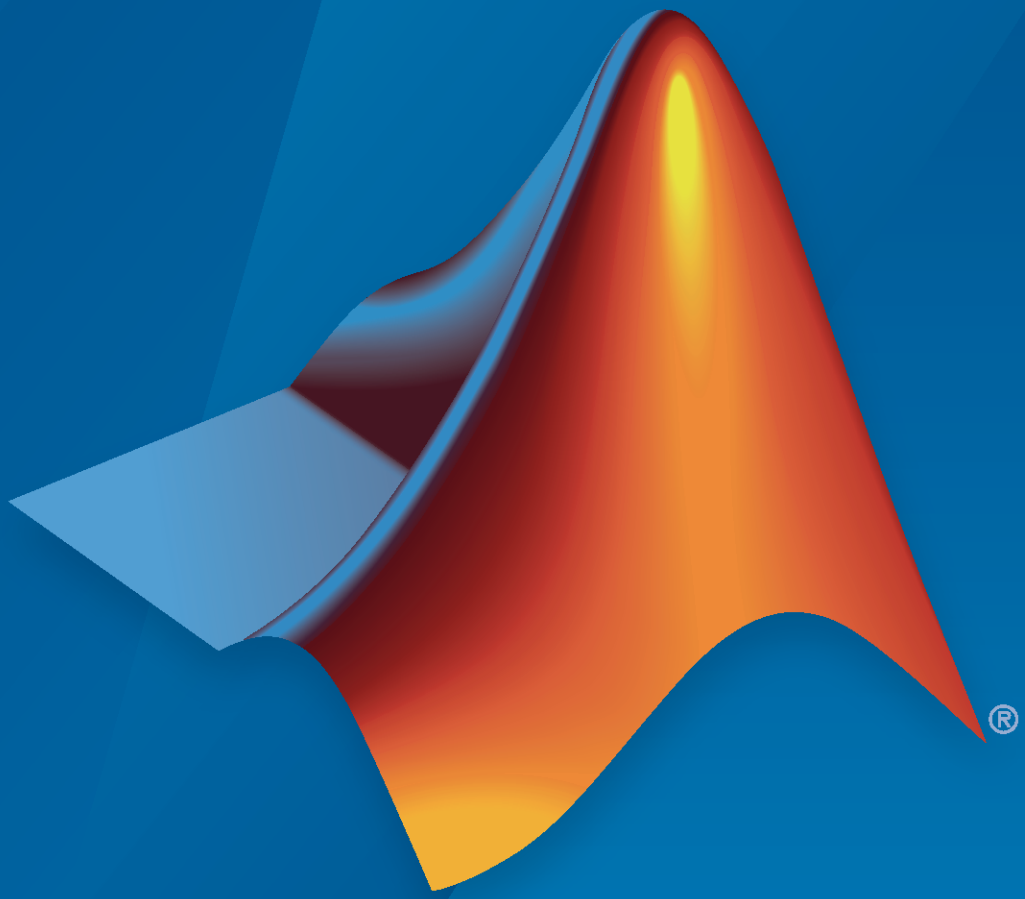
Fuzzy Logic Toolbox™ Release Notes



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

Fuzzy Logic Toolbox™ Release Notes

Contents

R2025b

R2025a

R2024b

R2024a

R2023b

R2023a

R2022b

R2022a

R2021b

R2021a

R2025b

Version: 25.2

Bug Fixes

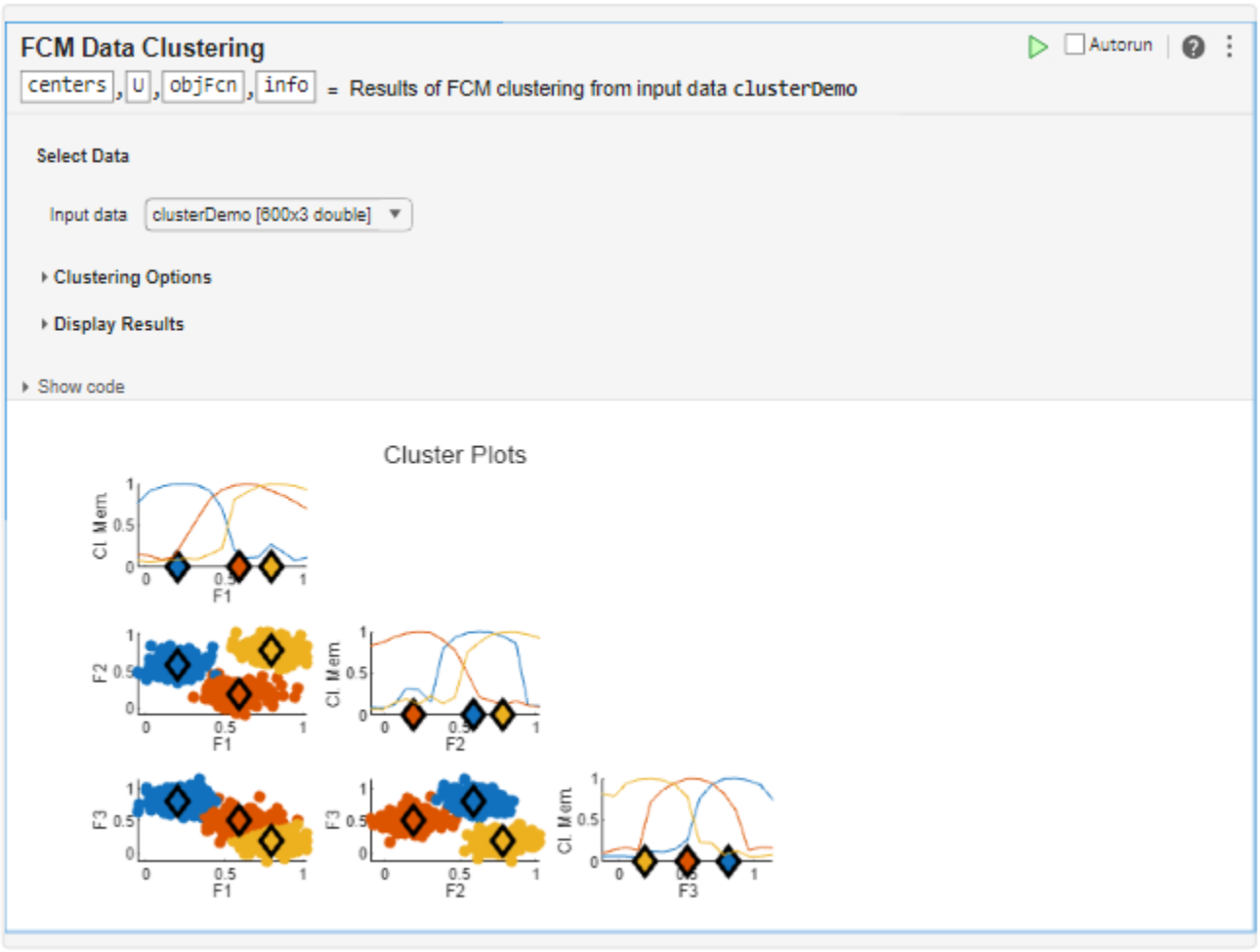
Quality and stability improvements

R2025b delivers quality and stability improvements, building on the new features introduced in R2025a.

R2025a

★FCM Data Clustering Live Editor Task: Perform fuzzy c-means clustering without writing code

You can now use the FCM Data Clustering Live Editor task to perform fuzzy c-means clustering without writing code. To add this task to a live script, on the **Live Editor** tab, click **Task** and select the **FCM Data Clustering** icon.



Fuzzy Logic Designer: Export fuzzy inference system to new Simulink model

When designing a FIS using Fuzzy Logic Designer, you can now export the active fuzzy system to a new Simulink® model that contains a corresponding Fuzzy Logic Controller or FIS Tree block. For more information, see Export FIS to Simulink.

★Fuzzy Logic Controller Block: Open Fuzzy Logic Designer from Simulink block

You can now open Fuzzy Logic Designer from the Fuzzy Logic Controller block and use the app to modify the FIS. You can then update the block to use the modified FIS design. For more information, see Design Fuzzy Inference System in Simulink.

