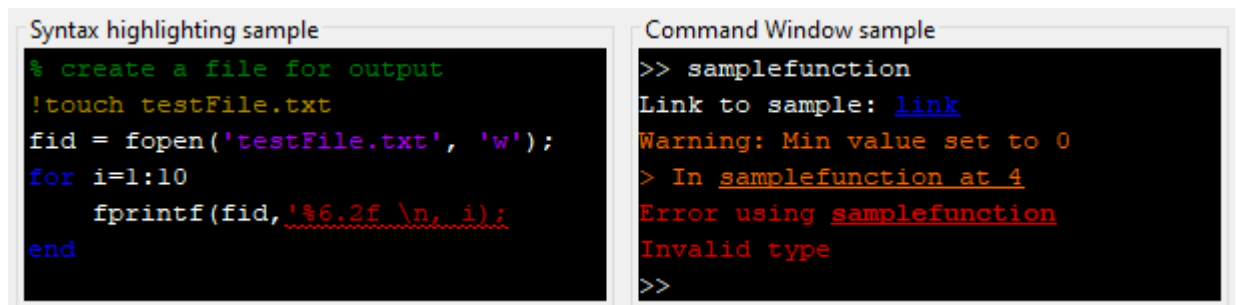
To restore the default text and background colors, click the **Restore Default Colors** button at the bottom of the Preferences Window.

Changing the colors of the toolstrip and Help browser is not supported. To change the text colors for the Live Editor, use settings. For more information, see matlab.fonts.
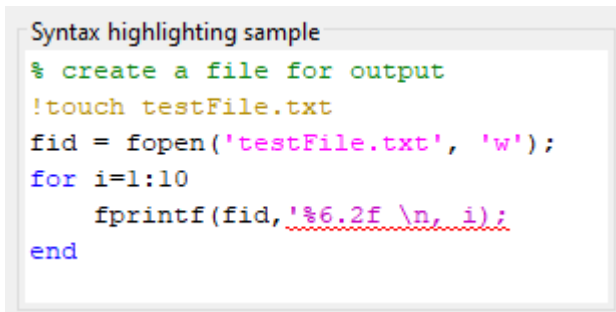
Change Syntax Highlighting Colors

MATLAB conveys syntax information in code using different colors. This feature, known as syntax highlighting, helps you to identify syntax elements, such as `if/then/else` statements, at a glance. Syntax highlighting appears in code files, in code you enter in the Command Window, and in error and warning messages. It does not appear in other kinds of output.

To change the colors used for syntax highlighting in MATLAB code:

1 On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**.

2 Select **MATLAB > Colors**.

3 Use the **Keywords**, **Comments**, **Strings**, **Unterminated strings**, **System commands**, **Syntax errors**, and **Validation sections** fields to change the colors. For example, select magenta for the strings color and purple for the unterminated strings color.

The **Syntax Highlighting sample** area shows a preview of the selected colors.

```
Syntax highlighting sample
% create a file for output
!touch testFile.txt
fid = fopen('testFile.txt', 'w');
for i=1:10
    fprintf(fid,'%6.2f \n, i);
end
```

To restore the default syntax highlighting colors, click the **Restore Default Colors** button at the bottom of the Preferences window.

To disable syntax highlighting, in the Preferences window, select **MATLAB > Editor/Debugger > Language**. Then, in the **Syntax Highlighting** section, clear the **Enable syntax highlighting** check box. In MATLAB Online, this option is located under **MATLAB > Editor/Debugger > MATLAB Language**.

The Editor also provides syntax highlighting for other languages. To change the syntax highlighting colors for other languages, in the **MATLAB > Editor/Debugger > Language** Preferences window, select a language in the **Language** field. Then, change the colors for that language. To disable syntax highlighting for the language, clear the **Enable syntax highlighting** check box. In MATLAB Online, these options are located under **MATLAB > Editor/Debugger > Other Languages**.

For more information about syntax highlighting, see the "Check Syntax as You Type" on page 3-19.

Change Output Colors
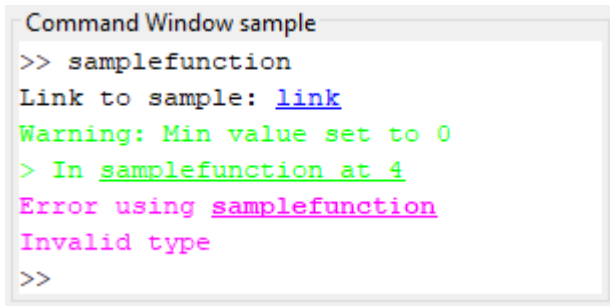
MATLAB uses different colors to identify errors, warnings, and links in Command Window and Live Editor output.

To change the colors used for errors, warnings and links in output:

1 On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**.

2   Select **MATLAB > Colors**.

3   Use the **Error text**, **Warning text**, and **Hyperlinks in Command Window** fields to change the colors. For example, select magenta for error text and green for warning text.

   The **Command Window sample** area shows a preview of the selected colors.

```
Command Window sample
>> samplefunction
Link to sample: link
Warning: Min value set to 0
> In samplefunction at 4
Error using samplefunction
Invalid type
>>
```

   To restore the default output colors, click the **Restore Default Colors** button at the bottom of the Preferences window.

Change Programming Tools Colors

In the Editor and Live Editor, MATLAB uses additional colors to identify Code Analyzer messages, variables, functions, and sections. To change these colors, go to the **Home** tab, and in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Colors > Programming Tools**.

Code Analyzer Colors

To change the colors used to identify Code Analyzer messages in the Editor and Live Editor, in the **Code analyzer colors** section, select colors from the **Warnings** and **Autofix highlight** fields. To disable highlighting autofixes in the Editor and Live Editor, clear the **Autofix highlight** check box.

For more information about using the Code Analyzer, see "Check Code for Errors and Warnings Using the Code Analyzer".

Variable and Function Colors

When you select a variable or function in the Editor or Live Editor, MATLAB highlights all occurrences of the specific variable or function. To change the color of this highlight, in the **Variable and function colors** section, select a color from the **Automatically highlight** field. To disable automatic highlighting in the Editor and Live Editor, clear the **Automatically highlight** check box. For more information about automatic highlighting, see "Find and Replace Functions or Variables in Current File".

MATLAB also highlights variables with shared scope in the Editor and Live Editor. To change the color of this highlight, in the **Variable and function colors**, select a color from the **Variables with shared scope** field. The text is colored, not shaded. For more information about variables with shared scope, see "Check Variable Scope in Editor".

See Also

matlab.fonts | **Preferences Window**

Related Examples

- "Change Desktop Fonts" on page 2-2
- "Set Print Options for Command Window and Editor" on page 2-38

Customize MATLAB Toolbars

In MATLAB, you can change the location of the quick access and the Current Folder toolbars. You also can choose which controls appear in the toolbars.

Quick Access Toolbar

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

By default, the quick access toolbar is located at the upper-right corner of the MATLAB desktop. To 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**. Moving the quick access toolbar is not supported in MATLAB Online.

To add a toolstrip control to the quick access toolbar, right-click the control and then select **Add to Quick Access Toolbar**. You also can select from a list of common controls by going to the quick access toolbar and clicking the ⊙ button. To remove a control from the quick access toolbar, right-click the control and select **Remove from Quick Access Toolbar**. To arrange controls in the quick access toolbar, drag them to the desired locations.

To further customize the quick access toolbar, click the ⊙ button on the toolbar and select **Customize Toolbar**. Then, select from the options described in "Toolbar Preferences" on page 2-10.

Current Folder Toolbar

You can use the Current Folder toolbar to view and change the current folder.

By default, the Current Folder toolbar is located under the toolstrip. To change the location of the Current Folder toolbar, on the **Home** tab, in the **Environment** section, click **Layout**, and then select an option for the **Current Folder Toolbar**. Moving the Current Folder toolbar is not supported in MATLAB Online.

To add, remove, or rearrange the controls that appear in the Current Folder toolbar, right-click the toolbar and select **Customize**. Then, select from the options described in "Toolbar Preferences" on page 2-10.

Toolbar Preferences

You can customize the quick access and Current Folder toolbars using the Toolbar preferences. To open the Toolbar preferences, go to the **Home** tab, and in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Toolbars** and in the **Toolbar** section, select the toolbar that you want to customize.

To rearrange controls in the selected toolbar, in the **Layout** section, drag the icon for a control or separator bar to a new location. You also can select a control icon or separator bar and then use the buttons below the control icons to move the selected control. For example, to move the MATLAB

desktop **Cut** button to the beginning of the quick access toolbar, select the **Cut** icon ✄, and then click the ⇤ button.
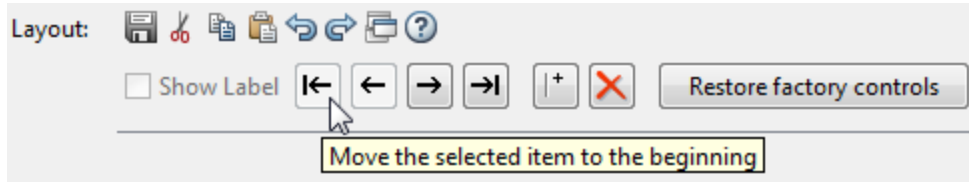
To add or remove controls from the selected toolbar, select or clear the check box for those controls. For example, to easily create new folders in the current folder, you can add a New Folder button to the Current Folder toolbar. To do so, select the 📁 **Folder** button and click **OK**. MATLAB adds the button to the toolbar.

See Also
Preferences Window

More About
- "Rerun Favorite Commands" on page 3-13
- "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23
- "Customize Keyboard Shortcuts" on page 2-33

Change Desktop Layout

When you start MATLAB, the desktop appears in its default layout.
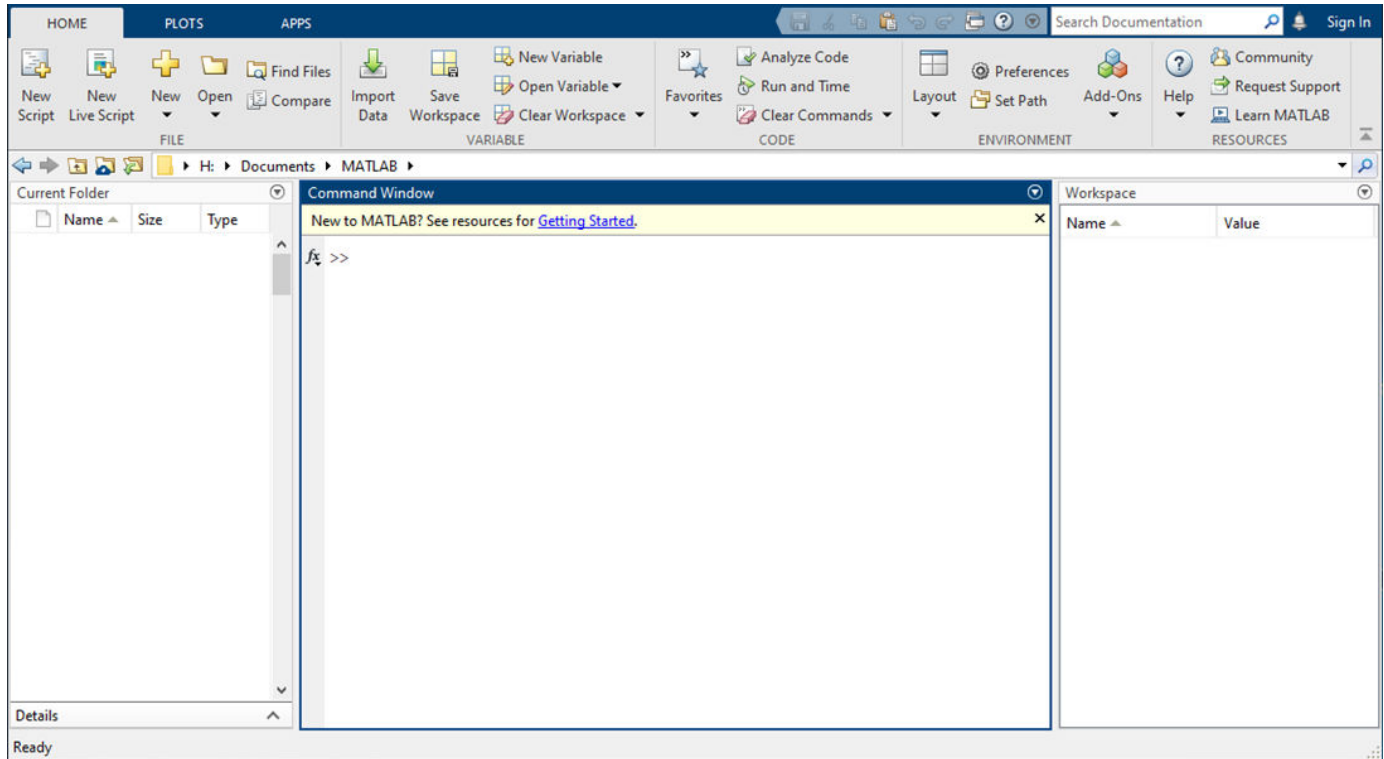
By default, the desktop includes these tools:

Tool	Position	Description
Toolstrip	Top of desktop	By default, the toolstrip includes three tabs, **Home**, **Plots**, and **Apps**.
Current Folder Toolbar	Top of desktop underneath toolstrip	View and change the current folder.
Current Folder Browser	Left side of desktop	Manage files and folders in MATLAB. Actions include viewing, creating, opening, moving, and renaming files and folders in the current folder.
Workspace Browser	Right side of desktop	View and manage the contents of the workspace in MATLAB.
Command Window	Middle of desktop	Enter individual statements at the command line and view the resulting output.

In MATLAB Online on page 10-2 and on systems with a lower screen resolution, MATLAB defaults to a two-column layout with the Workspace browser on the left side of the desktop, underneath the Current Folder browser.

To change the desktop layout, you can choose from a set of preconfigured desktop layouts or you can create your own layout by hiding, minimizing, and undock individual tools. You also can change the

layout of documents within a tool by undocking, reordering, or tiling them. Then, you can save your layouts and reuse them again at a later time.

Select Preconfigured Layout

MATLAB provides a set of preconfigured desktop layouts that are optimized for certain workflows. To select a preconfigured layout, on the **Home** tab, in the **Environment** section, click **Layout** and select a layout. To restore the MATLAB desktop to its default layout, select **Default**.

For example, if you work mostly in the Command Window, select the **All but Command Window Minimized** option (**Minimize All but Command Window** in MATLAB Online) to maximize the Command Window and minimize all other tools in the desktop.
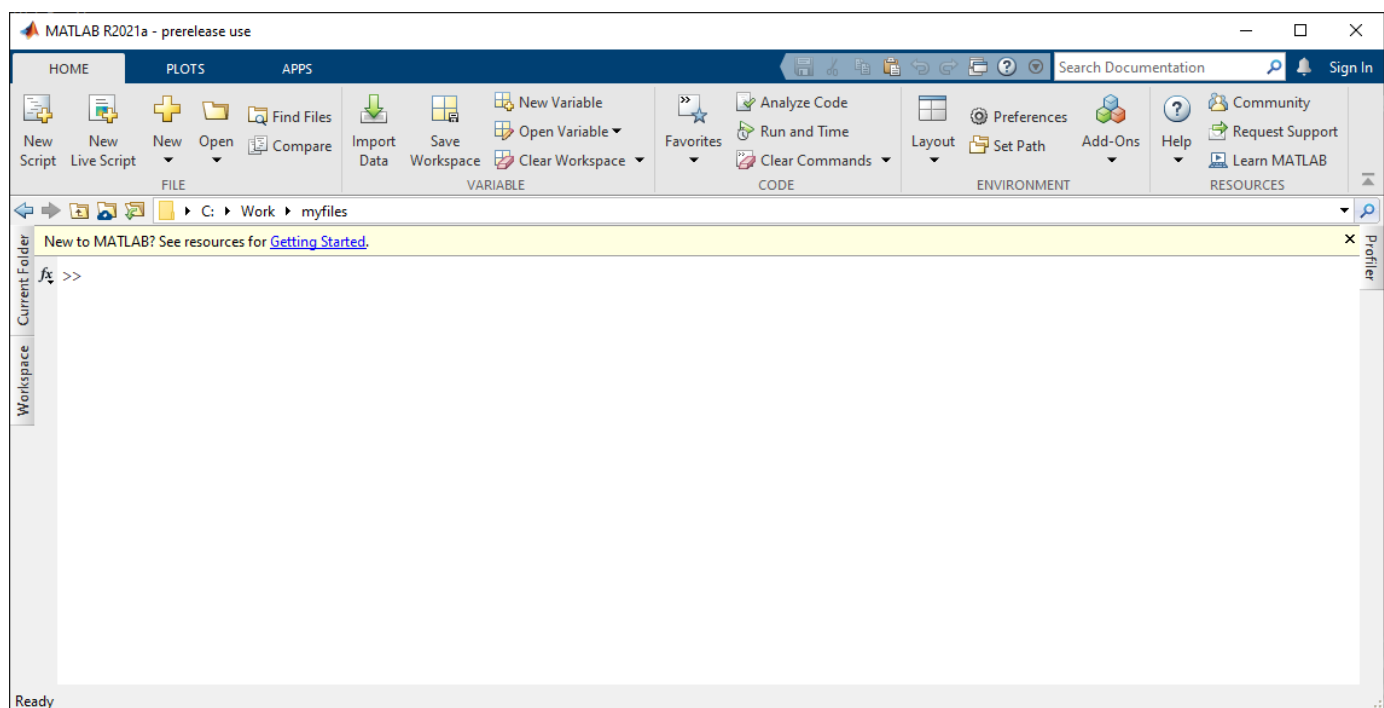
Hide Tools

To hide a tool, click the ⊙ button on the title bar of a tool and select **Close**. To hide just the title bar for a tool, on the **Home** tab, in the **Environment** section, click **Layout**. Then, in the **Show** section, clear **Panel Titles**.

In MATLAB Online, hiding tools is not supported. To optimize your desktop layout, minimize tools instead.

Minimize Tools

Minimize tools in the desktop to optimize the layout while still maintaining quick access to tools that you use periodically.

To minimize the toolstrip, at the lower right corner of the toolstrip, click the ⬆ button. The toolstrip minimizes but the tabs remain visible. To restore the toolstrip, select one of the tabs to display the toolstrip, and then, in the lower right corner of the toolstrip, click the 📌 button.
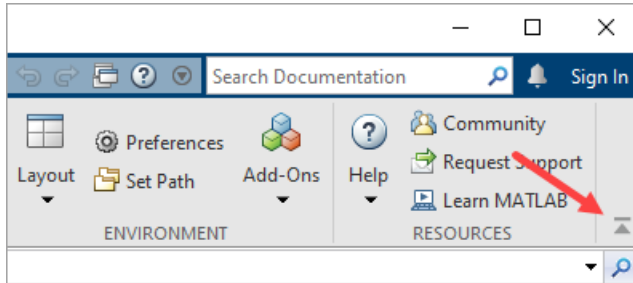
To minimize other tools including the Current Folder browser, Command Window, Editor, and Live Editor, on the title bar of the tool, click the 🔽 button and select **Minimize**. The minimized tool appears along the edge of the desktop indicated by the arrow in the Minimize icon. For example, the minimize icon ➡ indicates that the tool minimizes to the right edge of the desktop by default. To move a minimized tool to a different edge of the desktop, drag the tools to the desired location.

For example, to minimize the Current Folder browser, at top right of the Current Folder browser, click the 🔽 button and select **Minimize**. The Current Folder browser minimizes to the left side of the screen.
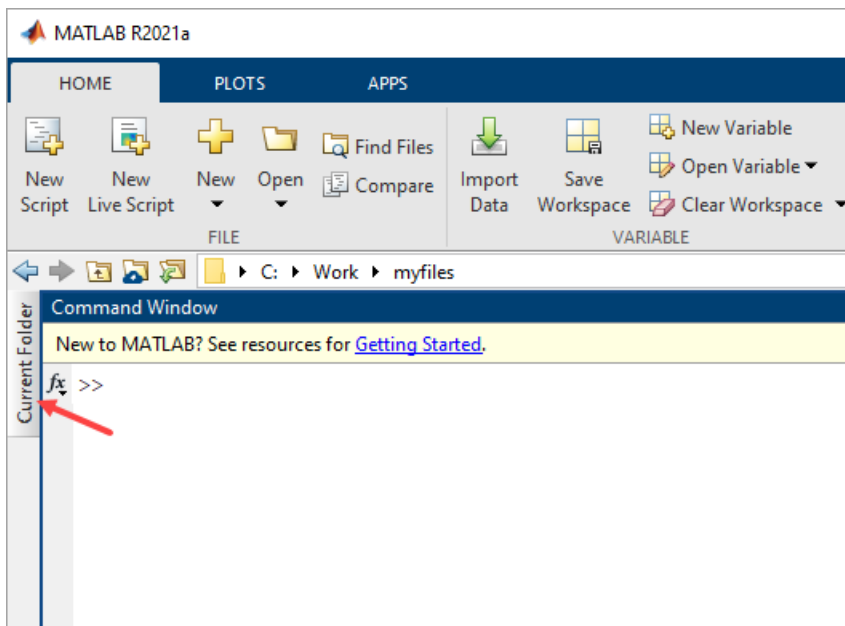
To open a minimized tool temporarily on the desktop, click the tool. To restore the tool to its original size, double click the tool. Alternatively, you can click the 🔽 button and select **Restore**.

In MATLAB Online, to minimize a tool, collapse the panel where the tool resides. For example, if the MATLAB Online desktop is in the default layout with the Current Folder browser in the left panel, hide the Current Folder browser by clicking the ◀ button at the bottom left corner of the panel. To restore it, click the ▶ button. If the left or right panel contains multiple tools, you can minimize an

individual tool by clicking the ⏷ button to the left of its title bar. Multiple tools in the bottom panel display as tabs and cannot be minimized.

Open Tools

To open tools that you previously hid, on the **Home** tab, in the **Environment** section, click **Layout**. Then, in the **Show** section, select the desktop tool that you want to show on the desktop.

You also can open some tools programmatically using a function. For example, to open the Editor, use the `edit` function. To open the Workspace browser, use the `workspace` function.

Undock Tools and Documents

Undock tools to open them in a separate window and move them outside the MATLAB desktop. To undock a tool, on the title bar of the docked tool, click the ⊙ button and select **Undock**. Alternatively, drag the tool by its title bar to a new location outside of the MATLAB desktop. To move an undocked tool back into the desktop, at the upper right of the undocked tool, click the ⊙ button, and select **Dock**.

You also can undock individual documents within a tool such as an individual script file open in the Editor. To undock an individual document, right-click the document tab and select **Undock**. Alternatively, you can click the ⊙ button on the title bar of the tool and select **Undock *document***.

For example, if you have the file `fact.m` open in the Editor, to undock just that file, on the title bar of the Editor, click the ⊙ button and select **Undock fact.m**. MATLAB opens `fact.m` in a separate window and leaves the Editor docked in the desktop. To move the file back into the desktop, at the top right of the undocked file, click the ⊙ button and select **Dock**. If you have multiple undocked documents and want to move them all back into the desktop, select **Dock All in *tool***.
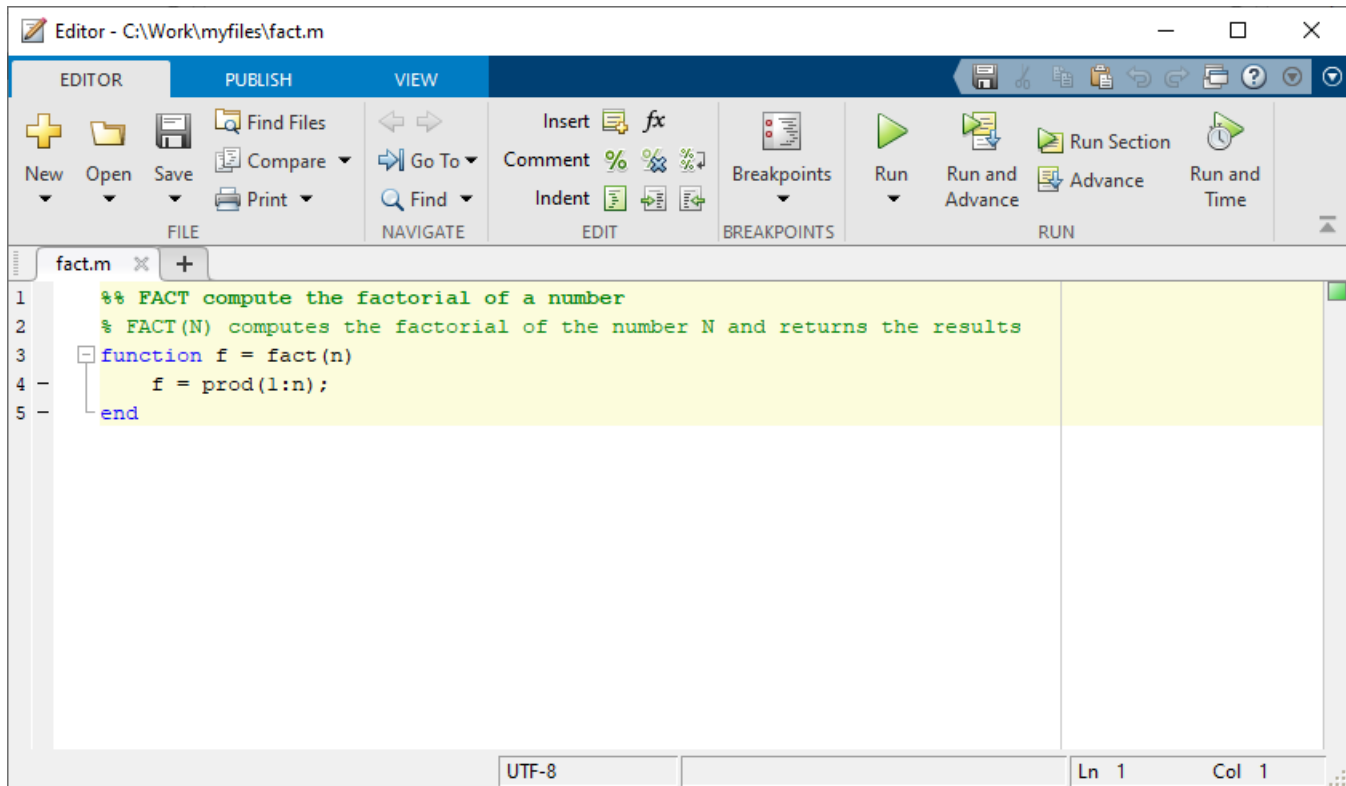
Undocked tools and documents appear on the Windows task bar, or the equivalent for your platform. Click the task bar icon for a tool or document to make it active.

Undocking tools and documents is not supported in MATLAB Online.

Reorder and Tile Documents

When you open MATLAB documents, they open in the associated tool, such as the Editor, Live Editor, or Variables editor. Individual documents open as separate tabs within the tool. To optimize the layout of multiple documents, you can reorder or tile them. You also can change where the tabs appear within a tool.

By default, the tabs appear at the top of the document. To reorder individual document tabs within a tool, drag the tabs to a different position. To reorder document tabs alphabetically, on the **View** tab, in the **Document Tabs** section, select **Alphabetize**.

To change the position of the tabs in a tool, on the **View** tab, click **Tabs Position** and select from the available options. For example, to display tabs on the side of the Editor instead of at the top, with the Editor open, go to the **View** tab, click **Tabs Position**, and select **Left**. MATLAB displays the Editor tabs of the left side of the tool.
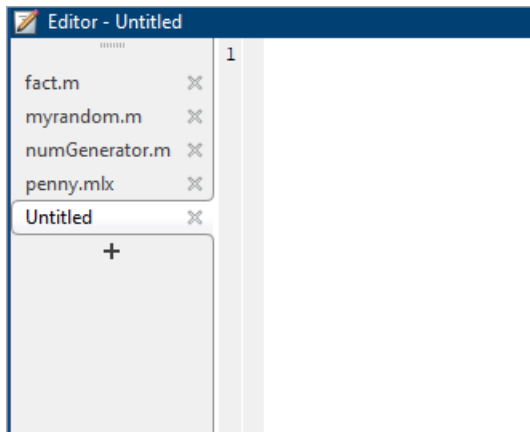
To view multiple documents at once in a tool, you can tile the documents. To tile documents in the Editor, Live Editor, and Variables editor, go to the **View** tab and in the **Tiles** section, select a tile option. For example, to view two files side by side in the Editor, go to the **View** tab and select the **Left/Right** button.
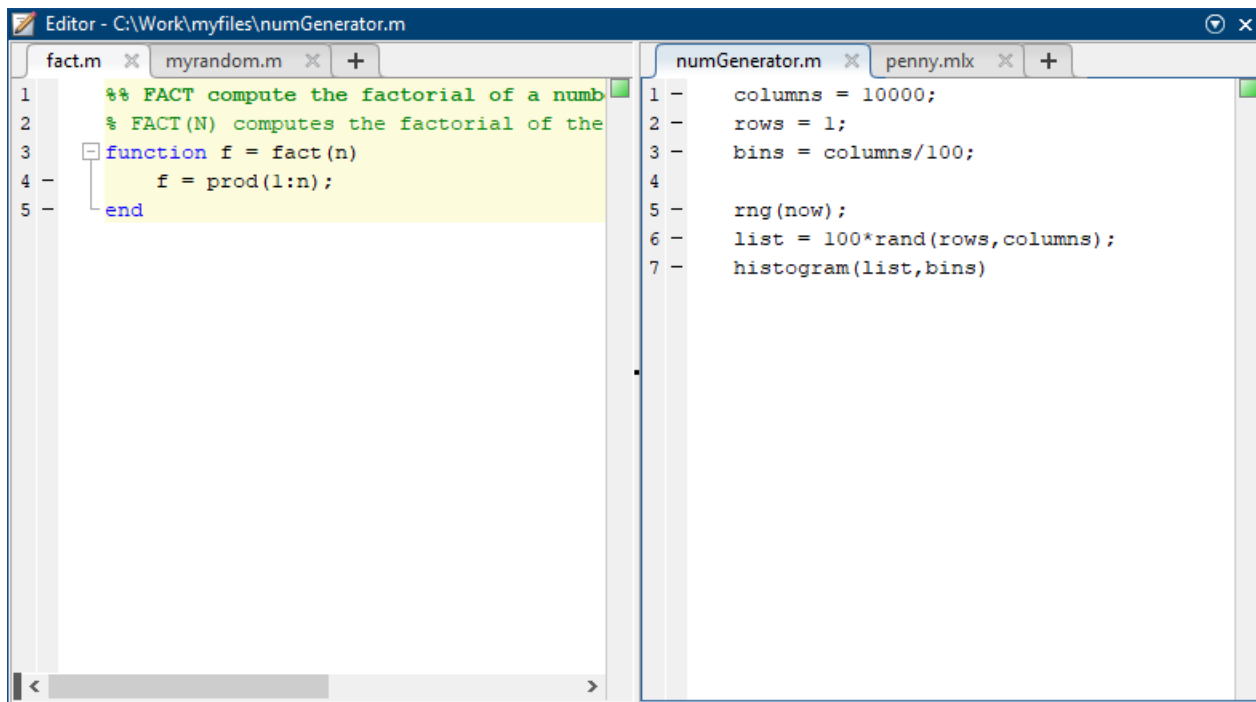
To move a tiled document, drag the document tab to another tile. If you drag it to a tile that already contains a document, the document you are dragging covers up the other document.

To tile documents in a Figure panel, Help browser, or web browser, on the right side of the title bar of the tool, select the ⊞, ⊟, ⊟, or ☐ option button.

In MATLAB Online, to tile documents, on the title bar of the tool, click the ⊙ button, select **Tile All**, and then select a tile option.

Save Desktop Layouts

When you end a session, MATLAB saves the current desktop layout. The next time that you start MATLAB, the desktop appears as you left it.

If you alternate between two or more customized desktop layouts, you can save them all to easily switch between them. To save a layout, on the **Home** tab, in the **Environment** section, click **Layout**, and select **Save Layout**. To use a saved layout, on the **Home** tab, click **Layout**, and select your saved layout. To delete or rename saved layouts, select **Manage Layouts**.

MATLAB stores all saved layouts in the preferences folder. MATLAB saves the current desktop layout at the end of a session in the file `MATLABDesktop.xml`.

Some tools, such as the Help browser, web browser, and Variables editor, do not reopen automatically, even if they were open when you ended the last session. You can use startup options to specify tools that you want to open on startup. For more information, see "Specify Startup Options" on page 1-18.

Saving layouts and specifying startup options is not supported in MATLAB Online.

See Also
Preferences Window

Related Examples
- "Customize MATLAB Toolbars" on page 2-10
- "Change Desktop Fonts" on page 2-2
- "Change Desktop Colors" on page 2-6

Accessibility in MATLAB

MATLAB provides several features to help make it accessible to all users. To navigate MATLAB using just a keyboard, you can use keyboard shortcuts. In MATLAB Online, you can use a screen reader and a keyboard to interact with the different tools in MATLAB. Finally, to make the desktop more visible, you can zoom in or change its colors.

For our best accessibility support, use MATLAB Online on page 10-2. MathWorks is committed to improving the accessibility of its software products, focusing first on the accessibility of MATLAB Online. See our Accessibility Statement for MATLAB for the latest information on accessible workflow support.

Navigate Using the Keyboard

You can use keyboard shortcuts to navigate MATLAB with just a keyboard. Navigating with a keyboard differs depending on whether you are using MATLAB Online or an installed version of MATLAB.

In MATLAB Online, to move forward through the different areas of the MATLAB Online desktop, press **Ctrl+F6**. To move backward, press **Ctrl+Shift+F6**. On macOS systems, use the **Command** key instead of the **Ctrl** key. To display a list of navigational keyboard shortcuts, press **Ctrl+/**.

In an installed version of MATLAB, to move forward through the different areas of the MATLAB desktop, press **Ctrl+Tab**. To move backward, press **Ctrl+Shift+Tab**. To select a toolstrip tab, press **Alt** to show the access keys for the tabs and then ***\<character\>*** to select the tab, where ***\<character\>*** is the displayed access key for the desired tab. After selecting a tab, press ***\<character\>*** to select an item on the tab, where ***\<character\>*** is the displayed access key for the desired toolstrip item. Accessing the toolstrip using keyboard shortcuts in an installed version of MATLAB is not supported on macOS or Linux systems.

For more information, see "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23.

Work with a Screen Reader

In MATLAB Online, you can use a screen reader and a keyboard to interact with the Command Window, create and edit scripts and functions in the Editor, and navigate through the different areas of the MATLAB desktop. MATLAB Online is tested with the NVDA screen reader on Windows 10 running MATLAB Online in the Google Chrome™ browser. For more information about NVDA, see Get Help on the NV Access website.

For example, on a Windows system with the NVDA screen reader and MATLAB Online running, enter individual statements in the Command Window.

1   With the cursor in the Command Window, create a variable named x by entering this statement in the Command Window.

```
x = 1
```

MATLAB adds x to the workspace and displays the results in the Command Window. You hear:

```
Greater greater
X equals one
```

When you hear `Greater greater`, the Command Window is ready for the next statement at the command-line prompt (>>).

2 Create another variable y by entering the following statement in the Command Window.

`y = cos(x)`

You hear the results:

`Y equals zero point five four zero three`

3 Clear all text from the Command Window by entering the following statement in the Command Window.

`clc`

You hear:

`Greater greater`

The Command Window is ready for the next statement.

For more information, see "Use a Screen Reader in MATLAB Online" on page 2-27.

Zoom

You can adjust the zoom level in MATLAB. The way you zoom differs depending on whether you are using MATLAB Online or an installed version of MATLAB.

In MATLAB Online, you can zoom in and out of the entire desktop by adjusting the zoom settings for your web browser.

In an installed version of MATLAB, you can change the zoom level in the Editor, Live Editor, and Help browser. To zoom in and out, hold the **Ctrl** key and move the scroll wheel. On macOS systems, use the **Command** key instead.

Alternatively, you can zoom in and out using the available keyboard shortcuts.

Action	Keyboard Shortcut
Zoom in	**Ctrl+Plus** On macOS systems, use **Command+Shift+Plus**
Zoom out	**Ctrl+Minus** On macOS systems, use **Command+Shift +Minus**
Reset Zoom (Not supported in Help browser)	**Ctrl+Alt+0** On macOS systems, use **Command+Alt+0**

For tools such as the Command Window, Current Folder browser, and Workspace browser, you can increase or decrease the displayed font size using font preferences. For more information, see "Change Desktop Fonts" on page 2-2.