Fuzzy Logic Controller Block: Implement fuzzy inference system as lookup table

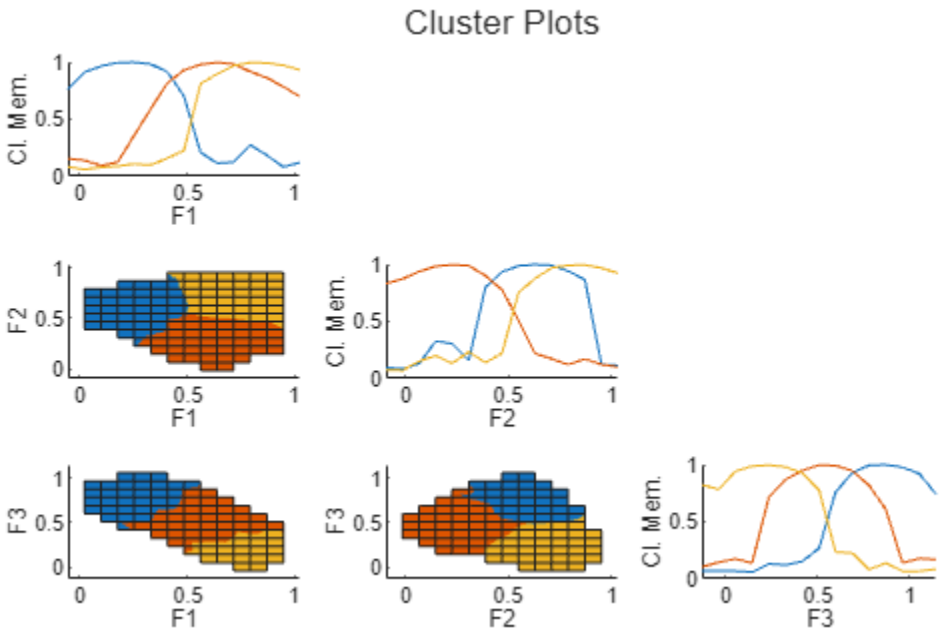
When simulating a fuzzy inference system using the Fuzzy Logic Controller block, you can now represent your FIS as a lookup table, which reduces processing time.

To use a lookup table, select the **Implement FIS as lookup table** parameter. The block divides the input range of each FIS input into equally sized regions based on the number of breakpoints, which you specify using the **Number of input breakpoints** parameter.

For each input combination, the block calculates a corresponding FIS output. During evaluation, the controller uses these precomputed outputs instead of performing the fuzzy inference process.

plotFuzzyClusters Function: Visualize clusters computed using fuzzy c-means clustering

After you compute fuzzy clusters using the `fcm` function, you can now visualize the clusters using the new `plotFuzzyClusters` function.



New Example: Control vehicle cabin temperature using fuzzy systems

The new Vehicle HVAC System Control Using Fuzzy Logic example shows how to control vehicle cabin temperature using the following fuzzy systems:

- Fuzzy inference system that calculates climate control setpoints to maintain a desired cabin temperature
- Fuzzy PID controller that maintains the cabin temperature

Functionality being removed or changed

Clustering tool not recommended
Still runs

Interactively clustering data using the Clustering tool is not recommended. Use the FCM Data Clustering Live Editor Task instead.

R2024b

★Fuzzy PID Controller Block: Implement fuzzy logic-based PID controller in Simulink

You can now implement a fuzzy logic-based PID controller in Simulink using the new Fuzzy PID Controller block. For examples, see:

- Implement Fuzzy PID Controller in Simulink
- Fuzzy PID Control with Type-2 FIS

★Fuzzy Logic Designer: Generate MATLAB code for simulating fuzzy systems

Once you interactively build or tune a fuzzy system using Fuzzy Logic Designer, you can now generate MATLAB® code to programmatically simulate that system. You can then use or modify the generated code for your applications. For more information, see Generate MATLAB Code for Simulating Fuzzy Systems.

writeFIS and readfis: Save and load FIS objects using MAT files

You can now save FIS objects to MAT files using `writeFIS` and load FIS objects from MAT files using `readfis`.

writeFIS and readfis: Save and load FIS trees

You can now save FIS trees using `writeFIS` and load FIS trees using `readfis`.

plotrule Function: View rule inference process for Mamdani and Sugeno systems

Using the new `plotrule` function, you can now view the inference process for a Mamdani or Sugeno FIS. The resulting rule inference diagram is similar to the **Rule Inference** document in Fuzzy Logic Designer.

comparefis Function: Simulate fuzzy systems and compare output values

Using the new `comparefis` function, you can now:

- Simulate a fuzzy system for specified input values and compare the resulting output values with reference output data.
- Simulate multiple fuzzy systems and compare their respective output values.

The resulting comparison plots are similar to the **System Validation** document in Fuzzy Logic Designer.

plotfiserr Function: View distribution of simulation errors across universe of discourse

Using the new `plotfiserr` function, you can now simulate a FIS for specified input values and view a distribution of simulation errors across the universe of discourse. The simulation errors are calculated based on specified output reference data. The resulting error distribution diagram is similar to the **Error Distribution** document in Fuzzy Logic Designer.

plotfis Function: View propagation of inference results through fuzzy system for specified input values

Using the `plotfis` function, you can now view the propagation of inference results through a fuzzy system for specified input values.

New Example: Control house heating system using fuzzy systems

The new Fuzzy Logic Control for House Heating System example shows how to control a house heating system using the following fuzzy systems:

- Fuzzy inference system for selecting the desired room temperature
- Fuzzy PID controller for regulating the heater power

New Example: PMSM speed control using fuzzy PI controller

The new Field-Oriented Control of PMSM Using Fuzzy PI Controller example shows how to use a fuzzy PI controller for speed control of a permanent magnet synchronous motor (PMSM) using field-oriented control (FOC) principles.

⚠Functionality being removed or changed

Support for representing fuzzy inference systems as structures has been removed

Errors

Support for representing fuzzy inference systems as structures has been removed. Use `mamfis` and `sugfis` objects instead. There are differences between these representations that require updates to your code. These differences include:

- Object property names that differ from the corresponding structure fields
- Objects that store text data as strings rather than as character vectors

All Fuzzy Logic Toolbox functions that accepted or returned fuzzy inference systems as structures now accept and return either `mamfis` or `sugfis` objects.

To convert existing fuzzy inference system structures to objects, use the `convertfis` function.

This table shows some examples of how to update your code to use the new object property names.

If your code has this form:	Use this code instead:
`fis.andMethod`	`fis.AndMethod`

If your code has this form:	Use this code instead:
`fis.aggMethod`	`fis.AggregationMethod`
`fis.input(1).name`	`fis.Inputs(1).Name`
`fis.output(1).mf(1).params`	`fis.Outputs(1).MembershipFunctions(1).Parameters`

addvar has been removed
Errors

`addvar` has been removed. To add input or output variables to a fuzzy system, use `addInput` or `addOutput`, respectively.

This table shows some typical usages of `addvar` and how to update your code to use `addInput` or `addOutput` instead.

If your code has this form:	Use this code instead:
`fis = addvar(fis,'input',...` `'service',[0 10])`	`fis = addInput(fis,[0 10],...` `'Name',"service")`
`fis = addvar(fis,'output',...` `'tip',[0 30])`	`fis = addOutput(fis,[0 30],...` `'Name',"tip")`

getfis has been removed
Errors

`getfis` has been removed. Access fuzzy inference system properties using dot notation instead. For more information on fuzzy inference system objects, see `mamfis` and `sugfis`.

This table shows some typical usages of `getfis` for accessing fuzzy inference system properties and how to update your code to use dot notation instead.

If your code has this form:	Use this code instead:
`get(fis,'andmethod')`	`fis.AndMethod`
`getfis(fis,'input',1)`	`fis.Inputs(1)`
`getfis(fis,'input',1,'name')`	`fis.Inputs(1).Name`
`getfis(fis,'input',2,'mf',1)`	`fis.Inputs(2).MembershipFunctions(1)`
`getfis(fis,'input',2,'mf',1,...` `params)`	`fis.Inputs(2).MembershipFunctions(1).Parameters`

mam2sug has been removed
Errors

`mam2sug` has been removed. Use `convertToSugeno` instead. To update your code, change the function name from `mam2sug` to `convertToSugeno`. The syntaxes are equivalent.

mf2mf has been removed
Errors

`mf2mf` has been removed. Convert membership functions using dot notation on `fismf` objects instead.

Previously, membership functions were represented as structures within a fuzzy inference system structure. Now, membership functions are represented as `fismf` objects within `mamfis` and `sugfis` objects.

Previously, to change the type of a membership function in a fuzzy inference system, you converted the parameters using `mf2mf`.

```
fis = readfis('tipper');
oldType = fis.input(1).mf(1).type;
oldParams = fis.input(1).mf(1).params;
fis.input(1).mf(1).type = newType;
fis.input(1).mf(1).params = mf2mf(oldParams,oldType,newType);
```

Now, when you change the type of membership function, the parameters are converted automatically.

```
fis = readfis('tipper');
fis.Inputs(1).MembershipFunctions(1).Type = newType;
```

newfis has been removed
Errors

`newfis` has been removed. To create a Mamdani or Sugeno FIS, use `mamfis` or `sugfis`, respectively.