Maximize Focus Area When Zooming

To maximize the space for your current area of focus when zooming, minimize the tools that you are currently not using. In MATLAB Online, to minimize a tool, collapse the panel where the tool resides.

For example, to minimize the Current Folder browser and Workspace browser, click the collapse button ◁ at the bottom left corner of the panel. To restore them, click the expand button ▷.

In an installed version of MATLAB, to minimize tools such as the Current Folder browser, Command Window, Editor, and Live Editor, on the title bar of the tool, click the action menu button ⦿ and select **Minimize**. For example, to minimize the Current Folder browser, at top right of the Current Folder browser, click the action menu button ⦿ and select **Minimize**.

For more information about minimizing tools, see "Change Desktop Layout" on page 2-12.

Change Colors in MATLAB

You can change the text and background colors for certain desktop tools in MATLAB, including the Editor, Command Window, Current Folder browser, Workspace browser, and Import Wizard. In MATLAB Online, changing the colors of the Current Folder browser and the Workspace browser is not supported.

To change the text and background colors:

1 In the Command Window, type `preferences Colors` to open the Preferences window with the MATLAB Colors Preferences page selected.

2 In the **Desktop tool colors** section, clear the **Use system colors** check box.

   In MATLAB Online, the **Use system colors** check box is not available and this step can be skipped.

3 Use the **Text** and **Background** fields to change the colors. For example, select white for the text color and black for the background color.

   The **Syntax Highlighting sample** and **Command Window sample** areas show a preview of the selected colors.
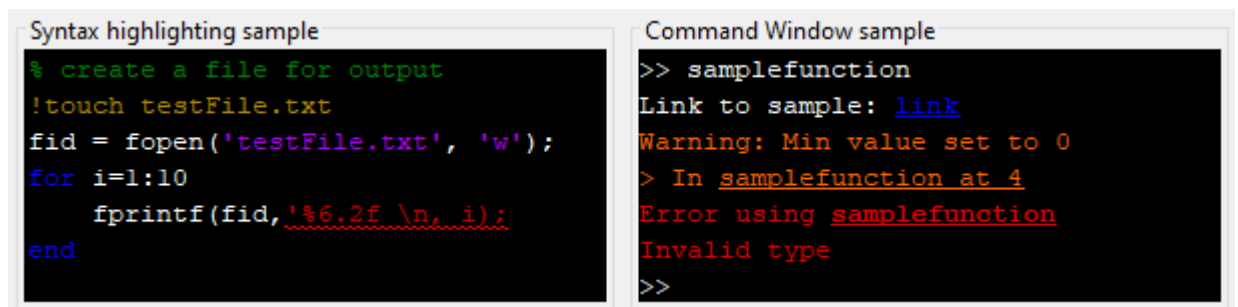
To restore the default text and background colors, click the **Restore Default Colors** button at the bottom of the Preferences window.

For more information, see "Change Desktop Colors" on page 2-6.

Changing the colors of the toolstrip and Help browser is not supported. To change the color of text in the Live Editor, use settings. For more information, see matlab.fonts.

See Also
matlab.fonts

Related Examples

- "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23
- "Use a Screen Reader in MATLAB Online" on page 2-27
- "Change Desktop Fonts" on page 2-2
- "Change Desktop Colors" on page 2-6
- Accessibility Statement for MATLAB

Use Keyboard Shortcuts to Navigate MATLAB

Using keyboard shortcuts to navigate MATLAB can increase productivity and is useful in situations where using a mouse is not an option.

Installed MATLAB

If you have MATLAB installed on your system, this table describes the actions and related keyboard shortcuts useful for navigating MATLAB without a mouse.

Action	Keyboard Shortcut
Move to the next visible panel.	**Ctrl+Tab**
Move to the previous visible panel.	**Ctrl+Shift+Tab**
Move to the next tab in a panel.	**Ctrl+Page Down**
Move to the previous tab in a panel.	**Ctrl+Page Up**
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** On macOS systems, use the **Command** key instead of the **Ctrl** key.
Show access keys for the toolstrip tabs.	**Alt** Not supported on macOS and Linux systems.

Action	Keyboard Shortcut
Select or open a toolstrip tab and show the access keys for each item on the tab.	Press **Alt** to show the access keys for the toolstrip tabs and then ***<character>*** to select or open a tab, where *<character>* is the displayed access key for the desired toolstrip tab. For example, press **Alt** followed by **H** to select or open the **Home** tab and display the access keys for the items on that tab. Not supported on macOS and Linux systems.
Select a toolstrip item.	Press **Alt** and then ***<character>*** to select the current toolstrip tab, where *<character>* is the displayed access key for the current tab. Then, press ***<character>*** to select a toolstrip item, where *<character>* is the displayed access key for the desired item. For example, press **Alt** followed by **H** to select the **Home** tab and then **S** to create a new script. Not supported on macOS and Linux systems.

You cannot customize most of these shortcuts. For information about customizable keyboard shortcuts and how to view and modify them, see "Customize Keyboard Shortcuts" on page 2-33.

MATLAB Online

MATLAB Online on page 10-2 provides access to MATLAB from a standard web browser. Because MATLAB Online runs in a browser, navigation using the keyboard is slightly different.

This table describes the actions and related keyboard shortcuts useful for navigating MATLAB Online without a mouse.

Action	Keyboard Shortcut
Move forward through the different areas of the MATLAB Online desktop, including the toolstrip, Current Folder toolbar, Current Folder browser, Workspace browser, and Command Window.	**Ctrl+F6** On macOS systems, use **Command+F6** instead.
Move backward through the different areas of the MATLAB Online desktop, including the toolstrip, Current Folder toolbar, Current Folder browser, Workspace browser, and Command Window.	**Ctrl+Shift+F6** On macOS systems, use **Command+Shift+F6** instead.
Move into a tool, for example, into the current toolstrip tab.	**Tab**

Action	Keyboard Shortcut
Move between controls within a tool, for example, between toolstrip tabs or between the items on a toolstrip tab.	**Up/Down Arrow**, **Left/Right Arrow**
Close controls within a tool, for example, documents in the Editor and Live Editor.	**Delete**
Open context menu.	**Shift+F10** Not supported on macOS systems.
Show access keys for the toolstrip.	**Alt** Not supported on macOS systems.
Open a toolstrip tab and show access keys for the toolstrip.	**Alt+<*character*>** For example, pressing **Alt** followed by **H** accesses the **Home** tab and displays access keys for the features available on that tab. Not supported on macOS systems.
Display a compact list of keyboard shortcuts.	**Ctrl+/**

Customizing shortcuts is not supported in MATLAB Online.

Navigating Figures in MATLAB Online

In MATLAB Online, plotted data can be explored using keyboard shortcuts. Once a figure is selected, the shortcuts described above are slightly modified.

This table describes the actions and related keyboard shortcuts useful for navigating figures in MATLAB Online without a mouse.

Action	Keyboard Shortcut
Navigate Forward	**Tab** When focus is on the figure, this means **Tab** steps into the axes. If if there are multiple axes in a figure, and focus is on the first axes, then **Tab** steps to the next axes.
Reverse Navigation	**Shift + Tab** Navigate axes in the reverse order as **Tab**.
Navigate Down	**Control + down** If focus is on a set of axes, then **Control + down** focuses the first focusable element within the axes. For example, if a title is present, focus jumps from the axes to the title. If no titles or labels are present, then focus jumps from the axes to an object within the axes.

Action	Keyboard Shortcut
Navigate up	**Control + up** Focus shifts from an element to its parent element. For example, if focus is on a line, then **Control + up** shifts focus to the axes.

Additional Keyboard Shortcuts

In addition to navigation, keyboard shortcuts are useful for accessing other frequently used actions in MATLAB.

This table describes several of these actions and their related keyboard shortcuts. For additional keyboard shortcuts, see the documentation for a specific tool or feature.

Action	Keyboard Shortcut
Cancel the current action.	**Esc** (escape) For example, if you click the name of the **Edit** menu, the whole menu appears. Pressing **Esc** hides the menu again. In the Function Browser, pressing **Esc** up to three times has the following effects: **1** Dismiss the search history. **2** Clear the search field. **3** Close the Function Browser.
Interrupt MATLAB execution.	**Ctrl+C** On Windows and Linux systems, you also can use **Ctrl +Break**. On macOS systems, you also can use **Command+.** (period).

You cannot customize these shortcuts. For information about customizable keyboard shortcuts and how to view and modify them, see "Customize Keyboard Shortcuts" on page 2-33.

See Also

More About

- "Customize Keyboard Shortcuts" on page 2-33
- "Rerun Favorite Commands" on page 3-13
- "Customize MATLAB Toolbars" on page 2-10

Use a Screen Reader in MATLAB Online

In MATLAB Online on page 10-2, you can use a screen reader and a keyboard to interact with the Command Window, create and edit scripts and functions in the Editor, and navigate through the MATLAB desktop tools. MATLAB Online is tested with the NVDA screen reader on Windows 10 running MATLAB Online in the Google Chrome browser. For more information about NVDA, see Get Help on the NV Access website.

Navigate MATLAB Using a Screen Reader

When you first start MATLAB Online, the cursor is in the Command Window. To move forward through the different areas of the MATLAB Online desktop, press **Ctrl+F6** (or **Command+F6** on macOS systems). To move backward, press **Ctrl+Shift+F6** (or **Command+Shift+F6** on macOS systems).

By default, MATLAB Online has five areas that you can navigate to:

Tool	Position	Description
Toolstrip	Top of desktop	By default, the toolstrip includes three tabs, **Home**, **Plots**, and **Apps**. To navigate between the toolstrip tabs, use the arrow keys. To enter and navigate within the toolstrip tabs, use the **Tab** key.
Current Folder Toolbar	Top of desktop underneath toolstrip	View and change the current folder.
Current Folder Browser	Left side of desktop	Manage files and folders in MATLAB. Actions include viewing, creating, opening, moving, and renaming files and folders in the current folder.
Workspace Browser	Left side of desktop underneath Current Folder browser	View and manage the contents of the workspace in MATLAB.
Command Window	Middle of desktop	Enter individual statements at the command line and view the resulting output.
Docked figures	Middle of desktop. Above Command Window	Plot data in embedded figures.

For example, you can use the toolstrip to send feedback about the current release:

1   On a Windows system with the NVDA screen reader running, open a Google Chrome browser and start MATLAB Online.
2   With the cursor in the Command Window, press **Ctrl+F6** to navigate to the toolstrip. You hear:

```
Toolstrip region
Toolstrip tab control
Home tab selected one of three
```
3   Press the **Tab** key to enter the **Home** tab and then continue pressing the **Tab** key until you reach the **Feedback** button. You hear:

```
Feedback button
Send us feedback or report a bug
```
4 Press **Enter**. The Send Feedback dialog opens.

For more information about the MATLAB desktop, see "Change Desktop Layout" on page 2-12.

Enter Statements in the Command Window Using a Screen Reader

As you work in MATLAB, you can enter individual statements in the Command Window using a screen reader and a keyboard.

For example:

1 On a Windows system with the NVDA screen reader running, open a Google Chrome browser and start MATLAB Online.
2 With the cursor in the Command Window, create a variable named x by typing this statement in the Command Window.

```
x = 1
```

MATLAB adds x to the workspace and displays the results in the Command Window. You hear:

```
Greater greater
X equals one
```

When you hear `Greater greater`, the Command Window is ready for the next statement at the command-line prompt (>>).
3 Create another variable y by entering the following statement in the Command Window.

```
y = cos(x)
```

You hear:

```
Y equals zero point five four zero three
```
4 Set the value of x to the value of y by entering the following statement in the Command Window.

```
x = y
```

You hear:

```
X equals zero point five four zero three
```
5 Save the variables from the current workspace in the file `test.mat` by entering the following statement in the Command Window.

```
save test.mat
```

MATLAB saves the file in the current folder. You hear:

```
Greater greater
```
6 Remove the variables from the workspace by entering the following statement in the Command Window.

```
clear
```

You hear:

```
Greater greater
```

7 Clear all text from the Command Window by entering the following statement in the Command Window.

```
clc
```

You hear:

```
Greater greater
```

The Command Window is ready for the next statement.

For more information about entering statements in the Command Window, see "Enter Statements in Command Window" on page 3-2.

Create Scripts Using a Screen Reader

Scripts are the simplest kind of program file because they have no input or output arguments. They are useful for automating series of MATLAB commands, such as computations that you have to perform repeatedly from the command line or series of commands you have to reference.

You can create and edit scripts using a screen reader and a keyboard.

For example:

1 On a Windows system with the NVDA screen reader running, open a Google Chrome browser and start MATLAB Online.
2 Navigate to the toolstrip by typing **Ctrl+F6** until you hear:

```
Toolstrip region
Toolstrip tab control
Home tab selected one of three
```

3 Press the **Tab** key to enter the **Home** tab. The **New Script** button is selected. You hear:

```
Home property page
File grouping
New script button
Create a new script in the current folder
```

4 Press **Enter** to create a new script. MATLAB creates the script and opens it in the Editor. You hear:

```
Untitled property page
Edit multiline, line one
```

5 Add this code that generates an array of random numbers and then calculates the size and mean of the array.

```
n = 50;
r = rand(n,1);

sz = size(r)
m = mean(r)
```

6 To save the file, navigate to the toolstrip by typing **Ctrl+F6** until you hear:

```
Toolstrip region
Toolstrip tab control
Editor tab selected four of seven
```

If the **Home** tab is selected instead, you hear:

```
Home tab selected one of seven
```

To select the **Editor** tab, press the **Right Arrow** key until you hear:

```
Editor tab selected four of seven
```

7   Press the **Tab** key to enter the **Editor** tab and then continue pressing the **Tab** key until you reach the **Save** button. You hear:

```
Clickable save button
Save all modified files
Ctrl plus S
```

8   Press **Enter** to save the script. Alternatively, you can save the script using the keyboard shortcut **Ctrl+S**. MATLAB opens the Save As dialog. You hear:

```
Save as dialog
Edit untitled.m
```

9   Type **Ctrl+A** to select the default file name and then enter a name for the script. For example, enter the name `myrandom.m`. Press **Enter** to save the file.

10 To run the file, navigate to the toolstrip by typing **Ctrl+F6** until you hear:

```
Toolstrip region
Toolstrip tab control
Editor tab selected four of seven
```

11 Press the **Tab** key to enter the **Editor** tab and then continue pressing the **Tab** key until you reach the **Run** button. You hear:

```
Run grouping
Clickable run button
Save and run the active file
F5
```

12 Press **Enter** to run the script. MATLAB runs the script and displays the output in the Command Window. You hear:

```
myrandom.m property page
Edit multiline, line 5
m equals mean r
sz equals 50 1
m equals zero point five one seven two
```

Alternatively, you can press **F5** to run the script. Before pressing **F5**, make sure that focus is inside the script. Otherwise, pressing **F5** reloads MATLAB Online in the browser.

For more information about creating scripts, see "Create Scripts".

Explore Plotted Data Using a Screen Reader

Plotting data in figures allows visualization of data. They can be used to quickly survey data and display the output of code. You can explore figures using a screen reader and a keyboard.

For example:

1   On a Windows system with the NVDA screen reader running, open a Google Chrome browser and start MATLAB Online.

2   With the cursor in the Command Window, create a variable named `x` by typing this statement in the Command Window.

```
x = 1:5
```

MATLAB adds `x` to the workspace and displays the results in the Command Window.

You hear:

```
Greater greater
X equals one two three four five
```

When you hear `Greater greater`, the Command Window is ready for the next statement at the command-line prompt (>>).

3 Create another variable `y` by entering the following statement in the Command Window.

```
y = 2*x
```

You hear:

```
Y equals two four six eight ten
```

4 Plot the data of `y` against the data of `x` by entering the following statement in the Command Window.

```
plot(x,y)
```

You hear:

```
Figure 1
The current axes object contains one objects of type line
```

5 Add a title and legend to the figure.

```
title("My Plot")
legend
```

You hear:

```
Figure 1
The current axes object contains one objects of type line
```

6 Navigate to the new figure tab by typing **Ctrl+F6** until you hear:

```
Tab control
Figure 1 tab selected
Figure 1, 1 of 1
```

7 Press **Tab** twice to select the figure axes.

You hear:

```
Figure 1
The current axes object contains one objects of type line
The axes object with title my plot contains one objects of type line
```

8 Type **NVDA Modifier Key+Space** to switch to focus mode for the figure axes. By default, the NVDA Modifier Key is the **Insert** key.

9 Type **Ctrl+Down Arrow** to select the title inside the axes.

You hear:

```
Text my plot
```

10 Press **Tab** to select the line object. If axes contain multiple objects, press **Tab** to navigate through them.

For more information about exploring plotted data, see "Interactively Explore Plotted Data".

Screen Reader Tips and Limitations

Reduce Blank Lines in Output

To improve the screen reader experience when working in the Command Window, suppress excess blank lines in output for the current session using the `format` command.

```
format compact;
```

To maintain the compact format across all future sessions, use settings to set the personal value for line spacing in output display.

```
s = settings;
s.matlab.commandwindow.DisplayLineSpacing.PersonalValue = 'compact';
```

Modifying Graphics Objects

When working with graphics objects, avoid using the property inspector to view and modify properties. Instead, view and modify graphics object properties programmatically. For example, the `plot` function returns a chart line object. You can use dot notation to view and set properties.

```
p = plot(1:10,1:10);
p.LineWidth = 3;
```

Alternatively, you can set properties using name-value pair arguments when creating the object, such as `plot(1:10,1:10,'LineWidth',3)`.

For more information about viewing and modifying graphics object properties, see "Modify Graphics Objects".

Accessing MathWorks Documentation

To ensure the best experience when viewing and searching the documentation using a screen reader, open it in a separate browser tab or window at https://www.mathworks.com/help/.

Live Editor

Using a screen reader is not supported in the Live Editor. To create scripts and functions, use the Editor instead.

See Also

More About
- "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23
- "Desktop Basics"

Customize Keyboard Shortcuts

Keyboard shortcuts are useful for accessing desktop features quickly. To optimize your experience, you can customize the current set of keyboard shortcuts in MATLAB.

There are several ways you can modify the current set of keyboard shortcuts. You can:

- Create new keyboard shortcuts or modify existing ones.
- Choose among available sets of keyboard shortcuts.
- Create customized sets of keyboard shortcuts or use existing customized sets.

To view or modify the current set of keyboard shortcuts, use the **Keyboard Shortcuts Preferences** options in the Preferences Window. To open the **Keyboard Shortcuts Preferences** page in the Preferences Window, go to the **Home** tab, and in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Keyboard > Shortcuts**.

View Keyboard Shortcuts

For actions performed with a menu item, keyboard shortcuts appear in the menu itself. This is true whether the menu item is in the toolstrip or in a context menu. If a menu item does not display a keyboard shortcut, then a keyboard shortcut for the action does not exist.

For actions that are not performed with a menu item, to find keyboard shortcuts, open the **Keyboard Shortcuts Preferences** page in the Preferences Window. Then, in the search box, enter the name of the tool or action for which you want to see the keyboard shortcuts. For example, type `Editor clear` to find the keyboard shortcut for clearing selected text in the Editor.
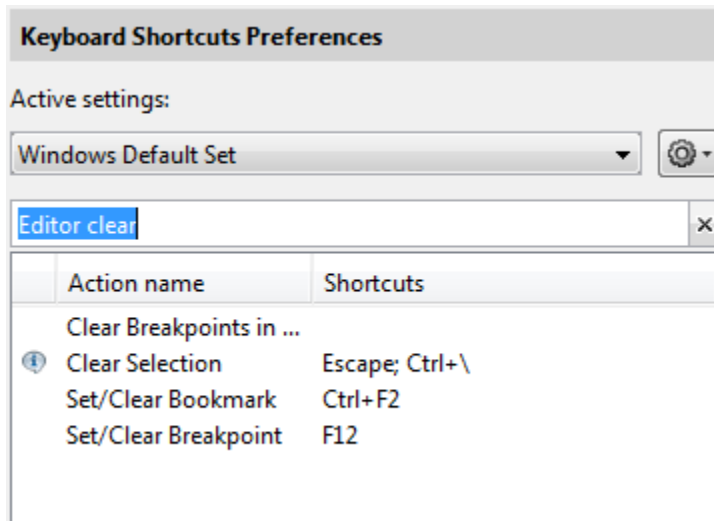
To view the shortcuts for an action, select the action name. For example, select **Clear Selection**. MATLAB displays the keyboard shortcuts for the selected action. For example, the keyboard shortcut for the **Clear Selection** action in the Editor is the **Escape** key.

Shortcuts for **Clear Selection**

	Shortcut	Tools with shortcut	
ⓘ 🔹▾	Escape	MATLAB Editor	▾
🔹▾	Ctrl+\	Command Window	▾

[+] [−]
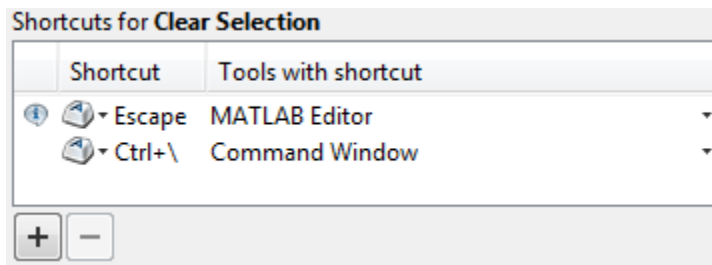
Create a List of All Keyboard Shortcuts in a Set

You can create a list of all the keyboard shortcuts in the current set by copying them to the clipboard and pasting them in a text file or spreadsheet application, such as Microsoft Excel®. For the best formatting, use a spreadsheet application. To copy all the keyboard shortcuts, open the **Keyboard Shortcuts Preferences** page in the Preferences Window. Click the Actions button ⚙▾ and from the drop-down menu, choose `Copy to Clipboard`. Paste the data into the desired application.

Customize Keyboard Shortcuts for Individual Actions

Customizing a keyboard shortcut is useful if you frequently perform an action and the action does not have a keyboard shortcut defined, or if the defined keyboard shortcut is hard to use or conflicts with a different shortcut. For an overview on customizing keyboard shortcuts, watch the Customizable Keyboard Shortcuts video. You must have an internet connection to watch the video.

To customize a keyboard shortcut:

1 Open the **Keyboard Shortcuts Preferences** page in the Preferences Window and, in the search box, type an existing keyboard shortcut or the name of an action, tool, or menu.

For example, you can type `Ctrl+R` (shortcut), `Delete` (action), `Command Window` (tool), or `File` (menu).

You can enter most keyboard shortcuts by either pressing keystrokes or typing the key names. If using keystrokes for a keyboard shortcut does not work, try typing the key names instead. For example, if pressing the **Ctrl** key and then **R** does not work, try typing the text `Ctrl+R`.

MATLAB displays the list of matching actions for which you can customize or define a keyboard shortcut. If you specify the name of a tool, MATLAB displays actions associated with the tool or menu, as well as any action names that contain the specified text.

2 Select the name of the action for which you want to define or modify a keyboard shortcut.

3 Click the Add button [+].

An editable text box opens in the **Shortcut** column.

Shortcuts for **Clear Command History**

	Shortcut	Tools with shortcut	
🔹▾		All Tools	▾

[+] [−]
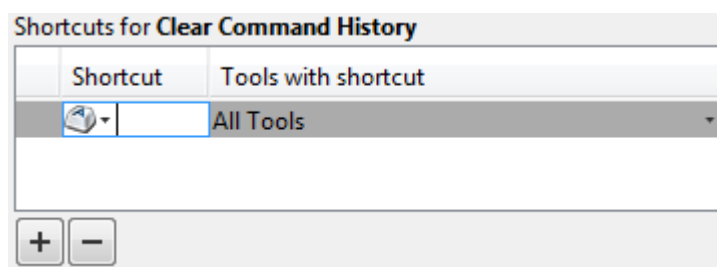
4   Specify the keystrokes you want to use for the keyboard shortcut.

A *keystroke* can be a single key or the combination of a modifier (**Alt**, **Shift**, or **Ctrl**) and another key. Specify a keystroke by performing it, not by typing the names of the keys character by character. For example, press the **Ctrl** key and the **Y** key. Do not type **C-t-r-l-+-Y**.

Alternatively, you can choose a shortcut from the drop-down menu.

To specify multiple keystrokes for one action, or to limit the number of keystrokes for an action, click the down arrow next to the key icon 🔑 in the **Shortcuts** box. Then, select either `Limit to 1 keystroke`, `Limit to 2 keystrokes`, or `Limit to 3 keystrokes`. For example, to specify **Ctrl+Y**, **Shift+Z**, **F9** as the three keystrokes for an action, select `Limit to 3 keystrokes` and then type the keystrokes.

5   In the **Tools with shortcut** column, click the down arrow and select which tools you want to assign the keyboard shortcut to.
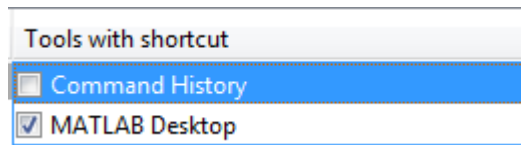
6   Evaluate and resolve any conflicts, indicated by the error icon 🔴 and informational icon 🔵. The error icon 🔴 indicates that two different actions within the same tool have the same shortcut. The informational icon 🔵 indicates that two different actions in two different tools have the same shortcut.

There is no requirement to resolve keyboard shortcut conflicts. However, if the same shortcut specifies two different actions, the shortcuts can be confusing to use. Resolving all conflicts indicated by the error icon 🔴 is recommended, although in some situations, resolving a conflict might be unnecessary. For example, resolving the conflict might be unnecessary if the conflict is temporary or the two actions are associated with different modes of the same tool. Resolving conflicts indicated by the informational icon 🔵 is recommended if you use both tools frequently and you perform both actions frequently.

To resolve a conflict, change or delete shortcuts such that there is a one-to-one correspondence between shortcuts and frequently used actions.

7   Click **OK** or **Apply**.

New or modified keyboard shortcuts become available immediately. Changed shortcuts that correspond to menu items immediately appear in the menu.

To delete a keyboard shortcut for a selected action, select the keyboard shortcut that you want to delete and click the Delete button 🔲.

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, click the Actions button ⚙️ ▾ and select **Undo Modifications to *<Set Name>* Default Set (modified)**, where *<Set Name>* is the name of the modified default set.

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

Restrictions When Customizing Keyboard Shortcuts

There are some actions and tools for which you cannot change the keyboard shortcuts. Actions for which the keyboard shortcut cannot be customized do not appear when you search for them in the **Keyboard Shortcuts Preferences** page in the Preferences Window. Examples of these actions include canceling the current action (**Esc**), interrupting MATLAB execution (**Ctrl+C**), and some of the navigational actions described in "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23.

In addition, you cannot change the keyboard shortcuts associated with these 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.

Manage Sets of Keyboard Shortcuts

Select Set of Keyboard Shortcuts

By default, MATLAB uses the keyboard shortcut settings for your current platform. To select a different set of keyboard shortcuts, open the **Keyboard Shortcuts Preferences** page in the Preferences Window and, in **Active settings**, select from the available options. To use a keyboard shortcut settings file that is on your system but not in the **Active settings** list, select `Browse...` to find and select the desired file.

This table lists the keyboard shortcut settings files installed with MATLAB.

Operating System	Keyboard Shortcut Settings Files Installed with MATLAB
Windows	• `Windows Default Set` (Default) • `Emacs Default Set`
Linux	• `Emacs Default Set` (Default) • `Windows Default Set`
macOS	• `Macintosh Default Set` (Default)

You also can download keyboard shortcut settings files from File Exchange. For example, to restore the MATLAB default keyboard shortcuts that were in place for MATLAB Version 7.9 (R2009a) and earlier releases, go to File Exchange and search for `MATLAB Desktop R2009a Default Keyboard Shortcut sets`. Download and extract the set and then select it as the active settings file. A valid keyboard shortcut settings file appears with a keyboard key icon .

Compare Sets of Keyboard Shortcuts

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

1 Open the **Keyboard Shortcuts Preferences** page in the Preferences Window and click the Actions button ⚙ ▾.

2 From the drop-down menu, choose the set of keyboard shortcuts to which you want to compare the current set.

3 The Comparison Tool opens and displays the two keyboard shortcut sets side-by-side. For more information about how to read the results, see "Compare Text Files" on page 6-8.

Save Set of Keyboard Shortcuts to File

Saving a set of keyboard shortcuts to a settings file is useful if you want to:

- Save the changes you make to a default keyboard shortcut set to a new set.
- Use a set of customized keyboard shortcuts on another system running MATLAB.
- Overwrite a previously saved set of keyboard shortcuts.
- Share a set of keyboard shortcuts with others, for example, on File Exchange.

To save a keyboard shortcut settings file, open the **Keyboard Shortcuts Preferences** page in the Preferences Window. Then, click the Actions button ⚙ ▾ and select `Save As`. MATLAB saves the file as an `.xml` file in the folder that you specify.

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)`.

Delete Set of Keyboard Shortcuts

To delete a previously saved set of keyboard shortcuts, open the **Keyboard Shortcuts Preferences** page in the Preferences Window and in **Active settings**, select the set of keyboard shortcuts that you want to delete. Click the Actions button ⚙ ▾ and select `Delete` *`filename`*, where *`filename`* is the name of a keyboard shortcut set you want to delete. You cannot delete default keyboard shortcut sets, such as `Windows Default Set`.

See Also

More About
- "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23
- "Rerun Favorite Commands" on page 3-13
- "Customize MATLAB Toolbars" on page 2-10

Set Print Options for Command Window and Editor

In this section...
"Specify Layout Options" on page 2-38
"Add Header" on page 2-38
"Change Fonts" on page 2-40

You can customize how pages look when printing from the Command Window and Editor.

To specify page setup options, open the Page Setup dialog box using one of these methods:

- Command Window — Right-click in the Command Window and select **Page Setup**.
- Editor — Go to the **Editor** tab, and in the **File** section, select **Print > Page Setup**.

The Live Editor has a different way of controlling how pages look when printing. To change the page setup options when exporting in the Live Editor, use settings. For more information, see the `matlab.editor.export` settings described in matlab.editor.

Specify Layout Options

To specify layout options when printing, in the Page Setup dialog box, select the **Layout** tab. (On macOS platforms, select **MATLAB** in the **Settings** menu to see the **Layout** tab.) Then, select from the available options as described in the table below. A preview pane shows a sample document with the selected layout options.

Layout Option	Description
Print header	Include a header at the top of each printed page.
Print line numbers	Print line numbers.
Wrap lines	Wrap any lines that are longer than the printed page width. Otherwise, the line is cut off at page boundaries.
Syntax highlighting	Specify how MATLAB elements appear when printed. Options include: - **Black and white text** — MATLAB elements print in black and white with no highlighting. - **Colored Text** — MATLAB elements print using the syntax highlighting colors displayed in the Command Window or Editor. - **Styled Text** — Keywords print in bold, comments print in italics, and all other text prints as plain text. Output is not styled.

Add Header

When printing from the Command Window or Editor, you can include a header on each printed page. The header includes information such as the current date and time, the name of the file you are printing, and the page number. To include a header, in the Page Setup dialog box, select the **Layout** tab and then select **Print header**.

To change the format and layout of the header, select the **Header** tab and then select from the available options, as described in the table below. (On macOS platforms, select **MATLAB** in the **Settings** menu to see the **Header** tab.) The preview area shows a sample of the header with the selected options.

Header Option	Description
Page number	Format for the page number. Options include **# of n**, **Page #**, and **# / n**
Border	Border style for the header. Options include **Single line**, **Double line**, **Box**, **Shaded box**, and **None**.
Layout	Layout style for the header. Options include: • **Standard one line** 1/4/21 11:39 AM C:\...\filename.m 1 of n • **Standard two lines** C:\...\filename.m 1 of n January 4, 2021 11:40:02 AM • **Simple one line** C:\...\filename.m 1 of n • **Simple two lines** filename.m 1 of n C:\...\ January 4, 2021 (This option is not available for Command Window printing.)

For example, in the Editor, print the example file `fourier_demo2.m` with a standard two line header and a double line border.

```
fourier_demo2.m                                            1 of 1
C:\Users\MATLAB                                   November 9, 2020

%% Square Waves from Sine Waves
% The Fourier series expansion for a square-wave is
% made up of a sum of odd harmonics, as shown here
% using MATLAB(R).

%% Add an Odd Harmonic and Plot It
t = 0:.1:pi*4;
y = sin(t);
plot(t,y);
```

Change Fonts

You can specify the font to use when printing. By default, MATLAB uses the Editor or Command Window font when printing.

To specify a different font to use when printing, follow these steps:

1 In the Page Setup dialog box, select the **Fonts** tab. (On macOS platforms, select **MATLAB** in the **Settings** menu to see the **Fonts** tab.)

2 In the **Choose font** field, select **Header** to change the font for the header text, or **Body** to change the font for all of text in Command Window or Editor except for header text.

3 Select **Use custom font** and then specify the font name, style, and size. The **Sample** area shows a preview of the selected font.

See Also
`print` | matlab.editor

Related Examples
- "Print Figure from File Menu"
- "Change Desktop Fonts" on page 2-2
- "Change Desktop Colors" on page 2-6

Web Browsers and MATLAB

In this section...

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:
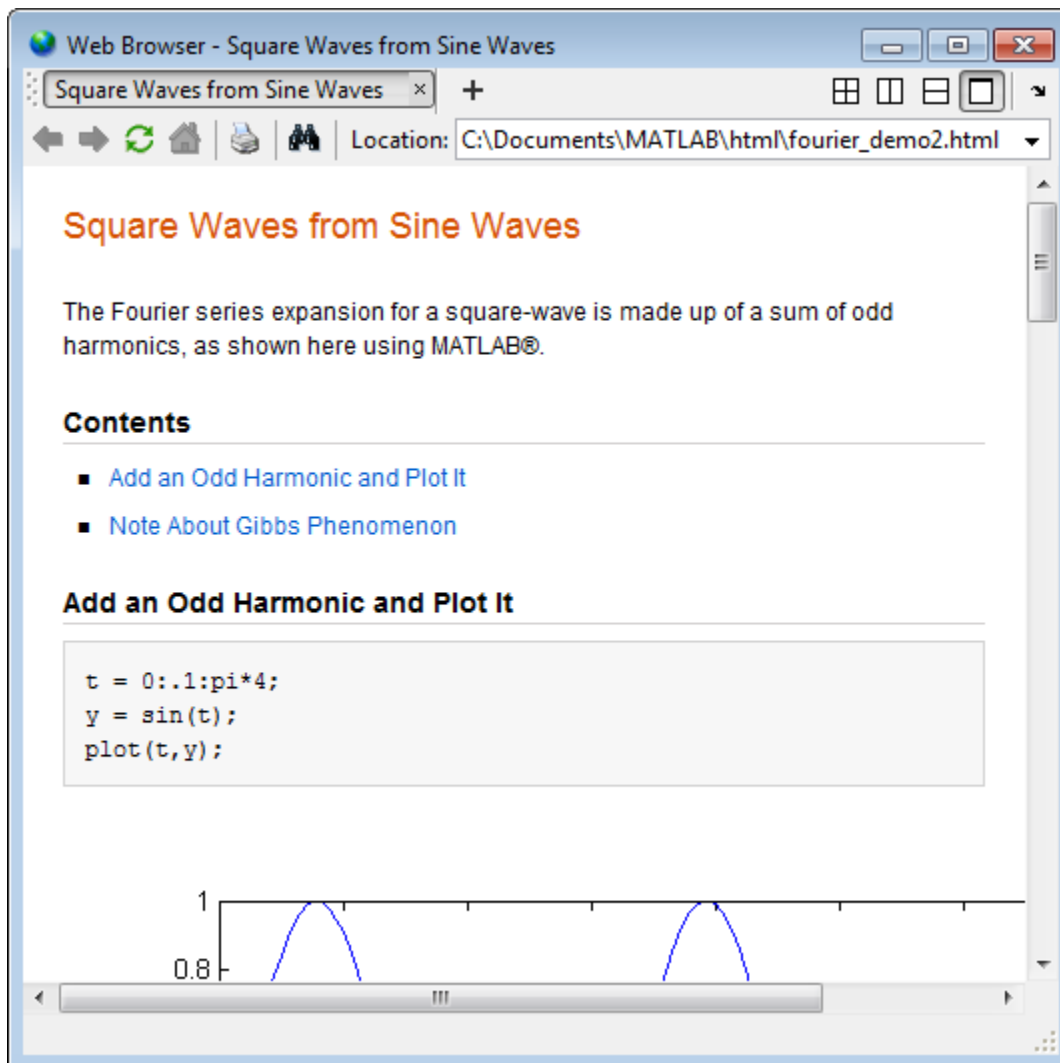
MATLAB Web and Help Browsers

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 on page 2-44.