This table shows some typical usages of `newfis` for creating fuzzy systems and how to update your code to use `mamfis` or `sugfis` instead.

If your code has this form:	Use this code instead:
`fis = newfis(name)`	`fis = mamfis('Name',name)`
`fis = newfis(name,'FISType','mamdani')`	`fis = mamfis('Name',name)`
`fis = newfis(name,'FISType','sugeno')`	`fis = sugfis('Name',name)`

`fis = newfis(name,...`
`'FISType','mamdani',...`
`'AndMethod','prod')`
	`fis = mamfis('Name',name,...`
	`'AndMethod','prod')`

`fis = newfis(name,...`
`'FISType','sugeno',...`
`'OrMethod','probor')`
	`fis = sugfis('Name',name,...`
	`'OrMethod','probor')`

parsrule has been removed
Errors

`parsrule` has been removed. Use `addRule` instead.

If you previously added rules using linguistic or symbolic expressions with `parsrule`, you can specify rules using the same expressions with `addrule`. `addRule` automatically detects the format of the strings or character vectors in your rule list. Therefore, it is no longer necessary to specify the rule format. To add a rule list using `addRule`, use the following command:

```
fis = addRule(fis,rules);
```

Previously, you could add rules using indexed expressions with `parsrule`.

```
rule1 = "1 2, 1 4 (1) : 1";
rule2 = "-1 1, 3 2 (1) : 1";
rules = [rule1 rule2];
fis = parsrule(fis,rules,'Format','indexed');
```

Now, specify these rules using arrays of indices.

```
rule1 = [1 2 1 4 1 1];
rule2 = [-1 1 3 2 1 1];
rules = [rule1; rule2];
fis = addRule(fis,rules);
```

If you previously specified rules using the `'Language'` name-value pair argument with `parsrule`, this functionality has been removed and there is no replacement. Specify your rules using `addRule` with a different rule format.

Previously, `parsrule` replaced the entire rule list in your fuzzy system. `addRule` appends your specified rules to the rule list.

rmmf has been removed
Errors

`rmmf` has been removed. Use `removeMF` instead.

The following table shows some typical usages of `rmmf` and how to update your code to use `removeMF` instead. Previously, you specified the index of the variable from which you wanted to remove the membership function and the index of the membership function that you wanted to remove. Now, to remove a membership function, specify the variable name and the membership function name.

If your code has this form:	Use this code instead:
`fis = rmmf(fis,'input',1,'mf',1)`	`fis = removeMF(fis,"service","poor")`
`fis = rmmf(fis,'output',1,'mf',1)`	`fis = removeMF(fis,"tip","cheap")`

Previously, you had to delete any references to a membership function you wanted to remove from the rule set. `removeMF` automatically removes these references from the rule set of your fuzzy system.

rmvar has been removed
Errors

`rmvar` has been removed. To remove input or output variables from a fuzzy system, use `removeInput` or `removeOutput`, respectively.

This table shows some typical usages of `rmvar` and how to update your code to use `removeInput` or `removeOutput` instead. Previously, you specified the index of the variable that you wanted to remove. Now, to remove a variable, specify the variable name.

If your code has this form:	Use this code instead:
`fis = rmvar(fis,'input',1)`	`fis = removeInput(fis,"service")`
`fis = rmvar(fis,'output',1)`	`fis = removeOutput(fis,"tip")`

Previously, you had to delete any rules from your fuzzy system that contained the variable you wanted to remove. `removeInput` and `removeOutput` automatically remove these variables from the rule set of your fuzzy system.

setfis has been removed
Errors

`setfis` has been removed. Set fuzzy inference system properties using dot notation instead. For more information on fuzzy inference system objects, see `mamfis` and `sugfis`.

This table shows some typical usages of `setfis` for setting fuzzy inference system properties and how to update your code to use dot notation instead.

If your code has this form:	Use this code instead:
`fis = setfis(fis,'andmethod','prod')`	`fis.AndMethod = 'prod'`
`fis = setfis(fis,'input',1,...` `'name','service')`	`fis.Inputs(1).Name = "service"`

`fis = setfis(fis,'input',2,...`
`'mf',1,...`
`params,[5 10 15])`
	`fis.Inputs(2).MembershipFunctions(1).Parameters = ...`
	`[5 10 15]`

showfis has been removed
Errors

`showfis` has been removed. View the properties of your FIS directly instead.