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.

Note **Security Considerations:** Avoid using the MATLAB Web browser for general web browsing. Use your system browser instead.

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.
- If a proxy with basic authentication is specified, MATLAB only supports HTTP connections. HTTPS connections are not supported.
- 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.

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

   MATLAB attempts to connect to `https://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.

6   Restart MATLAB to enable the changes.

Specify the System Browser for Linux Platforms

On Linux platforms, you can specify the system browser to use. Specifying the system browser is not supported on Windows and Mac 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 If you specify Firefox as the system browser using the system command `firefox`, specify a display option to ensure that the browser opens correctly.

See Also
web

More About
• "Web Preferences" on page 2-55

Manage Your Licenses

You can manage many licensing-related tasks, such as activating licenses, deactivating licenses, or updating licenses, through MATLAB or your MathWorks Account on www.mathworks.com.

Note Some options for license actions 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 2-43 for more information.

Update Current Licenses

The simplest way to update your license file is from within MATLAB. You should perform this task within your MathWorks Account only if you cannot access MathWorks from your computer. For additional details on updating licenses, see "Update Your License", or, if you are a license administrator, see "Update Network License".

User Interface	Description
MATLAB	On the **Home** tab, in the **Resources** section, click **Help > Licensing > Update Current Licenses**. MATLAB displays a list of all your MathWorks licenses on this computer, with their status. When you select a license and click **Update Selected License**, MATLAB contacts MathWorks to retrieve the most current version of the License File for the license. The update process overwrites the current License File on your system. You need to restart MATLAB.
MathWorks Account	1 Sign in to mathworks.com. If you are already signed in, select **My Account** from the drop-down menu under your account icon. 2 Under **My Software**, click the license you want to update. If you have more licenses than fit on the screen, click **View additional Licenses or Trials** and then click the license you want. Clicking a license takes you to the License Center. 3 In the License Center, click the Install and Activate tab. 4 Click **Update License File**. 5 Click one of the icons under **Get License File**. You can download the license file or have it emailed to you. 6 Follow all prompts. MathWorks retrieves the most current version of the License File for the license. You need to restart MATLAB.

Activate Software

The simplest way to activate your software is from within MATLAB. You should perform this task within your MathWorks Account only if your computer cannot communicate with MathWorks. For additional details on activation, see "Activate MATLAB Installation Manually".

User Interface	Description
MATLAB	On the **Home** tab, in the **Resources** section, click **Help > Licensing > Activate**. MATLAB 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**.
MathWorks Account	**1** Sign in to mathworks.com. If you are already signed in, select **My Account** from the drop-down menu under your account icon. **2** Under **My Software**, click the license you want to activate. If you have more licenses than fit on the screen, click **View additional Licenses or Trials**, and then click the license you want. **3** Click the Install and Activate tab. **4** Click **Activate to Retrieve License File** or **Activate a Computer**. **5** Follow all prompts.

Deactivate Software

Although you can deactivate the software from within MATLAB, if you cannot access the computer to deactivate, you can still deactivate the software from your MathWorks Account.

For more details on deactivating MathWorks software, see "Deactivate Your Installation".

User Interface	Description
MATLAB	On the **Home** tab, in the **Resources** section, click **Help > Licensing > Activate > Deactivate**. MATLAB displays a list of all your MathWorks licenses on this computer, with their status. When you select a license and click **Deactivate**, MATLAB deactivates all releases on this computer associated with the license, and updates the licensing information at the MathWorks website. You will not be able to use MathWorks software with that license on this computer. If you are not connected to the Internet, MATLAB deactivates the licenses on your computer but cannot update the corresponding license information stored at the MathWorks website. 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 website. There you can sign in to your MathWorks Account and enter the deactivation string.

User Interface	Description
MathWorks Account	**1** Sign in to mathworks.com. If you are already signed in, select **My Account** from the drop-down menu under your account icon. **2** Under **My Software**, click the license you want to deactivate. If you have more licenses than fit on the screen, click **View additional Licenses or Trials**, and then click the license you want. **3** Click the Install and Activate tab. **4** Find the activated computer in the list, then click the deactivate icon ⊗ on the right-hand side of the display. **5** Follow all prompts. MathWorks deactivates all releases on the specified computer, and updates the licensing information in License Center.

Link a License to Your Account

User Interface	Description
MATLAB	You cannot perform this task from within MATLAB.
MathWorks Account	**1** Sign in to mathworks.com. If you are already signed in, select **My Account** from the drop-down menu under your account icon. **2** Under **My Software**, click **Link an additional license**. **3** Follow all prompts. If you are linking a network license, you may need to get the license number from your organization's license administrator.

Get a Trial

Start a free 30-day trial of MathWorks software.

Although you can get a trial from within MATLAB or from your MathWorks Account, the simplest way is to go directly to Free MATLAB Trial on mathworks.com.

User Interface	Description
MATLAB	To get a trial, go to **Add-Ons > Get Add-Ons**. Click on a toolbox, and then click **Get Trial**. Follow all prompts.
MathWorks Account	**1** Sign in to mathworks.com. If you are already signed in, select **My Account** from the drop-down menu under your account icon. **2** Under **My Software**, click **Get a trial**. Follow all prompts.

See Also
license

Related Examples

- "Install Products"
- "Manage Products"
- "Administer Network Licenses"

Update an Existing Installation

You can download and install a more recent version of the release of MATLAB that you currently have installed. For example, if you have R2020a installed, you might be able to download a more recent version of R2020a. This version is called a MATLAB update. When you install a MATLAB update, you get not only the most recent update for your version but also all previous updates for your version as well.

Update from Notifications

Since R2019a

If a more recent version is available, a notification displays in MATLAB when you start the program.
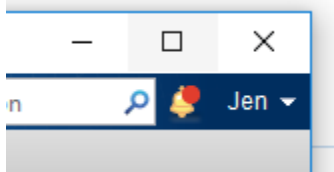
To install the MATLAB update, click the notification icon and then select **Install Update**. If you click the notification but decide not to install the update, the notification is cleared; however, you can still install the update at a later time.

Check for Updates

Since R2016b

You also can check for updates manually. You must have an active internet connection to check for and update your installation.

On the **Home** tab, click **Help > Check for Updates**. The Add-On Manager opens with the **Updates** tab selected. If there is an update available for your version of MATLAB, click the **Update** button to the right of the update name to install it. If there is no update available for your version or if you have already installed the latest update, MATLAB displays that information.

In R2019a: On the **Home** tab, click **Help > Check for Updates > Products**. If there is an update available, click **Install Update**.

Before R2019a: On the **Home** tab, click **Add-Ons > Check for Updates > Products**.

See Also

More About
* "Get New MATLAB Release"

macOS Platform Conventions

In this section...
"Mouse Instructions and macOS Platforms" on page 2-50
"Navigating Within the MATLAB Root Folder on macOS Platforms" on page 2-50
"MATLAB Dock Menu" on page 2-50

Mouse Instructions and macOS Platforms

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

On macOS 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 a folder within the MATLAB application package, for example, `/Applications/MATLAB_R2019a.app/bin`.

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'))
```

MATLAB Dock Menu

Starting in MATLAB R2012a, the MATLAB Dock menu lists open windows and documents associated with the running MATLAB. Choosing an open document from the Dock menu brings the selected document's window to the foreground. In addition, the MATLAB Dock menu includes the item **Open Additional Instance of MATLAB**. This option starts another instance of MATLAB. You can select **Options** > **Keep in Dock**, but only for the first MATLAB instance. Use **Open Additional Instance**

of MATLAB to start additional instances rather than trying to keep multiple MATLAB icons in the Dock.

Where MATLAB Stores Preferences

MATLAB and other MathWorks products store their preferences in the preferences folder. To see the full path for the preferences folder type `prefdir` in the MATLAB Command Window. The name of the preferences folder matches the name of the release. For instance, for release R2016b, the name of the preferences folder is `R2016b`.

On macOS and iOS, the preferences folder might be inside another folder that is hidden. If so, to access the hidden folder:

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

You must have write access to the preferences folder. Otherwise, MATLAB generates an error in the Command Window when you try to change preferences. MATLAB also can generate an error if the preferences folder is hidden.

Temporary Preferences Folder

In some situations, if MATLAB is unable to create or use the preferences folder when starting up, it creates a temporary preferences folder. If a preferences folder exists for any of the three releases of MATLAB that immediately precede the one you are starting up, MATLAB copies the preference files from that release to the temporary preferences folder. Otherwise, MATLAB creates the default preference files for the current release. MATLAB continues to use the temporary preferences folder until the issue is resolved.

If you make changes to preferences while MATLAB is using the temporary preferences folder, the preferences persist across sessions. But, once the issue is resolved that was preventing MATLAB from accessing the preferences folder, any changes made to the preferences while using the temporary preferences folder are lost.

Each version of MATLAB can have a temporary preferences folder. MATLAB does not migrate preferences from the temporary preferences folder across releases.

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 it starts up, 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.

See Also
`prefdir` | **Preferences Window**

Related Examples
*   "Import Preferences From Other Releases" on page 2-53

Import Preferences From Other Releases

When you start a newly installed version of MATLAB for the first time, it attempts to import your preferences from a previous release installed on your system. MATLAB looks for preferences folders according to these guidelines.

- MATLAB looks for a preferences folder for release R2019b or newer. If MATLAB finds one or more preferences folders matching that criteria, it migrates the files from the most recent version.

  For example, if you start R2021b and preferences folders exist for both the R2020b and R2019b releases, then MATLAB migrates the files from the R2020b preferences folder to the R2021b preferences folder.

  Note For releases before R2021b, MATLAB looks back three releases. For example, if you start R2019a, MATLAB looks for preferences from R2018b, R2018a, and R2017b.

- If a preferences folder does not exist from a valid previous release, then MATLAB creates the default preference files for the release starting up.

  For example, if you start R2021a, MATLAB looks for preferences for versions R2020b, R2020a, and R2019b. If none of those is installed, MATLAB creates the default files for the R2021a release. This is true even if a preferences folder exists for the R2019a release or earlier.

- If you install a release older than the current version, MATLAB does not import preferences from releases that are newer than that release.

To see the full path for the preferences folder, type `prefdir` in the MATLAB Command Window. The name of the preferences folder matches the name of the release you are running. For example, for MATLAB R2021a, the name of the preferences folder is `R2021a`.

```
prefdir
```

```
ans =
    'C:\Users\username\AppData\Roaming\MathWorks\MATLAB\R2021a'
```

On macOS and iOS, the preferences folder might be inside another folder that is hidden. If so, to access the hidden folder:

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

Use Default Preferences

Instead of importing preferences from a previous release, you can use the default preferences for the new release. To use the default preferences, delete all of the files in the preferences folder for the corresponding release. Use the `prefdir` command to determine the full path of the preferences folder.

For example, to use the default preferences for R2021a, start MATLAB, and in the Command Window, type `prefdir` to determine the location of the preferences folder.

```
prefdir
```

```
ans =
    'C:\Users\username\AppData\Roaming\MathWorks\MATLAB\R2021a'
```

Close MATLAB, delete all of the files in the folder `C:\Users\username\AppData\Roaming\MathWorks\MATLAB\R2021a`, and then start MATLAB again. MATLAB starts with all preferences set to their default values. One file to consider keeping is `history.m`. For more information, see "Set Command History Preferences" on page 3-24.

Alternatively, if you have not started the new release yet, create an empty preferences folder for it. This will prevent MATLAB from importing preferences when you do open the new release, and force it to use the default preferences instead. For example, to use the default preferences for MATLAB R2021a, if your MATLAB preferences are stored in the `C:\Users\username\AppData\Roaming\MathWorks\MATLAB` folder, create the folder `C:\Users\username\AppData\Roaming\MathWorks\MATLAB\R2021a`.

See Also
`prefdir` | **Preferences Window**

Related Examples
*   "Where MATLAB Stores Preferences" on page 2-52

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 this table.

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.
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.
System Web browser	Select **Use system web browser when opening links to external sites (recommended)** to open external sites in your system browser. Clear the option to use the MATLAB Web browser instead. Using the system browser is recommended when opening external sites. **Note Security Considerations:** Avoid using the MATLAB Web browser for general web browsing. Use your system browser instead.

Preference	Usage
	UNIX platforms only, excluding Macintosh: Specify the system **Command** to use to open the system browser. For example, `opera`, opens the Opera Web browser. Specify the **Options** to use for the system browser. For example, `geometry 1064x860` specifies the size of the window for Opera. If you specify Firefox as the system browser using the system command `firefox`, specify a display option to ensure that the browser opens correctly.

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:

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

```
hello
```

Find Functions to Use

This example shows how to find the name and description of a MathWorks function from the Command Window using the Function browser.

1 In the Command Window, click the f_x browse for functions button to the left of the prompt. The Function browser opens.

```
┌─────────────────────────────────────┐
│              ........                │
│  Search for functions          🔍   │
│  Categories                    ▲    │
│  📁 MATLAB                      ▤    │
│      📁 Language Fundamentals        │
│      📁 Mathematics                  │
│      📁 Graphics                     │
│      📁 Programming Scripts and Functions │
│      📁 Data and File Management     │
│      📁 GUI Building                 │
│      📁 Advanced Software Development │
│      📁 Desktop Environment          │
│  📁 Simulink                         │
│  📁 Aerospace Toolbox          ▼    │
│  📁 Bioinformatics Toolbox           │
│        All installed products        │
└─────────────────────────────────────┘
```
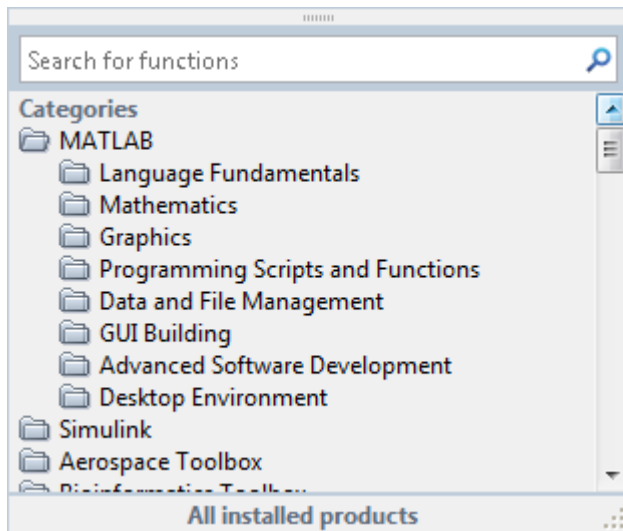
Tip The Function browser closes when you move the pointer outside of it. To keep the browser open, drag it by the top edge to a different location.

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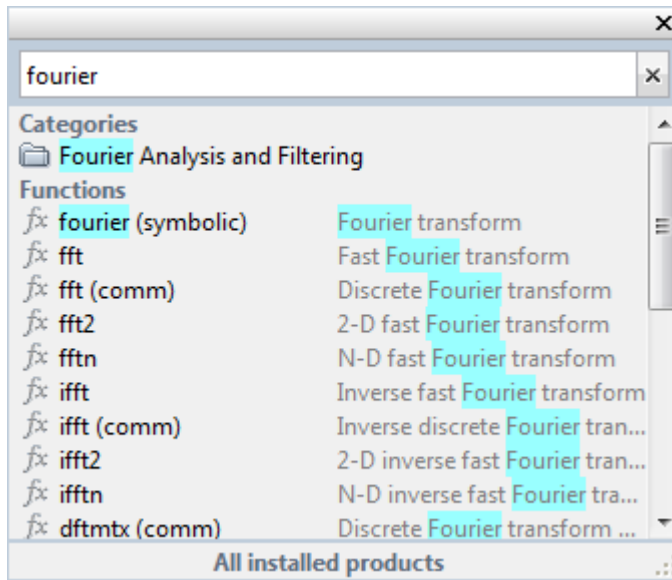
In the search results, a parenthetical term after a function name indicates either that the function is in a product folder other than MATLAB, or that there are multiple functions with the same name. For example, `fft (comm)` corresponds to the `fft` function in the Communications Toolbox™ folder.

4   Select a function that you would like to use or learn more about, as follows.

   •   Insert the function name into the current window by double-clicking the name. Alternatively, drag and drop the function name into any tool or application.

   •   View syntax information for the function by single-clicking its name. A brief description for each of the syntax options displays in a yellow pop-up window.
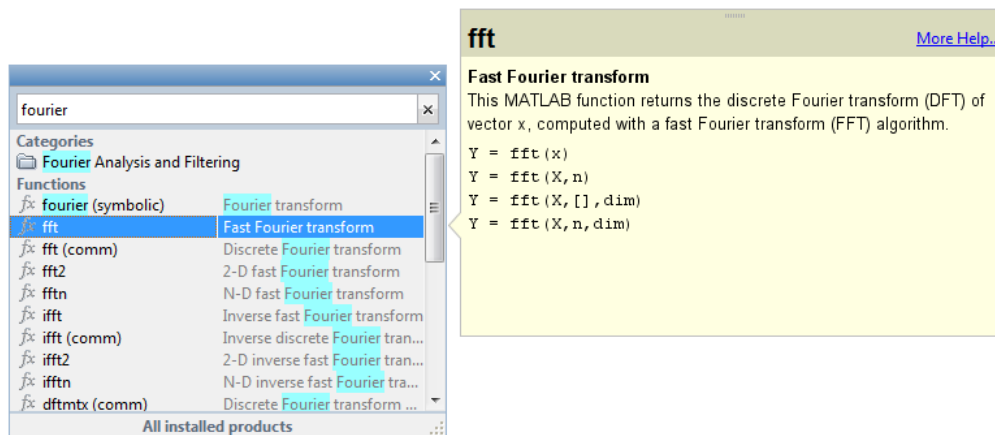
Tip The pop-up window automatically closes when you move your pointer to a new item in the results list. To keep the pop-up window open, drag it by the top edge to a different location.

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

Format Output

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

Format Line Spacing in Output

By default, MATLAB displays blanks lines in Command Window output.

You can select one of two line spacing 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 change the line spacing option, do one of the following:

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

  ```
  format loose
  format compact
  ```

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

Format Floating-Point Numbers

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

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

Numeric Display Format	Example Output
`short` (default)	`x = 1.3333 0.0000`
`short e`	`x = 1.3333e+00 1.2345e-06`
`long`	`x = 1.333333333333333 0.000001234500000`
`+`	`x = ++`

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

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

- On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Command Window**, and then choose a **Numeric format** option.

- Use the `format` function, for example:

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

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

Wrap Lines of Code to Fit Window Width

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

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

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

Suppress Output

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

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

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

View Output by Page

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

1. In the Command Window, type `more on` to enable paged output.
2. Type the command that generates large output.
3. View the output:

   - Advance to the next line by pressing **Enter**.

- Advance to the next page by pressing **Space Bar**.
- Stop displaying the output by pressing **q**.

To disable paged output, type `more off`.

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

Clear the Command Window

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

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

See Also
`clc` | `format` | `home` | `more`

Stop Execution

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

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

Ctrl+C does not always stop execution for files that run a long time, or that call built-ins or MEX-files that run a long time. If you experience this problem, include a `drawnow`, `pause`, or `getframe` function in your file, for example, within a large loop. Also, **Ctrl+C** might be less responsive if you start MATLAB with the `-nodesktop` option.

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

To programmatically stop execution of a function or script before reaching the end, use the `return` function. MATLAB returns control to the Command Window or the invoking function.

See Also
`drawnow` | `getframe` | `pause` | `return`

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 3-12

Find Text in the Command Window

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

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

Search Using Find Dialog Box

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

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

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

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

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

Search Using Keyboard Shortcuts

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

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

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

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

2 Begin typing your search term.

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

3 Perform incremental search actions using these keyboard shortcuts:

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

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

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

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

Find Text in the Command History Window

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

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

2 Begin typing your search term.

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

3 Select from the different search options using the buttons to the right of the search field. Options include Aa (match case), ☰ (match anywhere within command), and ☰ (match at beginning of command).

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

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

Rerun Favorite Commands

Create and Run Favorite Commands

MATLAB favorite commands (previously called command shortcuts) are an easy way to run a group of MATLAB language statements that you use regularly. For example, you can use a favorite command to set up your environment when you start working, or to set the same properties for figures you create.

To create a favorite command:

1   On the **Home** tab, in the **Code** section, click **Favorites** and then click **New Favorite**. The Favorite Command Editor dialog box opens.

2   In the **Label** field, enter a name for the favorite command. For this example, enter `Setup Workspace`.

3   In the **Code** field, type the statements you want the favorite command to run. You also can drag and drop statements from the Command Window, the Command History Window, or a file. MATLAB automatically removes any command prompts (>>) from the **Code** field when you save the favorite command.

    For example, enter these statements:

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

4   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 favorite command appears in the default **Favorite Commands** category.

5   In the **Icon** field, select an icon.

6   To add the favorite command to the quick access toolbar, select both the **Add to quick access toolbar** and **Show label on quick access toolbar** options.

7   To run the statements in the **Code** section and ensure that they perform the desired actions, click **Test**.

8   When you are done configuring the favorite command, click **Save**.

To run a favorite command, on the **Home** tab, click **Favorites** and then click the icon for the desired favorite command. All the statements in the **Code** field of the Favorite Command Editor execute as if you ran those statements from the Command Window, although they do not appear in the Command History window.

To edit a favorite command, click the ✎ icon to the right of the favorite command. To delete a favorite command, click the 🗑 icon to the right of the favorite command. You also can right-click the favorite command and select **Edit Favorite** or **Delete Favorite**.

Organize Favorite Commands

You can organize your favorite commands by storing them in different categories.

To create a new category:

1 On the **Home** tab, in the **Code** section, click **Favorites** and then click **New Category**. The Favorite Category Editor dialog box opens.

2 In the **Label** field, enter a name for the category. For this example, enter `My Favorite Favorites`.

3 In the **Icon** field, select an icon.

4 To add the category to the quick access toolbar, select both the **Add to quick access toolbar** and **Show label on quick access toolbar** options.

5 Click **Save**.

To move a category up or down in the list of categories, or to move a favorite command within a category, drag the category or favorite command to the desired location. You also can use the ▲ and ▼ buttons to the right of the category.

To change whether a single category or favorite command appears in the quick access bar, click the ⏷ and ⏷ icons to the right of the category or favorite command. In MATLAB Online, right-click the category or command and select `Add to quick access toolbar`. To add all favorite commands to the quick access bar, on the **Home** tab, right-click **Favorites** and select `Add to quick access toolbar`.

To further configure which favorite commands and categories appear in the quick access bar, on the **Home** tab, in the **Code** section, click **Favorites** and then click ⚙ **Quick Access**. Configuring the quick access bar is not supported in MATLAB Online.

See Also

More About

- "Customize Keyboard Shortcuts" on page 2-33
- "Set Command History Preferences" on page 3-24

Set Command Window Preferences

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

General Preferences for the Command Window

You can specify what is displayed in the Command Window and how it is displayed and the default format of `Datetime` objects.

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

Some Command Window preferences are not available in MATLAB Online on page 10-2.

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 3-7.
	Select a **Line spacing** option to specify whether blank lines appear in Command Window output. To suppress blank lines, select `compact`. To display blank lines, select `loose`.
Datetime format	Select a **Locale** option to specify the default input locale of the `Datetime` object. You also can enter a custom locale. For more information, including a list of common values, see `datetime`.
	Select a **Default date and time format** option to specify the default format of the `Datetime` object. You also can enter a custom format. For more information, see the `Format` property for datetime arrays.
	Select a **Default date-only format** option to specify the default date format of the `Datetime` object. You also can enter a custom format. For more information, see the `Format` property for datetime arrays.
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. If you also select **Wrap lines**, and the width of the Command Window is fewer than 80 characters, each row of 80 characters of matrix output wraps to fit within the width of the Command Window.
	Select **Show getting started message bar** to display the Command Window message bar that provides links to introductory information. New to MATLAB? See resources for Getting Started. ✕

Preference	Usage
	Select **Show function browser button** to display the Function Browser button f_x to the left of the prompt in the Command Window. You can use the Function Browser to search for MATLAB functions.
	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 that 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 see in the Command Window.
Tab key	**Tab size** specifies the number of spaces assigned to the tab key. This setting does not apply if you have enabled tab completion for the Command Window or Editor. To enable or disable tab completion, on the **Home** tab, select **Preferences > Keyboard**.

Command Window Automatic Completions Preferences

You can change how MATLAB suggests and completes names in the Command Window.

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

Preference	Usage
Suggestions and completions	Select **Enable tab completion** for MATLAB to suggest and complete names when you type the first few characters of the name and press the **Tab** key in the Command Window. For more information, see "Code Suggestions and Completions" on page 3-20.
	Select **Tab key narrows completions** for MATLAB to continue to reduce the list of possible names for completions as you type each additional character and press the **Tab** key.

Preference	Usage
	Select **Enable function hints** for MATLAB to display function hints in the Command Window 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: For more information, see "Function Syntax Hints" on page 3-22.

In MATLAB Online, for MATLAB to suggest and complete names as you type in the Command Window, select **Show automatically**. For MATLAB to suggest and complete names only after you press the **Tab** key, select **Show on tab**.

See Also

More About
- "Check Syntax as You Type" on page 3-19
- "Set Keyboard Preferences" on page 3-18
- "Change Desktop Fonts" on page 2-2

Set Keyboard Preferences

You can set delimiter matching options in the Command Window, Editor, and Live Editor.

On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Keyboard**, and then adjust the options as described in this table.

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. Select **Show matches and mismatches when typing** for MATLAB to alert you to matched and mismatched delimiters as you type. Select **Show matches and mismatches when using arrow keys** for MATLAB to alert you to matched and mismatched delimiters when you move the cursor over a delimiter by using an arrow key. For details, see "Delimiter Matching" on page 3-19.

See Also

Related Examples

Check Syntax as You Type

In this section...

Syntax Highlighting

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

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

```
% check to see if A is greater than B
if A > B
'greater'
elseif A < B
'less'
end
```

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

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

To change syntax highlighting preferences, on the **Home** tab, in the **Environment** section, select ⚙ **Preferences**. Then, select **MATLAB > Editor/Debugger > Language** and from the **Language** drop-down list, choose a language. To change syntax highlighting preferences in MATLAB Online, select **Editor/Debugger > MATLAB Language** or **Editor/Debugger > Other 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.

In the Editor and Live Editor, MATLAB indicates matching delimiters by briefly underlining both delimiters in the pair. In the Command Window, matching delimiters are indicated by highlighting instead of underlining. In MATLAB Online, the Command Window behavior matches the Editor and Live Editor behavior.

MATLAB indicates mismatching delimiters in the Editor, Live Editor, and Command Window by briefly crossing out the mismatched delimiter.

If a matching delimiter exists, but it is not visible on the screen, a window opens and displays the line containing the matching delimiter. Click in the window to go to that line.

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

You can change if and when MATLAB alerts you to matched and mismatched delimiters. On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Keyboard** and in the **Delimiter matching** section, select from the available options.

Code Suggestions and Completions

When you write code, MATLAB suggests and completes the names of functions, models, MATLAB objects, files, folders, variables, structures, graphics properties, parameters, and options. Use these automatic suggestions and completions to write commands faster and avoid typographical errors.

The Editor and Live Editor display possible matches as you type. To insert a match in your script, use the arrow keys to select the name that you want, and then press the **Tab** key. Press the **Esc** key to clear the list of possible completions without selecting anything.

The Command Window displays possible matches only after you press the **Tab** key. To complete names in the Command Window, type the first few characters of the name that you want to complete, and then press the **Tab** key. Use the arrow keys to select the name that you want, and then press the **Tab** key again. In MATLAB Online, the Command Window behavior matches the Editor and Live Editor behavior, displaying possible matches as you type.
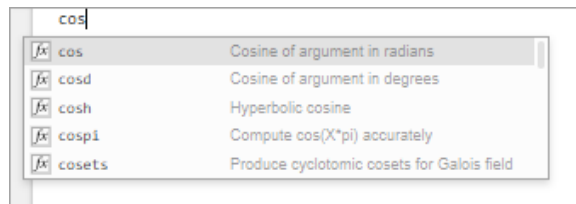
Automatic suggestions and completions are especially useful when completing the names and values of graphics properties. For example, `plot(x,y,'LineStyle','-')`.
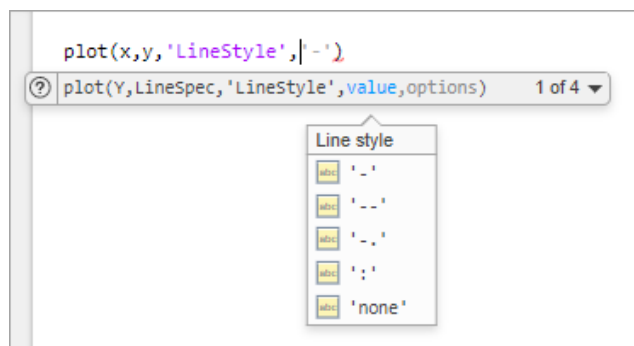
You can also use the suggestions to complete parameter names and options for certain functions. For example, `format('shortEng')`.
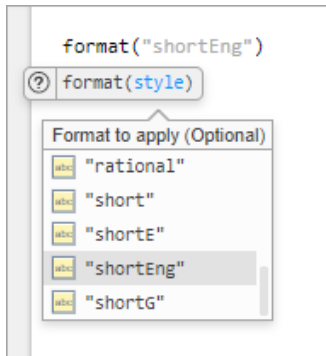
When entering a statement that includes a function, MATLAB might suggest multiple syntaxes for the function. To view the additional syntaxes, click the arrow to the right of the syntax. Alternatively, press the **Ctrl + Down Arrow** keyboard shortcut to show and hide the additional syntaxes. On macOS systems, use the **Command** key instead of **Ctrl**.

Similarly, if MATLAB suggests multiple arguments, click the arrow to the right of the argument description to view the additional arguments. Alternatively, you can use the **Ctrl + Alt + Right Arrow** and **Ctrl + Alt + Left Arrow** keyboard shortcuts.
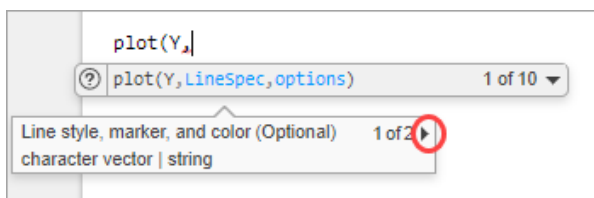
MATLAB displays suggestions based on its current state, including:

- Variables that are defined in the current workspace.
- Files and folders that are on the search path or in the current folder.
- Variables, functions, class properties, and class methods that are defined in the active document and available at the current location of the cursor.

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

Tip To insert a tab within a statement when suggestions and completions are enabled, first, add a space. Then, press the **Tab** key. Otherwise, MATLAB attempts to complete a name. In the Live Editor,

if suggestions and completions are enabled but there are no available completions, MATLAB inserts a tab.

Disable Automatic Suggestions and Completions

In the Editor and Live Editor, for MATLAB to suggest and complete names only after you press the **Tab** key, on the **View** tab, in the **Display** section, click the ⊞ **Automatic Completions** button off.

Or, you can go to the **Home** tab, and in the **Environment** section, click ⚙ **Preferences**. Then, select **Editor/Debugger > Automatic Completions** and in the **Suggestions and completions** section, select **Show on tab**.

In the Command Window, to disable suggestions and completions, in the Preferences window, select **MATLAB > Command Window > Automatic Completions** and clear the **Enable tab completion** option.

In the Command Window in MATLAB Online, for MATLAB to suggest and complete names only after you press the **Tab** key, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Command Window > Automatic Completions** and in the **Suggestions and completions** section, select **Show on tab**.

Function Syntax Hints

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

Function syntax hints appear for MATLAB installed functions and for functions that you create. The syntax hints for MATLAB functions come from the documentation. The syntax hints for functions that you create come 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. For more information about adding help to the functions that you create, see "Add Help for Your Program".

To use function syntax hints, type a function name with an opening parenthesis. A window opens showing the basic syntax for the function. In the Editor and Live Editor (and Command Window in MATLAB Online), you can use the drop-down arrow ▾ to the right of the syntax to display all available syntaxes.
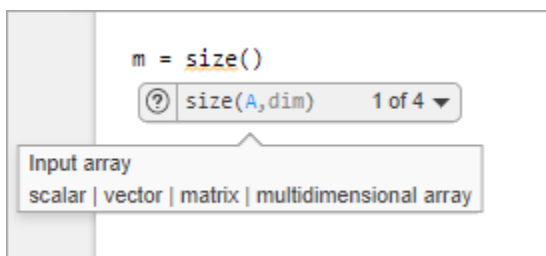
You can type an input for any argument that appears in blue. Enter your own input names, and not the argument names shown in the window. The displayed syntax options change based on the argument that you just entered.

In the Editor and Live Editor (and Command Window in MATLAB Online), MATLAB displays descriptions for each argument. Optional arguments are indicated by the **(optional)** keyword next to the argument description.
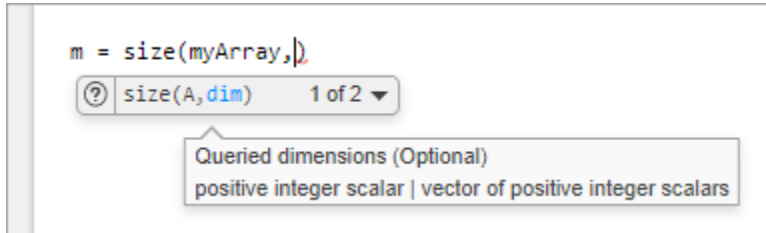
Some function names are overloaded. That is, there are different functions with the same name that support different types of input. Overloaded functions require that you pass an object as the first input. When you specify the object name, the syntax hints are updated to reflect the correct version of the function, as shown. The Live Editor might not display all overloads.
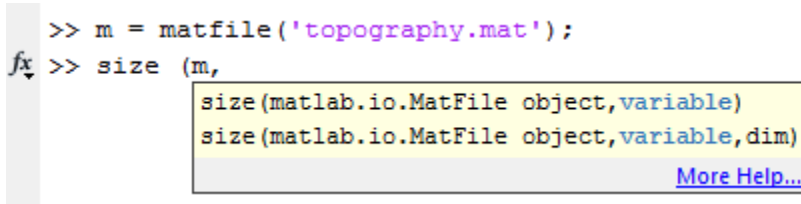
Function syntax hints are only suggestions. For detailed information about a function, see the function reference page in the documentation. To open the documentation for the function, click the ⓘ button to the left of the function syntax hint.

To disable function syntax hints in the Command Window, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Command Window > Automatic Completions** and in the **Suggestions and completions** section, clear **Enable function hints**. Disabling function syntax hints is not supported for the Editor, Live Editor, or for the Command Window in MATLAB Online.

Set Command History Preferences

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

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

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

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

Some Command History preferences are not available in MATLAB Online on page 10-2.

Option	Usage
Save	Select **Save exit/quit commands** to save `exit` and `quit` commands in the command history.
	Select **Save consecutive duplicate commands** to 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. • With this option cleared, the command history retains only one entry for `magic(5)` and displays a tally of consecutive executions to the left.
	Select **Don't save history file** to prevent saving the command history across sessions. This option is useful when multiple users share the same machine. For example, the option prevents each user from viewing statements others have run. Any entries predating the current session remain unless you first delete entries from the Command History window.
	Save last *n* commands specifies the number of commands to save.
Match	Select **Match anywhere** to retrieve statements that contain the search text in any location. Select **Match beginning** to retrieve statements that begin with the search text. This option is selected by default.
	Select **Match case** to retrieve statements that match the case of the search text.

Option	Usage
	Select **Filter matches** to display only statements that match the search text. Clear **Filter matches** to display all previously executed statements and highlight the statements that match the search text.
Show	Select **Show match toolbar** to display a search toolbar at the top of the Command History window. Search for previously executed statements using the search field and change **Match** preferences using the provided buttons.
	Select **Show match locations** to display yellow markers to the right of the scroll bar in the Command History window that indicate the location of matches throughout the command history.
	Select **Show execution time** to display an approximate execution time to the right of each statement. Times display for any statements that take longer than `0.1` seconds to execute.
Favorite Commands	Select **Don't show** to disable showing favorite commands in the Command History window. Select **Show default category** to only show the default **Favorite Commands** category in the Command History window. Select **Show all categories** to show all the favorite commands categories in the Command History window.

Change the Command History Date Format

MATLAB uses the short date format of your operating system 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.

See Also
Command History Window

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_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

Standalone Examples

A standalone example is a readable version of a MATLAB script or live 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.)

To access examples, click **Examples** at the top of a documentation page. The Help browser displays the examples for the current product category.

MATLAB includes examples that demonstrate various functionalities. For instance, to view examples demonstrating plotting in MATLAB, navigate to the **MATLAB > Graphics > 2-D and 3-D Plots** category and click **Examples** at the top of the page.

Each example combines comments, code, and output in a formatted document. If you are viewing the documentation in MATLAB, you can open the corresponding script or live script by clicking **Open Script** or **Open Live Script**.
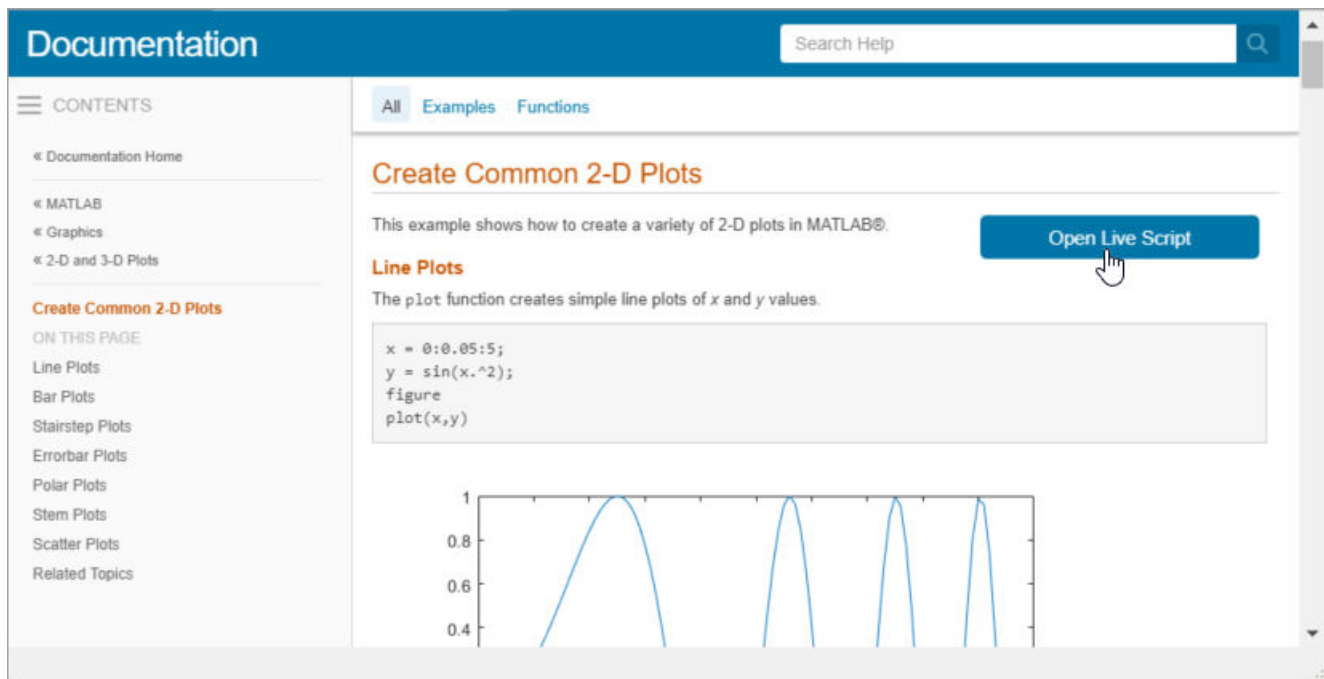
If you are viewing the documentation on the web, you can open live script examples using the Live Editor in a web browser. To open the example, click the **Try this Example** button to the right of the example.

Once it is open, run the example by clicking **Run** ▷. To run the example 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** 📊.

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. Evaluating a selection from the Help browser is not available in MATLAB Online.
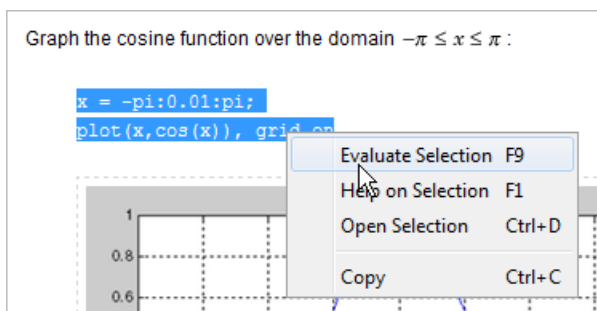
See Also
demo | echodemo

Related Examples

- "Create and Run Sections in Code"

External Websites

- File Exchange

Search Syntax and Tips

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

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

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

• Search for the word instead of the symbol or character, such as `plus` instead of `+`.

• View the documentation on "MATLAB Operators and Special Characters".

• Search the PDF documentation, available from the documentation home page.

Searches can include the following operators:

`""` Exact phrase

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

`*` Wildcard

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

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

`OR` Boolean `OR`

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

`NOT` Boolean `NOT`

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

`AND` Boolean `AND`

Implied when no operator is present between keywords.
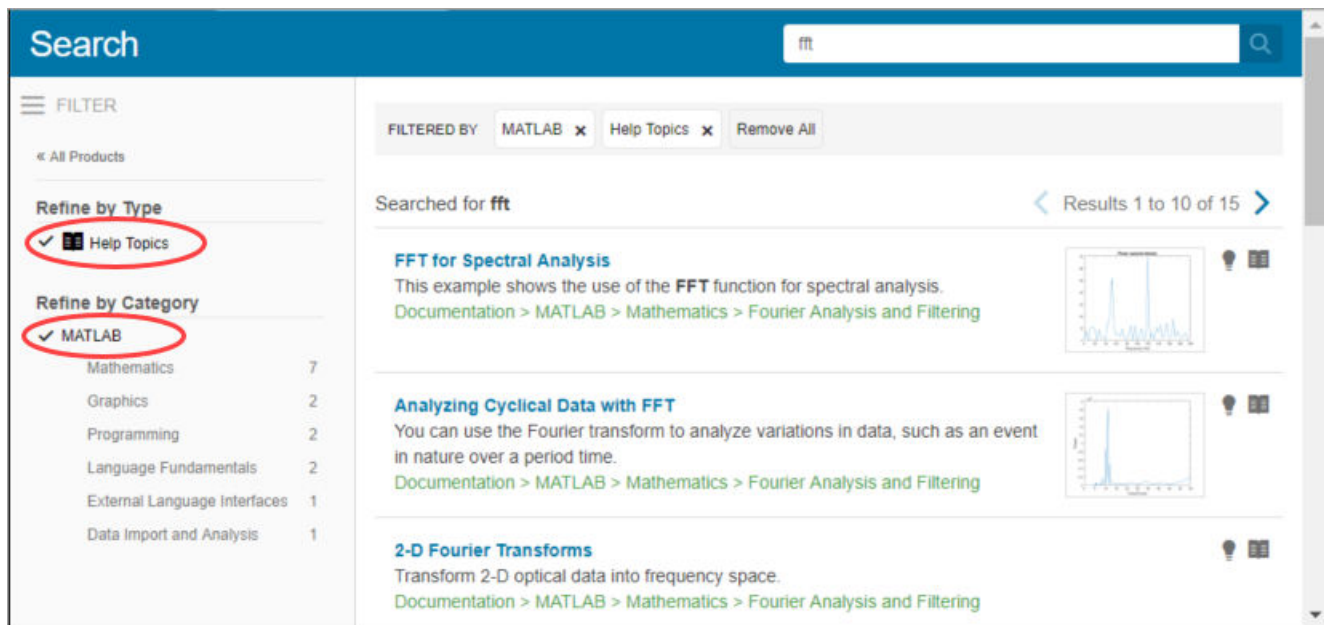
Example: `plot AND tools` is equivalent to `"plot" "tools"`.

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

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

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

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

The 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 4-8

Bookmark Favorite Pages

In the Help browser, bookmarks are called favorites. You can add, find, and organize favorites by clicking the Favorites button in the Help browser, [star icon].

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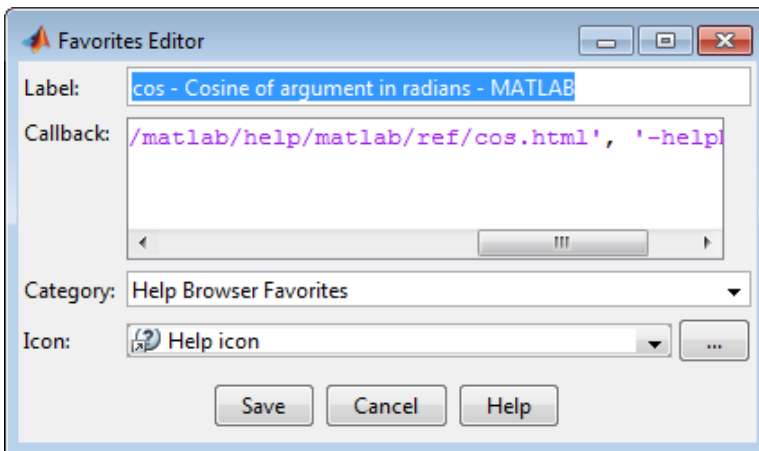
Note You cannot migrate favorites that you save in one MATLAB release to a new release.

View Page Locations

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

The Help Page Location dialog box provides two ways to access the page:
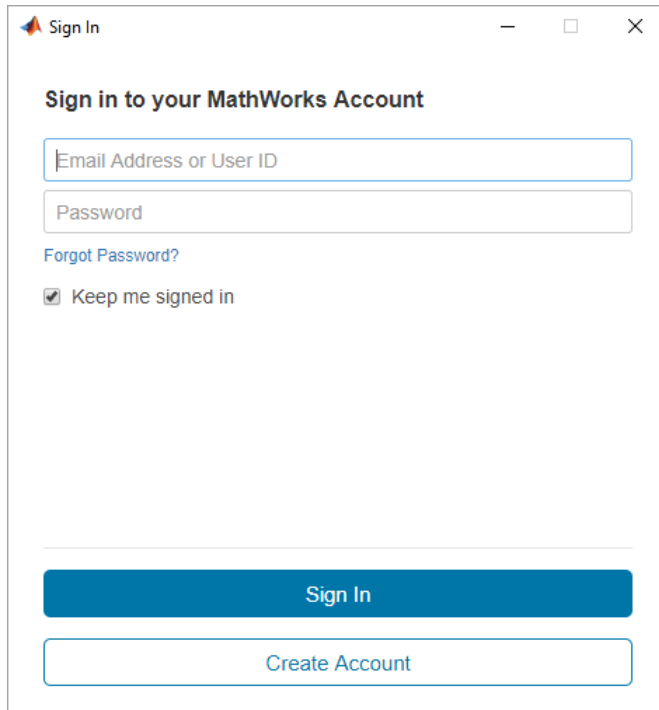
- A web command to run from the command line that opens the page from the installed documentation. This command is subject to change between releases, so it is not always accurate for someone running a different version of MATLAB.
- A URL for the page corresponding to your product version at the MathWorks website. This documentation is available to anyone, even if they do not have MathWorks products. However, to access archived documentation from previous releases, as well as translated documentation, you must sign in with a MathWorks Account.

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

Contact Technical Support

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

1 On the **Home** tab, in the **Resources** section, click ✉ **Request Support**.

2 When requested, sign 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 5 MB.

4 Specify the product that is related to the issue.

5 Submit the request.

See Also

External Websites

- 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 `https://www.mathworks.com/help`. Viewing the web documentation requires an internet connection. 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 Links to reference pages from the Current Folder browser always open in the Help browser.
Language (selected non-English systems only)	Specify whether you want documentation in the Help browser and context-sensitive help to appear in English. Installed non-English documentation is not always current. This option is only available if your **Documentation Location** is set to view documentation on the web, and for selected non-English systems only. For more information, see "Translated Documentation" on page 4-14.

To adjust the font size in the Help browser or MATLAB web browser, use the **Ctrl + Plus (+)** and **Ctrl + Minus (-)** keyboard shortcuts. On macOS systems, use the **Command** key instead of the **Ctrl** key.

See Also

More About

•   "Translated Documentation" on page 4-14

Translated Documentation

Many MathWorks products have translated documentation in Japanese. In addition, some features of MATLAB have translated documentation in Korean, simplified Chinese, Spanish, French, Italian, and German. Usually, if a product provides translated documentation, the new version of the product installs the translated documentation from the *previous* version and the English documentation for the *current* version.

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

Set Documentation Language

By default, if your system language is set to Japanese, Korean, Chinese, Spanish, French, Italian, or German, MATLAB opens the web documentation in that language.

If the documentation does not display in the language you want, check the preference settings using these steps:

1   Go to the **Home** tab and, in the **Environment** section, click ⚙ **Preferences** .

2   Select **MATLAB > Help**.

3   Make sure the **Language** is set to **Japanese**, **Korean**, **Chinese**, **Spanish**, **French**, **Italian**, or **German**.

The **Language** preference is only available when your system language is set to Japanese, Korean, Chinese, Spanish, French, Italian, or German. 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.

To view the English documentation, set the **Language** preference to English.

View Installed Documentation

To view the installed version of the documentation, follow these steps:

1   Go to the **Home** tab and in the **Environment** section, click ⚙ **Preferences** .

2   Select **MATLAB > Help**.

3   Set the **Documentation Location** to **Installed locally**.

4   Select the desired **Language**.

Spanish documentation is only available on the web, and is not installed with MATLAB.

View Documentation on the Web

To view the documentation directly on the web in a different language:

1   Open the MathWorks help website, https://www.mathworks.com/help.

2   Click the button with the globe icon 🌐 at the bottom-left corner of the web page, and select a country based on the language in which you want to view the documentation.

See Also

Related Examples

- "Set Locale on Microsoft Windows Platforms" on page 9-5
- "Set Locale on Linux Platforms" on page 9-8
- "Set Locale on macOS Platforms" on page 9-7

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 on page 4-10.

Type of Information You Want	To Get the Information
Version and license for Installed product	From the product, select **Help > About _Product Name_**. 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 > Licensing > Manage Licenses**.

Workspace Browser and Variable Editor

Create and Edit Variables

Create Variables

The MATLAB workspace consists of the variables you create and store in memory during a MATLAB session. You can create new variables in the workspace by running MATLAB code or using existing variables.

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

```
x = 5.71;
A = [1 2 3; 4 5 6; 7 8 9];
I = besseli(x,A);
```
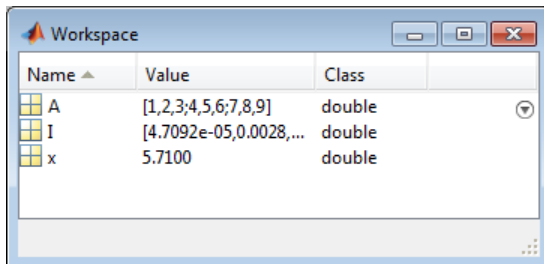

You do not have to declare variables before assigning values to them.

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

```
x = 5.71
```

```
x =
    5.7100
```

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

```
sin(1)
```

```
ans =
    0.8415
```

To view and edit variables, use the Workspace browser and Variables Editor. (Some editing options are not available in MATLAB Online on page 10-2.)

View Workspace Contents

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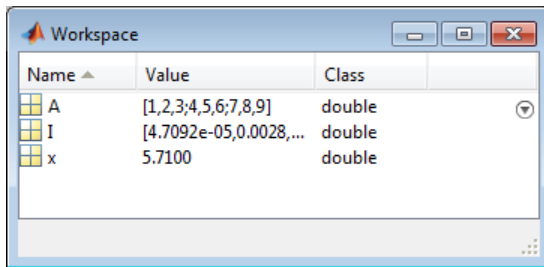
To open the Workspace browser if it is not currently visible, do either of the following:

- On the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select **Workspace**.

- Type `workspace` in the Command Window.

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

To display additional columns such as size and range, on the Workspace browser title bar, click ⊙, and then click **Choose Columns**. In MATLAB Online, to select which columns to display, right-click a column name in the Workspace browser and select or clear the desired column names.

You also can use the `who` command in the Command Window to view a list of variables. To list information about size and class, use the `whos` command. For example, if you have the variables `x`, `A`, and `I` in your workspace, you can run the `who` and `whos` commands to view your workspace contents:

```
who

Your variables are:

A  I  x

whos
  Name      Size            Bytes  Class     Attributes

  A         3x3                72  double
  I         3x3                72  double
  x         1x1                 8  double
```

View Variable Contents

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

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

  ```
  x

  x =
      5.7100
  ```

- Variables editor — In the Workspace browser, double-click a variable name. The Variables editor opens for that variable.
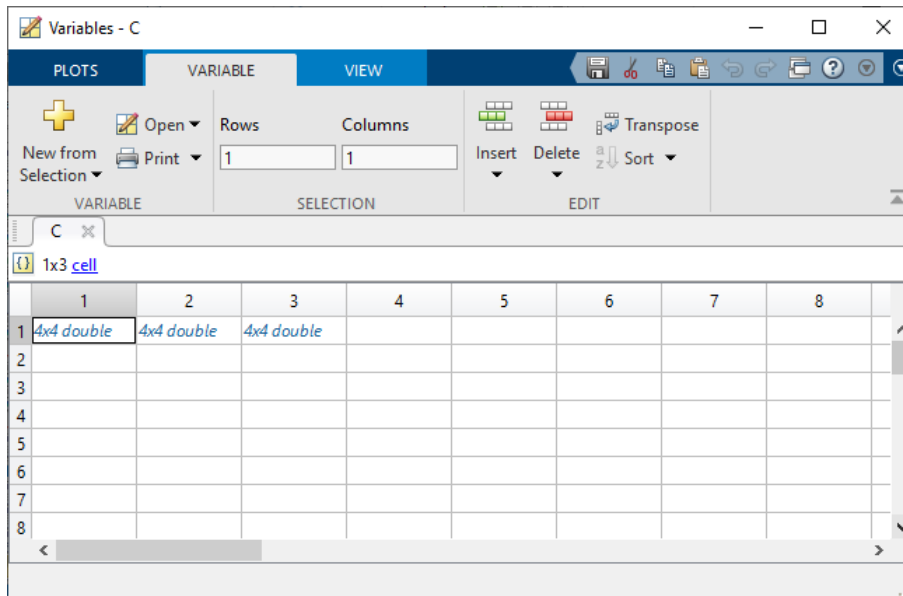
Some variables open a viewer or other tool appropriate for the type of value they have, such as `timeseries`. For details, see the documentation for that data or object type.

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

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

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

Edit Variable Contents

You can edit the contents of scalar (1-by-1) variables in the Workspace browser. To edit the contents, click the variable value to select it and then enter a new value.

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

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

In the Workspace browser, double-click the variable name `C` to open it in the Variables editor.
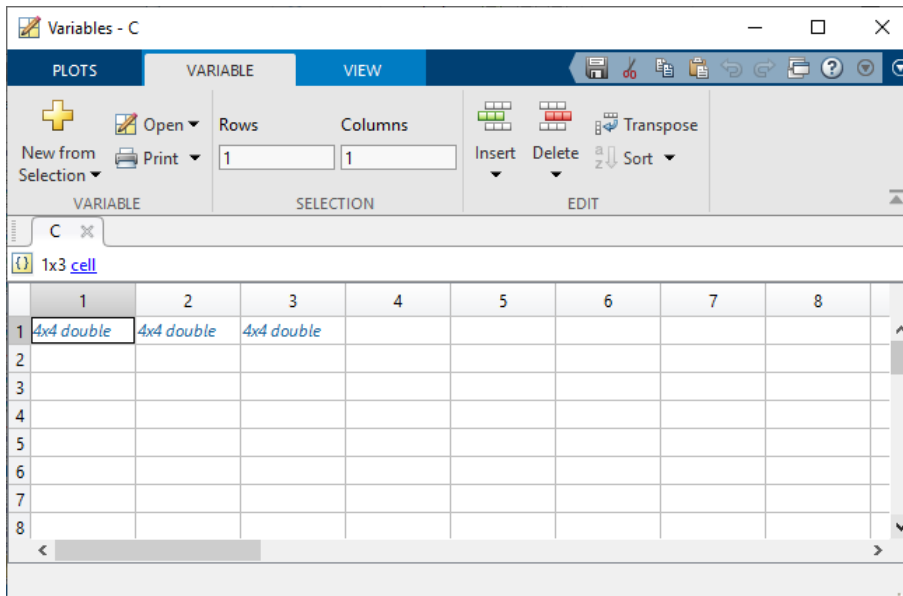
To edit an element of a variable, double-click the element. The element opens in a new document within the Variables editor. For example, if you double-click element `C{1,1}` in the Variables editor, the contents of that cell open in a new tab. You can edit the value of a variable element by clicking the element and typing a new value. Press **Enter** or click another element to save the change.
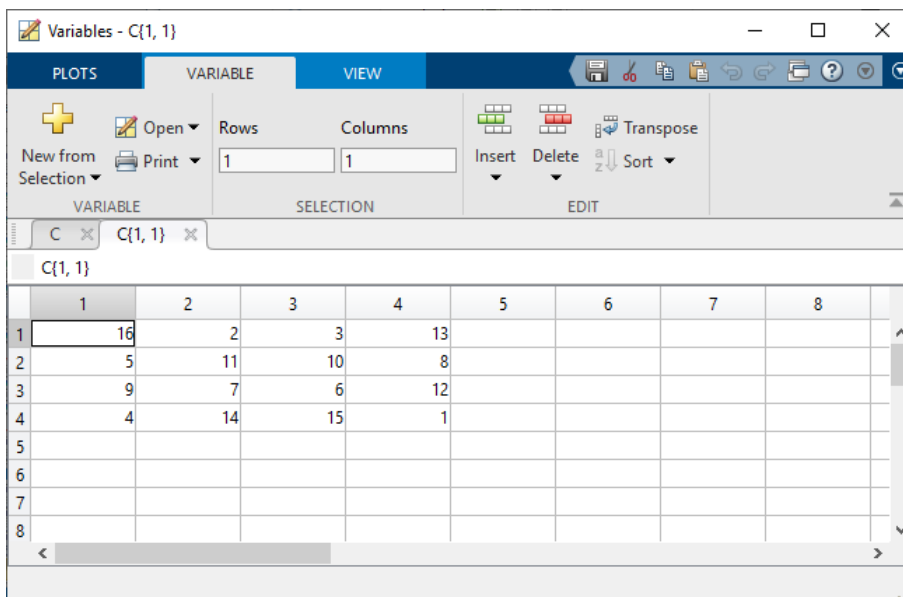
To return to the parent cell array or structure of an element, go to the **View** tab and click the **Go Up** button.
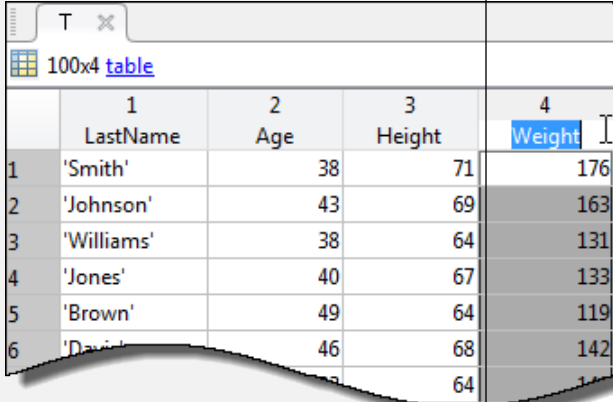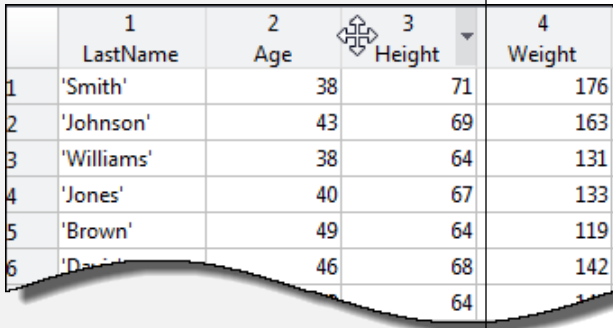
Changes you make in the Variables editor are automatically saved in the workspace. Changes you make to variables via the Command Window or other operations automatically update the information for those variables in the Workspace browser and Variables editor.

Note

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

Edit Table and Structure Array Variables

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

Action	Procedure	Example
Modify column or row name	Double-click the name and enter the new text.	
Reorder variables	Hover over the left side of a variable until a four-headed arrow appears. Then, click and drag the column to a new location.	

Action	Procedure	Example
Modify units and description of variables	Click the arrow that appears to the right of a variable name. Then, enter new text in the **Units** and **Description** fields.	
Sort variable data	Click the arrow that appears to the right of a variable name and select **Ascending** or **Descending**.	

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

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

Resize or Reshape Variables

To modify the size, shape, and order of variable elements in the Variable editor, use one of these procedures.

Action	Procedure
Delete row, column, or variable elements	Right-click the desired row header, column header, or selected elements and select **Delete Row** or **Delete Column**.
Insert new row or column	Right-click the desired row header, column header, or element and select **Insert Row Above**, **Insert Row Below**, **Insert Column to the Left**, or **Insert Column to the Right**. You can also add rows or columns simply by entering a value in an empty row or column. For example, to add a row and column to the array in `C{1,1}`, enter a value in element (5,5).
Cut variable elements	Right-click the desired row header, column header, or selected elements and select **Cut**. The cut values move to the clipboard and are replaced by the default value for empty elements. For more information, see "Empty Elements" on page 5-8.
Copy variable elements	Right-click the desired row header, column header, or selected elements and select **Copy**.
Paste variable elements	Right-click the row header, column header, or element where you want the insertion to begin and select **Paste**.
Paste cells from Microsoft Excel spreadsheet	Right-click the element where you want the insertion to begin, and then select **Paste Excel Data**.

Empty Elements

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

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

Copy, Rename, and Delete Variables

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

Action	Procedure
Copy variable to and from clipboard	Select the variables, right-click, and then select **Copy**. Then, you can paste the names, for example, into the Command Window or an external application. Multiple variables are comma-separated.
Duplicate variable	Select the variables, right-click, and then select **Duplicate**. MATLAB creates a copy of the selected variables.
Rename a workspace variable	Right-click the variable name, and then select **Rename**. Type the new variable name and press **Enter**.

Action	Procedure
Delete all variables in workspace	On the **Home** tab, in the **Variable** section, click **Clear Workspace**. You also can use the `clear` function in the Command Window.
Delete selected variables from workspace	Select the variables in the Workspace browser, right-click, and then select **Delete**. You can also use the `clear` function in the Command Window. For example, to clear variables A and B, use the command `clear A B`. To preserve specified variables, but delete others, use the `clearvars` function with the `-except` option. For example, to clear all variables except variable A, use the command `clearvars -except A`.

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

You can change the character that delimits decimals in the data when you cut and paste values from the Variables editor into text files or other applications. You might do this, for instance, if you provide data to a locale that uses a character other than the period (.). To change the delimiter character, specify a **Decimal separator for exporting numeric data via system clipboard** in the "Variables Preferences" on page 5-16.

Navigate Variable Contents

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

Action	Keyboard Shortcut
Commit changes to element and move to next element. "Variables Preferences" on page 5-16 enable you to specify what the next element is. The default is to move down.	**Enter**
Move right. Within a selection, **Tab** also moves from the last column in one row to the first column in the next row.	**Tab**
Move in opposite direction of **Enter** or **Tab**.	**Shift+Enter** or **Shift+Tab**
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**

Action	Keyboard Shortcut
Edit current element, positioning cursor at the end of the element.	**F2** (**Ctrl+U** on Apple Macintosh platforms)

See Also

openvar | **Workspace Browser**

More About

- "Variable Names"
- "Display Statistics in the Workspace Browser" on page 5-11
- "Workspace and Variable Preferences" on page 5-15
- "Save and Load Workspace Variables" on page 5-13

Display Statistics in the Workspace Browser

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

To display statistics, on the Workspace browser title bar, click ⊙, and then select **Choose Columns**. Select the statistics you want MATLAB to calculate. In MATLAB Online, to select which columns to display, right-click a column name in the Workspace browser and select or clear the desired column names.

Improve Workspace Browser Performance During Statistical Calculations

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

- Show only the statistics of interest to you.

  On the Workspace browser title bar, click ⊙, and then select **Choose Columns**. Clear the statistics that you do not want MATLAB to calculate. In MATLAB Online, to change which statistics display, right-click a column name in the Workspace browser and clear the statistics that you do not want MATLAB to calculate.

- Exclude large arrays from statistical calculations.

  On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Workspace**, and then use the arrow buttons under **Statistical calculations** to change the maximum array size for which the Workspace browser performs statistical calculations. Any variable exceeding the maximum array size reports `<Too many elements>` in Workspace browser statistics columns instead of statistical results.

  Changing Workspace preferences is not supported in MATLAB Online.

Include or Exclude NaN Values in Statistical Calculations

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

- **Use NaNs when calculating statistics**

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

- **Ignore NaNs whenever possible**

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

Changing Workspace preferences is not supported in MATLAB Online.

See Also

More About

- "Infinity and NaN"
- "Workspace and Variable Preferences" on page 5-15

Save and Load Workspace Variables

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

Save Workspace Variables

There are several ways to save workspace variables interactively:

- To save all workspace variables to a MAT-file, on the **Home** tab, in the **Variable** section, click **Save Workspace**.
- To save a subset of your workspace variables to a MAT-file, select the variables in the Workspace browser, right-click, and then select **Save As**. You also can drag the selected variables from the Workspace browser to the Current Folder browser.
- To save variables to a MATLAB script, click the **Save Workspace** button or select the **Save As** option, and in the **Save As** window, set the **Save as type** option to **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.

You also can save workspace variables programmatically using the `save` function. For example, to save all current workspace variables to the file `june10.mat`, use the command

```
save('june10')
```

To save only variables A and B to the file `june10.mat`, use the command

```
save('june10','A','B')
```

To store fields of a scalar structure as individual variables, use the `save` function with the `-struct` option. This can be useful if you previously loaded variables from a MAT-File into a structure using the syntax `S = load(filename)` and want to keep the original variable structure when saving to a new MAT-File.

To save part of a variable, use the `matfile` function. This can be useful if you are working with very large data sets that are otherwise too large to fit in memory. For more information, see "Save and Load Parts of Variables in MAT-Files".

In MATLAB Online on page 10-2, variables persist between sessions. Saving allows you to clear the workspace and load variables at a later time. To save variables, use the `save` or `matfile` functions.

Load Workspace Variables

To load saved variables from a MAT-file into your workspace, double-click the MAT-file in the Current Folder browser.

To load a subset of variables from a MAT-file on the **Home** tab, in the **Variable** section, click **Import Data**. Select the MAT-file you want to load and click **Open**. You also can drag the desired variables from the Current Folder browser Details panel of the selected MAT-file to the Workspace browser. The Details panel is not available in MATLAB Online.

To load variables saved to a MATLAB script into the workspace, simply run the script.

You also can load saved variables programmatically, use the `load` function. For example, load all variables from the file `durer.mat`

```
load('durer')
```

To load variables X and `map` from the file `durer.mat`

```
load('durer','X','map')
```

To load part of a variable, use the `matfile` function. This is useful if you are working with very large data sets that are otherwise too large to fit in memory. For more information, see "Save and Load Parts of Variables in MAT-Files".

Caution When you load data into the MATLAB workspace, the new variables you create overwrite any existing variables in the workspace that have the same name. To avoid overwriting existing variables, use the `load` function to load the variables into a structure. For example, `S = load('durer')` loads all the variables from the file `durer.mat` into the structure `S`.

View Contents of MAT-File

To see the variables in a MAT-file before loading the file into your workspace, click the file name in the Current Folder browser. Information about the variables appears in the **Details** pane.

Alternatively, use the command `whos -file filename`. This function returns the name, dimensions, size, and class of all variables in the specified MAT-file. For example, you can view the contents of the example file `durer.mat`.

```
whos -file durer.mat

  Name          Size              Bytes  Class      Attributes

  X           648x509           2638656  double
  caption       2x28                112  char
  map         128x3               3072  double
```

The byte counts represent the number of bytes that the data occupies in memory when loaded into the MATLAB workspace. Because of compression, data encoding, and metadata, the space occupied in the file by a variable may be different from the in-memory size. MATLAB compresses data in Version 7 or higher MAT-files. For more information, see "MAT-File Versions".

See Also
`save` | `load`

More About
- "MAT-File Versions"
- "Save and Load Parts of Variables in MAT-Files"

Workspace and Variable Preferences

In this section...
"Workspace Preferences" on page 5-15
"Variables Preferences" on page 5-16

Workspace Preferences

Workspace preferences enable you to configure options for saving workspace variables to MATLAB scripts. They also enable you to restrict the size of arrays on which MATLAB performs calculations, and to specify if you want those calculations to include or ignore NaNs.

To open Workspace preferences, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **MATLAB > Workspace**. Workspace preferences are not available in MATLAB Online.

Preference	Usage
Threshold for saving 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)**. Specify the dimensions of a 2-D slice as positive integers less than or equal to the dimensions of the n-D array. The second integer must be greater than the first.
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 "Improve Workspace Browser Performance During Statistical Calculations" on page 5-11.

Preference	Usage
Handling NaN values in calculations	Specify whether NaN values are included or excluded from calculations for the statistics displayed in the Workspace browser. You can select to either **Use NaNs when calculating statistics** or **Ignore NaNs when calculating statistics**.
MATLAB array size limit	By default, MATLAB can use up to 100% of the size of RAM (not including virtual memory) of your computer to allocate memory for each MATLAB array. To change this limit to a smaller percentage, select the **Limit the maximum array size to a percentage of RAM** check box. Then move the slider to adjust the percentage of RAM. To allow MATLAB to use both RAM and virtual memory when creating an array, clear the **Limit the maximum array size to a percentage of RAM** check box. If MATLAB attempts to allocate memory that exceeds the resources available on the computer, your system might become nonresponsive. This limit applies to the size of each array, not the total size of all MATLAB arrays.

Variables Preferences

When working in the Variables editor, Variables preferences enable you to specify array formatting, cursor movement, and the decimal separator for exporting data using the system clipboard.

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

Preference	Usage
Format	Select an option from the **Default array format** to specify the default array output format of numeric values displayed in the Variables editor. This format preference affects only how numbers display, not how MATLAB computes or saves them. For information on formatting options, see the reference page for the `format` function.

Preference	Usage
Editing	Specify where the cursor moves to after you type an element and press **Enter**. 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 and then paste 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.

Most Variables preferences are not available in MATLAB Online. To show generated code in the Command Window when performing operations in the MATLAB Online Variables Editor, select **Show MATLAB code for operations**.

See Also

More About

- "Create and Edit Variables" on page 5-2

6

Managing Files in MATLAB

Find Files and Folders

In this section...

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. To open the Current Folder browser if it is not open, on the **Home** tab, in the **Environment** section, click **Layout**. Then, under **Show**, select **Current Folder**.

Alternatively, to search for a series of characters anywhere in the name of a file:

1 Click the search button ![search] 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.

To clear the results and show all items in the current folder, press the **Esc** key.

In MATLAB Online, to search for files in the current folder or in the current project, go to the **Home** tab and click the ![icon] **Go to File** button.

Advanced Search for Files

To conduct an advanced search for files, you can use the Find Files tool. Use the Find Files tool to:

• Search for specific text in file names and within files
• Include or exclude specified file types from your search
• Search in specified folders
• Exclude large files from your search

To open the Find Files tool, on the **Home** tab, in the **File** section, click ![icon] **Find Files**. Enter your search criteria in the dialog box that opens.
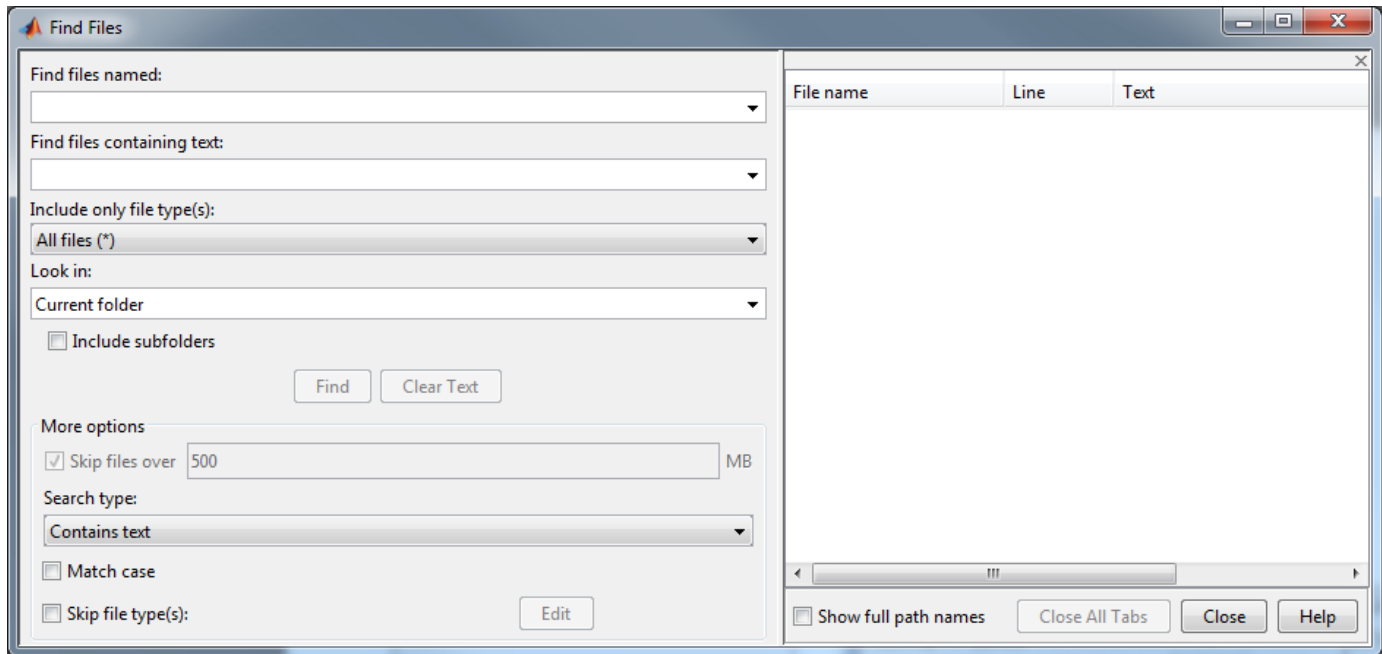
Use the **Look in** menu to specify the folders you want to search. Select **Entire MATLAB Path** to search all folders on the MATLAB search path. Alternatively, you can browse for a folder by selecting **Browse...** or you can enter the full path for one or more folders. Separate each path with a semicolon (**;**).

Click **Find** to begin the search. Search results appear in the right pane of the dialog box, with a summary at the bottom. For text searches, results include the line number and line of code. To see file locations, select **Show full path names**.

Open one or more files by right-clicking the files and selecting one of the **Open** options.

Include or Exclude File Types

You can search for files with only a specified extension, by selecting an option in the **Include only file type(s)** menu. For example, select ***.m, *.mlx** to limit the search to MATLAB program files.

To exclude some file types from the search:

1   In the **Include only file type(s)** menu, select **All files (*)**.

2   Under **More options**, select the **Skip file type(s)** box and click **Edit**. The Edit Skipped File Extensions dialog box opens.

3   Select the **State** check box for the file types to exclude from your search.

4   Click **OK** to accept your changes.

You can remove any file extension from the list by selecting the extension to highlight it. Then, click **Remove**.

Search Within File

Under **More options**, you can choose to search file contents for a partial word. From the **Search type** menu, select **Contains text**. To find an exact full-word match, select **Matches whole word**.

Searching within large files can be time consuming. To speed up your search, specify a file size in the **Skip files over** field. The Find Files tool ignores files larger than the size you specify.

Troubleshooting

If the Find Files tool does not find the file you want, try the following:

- When searching for file names, use the asterisk character (*) as a wildcard character to expand your search. For example, to show file names that begin with `coll` and have a `.m` extension, type `coll*.m`.
- Select the **Include Subfolders** check box if the file might be in a subfolder.
- If you select the **Skip file type(s)** check box, ensure that you do not exclude relevant file types from your search. Click **Edit** and review the Edit Skipped File Extensions dialog box. Ensure that relevant file types do not have the **State** check box selected.

See Also
dir | exist | what | which

More About

Compare Files and Folders and Merge Files

You can use the Comparison Tool to display the differences between selected pairs of files or folders. For some file types, you also can merge changes from one file to the other.

Comparison Process

The comparison process involves three steps:

1   Select the files or folders to compare.
2   Choose a comparison type.
3   Explore the comparison report.

Select the Files or Folders to Compare

You can select files and folders to compare using any of these methods:

*   MATLAB desktop — Go to the **Home** tab, and in the **File** section, click **Compare**. Click the button to select items to compare, or drag and drop files from your file browser into the **First file or folder** or **Second file or folder** fields.

*   Current Folder browser — Select a file or folder, right-click, and select **Compare Against**. Click the button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field. To select two files or subfolders to compare, **Ctrl**-click the file names. Then right-click and select **Compare Selected Files/Folders**.

*   Editor or Live Editor — Go to the **Editor** or **Live Editor** tab, and in the **File** section, click **Compare**. The Comparison Tool includes the currently open file in the **First file or folder** field.

    Click the button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field.

    To compare a file with the autosave version or the saved version on disk, go to the **Editor** tab and click **Compare** and select **Compare with Version on Disk** or **Save and Compare with Autosave**. If your file is modified, the Editor saves the file before comparing. The **Compare with Version on Disk** and **Save and Compare with Autosave** options are only available if the option for automatically saving changes to a file is disabled. To disable the option, go to the **Home** tab, and in the **Environment** section, click **Preferences**. Then, select **Editor/Debugger** and clear **Save changes upon clicking away from a file**. This option is not available in the Live Editor.

*   Command Window — Use the `visdiff` function. For example, to compare the two files `lengthofline.m` and `lengthofline2.m` using the `visdiff` function and the default text comparison, type `visdiff('lengthofline.m','lengthofline2.m')`. MATLAB opens the Comparison Tool and displays the resulting comparison report.

Tip   When selecting folders to compare, select **Include subfolders** to include subfolder content in the comparison.

Choose a Comparison Type

The Comparison Tool compares selected items using the default comparison type defined for the selected items. For some item types, you can select a different comparison type from the default. For

example, you can choose from a text, binary, file list, or XML comparison. To change the comparison type, select the files or folders to compare. Then, select from the available **Comparison type** options. The Comparison Tool displays only the valid options for the selected item types.

You cannot change the comparison type for an existing comparison report. To change the comparison type, start a new comparison.
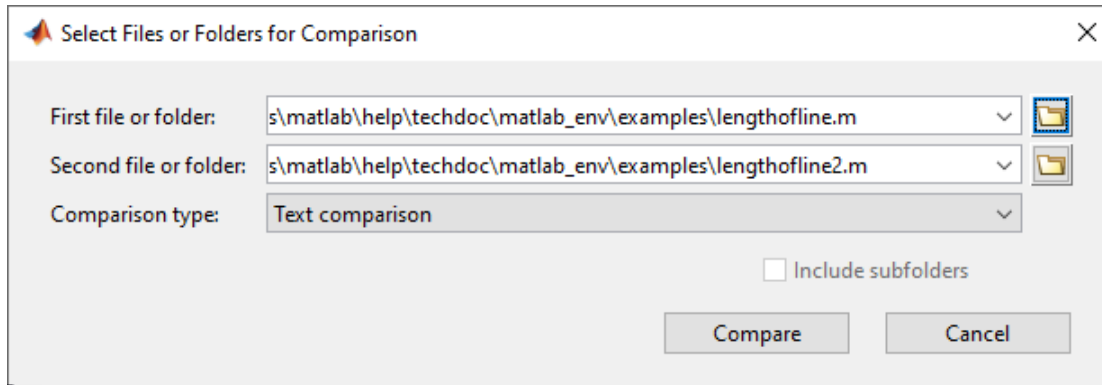
If you specify two files or folders to compare using either the Current Folder browser or the `visdiff` function, then the Comparison Tool automatically performs the default comparison type. For example, from the Current Folder browser, if you select two XML files to compare, the tool uses the default text comparison. To change the comparison type to a hierarchical comparison instead, create a new comparison using the Comparison Tool.

Explore the Comparison Report

After selecting the comparison type, click the **Compare** button. The comparison report opens. You can explore the comparison report in different ways depending on the items being compared. For more information, see the details described for each comparison type below.

Compare Folders and Zip Files

You can compare any combination of folders and ZIP files using the Comparison Tool. For example, you can compare the contents of a folder to the contents of a ZIP file. The Comparison Tool performs a file list comparison of the selected items. Comparing folders and ZIP files is not supported in MATLAB Online on page 10-2.

Create a folder comparison report to:

- Identify common files.
- Determine whether files or folders with identical names have identical content. If such files or folders exist, you can open a detailed comparison report for the items directly from the report.

When you perform a file list comparison, a new window opens and displays the contents of the specified lists side by side. For example, suppose that you have two folders, `curvefittings` and `curvefittings2`, with several differences. When you compare the two folders, the Comparison Tool displays the resulting report.

The Comparison Tool highlights files and subfolders that do not match using the colors listed in this table.

Highlight Color	Description
Purple	The contents of the files or folders differ. Click the **compare** link to investigate.
Blue	The file or folder only exists in the right list.
Yellow	The file or folder only exists in the left list.
None	The files or folders are identical.

You further can explore the results using several methods:

- To sort the results by name, type, size, or last modified timestamp, click the corresponding column header. For example, click the **Type** column header to sort by folder and file type.

- To open a detailed comparison report for items with differing content, click the **compare** link next to the item.

- 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** version of the 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. You can click **Skip Current** to skip the current item or **Cancel All** to stop further analysis.

Items still to be compared: 12   Skip Current   Cancel All

- To save time when reviewing differences, especially when comparing many subfolders, you can filter the report. To apply a filter, on the **View** tab, select **Filter > "*filterName*"**.

  To create a new filter, on the **View** tab, select **Filter > Add/Remove Filter**. You can specify filters to ignore certain files and folders, such as backup files or files created by a revision control system. For example, to ignore all files and folders in a folder named CVS, type CVS/. To ignore all files in a folder named CVS, but not ignore subfolders, type CVS/*.

  To edit existing filters, double click the filter.

For information about how to further configure the comparison report, see "Additional Comparison Tools" on page 6-12.

Compare Text Files

You can compare and merge lines in two text files using the Comparison Tool. When you perform a text comparison, a new window opens and displays the two files side by side. For example, run this code to view a text comparison of the example files lengthofline.m and lengthofline2.m.

```
file1 = fullfile(matlabroot,'help','techdoc','matlab_env',...
'examples','lengthofline.m');
file2 = fullfile(matlabroot,'help',...
'techdoc','matlab_env','examples','lengthofline2.m');
visdiff(file1,file2)
```

The Comparison Tool displays the resulting report.

The Comparison Tool displays the total number of differences in the bottom-right corner of the comparison report and highlights the lines that have changed with the colors listed in this table.

Highlight Color	Description
Purple	The line contains differences.
Dark Purple	The highlighted characters are different.
Blue	The line exists only in right file. This can happen when the line is inserted in the right file, or when it is deleted from the left file. The corresponding line in the left file is highlighted using a gray striped pattern.
Yellow	The line exists only in left file. This can happen when the line is inserted in the left file, or when it is deleted from the right file. The corresponding line in the right file is highlighted using a gray striped pattern.

The Comparison Tool attempts to match lines and detects text that is added, deleted, or changed. For example, in the text comparison of `lengthofline.m` and `lengthofline2.m`, the tool determines that `lengthofline2.m` has a line of code that does not exist in `lengthofline.m` and highlights it (line 23) in blue. Also, 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 in both files.

You further can explore and configure the results using several methods:

- Ignore whitespace differences — To help distinguish between functional changes and changes to indentation, you can hide whitespace differences. To hide differences that only involve white-space characters, click **Filter**, and select **Ignore White Space**.

- Show compared files details — To show the file details, click the expand arrow next to the file name.

For information about how to further configure the comparison report, see "Additional Comparison Tools" on page 6-12.

To step through the results one difference at a time, use the ⬇ **Next** and ⬆ **Previous** buttons.

By default, XML files are compared using a text comparison. To compare XML files with a hierarchical comparison instead, see "Compare XML Files" on page 6-26.

Merge Text Files

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. When merging changes, you only can merge from left to right. If you want to merge into the left file, click the ⬌ **Swap Sides** button before you start merging. Clicking the ⬌ **Swap Sides** button reverts any merges already made and creates a new comparison report for the original files.

To begin merging, on the **Comparison** tab, click the 🗔 **Merge Mode** button. Then, to replace content from the right pane with content from the left pane, click the 🗔 button located next to the line you want to merge. You also can select a difference and, in the **Merge** tab, click the 🗔 **Replace Content** button.

The right pane contains the merged result. An asterisk next to the merged file name in the right pane (*filename*_modified.m*) indicates that the file contains unsaved changes.

To undo a replacement, click the ↩ button between the modified lines or on the **Comparison** tab. To revert all merges and start again, click the ↻ **Refresh** button. You also can click the ↻ **Refresh** button to update the comparison report after making changes to and saving the files. Refreshing discards all unsaved merged changes.

To save your changes and return to the comparison report, click the ✔ **Save Result** button. To return to the comparison report without saving changes, click the ✖ **Return to Comparison** button.
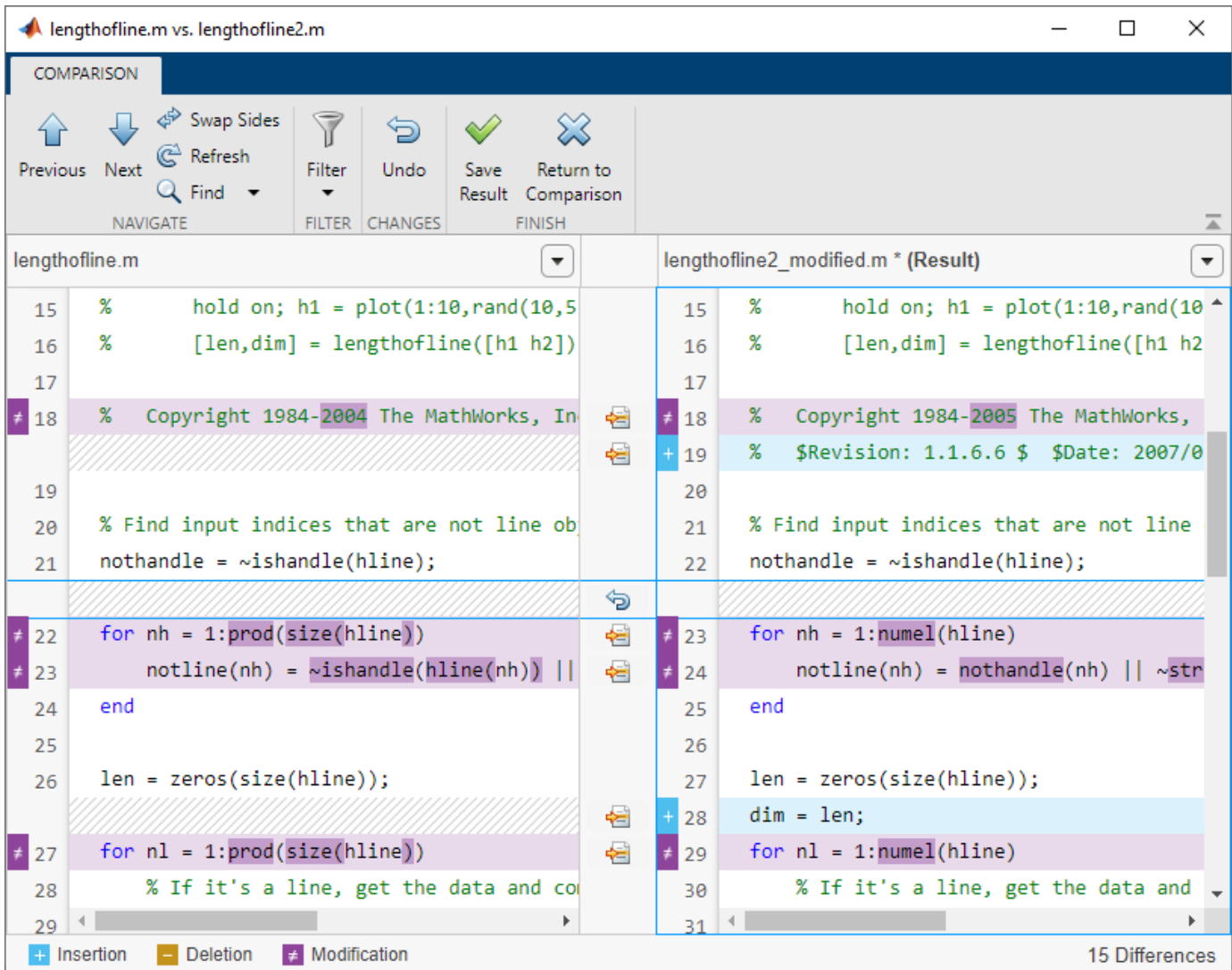
Compare Binary Files

You can compare two binary files, such as DLL files or MEX files, using the Comparison Tool. You also can perform a binary comparison on any two selected files, instead of the default comparison.

To compare any two files using a binary comparison, in the **Comparison type** menu, select `Binary comparison`. When you perform a binary comparison, a new window opens and indicates whether the two files are identical or different. If the files are different, click the **Show Details** link to view the binary files and the byte offset of the first difference.

Compare Other File Types

You can use the Comparison Tool to compare other file types:

- Live Code — Compare and merge the code and text in two live code files. For more information, see "Compare and Merge Live Scripts and Functions" on page 6-14.

- Apps — Compare and merge the code in two apps. For more information, see "Compare and Merge Apps" on page 6-18.
- MAT-Files — Compare and merge variables in two MAT-files. For more information, see "Compare and Merge MAT-Files" on page 6-22.
- XML Files — Compare two XML files using a text comparison or a hierarchical XML comparison. For more information, see "Compare XML Files" on page 6-26.
- Simulink Models — If you have Simulink, you can compare and merge Simulink models. For information, see "Model Comparison" (Simulink).

Additional Comparison Tools

You can further explore and configure a comparison report using these available tools:

- **Swap Sides** button — Switch the left side file or folder with the right side file or folder.
- **Refresh** button — Update the results in the Comparison Tool after making changes to and saving the files in the Editor.
- **Find** button — Find a phrase in the current display. For more information, see "Find Text in Command Window or History" on page 3-11.

Comparison Preferences

You can customize the Comparison Tool using Comparison preferences. To change the Comparison preferences, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Comparison**.

Color Preferences

You can change the colors that the Comparison Tool uses to highlight lines that have changed. Color preferences apply to all comparison types.

To change the color preferences, in the Comparison preferences **Colors** section, change the colors for differences, modified lines, modified content, and merged lines. The **Sample** area shows a preview of the selected colors. After changing comparison colors, you must refresh any open comparison reports to see the updated colors.

To save the modified color preferences for use in future MATLAB sessions, click **Save As** and enter a name for your color settings profile. After saving a color profile, you can select the profile in the **Active Settings** list. To restore the default color profile, click **Reset**.

External Source Control Integration Preferences

You can use the Comparison Tool to compare and merge files and folders in external source control tools. To enable external source control tools to reuse open MATLAB session, in the Comparison preferences **External Source Control Integration** section, select the **Allow external source control tools to use open MATLAB sessions for diffs and merges** option. When you configure your source control tool to use the MATLAB Comparison Tool, the Comparison Tool prompts you to select this option.

For more information, see "Customize External Source Control to Use MATLAB for Diff and Merge" (Simulink).

See Also
visdiff

Related Examples
- "Customize External Source Control to Use MATLAB for Diff and Merge"

Compare and Merge Live Scripts and Functions

You can compare two live code files and merge changes between one file and another using the Comparison Tool. The Comparison Tool highlights differences in code, text, and text formatting.

Select Files to Compare

There are multiple ways to select two files and start the Comparison Tool:

- Live Editor — Go to the **Live Editor** tab and, in the **File** section, click **Compare**. The Comparison Tool includes the currently open file in the **First file or folder** field. Click the button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field.

-
  Current Folder browser — Select a file, right-click, and select **Compare Against**. Click the button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field. To select two files or subfolders to compare, **Ctrl**-click the file names. Then right-click and select **Compare Selected Files/Folders**.

- Command Window — Use the `visdiff` function. For example, to compare the two files `averageweight.mlx` and `averageweight2.mlx` using the `visdiff` function and the default text comparison, type `visdiff('averageweight.mlx','averageweight2.mlx')`. MATLAB opens the Comparison Tool and displays the resulting comparison report.

Explore Differences

When you perform a live code comparison, a new window opens and displays the two files side by side. For example, suppose that you have two files, `averageweight.mlx` and `averageweight2.mlx`, with several differences. When you compare the two files, the Comparison Tool displays the resulting report.
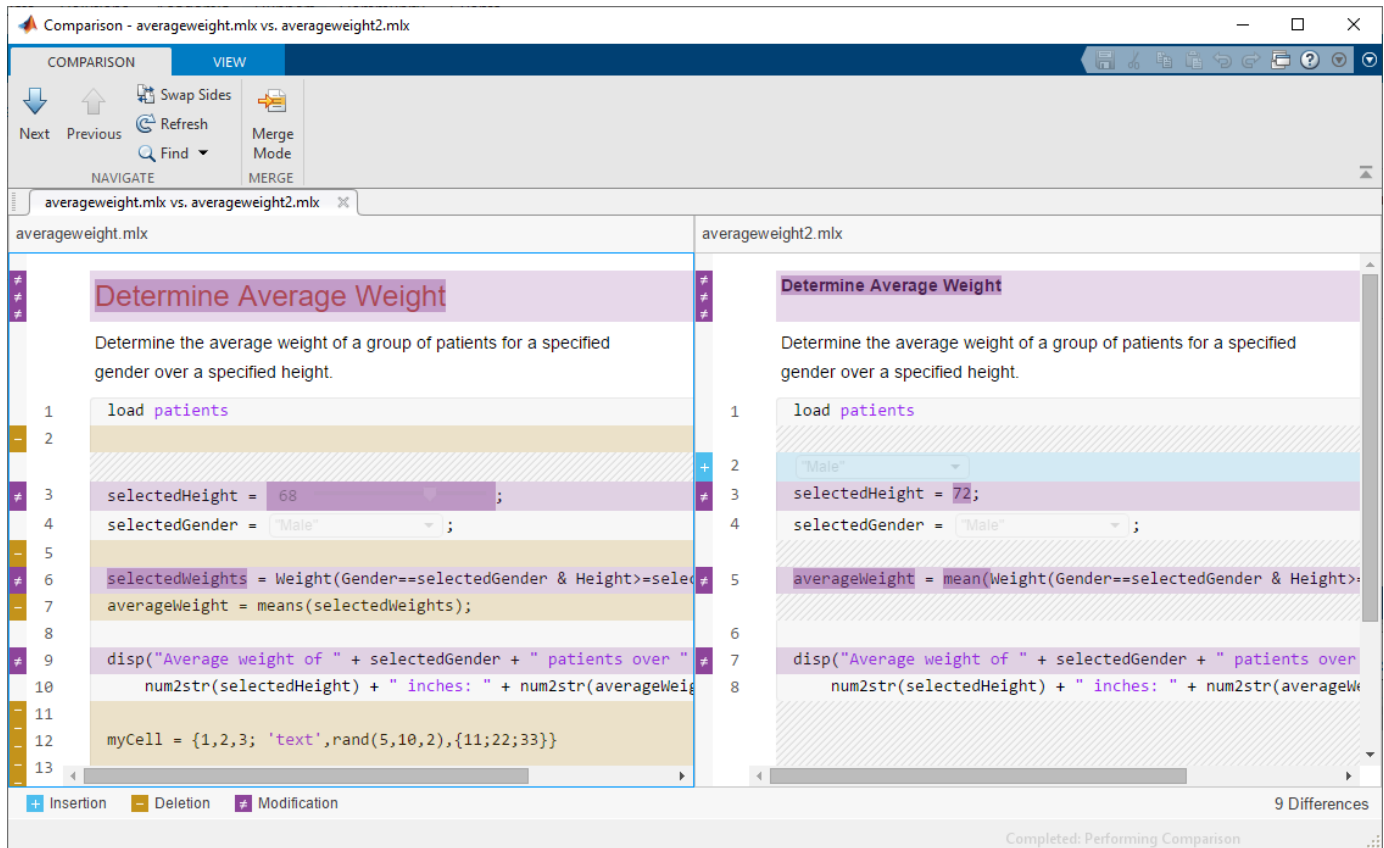
When comparing live code files, the Comparison Tool highlights differences in code, text, and text formatting. The Comparison Tool ignores output, even if the file contains saved output. If the tool detects a difference within an equation, image, or control, the entire item is highlighted. The Comparison Tool displays the total number of differences in the bottom-right corner of the comparison report, and highlights the lines that do not match using the colors listed in this table.

Highlight Color	Description
Purple	The line contains differences.
Dark Purple	The highlighted characters are different.
Blue	The line exists only in right file. This can happen when the line is inserted in the right file, or when it is deleted from the left file. The corresponding line in the left file is highlighted using a gray striped pattern.
Yellow	The line exists only in left file. This can happen when the line is inserted in the left file, or when it is deleted from the right file. The corresponding line in the right file is highlighted using a gray striped pattern.

The Comparison Tool attempts to match lines and detects code and text that is added, deleted, or changed. For example, in the live code comparison of averageweight.mlx and averageweight2.mlx, the tool determines that averageweight.mlx has two lines of code that do

not exist in `averageweight2.mlx` and highlights them (line 5 and line 7) in yellow. Also, the tool takes the additional lines into account and determines that the line containing the `disp` statement in each file matches, even though the `disp` statement does not occur on the same line number.

To step through the results one difference at a time, use the ⬇ **Next** and ⬆ **Previous** buttons.

For information about how to configure the comparison report, see "Compare Files and Folders and Merge Files" on page 6-5.

Merge Changes

When comparing live code files, you can merge changes from one file to the other. Merging changes can be useful when resolving conflicts between different versions of files. You only can merge from left to right. If you want to merge into the left file, click the 🔲 **Swap Sides** button before you start merging. Clicking the 🔲 **Swap Sides** button reverts any merges already made and creates a new comparison report for the original files.

To begin merging, on the **Comparison** tab, click the 🔲 **Merge Mode** button. Then, to replace content from the right pane with content from the left pane, click the 🔲 button located next to the line you want to merge. You also can select a difference and, in the **Merge** tab, click the 🔲 **Replace Content** button.

The right pane contains the merged result. An asterisk next to the merged file name in the right pane (*filename*_modified.mlx*) indicates that the file contains unsaved changes.
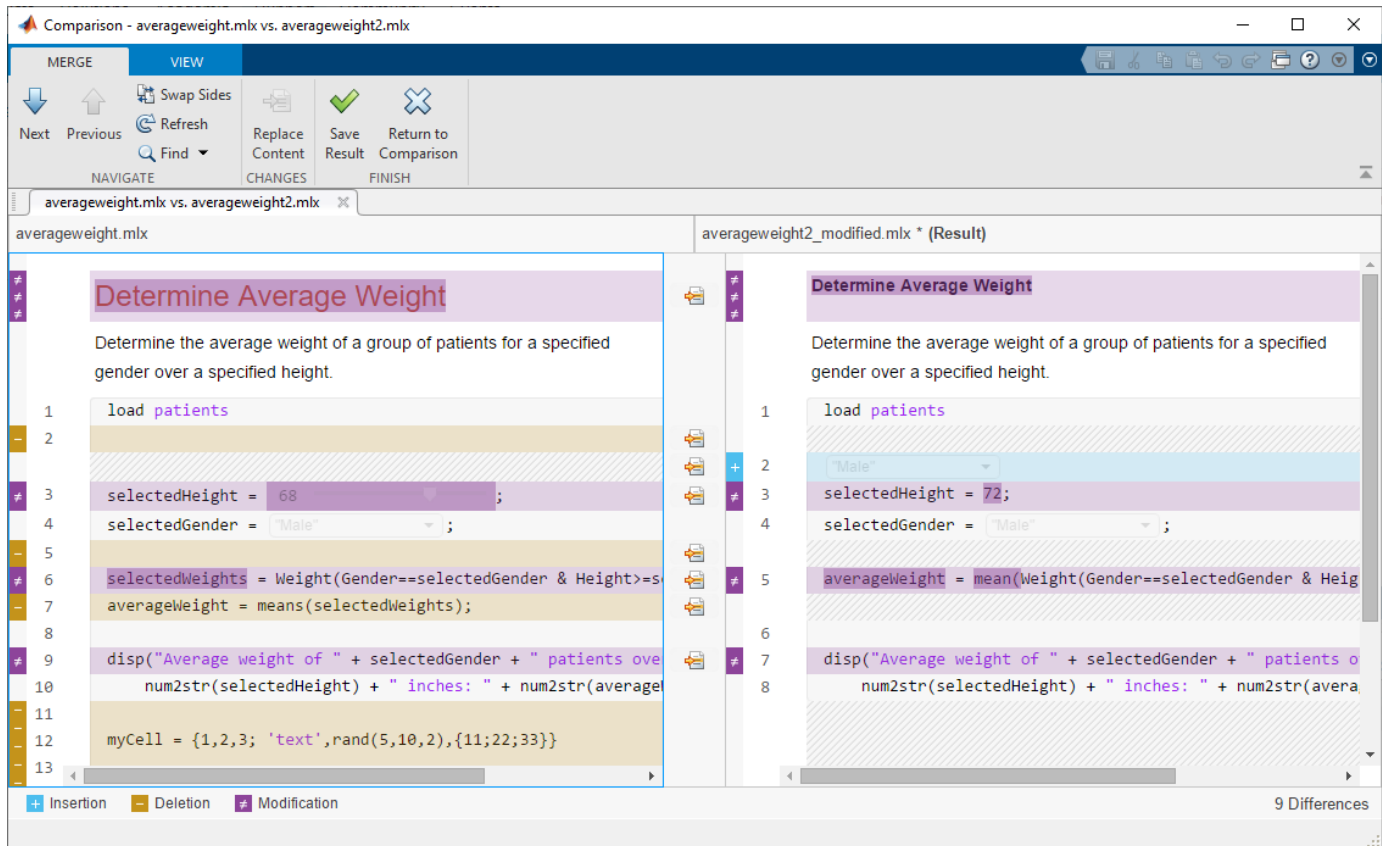
To undo a replacement, click the ⟲ button between the modified lines or on the **Comparison** tab. To revert all merges and start again, click the ⟳ **Refresh** button. You also can click the ⟳ **Refresh** button to update the comparison report after making changes to and saving the files in the Live Editor. Refreshing discards all unsaved merged changes.

To save your changes and return to the comparison report, click the ✓ **Save Result** button. To return to the comparison report without saving changes, click the ✗ **Return to Comparison** button.

See Also

visdiff

More About

- "Compare Files and Folders and Merge Files" on page 6-5
- "Customize External Source Control to Use MATLAB for Diff and Merge"

Compare and Merge Apps

You can compare two apps and merge changes between one file and another using the Comparison Tool. The Comparison Tool highlights differences in the code of two apps.

Select Files to Compare

There are multiple ways to select two files and start the Comparison Tool:

- App Designer — Go to the **Designer** or **Editor** tab and, in the **File** section, click **Compare**. The

  Comparison Tool includes the currently open file in the **First file or folder** field. Click the 🗀 button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field.

- Current Folder browser — Select a file, right-click, and select **Compare Against**. Click the 🗀 button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field. To select two files or subfolders to compare, **Ctrl**-click the file names. Then right-click and select **Compare Selected Files/Folders**.

- Command Window — Use the `visdiff` function. For example, to compare the two files `ExplorerE.mlapp` and `ExplorerF.mlapp` using the `visdiff` function and the default text comparison, type `visdiff('ExplorerE.mlapp','ExplorerF.mlapp')`. MATLAB opens the Comparison Tool and displays the resulting comparison report.

Explore Differences

When you perform an app comparison, a new window opens and displays the two files side by side. For example, suppose that you have two files `ExplorerE.mlapp` and `ExplorerF.mlapp` with several differences. When you compare the two files, the Comparison Tool displays the resulting report.
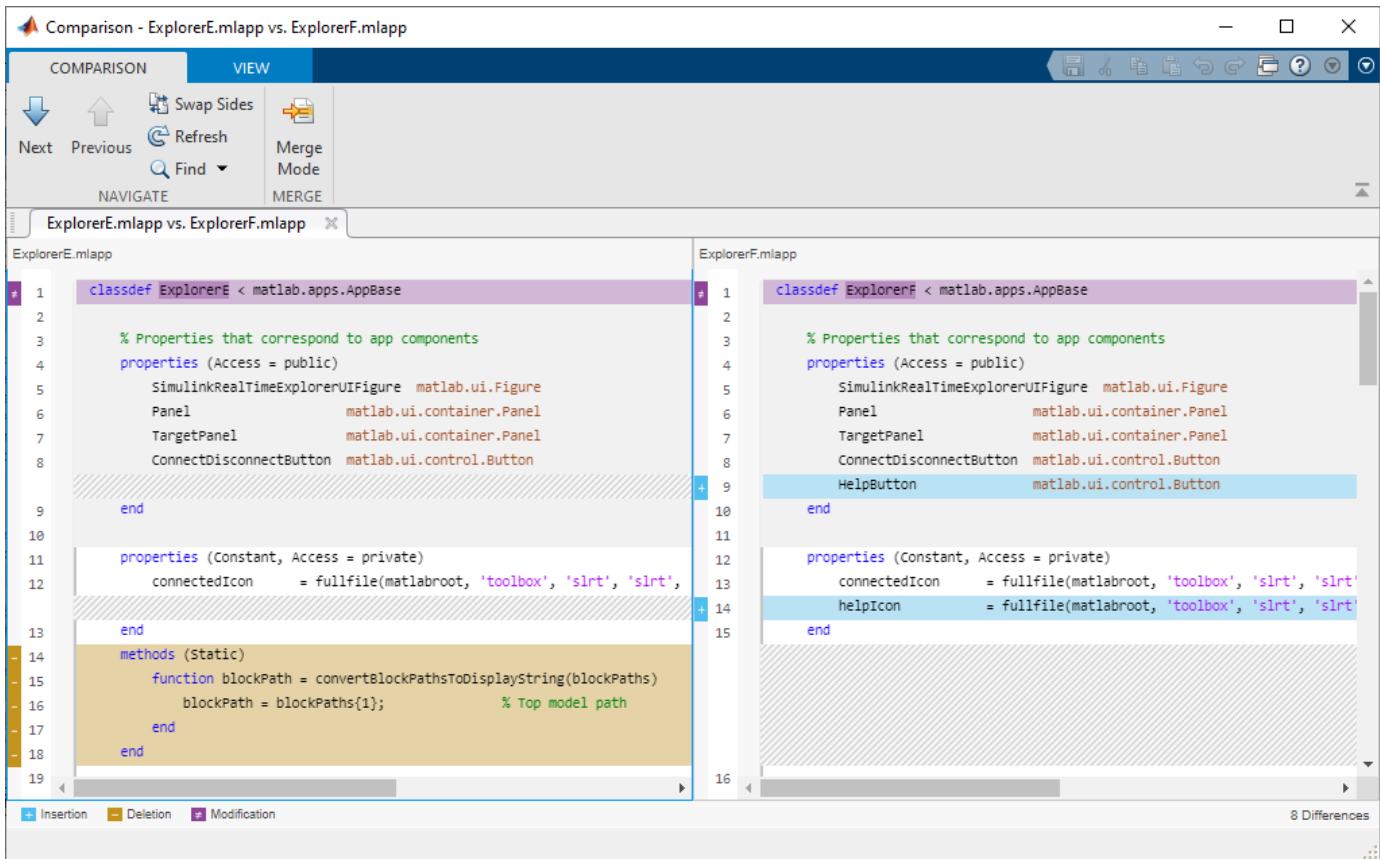
When comparing apps, the Comparison Tool attempts to match lines and detects code that is added, deleted, or changed. The Comparison Tool displays the total number of differences in the bottom-right corner of the comparison report and highlights the lines that do not match using the colors listed in this table.

Highlight Color	Description
Purple	The line contains differences.
Dark Purple	The highlighted characters are different.
Blue	The line exists only in right file. This can happen when the line is inserted in the right file, or when it is deleted from the left file. The corresponding line in the left file is highlighted using a gray striped pattern.
Yellow	The line exists only in left file. This can happen when the line is inserted in the left file, or when it is deleted from the right file. The corresponding line in the right file is highlighted using a gray striped pattern.

To step through the results one difference at a time, use the ⬇ **Next** and ⬆ **Previous** buttons.

For information about how to configure the comparison report, see "Compare Files and Folders and Merge Files" on page 6-5.

Merge Changes

When comparing apps, you can merge changes from one app to the other. Merging changes can be useful when resolving conflicts between different versions of an app.

To begin merging, on the **Comparison** tab, click the ⬜ **Merge Mode** button. Then, to replace content from the right pane with content from the left pane, click the ⬜ button located next to the line you want to merge. You also can select a difference and, in the **Merge** tab, click the ⬜ **Replace Content** button.
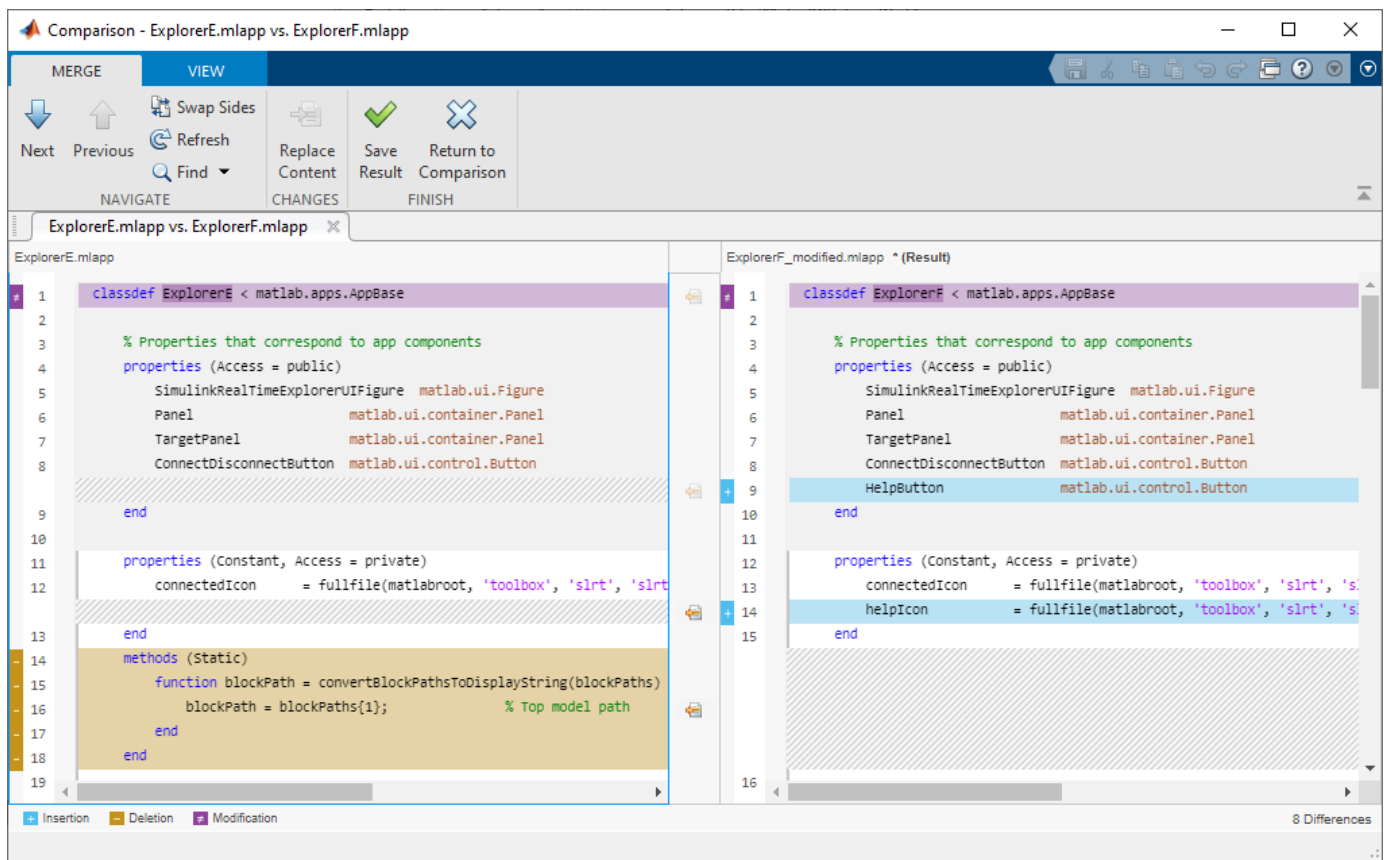
The right pane contains the merged result. An asterisk next to the merged file name in the right pane (*ExplorerF_modified.mlapp**) indicates that the file contains unsaved changes.

To undo a replacement, click the ↩ button between the modified lines or on the **Comparison** tab. To revert all merges and start again, click the ↻ **Refresh** button. You also can click the ↻ **Refresh** button to update the comparison report after making changes to and saving the files in the Live Editor. Refreshing discards all merged changes.

To save your changes and return to the comparison report, click the ✅ **Save Result** button. To

return to the comparison report without saving changes, click the ✖ **Return to Comparison**
button.

Tips for Successful Merging

You can merge changes only in editable code. Sections of code with a white background are editable.
Examples of editable sections include the body of functions that you define (such as callbacks and
helper functions) and the definitions of custom properties. Sections with a gray background contain
non-editable generated code and therefore cannot be merged. If a change spans both an editable and
non-editable section, you cannot merge it.

You can merge changes only from left to right. This makes the locations of the files in the Comparison
Tool important.

If neither app contains changes in the generated code, place the file containing the changes that you
want to merge in the left pane. You only can merge from left to right. To swap a file from the left pane

to the right pane, click the ⬜ **Swap Sides** button before you start merging. Clicking the ⬜ **Swap
Sides** button reverts any merges already made and creates a new comparison report for the original
files.

If one of the apps contains changes in the generated code, place the file containing the generated
code that you want in the merged result in the right pane of the Comparison Tool. To swap a file from

the left pane to the right pane, click the ⬜ **Swap Sides** button before you start merging.

If both apps contain changes in the generated code, the files cannot be merged in the Comparison
Tool. To merge the files, use App Designer to interactively apply the desired changes to the app

See Also
visdiff

More About
- "Compare Files and Folders and Merge Files" on page 6-5
- "Customize External Source Control to Use MATLAB for Diff and Merge"

Compare and Merge MAT-Files

You can use the Comparison Tool to compare two MAT-files. Compare two MAT-files to:

- Determine which variables are common to both files and which appear in only one file.
- Identify differences between individual variables. For example, you can determine which fields of a structure are different or which elements of a array differ.

Select Files to Compare

There are multiple ways to select two files and start the Comparison Tool:

- MATLAB desktop — Go to the **Home** tab and, in the **File** section, click **Compare**. Click the button to select items to compare, or drag and drop files from your file browser into the **First file or folder** or **Second file or folder** fields.

- Current Folder browser — Select a file, right-click, and select **Compare Against**. Click the button to select a second item to compare, or drag and drop a file from your file browser into the **Second file or folder** field. To select two files or subfolders to compare, **Ctrl**-click the file names. Then right-click and select **Compare Selected Files/Folders**.

- Command Window — Use the `visdiff` function. For example, to compare the two files `data1.mat` and `data2.mat` using the `visdiff` function, type `visdiff('data1.mat','data2.mat')`. MATLAB opens the Comparison Tool and displays the resulting comparison report.

Explore Differences

When you perform a MAT-file comparison, a new window opens and displays the two files side by side. For example, suppose that you have two MAT-files, `data1.mat` and `data2.mat`. When you compare the two files, the Comparison Tool displays the resulting report.

The Comparison Tool highlights changes in variables. If the variables in both files are equivalent but the files are not identical, the Comparison Tool displays a message. Possible causes of the differences between the files include file formats, file timestamps, the order in which the variables are stored, or variables that contain ignored differences. The Comparison Tool uses these colors to highlight differences.

Highlighting	Description
Purple	Values of the variable differ between the two files. Click the **compare** link to investigate.
Blue	The variable only exists in the right file.
Yellow	The variable only exists in the left file.
None	The variables in both files are equivalent. The tool ignores differences in NaN patterns, field ordering in structs, and the difference between negative zero or positive zero.
Purple (only in Class columns)	The variable data class changed. Click the **compare** link to investigate.

You can further explore the results in the report using several methods:

- Sort results — To sort the results by name, size, class, or difference summary, click the corresponding column header. For example, click the **Class** column header to sort by class type.
- Load variable — To load the contents of a variable into the Variable Editor, click the name of that variable.
- Load MAT-file — To load one of the MAT-files into the workspace, click the corresponding **Load** link at the bottom of the report.
- Save HTML report — To save a copy of the report as an HTML file, go to the **Comparison** tab and click **Save As > HTML**.

For information about how to further configure the comparison report, see "Compare Files and Folders and Merge Files" on page 6-5.

Compare Variables

To investigate differences between instances of a variable in both files, click the **compare** link in the **Difference Summary** column of the comparison report table. The Comparison Tool displays a report for the specified variable. To further investigate differences in individual array elements or fields of a structure, double-click a highlighted row.

Merge Changes

When comparing MAT-files, you can merge changes from one file to the other. There are no undo capabilities when merging MAT-files, so merge with caution.

To copy modified variables from one file to the other, on the **Comparison** tab, in the **Merge** section, click the ⬇ merge button.

See Also
visdiff

More About

- "Compare Files and Folders and Merge Files" on page 6-5
- "Customize External Source Control to Use MATLAB for Diff and Merge"

Compare XML Files

In this section...
"Choose XML Files to Compare" on page 6-26
"Change Comparison Type" on page 6-27
"Navigate the XML Comparison Report" on page 6-27
"Save Comparison Log Files in a Zip File" on page 6-28
"Export Results to the Workspace" on page 6-29

You can use the Comparison Tool to compare a pair of XML text files. The tool processes the results into a report that you can use to explore the file differences. You can compare XML files with a text comparison or a hierarchical XML comparison.

You can access the XML Comparison Tool from the Current Folder browser, the Comparison Tool, or programmatically using the `visdiff` function.

The XML comparison compares the files using the "Chawathe" algorithm, as described in this paper: *Change Detection in Hierarchically Structured Information*, Sudarshan Chawathe, Anand Rajaraman, and Jennifer Widom; SIGMOD Conference, Montreal, Canada, June 1996, pp. 493-504.

This conference paper is based on work published in 1995: see `http://dbpubs.stanford.edu:8090/pub/1995-45`.

XML comparison reports display in the Comparison Tool. For more information about the Comparison Tool, see "Compare Files and Folders and Merge Files" on page 6-5.

The XML comparison report shows a hierarchical view of the portions of the two XML files that differ. The report does not show sections of the files that are identical.

If the files are identical, you see a message reporting there are no differences.

Note It might not be possible for the analysis to detect matches between previously corresponding sections of files that have diverged too much.

Change detection in the Chawathe analysis is based on a scoring algorithm. Items match if their Chawathe score is above a threshold. The implementation of Chawathe's algorithm uses a comparison pattern that defines the thresholds.

Choose XML Files to Compare

From the Current Folder Browser

To compare two files from the Current Folder browser:

- For two files in the same folder, select the files, right-click and select **Compare Selected Files/Folders**.
- To compare files in different folders:

  1   Select a file, right-click, and select **Compare Against**

2   Select the second file to compare in the Select Files or Folders for Comparison dialog box.

3   Select the **Comparison type** `XML text comparison`.

4   Click **Compare**.

The XML Comparison Tool performs an analysis on the files and displays a report in the Comparison Tool. The file you right-click to open the XML Comparison Tool displays on the right side of the report.

For more information about comparisons of other file types with the Comparison Tool, such as text, MAT, or binary, see "Compare Files and Folders and Merge Files" on page 6-5.

Compare from the Comparison Tool

To compare files using the Comparison Tool, from the MATLAB toolstrip, in the **File** section, select the **Compare** button. In the dialog box, select files to compare.

If the files you select to compare are XML files and you select an `XML text comparison`, the XML Comparison Tool performs a Chawathe analysis of the XML files, and generates a report.

Compare from the Command Line

To compare XML files from the command line, enter

```
visdiff(filename1, filename2, 'xml')
```

where `filename1` and `filename1` are XML files.

Change Comparison Type

If you specify two XML files to compare using either the Current Folder Browser or the `visdiff` function, then the Comparison Tool performs the default comparison type for the selected files. You can compare XML files with a text comparison or a hierarchical XML comparison.

To change comparison type, either create a new comparison from the Comparison Tool, or use the **Compare Against** option from the Current Folder browser. You can change comparison type in the Select Files or Folders for Comparison dialog box. Select `XML text comparison` before clicking **Compare**.

Navigate the XML Comparison Report

The XML comparison report shows changes only. The report is a hierarchical view of the differences between two XML text files, and is not a hierarchical view of the original XML data.

To *step through differences*, use the **Comparison** tab on the toolstrip. To move to the next or previous group of differences, on the **Comparison** tab, in the **Navigate** section, click the arrow buttons to go to the previous or next difference.

You can also click to select items in the hierarchical trees.

- Selected items appear highlighted in a box.

- If the selected item is part of a matched pair, it is highlighted in a box in both left and right trees.

Report item highlighting indicates the nature of each difference as follows:

Type of report item	Highlighting	Notes
Modified	Purple	Modified items are matched pairs that differ between the two files. When you select a modified item it is highlighted in a box in both trees. Changed parameters for the selected pair are displayed underneath.
Inserted	Blue	When you select an unmatched item it is highlighted in a box in one tree only.
Deleted	Yellow	
Container	None	Rows with no highlighting indicate a container item that contains other modified or unmatched items.

Use the **Comparison** tab for the following functions:

- **Refresh** — Run Chawathe analysis again to refresh the comparison report.
- **Swap Sides** — Swap sides and rerun comparison. Runs the Chawathe analysis again.
- **Save As > HTML** — Opens the Save dialog box, where you can choose to save a printable version of the XML comparison report. The report is a noninteractive HTML document of the differences detected by the Chawathe algorithm for printing or archiving a record of the comparison.
- **Save As > Workspace Variable** — Export XML comparison results to workspace.
- In the **Navigate** section, click the arrow buttons (or press Up or Down keys) to go to the previous difference or go to the next difference.
- **Compare Selected Parameter** — Open a new report for the currently selected pair of parameters. Use this when the report cannot display all the details in the Parameters pane, e.g., long character vectors or a script.

Use the **View** tab controls on the toolstrip for the following functions:

- **Expand All** — Expands every item in the tree.

> **Tip** Right-click to expand or collapse the hierarchy within the selected tree node.

- **Collapse All** — Collapses all items in the tree to the most compact view possible.

> **Note** It may not be possible for the analysis to detect matches between previously corresponding sections of files that have diverged too much.

Save Comparison Log Files in a Zip File

Temporary XML comparison files accumulate in *tempdir*/MatlabComparisons/ XMLComparisons/TempDirs/. These temporary files are deleted when you close the related comparison report.

You can zip the temporary files (such as log files) created during XML comparisons, for sharing or archiving. While the comparison report is open, enter:

```
xmlcomp.zipTempFiles('c:\work\myexportfolder')
```

The destination folder must exist. The output reports the zip file name:

```
Created the zipfile "c:\work\myexportfolder\20080915T065514w.zip"
```

To view the log file for the last comparison in the MATLAB Editor, enter:

```
xmlcomp.showLogFile
```

Export Results to the Workspace

To export the XML comparison results to the MATLAB base workspace,

1   On the **Comparison** tab, in the **Comparison** section, select **Save As > Workspace variable**.

    The Input Variable Name dialog box appears.

2   Specify a name for the export object in the dialog box and click **OK**. This action exports the results of the XML comparison to an `xmlcomp.Edits` object in the workspace.

The `xmlcomp.Edits` object contains information about the XML comparison including file names, filters applied, and hierarchical nodes that differ between the two XML files.

To create an `xmlcomp.Edits` object at the command line without opening the Comparison Tool, enter:

```
Edits = xmlcomp.compare(a.xml,b.xml)
```

Property of `xmlcomp.Edits`	Description
`Filters`	Array of filter structure arrays. Each structure has two fields, Name and Value.
`LeftFileName`	File name of left file exported to XML.
`LeftRoot`	`xmlcomp.Node` object that references the root of the left tree.
`RightFileName`	File name of right file exported to XML.
`RightRoot`	`xmlcomp.Node` object that references the root of the right tree.
`TimeSaved`	Time when results exported to the workspace.
`Version`	MathWorks release-specific version number of `xmlcomp.Edits` object.

Property of `xmlcomp.Node`	Description
`Children`	Array of `xmlcomp.Node` references to child nodes, if any.
`Edited`	Boolean — If `Edited = true` then the node is either inserted or part of a modified matched pair.
`Name`	Name of node.
`Parameters`	Array of parameter structure arrays. Each structure has two fields, Name and Value.
`Parent`	`xmlcomp.Node` reference to parent node, if any.

Property of `xmlcomp.Node`	Description
`Partner`	If matched, `Partner` is an `xmlcomp.Node` reference to the matched partner node in the other tree. Otherwise empty `[]`.

See Also

Related Examples

- "Compare Files and Folders and Merge Files" on page 6-5

Manage Files and Folders

This table shows how to create, open, move, and rename files and folders on local and network drives. If you have MATLAB Drive Connector installed on your system, you also can access files and folders in MATLAB Drive from MATLAB.

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. You also can double-click the file in the Current Folder browser. 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 **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. Not supported for live scripts and functions.	none

Action	Tools Workflow	Function Alternative
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`.
View MATLAB Drive recent activity	In the Current Folder browser, right-click any MATLAB Drive file or folder and select **MATLAB Drive > View Recent Activity…**.	none
Open MATLAB Drive online	In the Current Folder browser, right-click any MATLAB Drive file or folder and select **MATLAB Drive > Go to MATLAB Drive Online…**.	none

To open the Current Folder browser if it is not visible, go to the **Home** tab, and 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.

Manage Files and Folders in MATLAB Online

MATLAB Online on page 10-2 provides access to MATLAB from a standard web browser. In MATLAB Online, you only have access to the files and folders in your MATLAB Drive, and some file actions are not available.

A special purpose folder named `Published` stores the paths to your shared published program files and results. This folder is publicly available to others on the web.

To transfer a file from MATLAB Online to the local system, select the file and on the **Home** tab, select **Download**.

See Also
edit | mkdir | open | movefile | rmdir | delete | recycle

More About
- "Share Folders in MATLAB" on page 6-34

- "Save and Back Up Code"
- "Save and Load Workspace Variables" on page 5-13
- "Errors When Updating Folders on Search Path" on page 6-59

Share Folders in MATLAB

Starting in R2019b, if you want other people to be able to view or edit your files, you can share the folder in MATLAB and control the file permissions. You can share a folder by inviting individual members to have access to the folder or by sharing a link to the folder.

To share a folder from MATLAB, MATLAB Drive Connector must be running and the folder you choose to share must be in your MATLAB Drive. If you are working in MATLAB Online on page 10-2, you do not need MATLAB Drive Connector to share folders since sharing is always enabled.

For more information about sharing with MATLAB Drive, including limitations, see the MATLAB Drive documentation.

Share a Folder

To share a folder by invitation, in the Current Folder browser, right-click the folder and select **Share > Manage Members**. You can select to share with view-only or can-edit permission for each person individually. To enter several addresses at one time, use a semi-colon or comma separator. You can only do this if they all have the same access level.

People you invite to a shared folder in this way will receive an email invitation. They must accept the invitation to have access to the shared folder. When inviting individual participants, you can make changes to each member's access at any time after the initial invite.

To share a folder with a view-only link, in the Current Folder browser, right-click the folder and select **Share > Manage Link**. Click **Copy Link** to copy the link to the clipboard.

In MATLAB Online, to share a folder that you have not shared before by invitation, right-click the folder and select **Share > Invite Members**. To share a folder by link, right-click the folder and select **Share > Create Link**.

Remove Access to Shared Folder

You can rescind invitations to a shared folder when you no longer want the content to be shared. To rescind an invitation, right-click the shared folder and select **Share > Manage Members**. Click the **X** next to each member name to delete shared access. Changing the access of invited members does not change access for people who joined using a shared link.

To remove access to a shared folder from a view-only link, right-click the folder and select **Share > Manage Link**. Then, Click **Delete Link**.

People who joined using the link will no longer have access to the shared folder, and anyone who then clicks the link gets an error. Deleting the link does not cause any change in access by members who received personal invitations.

Access Folders Shared with You

When you are personally invited to a shared folder, you will receive an email invitation. After you accept the invitation, you can view the shared folder in the Current Folder browser.

When you are provided with a link to a shared folder, click the link and an invitation shows up in MATLAB Drive online. When you accept the invitation, the shared folder appears in the Current Folder browser.

See Also

More About

- "Manage Files and Folders" on page 6-31
- "Restore Files in MATLAB Online" on page 10-4

Files and Folders that MATLAB Accesses

In this section...
"Where Does MATLAB Look for Files?" on page 6-36
"Files and Folders You Should Add to the Search Path" on page 6-36
"When Multiple Files Have the Same Name" on page 6-36
"Locations of MathWorks Products" on page 6-37

Where Does MATLAB Look for Files?

When you do not specify a path to a file, MATLAB looks for the file in the current folder or on the search path. Functions in the current folder take precedence over functions with the same file name that reside anywhere on the search path. To identify the current folder, type `pwd` in the Command Window.

To make files accessible to MATLAB, do one of the following:

- Change the current folder to the folder that contains the files. Use the `cd` function or browse to a different folder in the Current Folder toolbar:
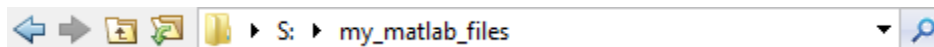
- Add the folder that contains the files to the search path. Changes you make to the search path apply to the current MATLAB session. To reuse the modified search path in future MATLAB sessions, save your changes.
- Store individual files in the *userpath* MATLAB folder, which is on the search path. To determine the location of this folder, run the `userpath` function.

Files and Folders You Should Add to the Search Path

The MATLAB search path should include:

- Folders containing files that you run.
- Folders containing files that are *called by* files you run.
- Subfolders containing files that you run. Making a folder accessible does not make its subfolders accessible.

For files in @ (class) and + (package) folders, make the parent folder accessible. For details, see "Folders Containing Class Definitions".

If files call other files that are in multiple folders, determine the location of all the called files by creating a Dependency Report. See "Dependencies Within a Folder"

When Multiple Files Have the Same Name

Name conflicts arise when MATLAB has access to multiple files with the same name, and when a file has the same name as a variable in the base workspace or a built-in function for a MathWorks product.

When there are name conflicts, MATLAB follows these precedence rules:

- "Function Precedence Order"
- "Class Precedence and MATLAB Path"

The file that MATLAB does *not* use is called a shadowed file. In some cases, MATLAB warns you that a shadowed file exists.

Locations of MathWorks Products

Files and folders for products provided by MathWorks are in *matlabroot*/toolbox. The files and folders under *matlabroot* are important to your installation. In particular:

- Do *not* store your personal files and folders in *matlabroot*/toolbox.
- Do *not* change files, folders, and subfolders in *matlabroot*/toolbox. The exception is the pathdef.m file, which you can update and save in its default location, *matlabroot*/toolbox/ local.

To see a list of all toolbox folder names supplied with MathWorks products, run:

```
dir(fullfile(matlabroot,'toolbox'))
```

See Also
userpath | cd | pwd

More About
- "What Is the MATLAB Search Path?" on page 6-45
- "MATLAB Startup Folder" on page 1-14
- "Toolbox Path Caching in MATLAB" on page 1-21

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**.

Some Current Folder Browser preferences are not available in MATLAB Online on page 10-2.

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.
	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	Use the Toolbar preferences to adjust the layout and controls of the Current Folder toolbar. For more information about how to access Toolbar preferences, see "Customize MATLAB Toolbars" on page 2-10.
Initial working folder	Use the **General Preferences** options in the Preferences Window to specify the current folder in MATLAB at startup. For more information about **General Preferences** options, see **Preferences Window**.

Preference	Usage
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.

See Also
Preferences Window

More About
- "Manage Files and Folders" on page 6-31
- "Customize MATLAB Toolbars" on page 2-10

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 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` or /users/myuserid/Homework/myfile.m. An absolute path name can start with any of the following:

- UNC path '\\'.

- Drive letter, on Microsoft Windows platforms, such as `C:\`.
- `'/'` character on Linux platforms.

Some MATLAB functions also support relative path names. Unless otherwise noted, the path name is relative to the current folder. For example:

- `myfile.m` refers to the `myfile.m` file in the current folder.
- `myfolder` refers to the `myfolder` folder in the current folder.
- `../myfolder/myfile.m` refers to the `myfile.m` file in the `myfolder` folder, where `myfolder` is at same level as the current folder. Each repetition of `../` at the beginning of the path moves up an additional folder level.

Tip If multiple documents are open and docked in the Editor, you can copy the absolute path of any of these documents to the clipboard. This practice is useful if you need to specify the absolute path in another MATLAB tool or an external application. Right-click the document tab, and then select **Copy Full Path to Clipboard**.

Partial Path Names in MATLAB

A partial path name is the last portion of a full path name for a location on the MATLAB search path. Some functions accept partial path names.

Examples of partial path names are: `matfun/trace`, `private/cancel`, and `demos/clown.mat`.

Use a partial path name to:

- Specify a location independent of where MATLAB is installed.
- Locate a function in a specific toolbox when multiple toolboxes contain functions with that name. For example, to open the file for the `set` function in the Database Toolbox™ product, type:

  `open database/set`
- Locate method files. For example, to check if a `plot` method exists for the time series object, type:

  `exist timeseries/plot`

  Specifying the at sign character (@) in method folder names is optional.
- Locate private and method files, which sometimes are hidden.

Be sure to specify enough of the path name to make the partial path name unique.

Maximum Length of Path Names in MATLAB

The maximum length allowed for a path name depends on your platform.

For example, on Microsoft Windows platforms:

- The maximum length is known as `MAX_PATH`.
- You cannot use an absolute path name that exceeds 260 characters.
- For a relative path name, you might need to use fewer than 260 characters. When the Windows operating system processes a relative path name, it can produce a longer absolute path name, possibly exceeding the maximum length.

If you get unexpected results when working with long path names, use absolute instead of relative path names. Alternatively, use shorter names for folders and files.

Case Sensitivity of File Names

In general, it is best to specify path and case precisely when specifying a file name.

Case Sensitivity When Calling Functions

You call function files by specifying the file name without the file extension. MATLAB returns an error if it cannot find a case-sensitive match on the search path. By default, MATLAB suggests a function with the correct case.

When multiple files have the same name, MATLAB follows precedence rules to determine which to call. For more information, see "Function Precedence Order".

Case Sensitivity When Loading and Saving Files

Linux platforms — File names are case sensitive.

- When loading or reading from a file, specify the file name using the correct case.
- When saving or writing to a file, MATLAB saves the file in the case you specify. Two files with the same name, but different cases can exist in the same folder.

Windows platforms — File names are case insensitive. The Windows operating system considers two files with the same name to be the same file, regardless of case. Therefore, you cannot have two file names that differ only by case in the same folder.

- When loading or reading from a file, MATLAB accesses the file with the specified name that is higher on the search path, regardless of case. For example, if you attempt to load `MYFILE` and `myfile.mat` is higher on the search path than `MYFILE.MAT`, then MATLAB loads `myfile.mat` without warning that there is a case mismatch.
- When saving or writing to a file, if you specify a file name that already exists in the folder, MATLAB accesses the existing file without warning. For example, if you save data to a file named `myfile` using the `save` function, and `MYFILE.mat` already exists in the folder, the data replaces the contents of `MYFILE.mat`. However, the file name remains `MYFILE.mat`.

See Also
`filesep` | `fullfile` | `which` | `ismac` | `ispc` | `isunix`

More About
-

Create and Extract from Zip Archives

In this section...

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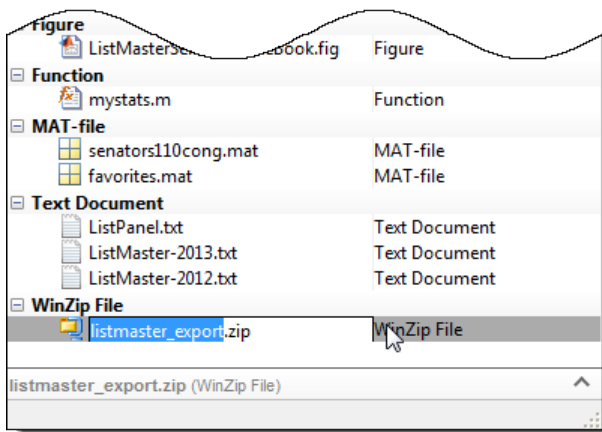
You also can create zip archives programmatically using the `zip` function. For example, to zip all files with a `.m` and `.mat` extension in the current folder to a zip file archive named `backup.zip`, call:

```
zip('backup',{'*.m','*.mat'});
```

Add Files to a Zip Archive

To add files and folders to a zip file archive in the Current Folder browser, do one of the following:

- Select, and then drag the file that you want to add onto the archive.

- Copy the file that you want to add to the archive. Then, select the archive to which you want to add the file and paste the file into the archive.

Extract Files from a Zip Archive

To extract a single file from within a zip file archive in the Current Folder browser:

1 Expand the zip file archive to view the archive contents, by clicking the associated + (expand) button. By default, files within a zip file archive appear dimmed to indicate that they are not on the MATLAB path.

2 Drag the file into a folder in the Current Folder browser.

MATLAB extracts the file and saves it to the folder where you dragged or pasted it.

To extract all the contents of a zip file, double-click the zip file in the Current Folder browser. MATLAB creates a folder with the same name as the zip file, and extracts the entire contents of the zip file into this folder.

To extract the contents of a zip file programmatically, use the `unzip` function. `unzip` also allows you to specify a target folder. For example, to unzip the file, `examples.zip`, to a folder named `myfolder`, call:

```
unzip('examples.zip','myfolder')
```

Note Archives created outside of MATLAB can be encrypted or password-protected. You cannot add files to, or extract files from, protected archives from within MATLAB.

Compare Zip Archive to Unzipped Files

To determine differences between archived and unarchived files, use the Comparison Tool from within the Current Folder browser as you would for any other files and folders.

- Right-click a zip archive, and then from the context menu select **Compare Against** and specify the folder to which you want to compare the contents of the zip archive.

- Expand a zip archive, right-click a file within it, and then from the context menu select **Compare Against**. Specify the file to which you want to compare the archived file.

See Also
`unzip` | `zip`

More About
- "Compare Files and Folders and Merge Files" on page 6-5

What Is the MATLAB Search Path?

The MATLAB search path is a subset of all the folders in the file system. MATLAB uses the search path to locate files used with MathWorks products efficiently.

The order of folders on the search path is important. When files with the same name appear in multiple folders on the search path, MATLAB uses the one found in the folder nearest to the top of the search path.

By default, the search path includes

- The MATLAB *userpath* folder, which is added to the search path at startup, and is the default location for storing user files
- The folders defined as part of the `MATLABPATH` environment variable
- The folders provided with MATLAB and other MathWorks products, which are under *matlabroot*/`toolbox`, where *matlabroot* is the folder displayed when you run `matlabroot` in the Command Window

Class, package, `private`, and `resources` folders are special folders that cannot be specified explicitly as part of the search path. A special folder is added to the search path implicitly when its parent folder is specified as part of the path. To access the files and folders within a special folder, you must add its parent folder to the path.

You can explicitly add folders to the search path for the files you run. For more information about adding files to the search path, see "Change Folders on Search Path" on page 6-49.

userpath Folder on the Search Path

The *userpath* folder is first on the search path. By default, MATLAB adds the *userpath* folder to the search path at startup. This folder is a convenient place for storing files that you use with MATLAB.

The default *userpath* folder is platform-specific.

- Windows platforms — `%USERPROFILE%/Documents/MATLAB`.
- Mac platforms — `$home/Documents/MATLAB`.
- Linux platforms — `$home/Documents/MATLAB` if `$home/Documents` exists.
- MATLAB Online — `/users/youruserid`.

Alternatively, to determine or change the current *userpath* folder, call `userpath`.

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

MATLABPATH Environment Variable

The `MATLABPATH` environment variable can contain a list of additional folders to be added to the MATLAB search path at startup. These folders are placed after the *userpath* folder, but ahead of the folders supplied by MathWorks. By default, the `MATLABPATH` environment variable is not set. For more information on how to set the `MATLABPATH` environment variable, see "Set the MATLABPATH Environment Variable" on page 6-54.

Determine If Files and Folders Are on the Search Path

There are several ways to determine if files and folders are on the search path. You can either check whether an individual file or folder is on the search path, or you can view the entire search path.

Check Whether File or Folder on Search Path

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

To determine whether a file or folder is on the search path, use the Current Folder browser. Files and folders not on the path are dimmed. To ensure that the Current Folder browser is set to indicate whether a file or folder in on the search path, right-click any file or folder, and select the **Indicate Files Not on Path** option.

Hover the pointer over any dimmed file or folder in the Current Folder browser to find out why it is dimmed. A tooltip opens with an explanation. Frequently, the tooltip indicates that the file or folder is not on the MATLAB path. If a tooltip does not appear, it may be disabled. To enable it, go the **Home** tab and in the **Environment** section, click ⚙ **Preferences**. Then, select **MATLAB > Current Folder**. Select **Show tooltip explaining why files are inaccessible** to display the tooltip.

View Entire Search Path

Run the `path` command to view all the folders on the MATLAB search path.

Alternatively, use the Set Path dialog box to view the entire MATLAB search path. On the **Home** tab, in the **Environment** section, click **Set Path**. The Set Path dialog box opens, listing all folders on the search path. For more information on using the Set Path dialog box, see "Change Folders on Search Path" on page 6-49.
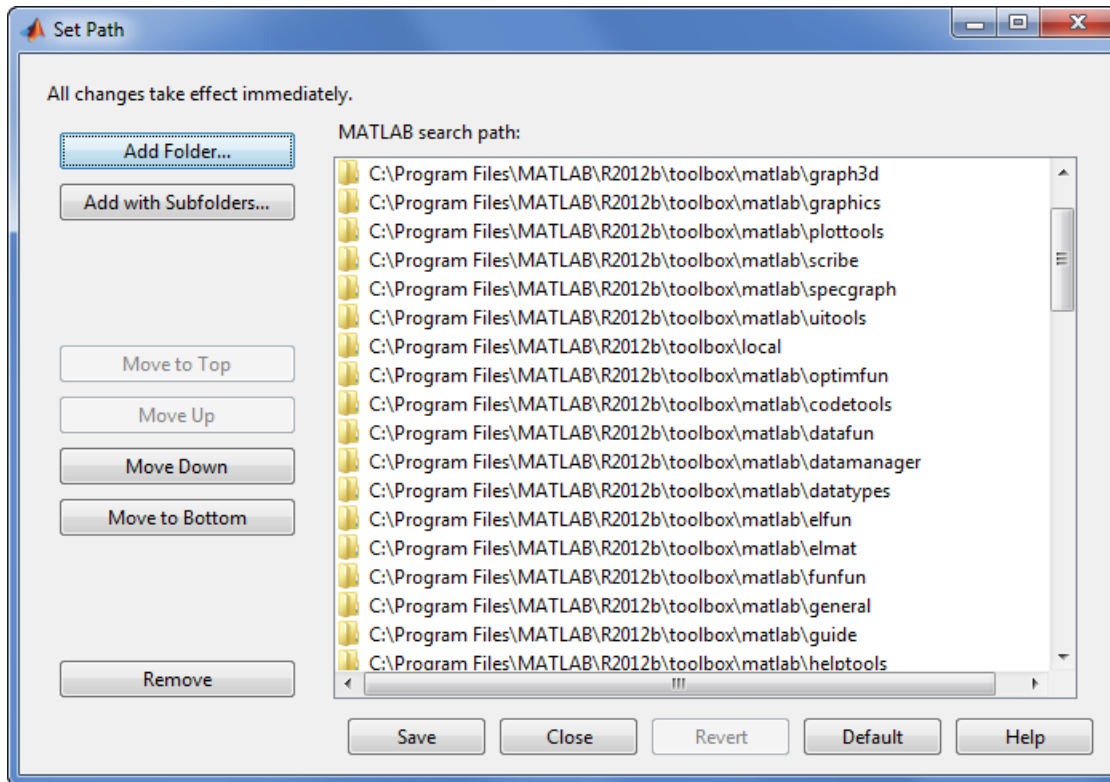
The Search Path Is Not the System Path

The search path is *not* the same as the system path. Furthermore, there is no explicit relationship between the MATLAB search path and the system path. However, both paths help in locating files, as follows:

- MATLAB uses the search path to locate MATLAB files efficiently.
- The operating system uses a system path to locate operating system files efficiently.

How MATLAB Stores the Search Path

MATLAB saves search path information in the `pathdef.m` file. This file contains a series of full path names, one for each folder on the search path.

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

When you change the search path, MATLAB uses it in the current session, but does not update `pathdef.m`. To use the modified search path in the current and future sessions, save the changes using `savepath` or the **Save** button in the Set Path dialog box. This updates `pathdef.m`.

In MATLAB Online, all changes to the search path are automatically saved.

See Also
`userpath`

More About

Change Folders on Search Path

In this section...

You can programmatically and interactively add or remove folders and change the order of MATLAB folders on the search path, for the current session and future sessions. When files with the same name appear in multiple folders on the search path, MATLAB uses the one in the folder nearest to the top of the search path.

Add or Remove Folders on the Search Path Programmatically

You can programmatically add one or more folders on the path using the `addpath` function. This example adds the folder to the top of the search path. See `addpath` for other options.

```
addpath("c:\matlab\MyFolder")
```

You can remove one or more folders from the path using the `rmpath` function.

```
rmpath("c:\matlab\MyFolder")
```

To save changes to the search path across MATLAB sessions, use the `savepath` function. This function also saves changes that you make interactively with the Set Path dialog box, from the Current Folder browser, or in the MATLAB Editor.

In MATLAB Online, changes to the path are automatically saved. Therefore, calling `savepath` is not necessary.

Change Folders on Search Path Interactively

Use the Set Path dialog box to interactively make changes to the search path.

1   On the **Home** tab, in the **Environment** section, click **Set Path**. The Set Path dialog box appears.

Alternatively, you can access this dialog box using the `pathtool` function.
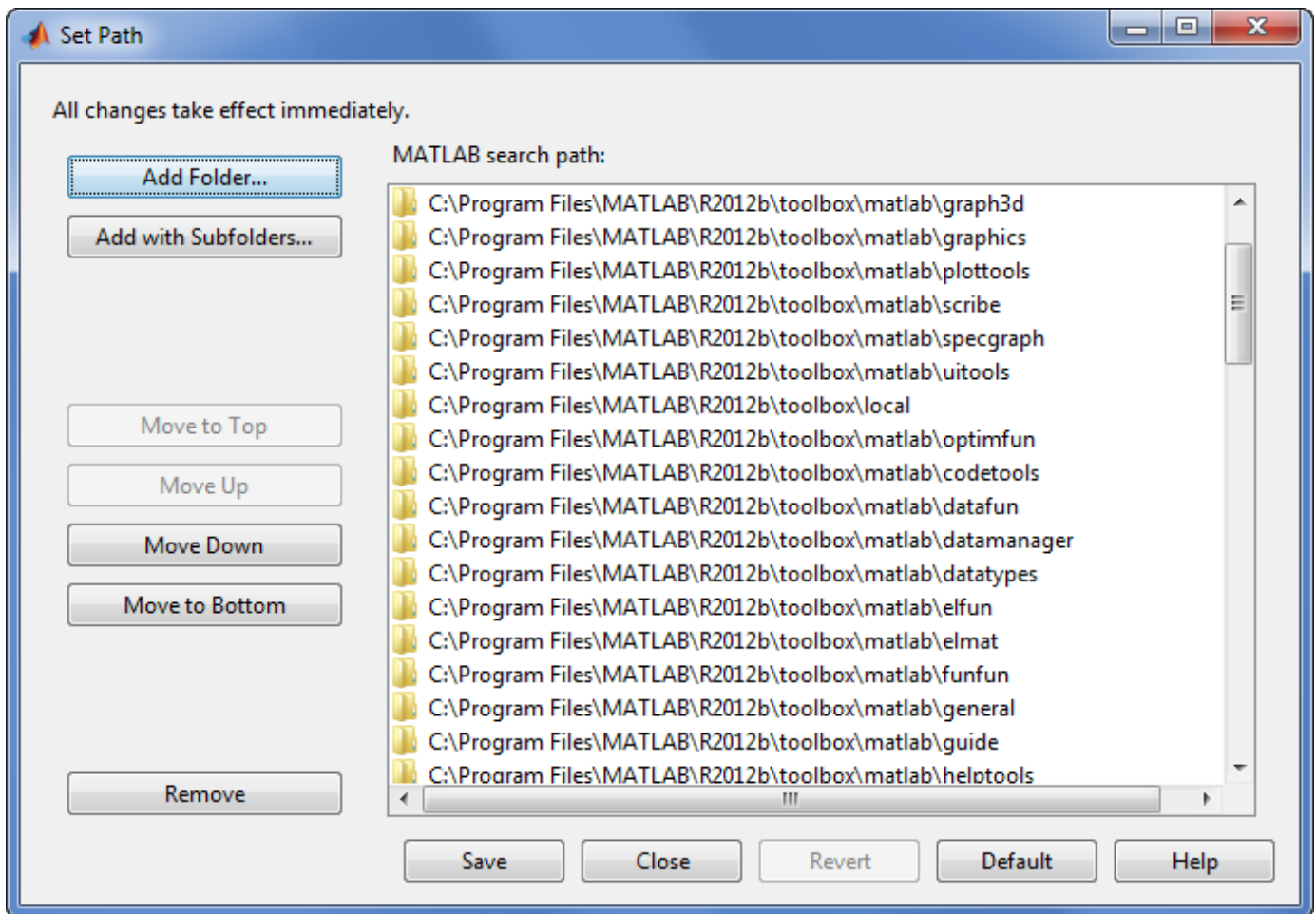
2   Use the **Add Folder** or **Add Folder with Subfolders** button to add new folders to MATLAB search path.

3   Use the **Move Down** and **Move Up** buttons to change the order of files on the search path. Files contained in folders at the top of the search path have precedence over those in folders farther down. For more information, see "Function Precedence Order".

4   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 6-45.

Change Folders Using Current Folder Browser

you can add or remove folders on the search path from the Current Folder browser. Folders that are not on the path appear transparent, while those on the path appear solid.

1   From the Current Folder browser, right-click the folder or select and then right-click multiple 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**
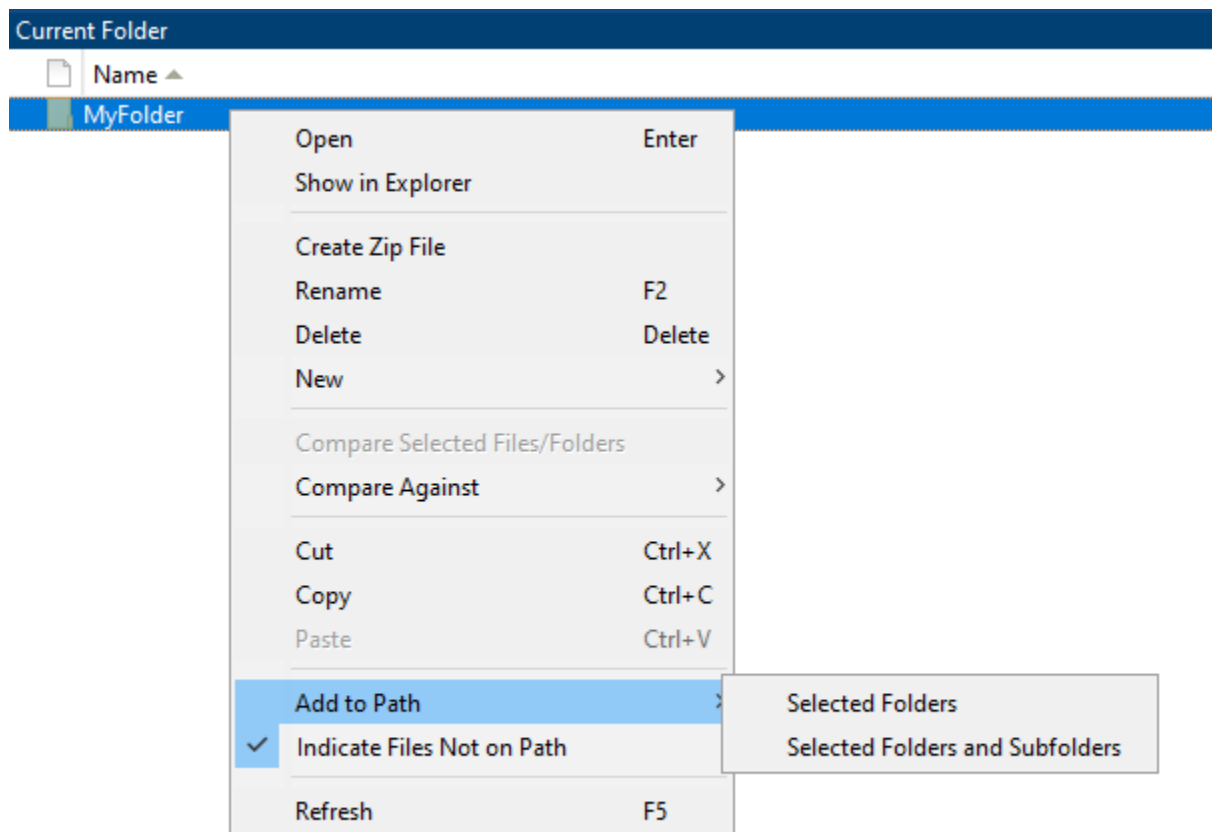
Changes to the search path using this method do not persist in future MATLAB sessions. To save these changes, use `savepath`.

Change Folders on Search Path Using MATLAB Editor

In the MATLAB Editor, you also can add or remove the folder that contains a script on the search path. Right-click the document tab, and then select an option to add or remove the folder from the search path.
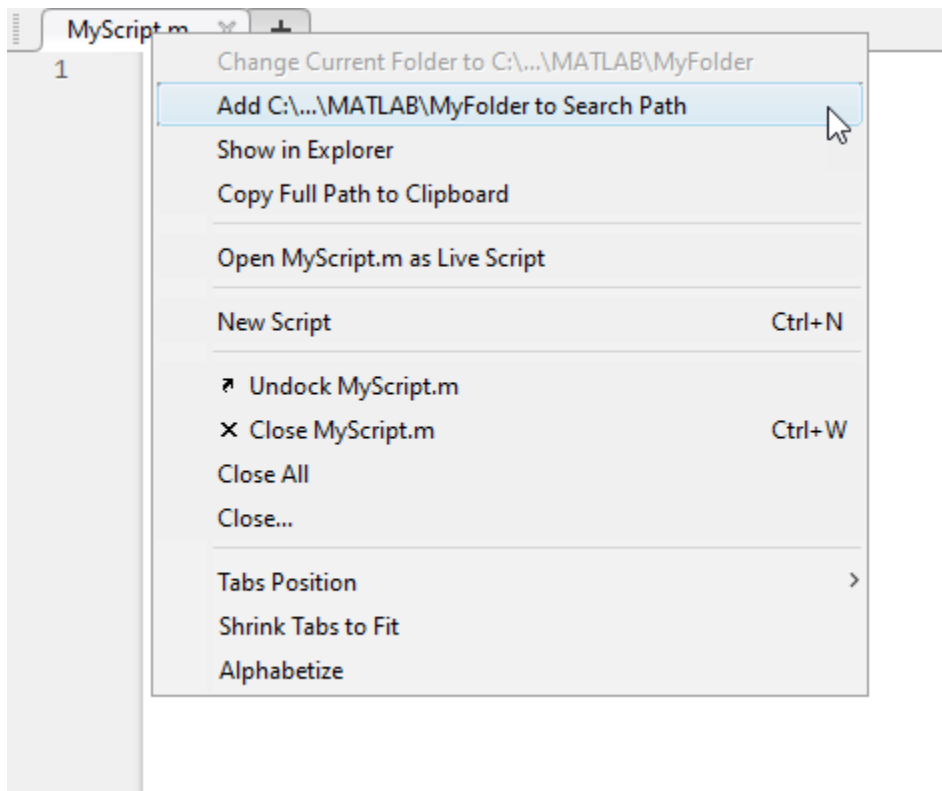
Changes to the search path using this method do not persist in future MATLAB sessions. To save these changes, use `savepath`.

See Also
addpath | rmpath | savepath

More About
*   "What Is the MATLAB Search Path?" on page 6-45

Use Search Path with Different MATLAB Installations

The default search path changes for each MATLAB version because the default folders that come with the products change. Different MATLAB versions cannot use the same `pathdef.m` file.

To use your files with a new MATLAB version or with multiple versions, do one of the following:

- For each version, add the folders containing your files to the search path. Save the search path (that is, save the `pathdef.m` file) where that version of MATLAB can access it.
- Include `addpath` statements in the `startup.m` file. Use the same `startup.m` file with the multiple versions of MATLAB.

Including `addpath` statements in the `startup.m` file also allows you to use your files with MATLAB on different platforms.

See Also
addpath

More About
- "Startup Options in MATLAB Startup File" on page 1-19

6-53

Add Folders to the MATLAB Search Path at Startup

There are two ways to add folders to the MATLAB search path at startup. You can either use a `startup.m` file or you can set the MATLABPATH environment variable.

Use a startup.m File

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

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

For more information on creating a `startup.m` file with `addpath` statements, see "Startup Options in MATLAB Startup File" on page 1-19.

Set the MATLABPATH Environment Variable

You can also add folders to the search path at startup by setting the MATLABPATH environment variable. :

Windows

To set the MATLABPATH environment variable in Windows, from the Windows **Control Panel**, go to **System** and select **Advanced system settings**. Click the **Environment Variables...** button. Click **New...** or **Edit...** to create or edit the MATLABPATH environment variable. In the dialog box that appears, set the variable name to MATLABPATH and the variable value to a semicolon-separated list of folders you want to add to the search path. For example, to add two folders, `c:\matlab_files\myfolder1` and `c:\matlab_files\myfolder2`, to the MATLABPATH environment variable, enter `c:\matlab_files\myfolder1;c:\matlab_files\myfolder2` as the variable value. Click **OK** to set the variable and exit the dialog box. Restart MATLAB for the new settings to take effect.

To set the environment variable from a command window, run the command `set MATLABPATH=`*folders*, where *folders* is a semicolon-separated list of folders. For example, suppose that you want to add two folders, `c:\matlab_files\myfolder1` and `c:\matlab_files\myfolder2`, to the MATLABPATH environment variable. Run the command

```
set MATLABPATH=c:\matlab_files\myfolder1;c:\matlab_files\myfolder2
```

Once the environment variable is set, you must start MATLAB from the same command window for the settings to take effect. The environment variable persists only as long as the command window is open.

UNIX and Mac

To set the MATLABPATH environment variable in UNIX and Mac, in a terminal, run the command `export MATLABPATH=`*folders*, where *folders* is a colon-separated list of folders.

For example, suppose that you want to add two folders, `/home/j/Documents/MATLAB/mine` and `/home/j/Documents/MATLAB/research`, to the MATLABPATH environment variable on a UNIX platform. Run the command

```
export MATLABPATH=/home/j/Documents/MATLAB/mine:/home/j/Documents/MATLAB/research
```

Once the environment variable is set, you must start MATLAB from the same shell for the settings to take effect. The environment variable persists only as long as the shell remains open.

Note If you are using a C shell (`csh` or `tcsh`), the command for setting the `MATLABPATH` environment variable is `setenv MATLABPATH` *`folders`*.

To add the folders for all future MATLAB sessions, set the `MATLABPATH` environment variable as part of your shell configuration script.

See Also

More About

- "What Is the MATLAB Search Path?" on page 6-45
- "Change Folders on Search Path" on page 6-49

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 Search Path" on page 6-49. 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 1-14.

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.

Troubleshoot Invalid or Unresponsive Windows Change Notification Handles

MATLAB uses a Windows operating system feature called change notification handles to detect when files in an associated folder are modified. Under certain circumstances, Windows fails to provide MATLAB with a valid or responsive Change Notification Handle. The three most common causes for invalid or unresponsive change notification handles are:

- Windows has exhausted its supply of notification handles.
- The specified folder resides on a file system that does not support change notification. (Syntax TAS file server, SAMBA file server, and many NFS file servers are known to have this limitation.)
- Network or file server latency delays the arrival of the change notification so that changes are not detected on a timely basis.

When MATLAB is unable to obtain a valid or responsive Change Notification Handle, it cannot automatically detect changes to files and folders. For example, new functions added to an affected folder might not be visible, and changed functions in memory might not be reloaded.

Control How MATLAB Detects Changes in Files and Folders

If MATLAB is unable to detect changes to your files and folders due to invalid or unresponsive change notification handles, you can control how MATLAB detects changes in folders.

If your file system updates folder timestamps when files are added to folders, you can set your system remote path policy to detect changes by testing the timestamps of folders. To do so, include these commands in a `startup.m` file. For more information on creating a `startup.m` file, see "Startup Options in MATLAB Startup File" on page 1-19.

```
system_dependent('RemotePathPolicy', 'TimecheckDir');
rehash path;
```

You might notice a performance degradation due to the time required to check the timestamps.

If your file system does not update folder timestamps (such as an NT file system), you can set your system remote path policy to detect changes by rereading the affected folders at frequent intervals. To do so, include these commands in a `startup.m` file.

```
system_dependent('RemotePathPolicy', 'Reload');
rehash path;
```

You might notice a significant performance degradation due to the time required to reread the folders.

To determine how MATLAB detects changes in folders, use this command:

```
system_dependent('RemotePathPolicy', 'Status');
```

In some cases, detecting new files or changed files in folders affected by the invalid or unresponsive change notification handles is not necessary. If detecting changes is not necessary, to provide maximum performance, you can disable your system remote path policy. To do so, include these commands in a `startup.m` file.

```
system_dependent('RemotePathPolicy', 'None');
rehash path;
```

Clear Functions from Memory

Sometimes, changing your system remote path policy is not effective in preventing problems related to remote file system caching or network latency. If MATLAB is still unable to detect the changes that you have made to a function, you can clear the old copy of the function from memory using the `clear` function. MATLAB reads the updated function the next time it is called.

Configure Change Notification Handle Warnings

To analyze which folders are affected by invalid or unresponsive Windows change notification handles, you can configure the number of warning messages issued by Windows. To do so, include one of these commands in a `startup.m` file. For more information on creating a `startup.m` file, see "Startup Options in MATLAB Startup File" on page 1-19.

- Show warning messages for all invalid or unresponsive change notification handles.

  ```
  system_dependent('DirChangeHandleWarn', 'Always');
  ```

- Only show the first warning message about invalid or unresponsive change notification handles.

  ```
  system_dependent('DirChangeHandleWarn', 'Once');
  ```

- Never show warning messages about invalid or unresponsive change notification handles.

  ```
  system_dependent('DirChangeHandleWarn', 'Never');
  ```

To determine when Windows warns about invalid or unresponsive change notification handles, use this command.

```
system_dependent('DirChangeHandleWarn', 'Status');
```

See Also
addpath | rehash

More About
- "What Is the MATLAB Search Path?" on page 6-45
- "Startup Options in MATLAB Startup File" on page 1-19

Editor Preferences

Editor/Debugger Preferences

In this section...

You can customize the visual display and functionality of the Editor, Live Editor, and Debugger using Editor/Debugger preferences.

General Preferences for the Editor/Debugger

You can specify which editor MATLAB uses, as well as how the MATLAB Editor and Live 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.

Some general preferences are not available in MATLAB Online.

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`. This preference only applies to plain text files such as files with a `.m` or `.txt` extension.
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 reopen files from previous MATLAB sessions** if you want the Editor and the files it contained during your last MATLAB session to reopen when you restart MATLAB.
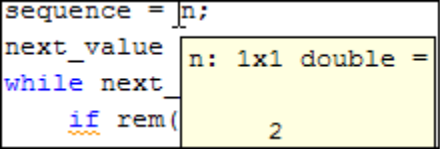
Preference	Usage
	Select **Automatically open files when MATLAB reaches a breakpoint** to open a running program file when MATLAB encounters a breakpoint in that file.
Automatic file changes	Select **Save changes upon clicking away from a file** if you want the Editor to automatically save changes to 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. This preference only applies to plain text files such as files with a `.m` or `.txt` extension.
Debugging in the Live Editor	Select **Only show Step in button for user-defined functions** to have the Live Editor only step into user-defined functions and scripts. To have the Live Editor step into MathWorks functions as well, clear the option.

Editor/Debugger Display Preferences

You can change the appearance of the Editor and Live 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). This preference does not apply in the Live Editor.
	Select **Show line numbers** to display line numbers along the left edge of the Editor and Live Editor window.

Preference	Usage

	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.)
	
	 `sequence = n;`
	 `next_value` `n: 1x1 double =`
	 `while next_`
	 `if rem(` `2`
	
	 For details, see "View Variable Value".
	
	 This preference does not apply in the Live Editor.

	Select an option from **Live Editor default view** to specify the default location of output when creating new live scripts: • **Output on right** — Output displays to the right of the code. Each output displays with the line that creates it. This option is ideal when writing code. To disable the alignment of output to code when output is on the right, right-click the output section and select **Disable Synchronous Scrolling**. • **Output inline** — Output displays inline with the code. Each output displays underneath the line that creates it. This option is ideal for sharing.
	Select **Display the Open as a Live Script banner** if you want the Open as a Live Script banner to appear at the top of documents in the Editor.
Right-hand text limit	Select **Show line** to display a vertical line with the specified **Color** at the specified column (**Placement**) in the Editor. For details, see "Change the Right-Side Text Limit Indicator". This preference does not apply in the Live Editor.

Editor/Debugger Tab Preferences

You can specify the size of tabs and indents and details about how tabs behave in the Editor and Live 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. 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 automatic indenting. To set automatic indenting preferences, see "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 automatic indenting preferences when you position the cursor in a line or select a group of lines, and then press the **Tab** key. To set automatic indenting preferences, see "Editor/Debugger Language Preferences" on page 7-5. If you select this preference, you cannot insert tabs within a line.
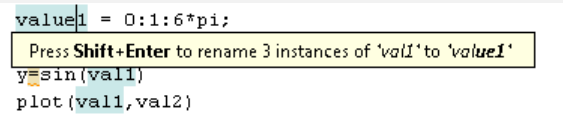
Editor/Debugger Language Preferences

You can specify how various languages appear in the Editor and Live Editor. MATLAB applies language preferences based on the file extension of the file that is open.

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. In MATLAB Online, to set language preferences, select **Editor/Debugger > MATLAB Language** or **Editor/Debugger > Other Languages**.

Not all preferences are available for all languages.

Preference	Usage
Language	Select the language for which you want to set preferences.

Preference	Usage
Syntax highlighting	Select **Enable syntax highlighting** to have the Editor and Live Editor use different colors for different language constructs. Then, adjust the colors you want to use for each language element. To change the MATLAB syntax highlighting colors, click **Set syntax colors**. For more information, see "Syntax Highlighting" on page 3-19. To change syntax highlighting colors for all other languages, select from the options under **Enable 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. For details on when MATLAB prompts you, see "Automatically Rename All Variables or Functions in a File".
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 or Live Editor document. If you clear this option, you can still wrap comments manually, as described in "Add Comments to Code".

Preference	Usage
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. You also can manually apply automatic indenting after you type the code. For more information, see "Indent Code".
	Select an option from **Function Indenting Format** (MATLAB Language only) to specify how functions indent in the Editor and Live Editor, as follows: • **Classic** — The Editor and Live Editor align the function code with the function declaration. • **Indent nested functions** — The Editor and Live Editor indent the function code within a nested function. • **Indent all functions** — The Editor and Live Editor indent the function code for both main and nested functions. For more information and examples of each indenting format, see "Indent 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 "Fold Code".

Code folding preferences do not apply in the Live Editor.

Option	Usage
Enable Code Folding	Specifies whether you want code folding enabled for the programming constructs that have their corresponding **Enable** check box selected.

Option	Usage
Enable	Specifies whether you want code folding enabled for the corresponding **Programming Construct**. 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.

Backup files preferences do not apply in the Live Editor and are not available in MATLAB Online.

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 the backup 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).
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. To ensure clear and current backup-to-file relationships if you disable this option, when you rename or remove a file, consider deleting or renaming the corresponding backup file.
File name	Select the naming convention that you want MATLAB to use for autosave files. For example: • If you specify **Replace extension with: 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~`

Preference	Usage
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. If you edit a file in a read-only folder and the back up **Location** is **Source file directories**, then the Editor does not create a backup copy of the file.

For more information about automatically creating backup copies of modified files in the Editor, see "Save and Back Up Code".

Editor/Debugger Autoformatting Preferences

You can quickly format live scripts and functions using autoformatting markup in the Live Editor. You can enable and disable autoformatting as a whole, or as individual options.

On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **Editor/Debugger > Autoformatting**, and then adjust preference options as described in this table.

Autoformatting preferences do not apply in the Editor.

Preference	Usage
Enable autoformatting in Live Editor	Select to enable autoformatting options in the Live Editor. Once enabled, options can then be enabled or disabled individually.
Text format	Select ***italic*, **bold**** to enable bold and italic formatting using asterisks (*).
	Select **_italic_, __bold__** to enable bold and italic formatting using underscores (_).
	Select **`monospace`, |monospace|** to enable monospace autoformatting.
Section break	Select **%%, ***, or --- for section break** to enable inserting section breaks by entering %%, ***, or - - - and then **Enter**.
	Select **%% text for section break and heading** to enable inserting section breaks with heading 1 style headings by entering %%text and then **Enter**.
Insert Options	Select **$LaTeX$ for LaTeX equation** to enable converting LaTeX expressions into equations using the format $LaTeX$.
	Select **URL for hyperlink** to enable converting internet paths automatically to hyperlinks.

Preference	Usage
	Select **<URL> for hyperlink** to enable converting internet paths to hyperlinks using the format <URL>.
	Select **[Label](URL) for labeled hyperlink** to enable converting internet paths to labeled hyperlinks using the format [Label](URL).
	Select **(TM), (SM), (R), and (C) for trademark, service mark, and copyright symbols** to enable inserting trademark and copyright symbols (™, ℠, ®, and ©) using the format (TM), (SM), (R), and (C).
Text Style	Select **#text for title** to enable inserting titles using the format #text.
	Select **##text, ###text, or ####text for heading** to enable inserting headings using the format ##text for heading 1, ###text for heading 2, or ####text for heading 3.
	Select **Automatic bulleted lists (*, +, or -)** to enable creating bulleted lists by entering *, +, or - followed by a space.
	Select **Automatic numbered lists (1., 2., etc.)** to enable creating numbered lists by entering 1., 2., and so on, followed by a space.

For more information about autoformatting in the Live Editor, see "Autoformatting".

Editor/Debugger Automatic Completions Preferences

You can change how MATLAB suggests and completes names in the Editor, Live Editor, and in App Designer. You also can change whether MATLAB autocompletes block endings, parentheses, quotes, comments, character vectors, and strings in the Editor, Live Editor, and in App Designer.

On the **Home** tab, in the **Environment** section, click ⚙ **Preferences**. Select **Editor/Debugger > Automatic Completions**, and then adjust the preference options as described in this table.

Preference	Usage
Suggestions and completions	Select **Show automatically** to have MATLAB automatically suggest and complete names as you type in the Editor, Live Editor, and in App Designer. Select **Show on tab** to have MATLAB suggest and complete names only after you press the **Tab** key. For more information, see "Code Suggestions and Completions" on page 3-20.
Autocoding options	Select **Enable autocoding** to enable autocoding options in the Editor, Live Editor, and in App Designer. Once enabled, options can be enabled or disabled individually.

Preference	Usage
	Autocomplete pairs • Select **Parentheses ( )** to enable autocompletion of parentheses. • Select **Curly braces { }** to enable autocompletion of curly braces. • Select **Square brackets [ ]** to enable autocompletion of square brackets. • Select **Single quotes (' ')** to enable autocompletion of single quotes. • Select **Double quotes (" ")** to enable autocompletion of double quotes.

Preference	Usage
	Autocomplete when splitting
	• Select **Comments (%)** to enable autocompletion of comments when split across two lines. For example, place your cursor after the first period in the comment below and press **Enter**.
	`% This is my first comment. This is my second comment.`
	The comment splits into two lines.
	`% This is my first comment.` `% This is my second comment.`
	• Select **Character vectors (' ')** to enable autocompletion of character vectors when split across two lines. For example, place your cursor after the word `Random` in the character vector below and press **Enter**.
	`title('Mean of Random Uniform Data')`
	The character vector splits into two character vectors across two lines.
	`title(['Mean of Random' ...` `    ' Uniform Data'])`
	• Select **Strings (" ")** to enable autocompletion of strings when split across two lines. For example, place your cursor after the word `long` in the string below and press **Enter**.
	`exist("exceptionally_long_file_name.txt")`
	The string splits into two strings on two lines.
	`exist("exceptionally_long" + ...` `    "_file_name.txt")`
	• Select **Parentheses ()** to enable autocompletion of parentheses when split across two lines. For example, place your cursor after the number 10 and the comma in the command below and press **Enter**.
	`plot(x,y,'--gs','LineWidth',2,'MarkerSize',10,'MarkerE`
	The command splits into two lines.
	`plot(x,y,'--gs','LineWidth',2,'MarkerSize',10, ...` `    'MarkerEdgeColor','b')`

Preference	Usage
	Autocomplete block endings • Select **Control flow statements (if, for, etc.)** to enable autocompletion of control flow statements. To autocomplete a statement, enter the statement and press **Enter**. For a list of control flow statements, see "Loops and Conditional Statements". • Select **Function and class definitions** to enable autocompletion of functions and class definitions. To autocomplete a definition, enter either `function` or `class` and press **Enter**.

Code Analyzer Preferences

In this section...
"Code Analyzer Preferences" on page 7-14
"Search for Messages in the Code Analyzer Preferences" on page 7-15
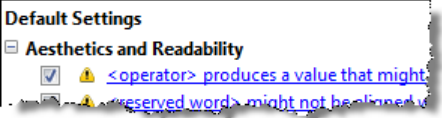
Code Analyzer Preferences

You can change how Code Analyzer messages appear in the Editor and Live Editor. With a few exceptions, these preferences apply to messages in the Editor, Live Editor, MATLAB Function Block Editor (if your products use that tool), and 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 "Check Code for Errors and Warnings Using the 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.
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.

Option	Usage
Search field Search Code Analyzer warning/error messages 🔍 ▾	Searches the list of Code Analyzer messages that display below the search field. For details, see "Search for Messages in the Code Analyzer Preferences" on page 7-15.

Code Analyzer message settings
 	
Default Settings
⊟ Aesthetics and Readability
   ☑ ⚠ <operator> produces a value that might
   ☑ ⚠ <reserved word> might not be aligned v
	Select or clear messages to enable or suppress their appearance in your Editor documents.
	
	To expand or collapse all categories in the list, right-click anywhere in the list and select **Expand All** or **Collapse All**.
	
	To suppress a message on a line-by-line or file-by-file basis, see "Adjust Code Analyzer Message Indicators and Messages".

Search for Messages in the Code Analyzer Preferences

You can search the list of Code Analyzer messages in the Code Analyzer preferences 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 specified text in the: • Short message • Extended message • Message category • Message ID In MATLAB Online, the extended message is not included in the search.	Type the text in the search field.	You recall seeing a message containing some text that 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 text in the short message, extended message, or message ID.

To See a List of Messages ...	Perform this action...	Example Scenario
Corresponding to a given message ID	Type `msgid:` followed by a space and 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 some text or a given 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.
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.

To See a List of Messages ...	Perform this action...	Example Scenario
Are errors	Click the down arrow to the right of the search field, and then select **Show All Errors**. By default, an X in a red dot indicates an error message, ⊗.	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`.
Are disabled	Click the down arrow to the right of the search field, and then select **Show Disabled Messages**.	You want to see the messages that are disabled by default or you have previously disabled.

Example of Searching Messages

To display Code Analyzer error messages that contain the word `variable` and are disabled:

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

   The search field contains `severity:error`.

2. At the end of the text `severity:error`, press the **Space** key, and then type `variable`.

3. Click the down arrow in the search field and select **Show Disabled Messages**.

The search field now contains `severity:error variable enabled:false`. Only the messages that fulfill those requirements appear in the Preferences pane. To restore the list of all messages, click the clear search button ⊠.

The **Show All Errors** and **Show Disabled Messages** options are not available in MATLAB Online.

See Also

More About

- "Check Code for Errors and Warnings Using the Code Analyzer"

Add-Ons

Get and Manage Add-Ons

In this section...
"Get Add-Ons" on page 8-2
"Install Add-Ons from File" on page 8-3
"Default Add-On Installation Folder" on page 8-3
"Manage Add-Ons" on page 8-4
"Update Add-Ons" on page 8-4
"Run Apps" on page 8-4

Add-ons encompass a wide variety of resources, including products, apps, support packages, and toolboxes. They extend the capabilities of MATLAB with functionality for additional tasks and applications.

Get Add-Ons

To find and install add-ons, go to the **Home** tab and, in the **Environment** section, click the **Add-Ons** icon. The Add-On Explorer opens and displays the list of available add-ons.
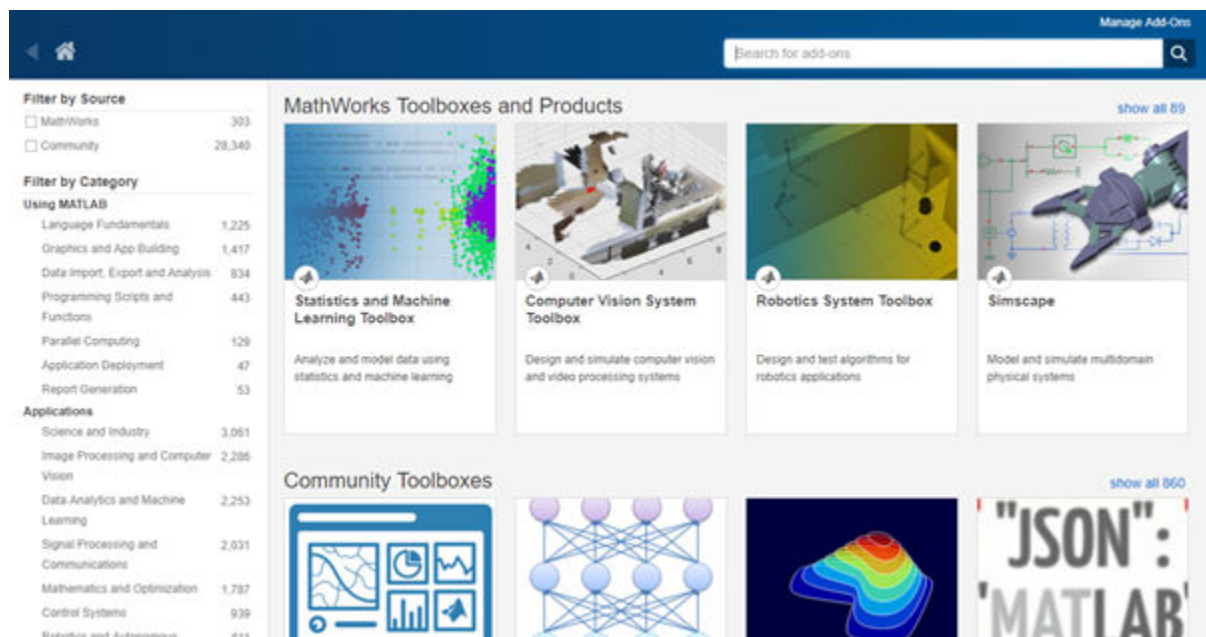
To find add-ons, use the search bar, or browse through the available categories on the left side of the Add-On Explorer window. Select an add-on from the list to view additional information about it, such as what files it includes and what documentation is available

To install an add-on, select it from the list. Then, select from the available install actions. After you install an add-on, MATLAB manages the MATLAB path for you. This lets you start using it without adjusting your desktop environment.

For some add-ons, you also can download the files from the Add-on Explorer without installing them. This is useful if you want to install the add-on on a system without an internet connection. To download the add-on without installing, select **Install > Download Only**.

Install Add-Ons from File

You can install some add-ons manually using an installation file. This is useful in several situations:

- The add-on is not available for installation through the Add-On Explorer, for example, if you create a custom add-on yourself or receive one from someone else.
- You downloaded the add-on from the Add-On Explorer without installing it.
- You downloaded the add-on from the File Exchange at MATLAB Central™.

Valid add-on installation files include `.mltbx` files (for toolboxes), `.mlappinstall` files (for apps), and `.mlpkginstall` files (for hardware support packages). To install an add-on from one of these files, in the MATLAB Current Folder browser, double-click the file. An installer opens to guide you through the installation process.

Note Make sure to double-click the installation file in the MATLAB Current Folder browser. Double-clicking to install the add-on from the system file browser is not supported on all operating systems.

You also can install some add-ons programmatically. For more information, see `matlab.addons.install`.

Default Add-On Installation Folder

The install location of an add-on depends on its type. MATLAB installs MathWorks products in the `matlabroot` folder and hardware support packages in the folder returned by `matlabshared.supportpkg.getSupportPackageRoot`. For more information about a specific support package install location, see the documentation for the package.

MATLAB installs all other add-ons in a default installation folder specific to the platform.

- Windows— `C:\Users\`*username*`\AppData\Roaming\MathWorks\MATLAB Add-Ons`
- Linux — `~/MATLAB Add-Ons`.
- macOS — `~/Library/Application Support/MathWorks/MATLAB Add-Ons`
- MATLAB Online — `/MATLAB Add-Ons`

To change the default installation folder, on the **Home** tab, in the **Environment** section, click **Preferences > MATLAB > Add-Ons**. Then, in the **Installation Folder** field, specify a folder name to which you have write access.

If you change the default installation folder, add-ons installed in the previously selected folder are no longer accessible from within MATLAB. Changing the default installation folder or customizing installed add-ons is not supported in MATLAB Online.

Manage Add-Ons

To view and manage installed add-ons, go to the **Home** tab and select 🔷 **Add-Ons > Manage Add-Ons**. The Add-On Manager displays all add-ons that are installed in the add-ons installation folder, as well as MATLAB products and hardware support packages.

To manage an individual add-on, click the options button ⋮ to the right of the add-on and select from the available actions:

- **Open Folder**
- **Open Documentation**
- **View in Add-on Explorer**
- **View Getting Started Guide**
- **Setup**
- **Version**
- **Enabled**
- **Uninstall**

Not all actions are available for all add-ons.

To customize your MATLAB environment, use the **Enabled** action to enable and disable add-ons. You also can enable or disable add-ons using the `matlab.addons.enableAddon` and `matlab.addons.disableAddon` functions.

If an add-on has multiple versions installed, only one version can be enabled. To view the currently enabled version, select **Version**. MATLAB displays a check mark next to the currently enabled version. To enable a different version, click it. For more information about installing multiple versions of an add-on, see `matlab.addons.install`.

Update Add-Ons

You can update installed add-ons as well as your current installed version of MATLAB using the Add-On Manager. On the **Home** tab, click **Help > Check for Updates**. The Add-On Manager opens with the **Updates** tab selected. View and install any available updates.

Run Apps

An app is a self-contained MATLAB program with a user interface. To view and interact with the user interface after installing the app, run it. Go to the **Apps** tab and expand the apps gallery by clicking the down arrow ▾ on the far right. Then, navigate to your installed app and click the icon. Custom apps that you write or receive from someone else appear in the **My Apps** section. You can run multiple custom apps concurrently, including multiple instances of the same app.

See Also
`matlab.addons.install` | `matlab.addons.installedAddons` | `matlab.addons.isAddonEnabled` | `matlab.addons.enableAddon`

Related Examples

- "Package Apps From the MATLAB Toolstrip"
- "Create and Share Toolboxes"

9

Internationalization

Locale Setting Concepts for Internationalization

In this section...
"Default Locale Setting" on page 9-2
"Supported Character Sets and Encodings" on page 9-2
"Platform-Specific Localized Formats for Current Folder Browser" on page 9-3

The locale setting defines the language of your user interface and the display formats for information like time, date, and currency. MATLAB uses the user-specified locale on all platforms.

If MATLAB does not correctly display characters in your language, you might have a locale setting problem. Locale is composed of individual settings that you can control. Each platform uses different parameters to specify the locale setting. These terms are relevant to understanding locale settings:

- locale — Format: *language_territory.codeset*

  For example, for the U.S. English locale setting en_US.UTF-8, en means that the display language is English. US indicates that time and date displays use U.S. conventions. UTF-8 is the character encoding (codeset) used to display text.

- ASCII (American Standard Code for Information Interchange) — Nearly universal character encoding for US English and control characters.

- Unicode — A universal character set that "provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language."

- UTF-8 — A variable width character encoding tor Unicode that is backwards compatible with ASCII. It has become the dominate encoding for the World Wide Web and anywhere else that cross-platform, cross-locale interoperability is required.

- user locale setting — Setting on your computer that specifies the locale that you want to use when running MATLAB. If your user-specified locale is not supported, MATLAB uses the default locale en_US_POSIX.US-ASCII.

- system locale setting — Setting on Microsoft Windows platforms. The user locale and system locale must be the same value. If these values are not the same, you might see garbled text or incorrectly displayed characters.

- i18n — Short for the word internationalization, where 18 stands for the number of letters between the letters i and n.

Default Locale Setting

MATLAB does not support every locale setting. If the user-specified locale is unsupported, MATLAB uses the default locale en_US_POSIX.US-ASCII, also known as C locale.

Supported Character Sets and Encodings

MATLAB uses Unicode as its internal character set so that it can represent all letters and symbols, regardless of platform, language, or locale. MATLAB uses UTF-8 as its default character encoding to ensure that all Unicode code points can be correctly represented in files and byte streams. MATLAB also supports other character encodings for backwards compatibility and interoperability.

Platform-Specific Localized Formats for Current Folder Browser

In the Current Folder Browser, MATLAB usually uses platform-neutral localized formats and rules. You can, however, use the operating system short date format to control the format for displaying file date and time data.

See Also

Related Examples

- "Set Locale on Microsoft Windows Platforms" on page 9-5
- "Set Locale on macOS Platforms" on page 9-7
- "Set Locale on Linux Platforms" on page 9-8

Limitations to International Character Support

MATLAB does not support non 7-bit ASCII characters in the following features:

- MATLAB C/C++ and Fortran engine library functions and the `loadlibrary` function cannot find files located in folder names that contain non 7-bit ASCII characters.
- MATLAB C/C++ and Fortran engine library functions and the `calllib` function used to call C library functions cannot convert Unicode®-compatible input arguments of type `char*` to MATLAB character arrays.
- The `matlab.wsdl.createWSDLClient` function cannot read non 7-bit ASCII characters in Web Services Description Language (WSDL) documents.

When you have a file containing text that has characters in a different encoding than that of your platform, when you save or publish your file, MATLAB displays those characters as garbled text.

Set Locale on Microsoft Windows Platforms

MATLAB reads the user locale and system locale on Windows platforms. The user locale and system locale must be the same value. If these values are not the same, you might see garbled text or incorrectly displayed characters.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale en_US_POSIX.US-ASCII, also known as C locale.

When you change the system locale, restart your system. Otherwise, you might see unexpected behaviors.

Locale on Windows 10 Platforms

These instructions are based on Windows 10 Version 1809. Refer to your version of Windows 10 documentation for information about setting region information using the **Settings** app.

1 From **Time & Language**, select **Region**.
2 Under **Related Settings**, select **Additional date, time, & regional settings**.
3 Select **Region** to change location.
4 On the **Formats** tab, select a target locale from the **Format** drop-down list, and then click **Apply**. This action sets the user locale.
5 On the **Administrative** tab, click **Change system locale...**.
6 Select a target locale from the **Current system locale** drop-down list. This action sets the system locale.
7 Exit each dialog box by clicking **OK**.
8 Restart the system.

Using UTF-8 on Windows 10 Platforms

As of Windows 10 1803, Microsoft added the ability to specify UTF-8 as the character encoding used by Windows itself. This feature can be enabled using the instructions shown above, however, after step 6, select the check box next to **Use Unicode UTF-8 for worldwide language support**. If this setting is enabled, MATLAB uses UTF-8 as both its default and its locale-specific character encoding.

Note It is not necessary to enable this option in order to use UTF-8 with MATLAB. MATLAB uses UTF-8 as the default encoding independent of this setting. Therefore, most users do not need to enable this Windows option; however, MATLAB works correctly if it is enabled.

Locale on Windows 7 Platforms

User Locale

1 Select **Start > Control Panel > Clock, Language, and Region > Region and Language**.
2 Open the **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   In the **Language for non-Unicode programs** section, click **Change system locale...**.
4   Select a target locale from the **Current system locale** drop-down list.
5   Restart the system.

Set Locale on macOS Platforms

On the Apple macOS platform, MATLAB reads the user locale setting. MATLAB automatically chooses a codeset for each combination of language and territory.

If you customize the locale setting, MATLAB ignores the customized portion. MATLAB ignores the `LANG` environment variable and the Terminal application locale setting.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

See Also

Set Locale on Linux Platforms

Use the `LANG` environment variable to specify the locale to be used by MATLAB.

MATLAB does not support every locale setting. If your locale is not supported, MATLAB uses the default locale `en_US_POSIX.US-ASCII`, also known as C locale.

See Also

More About

*   "Numbers Display Period for Decimal Point" on page 9-12

Unexpected Behavior on macOS Platforms

If you customize the locale setting, MATLAB ignores the customized portion.

MATLAB ignores the `LANG` environment variable and the Terminal application locale setting.

See Also

Related Examples

- "Set Locale on macOS Platforms" on page 9-7

Characters Incorrectly Displayed on Windows Systems

The user locale and system locale must be the same value on the Microsoft Windows platform. If these values are not the same, you might see garbled text or incorrect characters.

See Also

Related Examples

- "Set Locale on Microsoft Windows Platforms" on page 9-5

datenum Might Not Return Correct Value

The results of the `datenum` function vary depending on the locale. To ensure the correct calculation of functions using date values associated with files and folders, replace `datenum` function calls with the use of the `dir` function `datenum` field.

For example, look at the modification date of your MATLAB `license_agreement.txt` file:

```
cd(matlabroot)
f = dir('license_agreement.txt')
```

MATLAB displays information similar to:

```
f =

       name: 'license_agreement.txt'
       date: '10-May-2015 17:48:22'
      bytes: 5124
      isdir: 0
    datenum: 7.3317e+005
```

If your code uses the `date` field of the `dir` command, similar to:

```
n = datenum(f.date);
```

replace it with the `datenum` field:

```
n = f.datenum;
```

See Also
dir

Numbers Display Period for Decimal Point

MATLAB reads the user locale for all categories except for the `numeric` category (which is equivalent with `LC_NUMERIC`). This category controls numeric data formatting and parsing. MATLAB always sets `LC_NUMERIC` to the `C` locale.

MATLAB uses a period for a decimal point, regardless of the format specified by the user locale. For example, the value of pi can be displayed as `3,1416` or `3.1416`, depending on the format used by a locale. MATLAB always displays `3.1416`.

The MATLAB language reserves the use of commas to the cases described in the Symbol Reference topic.

See Also

More About

- Symbol Reference: Comma

Script Compatibility

As of R2020a, plain text files, such as MATLAB scripts or functions, that are encoded using UTF-8 (note that all 7-bit ASCII files *are* de facto UTF-8 encoded), work seamlessly across all platforms and all locales. Plain text files created with an earlier version of MATLAB that are not UTF-8 encoded also work if they were created in the same locale that the current MATLAB is using. Under most circumstances, MATLAB also operates correctly with plain text files encoded in one of the locale-specific CJK (Chinese, Japanese, or Korean) encodings.

Change the MATLAB Desktop Language

MathWorks offers MATLAB translated into Japanese, Korean, and Chinese. If you have your locale setting set to Japanese, Korean, or Chinese, you can choose the language in which the MATLAB desktop appears. Desktop items (such as dialog boxes, button names, and menu items) and error and warning messages appear in the language that you select.

To change the desktop language, on the **Home** tab, in the **Environment** section, click ⚙ **Preferences** and select **MATLAB > General**. Then, specify the desktop language in which the MATLAB desktop appears. If the **Desktop language** drop-down list is not displayed, then this option is not supported on your system.

Note You must restart MATLAB for the preference change to take effect.

Most desktop elements and apps use the language selected in the **Desktop language** preference drop-down list. However, system dialog boxes, such as file selectors or color pickers, use the operating system display language.

See Also
Preferences Window

More About
- "Translated Documentation" on page 4-14
- "Locale Setting Concepts for Internationalization" on page 9-2

X Servers and International Keyboard Layouts

Simulink requires the XKEYBOARD extension to detect the keyboard layout that is currently active. Some older X servers do not support this extension. If this extension is absent or disabled, then Simulink assumes a default layout. When the default layout does not match the keyboard layout, some keys might not produce the expected characters.

Suppose you are using a Linux based computer with a German keyboard. Without the XKEYBOARD extension, Simulink uses the default layout, which is English, and can therefore produce incorrect characters. MATLAB correctly recognizes the keyboard as German.

If you encounter this issue, try enabling the XKEYBOARD extension, or use an X server that supports this extension.

MATLAB Online

What Is MATLAB Online?

MATLAB Online provides access to the latest version of MATLAB from a standard web browser using MathWorks hosted computing resources and storage.

With MATLAB Online, you can:

- Access the latest version of MATLAB with all the latest features available to you through your current license.
- Use MATLAB in your web browser without any downloads, installation, or maintenance.
- Store up to 5 GB of MATLAB files in your MATLAB Drive.
- Collaborate with other MATLAB users by giving them view-only or can-edit access to your MATLAB files.
- Create a link to your work and share it globally.
- Synchronize your desktop files with MATLAB Online and MATLAB Drive using the MATLAB Drive Connector.

To start using MATLAB Online, go to https://matlab.mathworks.com/.

For more information about MATLAB Online, including supported license types, specifications, limitations, and system requirements, see MATLAB Online Overview.
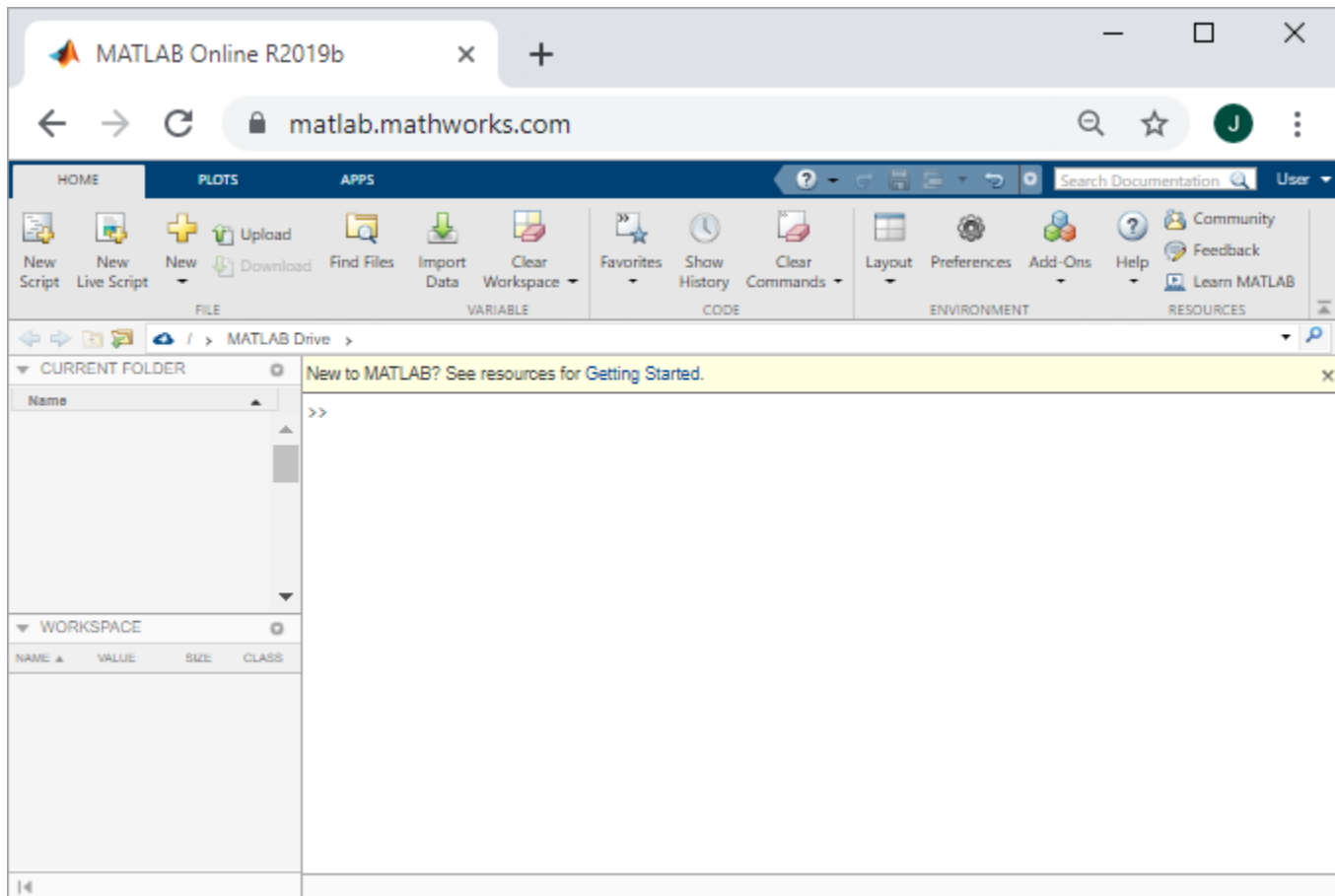
See Also

More About

- "Share Folders in MATLAB" on page 6-34
- "Use Keyboard Shortcuts to Navigate MATLAB" on page 2-23
- MATLAB Drive

Restore Files in MATLAB Online

MATLAB Online on page 10-2 provides access to MATLAB from a standard web browser. In MATLAB Online, each time you save a program file, MATLAB stores the contents of your program in the version history. To recover a previous version of a file, select the **File Versions** tab.

The file version history appears, as shown in the following figure.

Browse for the file you want by selecting from the list, or by clicking **Older** or **Newer**. When you find the file, click **Restore as Latest**. MATLAB adds the restored file to the top of your history with a new timestamp and version number.

See Also

More About

- "Share Folders in MATLAB" on page 6-34
- "Manage Files and Folders" on page 6-31

Post-Installation Tasks

Configure the MATLAB Startup Accelerator

When you install MathWorks products on your computer, the installation includes a utility program that can speed up MATLAB startup, called the MATLAB Startup Accelerator. By default, the installer configures this utility as a scheduled task on your computer that runs several times each day.

While you are not required to modify this configuration, the startup accelerator works best if you coordinate when it runs with your particular MATLAB usage patterns. For example, by default, the startup accelerator runs at 8:00 a.m. and again at 1:00 p.m. every day, or whenever you log on to your computer. To take full advantage of the startup accelerator, schedule it to run at a time just before you typically start MATLAB.

To modify this configuration, use the Windows Task Scheduler. In the Windows Task Scheduler, these start times are called triggers. You can modify these triggers to customize when the startup accelerator runs. You can also add triggers, delete triggers, or delete the startup accelerator task completely. The following section shows how to change when the startup accelerator runs using the Windows Task Scheduler. For more information about the Task Scheduler, see Windows documentation.

Change When the MATLAB Startup Accelerator Runs

To change when the MATLAB Startup Accelerator runs on your computer, use the Windows Task Scheduler.

1   Start the Windows Task Scheduler.

On the Windows Start menu, search for the phrase "task scheduler" and press **Enter**.

2   Click **Task Scheduler Library** in the Console tree to view scheduled tasks and double-click the name of the startup accelerator to change the configuration.

Click here to view
scheduled tasks.

Double-click to modify
configuration.
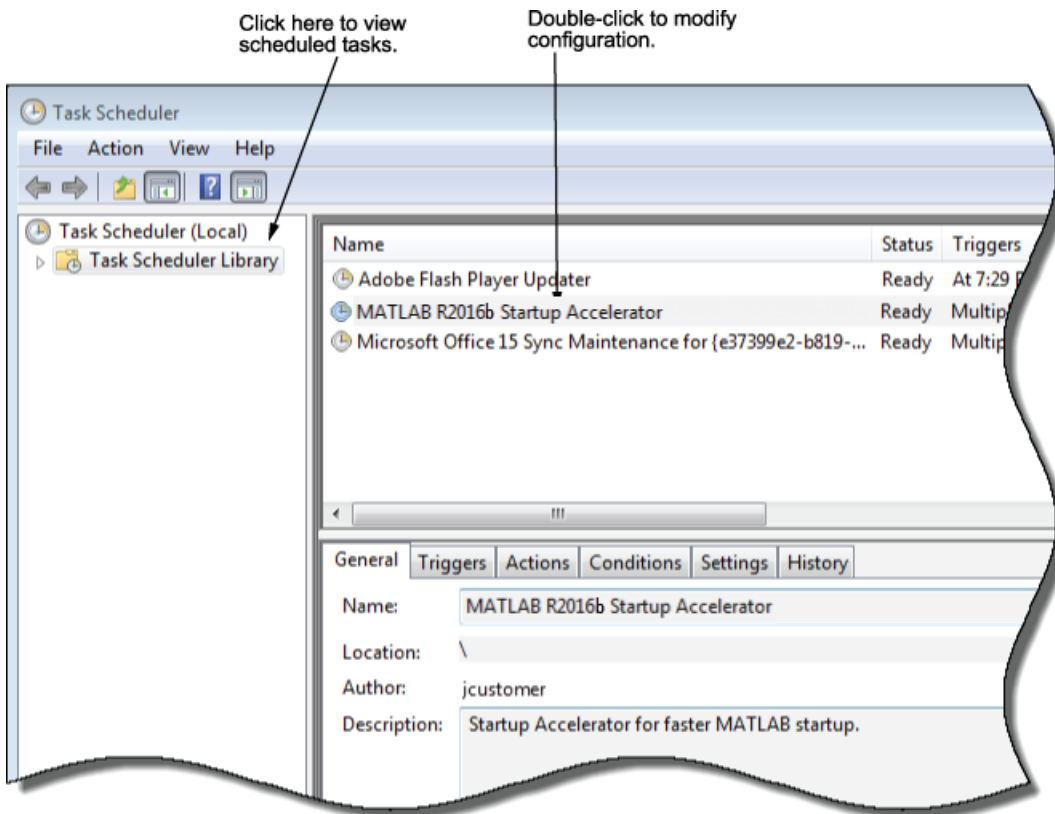
3   To change when the task is scheduled to start, click the Trigger tab in the task properties dialog box. In the list of triggers displayed, select the trigger you want to change, and click **Edit**.

Select trigger that
you want to edit.

MATLAB R2016b Startup Accelerator Properties (Local Computer)

General | Triggers | Actions | Conditions | Settings | History

When you create a task, you can specify the conditions that will trigger the task.

Trigger	Details	Status
At log on	At log on of any user	Enabled
Daily	At 8:07 AM every day	Enabled
Daily	At 1:07 PM every day	Enabled

New... | Edit... | Delete

OK | Cancel

Click Edit to modify
the trigger.

4 To change the time the task is scheduled to run, enter a new time and click **OK**.

Change the trigger
schedule time.
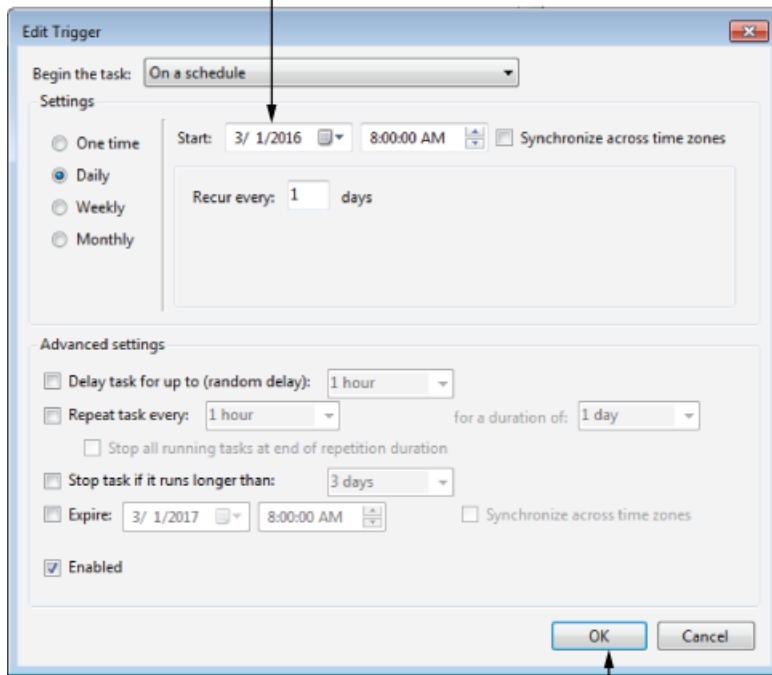
Click OK.

MATLAB Cloud Computing

Run MATLAB from Azure Marketplace

This topic describes how to use the Azure Marketplace to quickly deploy and run MATLAB on Azure. The elasticity of the cloud infrastructure combined with MATLAB enables you to leverage greater computing resources and keep your calculations close to your data.

Anyone with a valid MATLAB license and an Azure account can run MATLAB on the cloud. This is called *MATLAB (BYOL)* because you "Bring Your Own License."

Deploy MATLAB Resources on Azure

Requirements

To complete these instructions, you need:

- MATLAB License on page 12-2
- Microsoft Azure Account
- Remote Desktop client for Windows Desktop (typically preinstalled on most Windows systems)
- A working knowledge of Azure Resource Manager

Licensing

By default, the MATLAB (BYOL) offering on Azure Marketplace uses online licensing. This table summarizes which licenses support running MATLAB on the cloud.

License Type	Cloud Availability
Individual (a license that only you use that is in your name; not a Home or Student license)	Your license is already configured for MATLAB on the cloud.
Campus-Wide License (a license that you use that belongs to your academic institution)	Your license is already configured for MATLAB on the cloud.
Home and Student (that only you use and is in your name)	These license types are not eligible to use MATLAB on the cloud. However, you can use MATLAB Online on page 10-2.
All other license types, including Concurrent and Network Named User	Contact your license administrator.* The license administrator might need to make some changes to the license to enable running MATLAB on the cloud. **Administrators**: For more information on this process, see "MATLAB Licensing in the Cloud".
*Not sure who the license administrator is? Sign in to your MathWorks Account, click the license you are using, then click the tab marked "Contact Administrators".	

Configure and Deploy Template from Marketplace

To configure and deploy MATLAB (BYOL) resources in a resource group:

1 Navigate to https://portal.azure.com and login to your Azure account.

2 From the Portal, click **Create a Resource**.

3 Search for "matlab" in the marketplace and click on the **MATLAB (BYOL)** offering.

4 Click **Create** on the offering page to begin setup.

5 The setup process uses a Resource Manager template to help you configure the virtual machine and networking settings. To make setup easy, many of the fields come prepopulated with acceptable values. This table describes how to set the various options in each menu of the template. Click **Next** at the end of each step to proceed to the next menu.

Menu	Option	Description
Basics	Subscription	Select an Azure subscription to use.
	Resource group	You can select an existing resource group from the drop-down menu, or click **Create New** to create a new resource group. If you select an existing resource group, then it must not have any currently deployed resources.
	Location	Select a location from the drop-down list.
Virtual Machine Settings	Virtual Machine name	Select a name for the VM resource. The default name is `matlab-desktop`.
	Virtual Machine size	Select the VM size. The default size (D3 v2) is sufficient for most purposes and safely satisfies the minimum requirements for MATLAB, but you can select a different size if necessary. Note that the size of the VM you choose affects your costs.
	Username	Select a username for the VM. You will use this username to login to the VM with Remote Desktop later.

Menu	Option	Description
	Password	Select a password for the VM. Note that the password must have at least 8 characters and can only include letters and numbers.
	Confirm password	Retype your chosen password.
	IP addresses	Select the range of IP addresses that are authorized to connect to the VM. This is the public IP address for your computer which can be found by searching for "What is my IP address" on the web. This field uses CIDR notation of the form: *100.100.100.100/20*. The mask determines the number of IP addresses to include.
	Network License Manager port@server	If you are using a Network Licensing Manager, enter the port and hostname or IP address. For more information, see "MATLAB Licensing in the Cloud".
Networking	Virtual network	The **Virtual network** field is prepopulated with a new virtual network resource named `vnet01`. • You can configure the name, address space, or subnets of the new virtual network resource by clicking **Create new**. • You can select an existing virtual network resource in the drop-down menu. If you choose to use an existing resource, then the template does not create any new virtual network resources.

Menu	Option	Description
	Subnets	The **Subnet** field is prepopulated with a new subnet named `subnet-1` and an associated subnet address. • You can select a different subnet of the selected virtual network in the drop-down menu. • If you selected an existing virtual network rather than creating a new one, then you can click **Manage subnet configuration** to configure the subnets for the network.
	Public IP Address for the VM	The **Public IP Address for the VM** field is prepopulated with a new Public IP resource named `matlab-desktop-ip`. The resource uses a basic SKU and dynamic assignment. • You can configure the name, SKU, and assignment protocol by clicking **Create new**. • You can select an existing Public IP resource in the drop-down menu. If you choose to use an existing resource, then the template does not create any new Public IP resources.

Menu	Option	Description
	DNS Prefix for the Public IP Address	The **DNS Prefix for the Public IP Address** field is prepopulated with an acceptable DNS prefix, but you can edit it if necessary. The selected DNS name is used to connect to the VM with Remote Desktop. • If you edit the DNS prefix, then note that it must be globally unique on Azure.

Menu	Option	Description
Review and Create	—	When you advance to the **Review and Create** menu, Azure automatically runs some final validation checks on the information entered on previous screens. If Azure finds any errors, then you need to fix them before proceeding. After the validation completes successfully, review the MathWorks Terms of use and Privacy policy. You are responsible for all associated costs once you deploy MATLAB on Azure. Once you are satisfied with the values you have entered, click **Create** to finalize setup and begin deploying the selected resources, or **Download a template for automation** to get a copy of the completed template. • The deployment can take several minutes. You will get a notification in the Azure Portal once the deployment is complete. • When the deployment is complete, follow the instructions in "Connect to VM and Run MATLAB" on page 12-7 to use the virtual machine you set up.

Connect to VM and Run MATLAB

Once you have successfully configured and deployed the MATLAB (BYOL) resources, use these steps to connect to the virtual machine you set up. You can either create a prepopulated Remote Desktop file, or run Remote Desktop and manually fill out the DNS name.

1   From the Azure Portal, navigate to the resource group with the resources you deployed.

2   Click the `matlab-desktop` virtual machine resource to view its details. (If you renamed the virtual machine resource, then select the resource corresponding to that name.)

3   Click the **Connect** button to see a list of connection options, and then select **RDP** from the drop-down menu.

4 On the **Connect with RDP** screen, the **IP address** and **Port number** fields are prepopulated. Click **Download RDP file** to download a Remote Desktop file that automatically connects to the virtual machine.

5 Run the RDP file you downloaded to connect to the virtual machine, accepting any intermediate dialogs.

6 Once Remote Desktop Connection connects to the VM, sign in to the virtual machine using the username and password that you previously configured in **Virtual Machine Settings**. (See "Configure and Deploy Template from Marketplace" on page 12-2 for more information.)

7 You are now at the desktop of the Linux VM. Double click the MATLAB icon to launch MATLAB.

Note Launching MATLAB for the first time in this setting can take several minutes.

8 After you launch MATLAB, you are prompted to login to your MathWorks Account and select a MATLAB license to use.

Manual Connection

1 To manually connect to the virtual machine with Remote Desktop, you need the full DNS name of the machine. The `matlab-desktop` and `matlab-desktop-ip` resources both list the full DNS name. Open one of those resources and copy the DNS name.

2 Open the **Remote Desktop Connection** software.

3 To populate the **Computer** field, paste the DNS name that you copied.

4 Click **Connect** to proceed.

5 Follow steps 6–8 in "Connect to VM and Run MATLAB" on page 12-7 to sign in to the VM and launch MATLAB.

Delete MATLAB Resources from Azure

You can remove the resource group and all associated resources when you are done with them to help save on costs.

1 Login to the Azure Portal.

2 Select the resource group containing the MATLAB (BYOL) resources you deployed.

3 Select the **Delete resource group** icon to delete all resources deployed in the group.

4 You will be prompted to confirm the deletion.

The deletion is final, and files do not persist between deployments. (See "File Persistence on Azure" on page 12-8 for more information.) You need to create the resources to make them available again.

File Persistence on Azure

As you use MATLAB on Azure, all your files and changes are stored locally on the virtual machine. The files persist until you terminate the virtual machine or delete the resource group. However, stopping or shutting down the instance does not destroy the data on the virtual machine, so it is safe to shutdown the instance when you are not using it.

If you use MATLAB Drive to store your files on the cloud, you can install the MATLAB Drive Connector on the virtual machine to gain access to your files. See "Install MATLAB Drive Connector" (MATLAB Drive) for instructions to install the Connector on Linux.

Azure Architecture and Resources

Deploying MATLAB on Azure creates several resources in your resource group. The deployment sets up a single Azure virtual machine (VM) running Linux and MATLAB, a network interface with a public IP address to connect to the VM, a network security group that controls network traffic, and a virtual network for communication between resources. This table summarizes the resources that are created.

Resource Name	Default Resource Name in Azure	Description
Virtual machine	`matlab-desktop`	The VM instance with pre-installed desktop MATLAB. The name you choose for this virtual machine resource acts as the root of the names for the Public IP address and Disk resources.
Public IP address	`matlab-desktop-ip`	Public IP address to connect to the VM running MATLAB. If you select an existing Public IP resource, then the template does not create this resource.
Disk	`matlab-desktop-<unique id>`	The disk attached to the VM.
Network interface	`matlab-nic`	Enables the VM to communicate with the internet, Azure, and on-premises resources.
Network security group	`matlab-rdp-nsg`	Allows or denies traffic to and from sources and destinations.
Virtual network	`vnet01`	Enables resources to communicate with each other. If you select an existing virtual network resource, then the template does not create this resource.

The architecture of these resources is summarized in this diagram.
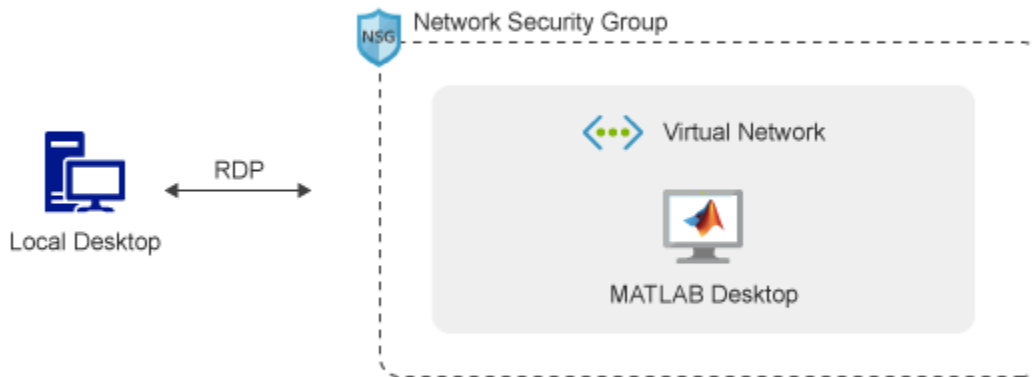
VM Software

A preconfigured Ubuntu VM is provided to make deployment easy. The VM contains this software:

- MATLAB, Simulink, toolboxes, and support for GPUs.

  The license used to run MATLAB determines which products and toolboxes are available for you to use. However, all products are installed on the VM, so the `ver` command returns a list of all products. To add new products to your license, see Products and Services.

- Add-Ons: Deep Learning Toolbox™ Model for AlexNet Network, Deep Learning Toolbox Model for GoogLeNet Network, and Deep Learning Toolbox Model for ResNet-50 Network.

Marketplace vs. Reference Architecture

The Azure Marketplace offering only runs the latest release of MATLAB. If you need to run older releases of MATLAB, or customize the templates and automation scripts more extensively, then you can use the Reference Architecture for Microsoft Azure instead.

Support

If you encounter an unexpected problem, search MATLAB Answers for solutions first. Most likely, other people have run into the same problem and resolved it already.

If the problem persists, or there are not any relevant posts on MATLAB Answers for the issue, contact Technical Support.

See Also

External Websites
- "MATLAB Licensing in the Cloud"
- Use MATLAB in the Cloud
- Reference Architecture for Microsoft Azure