Previously, you could view the properties of your fuzzy system, `myFIS`, using the `showfis` function.

```
showfis(myFIS)
```

Now, you can view the properties directly instead.

```
myFIS
```

To view additional FIS properties, use dot notation. For example, view information about the membership functions of the first input variable.

```
myFIS.Inputs(1).MembershipFunctions
```

For more information on fuzzy inference systems and their properties, see `mamfis` and `sugfis`.

R2024a

Version: 24.1

New Features

Bug Fixes

★FIS Tree Block: Simulate trees of interconnected fuzzy systems in Simulink

You can now simulate FIS trees in Simulink using the new FIS Tree block.

★Fuzzy Logic Designer: Generate MATLAB code for building or tuning fuzzy systems

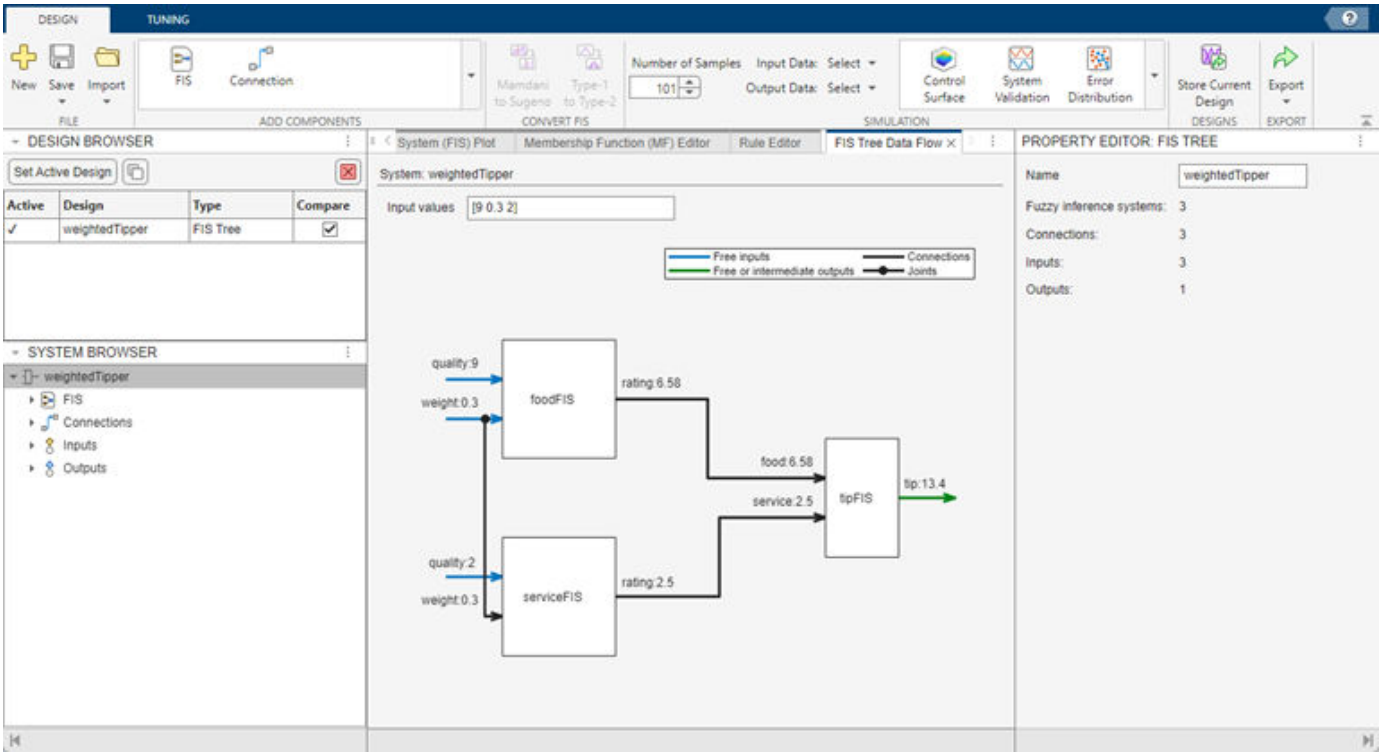
Once you interactively build or tune a fuzzy system using Fuzzy Logic Designer, you can now generate MATLAB code to programmatically build or tune that system. You can then use or modify the generated code for your applications.

For more information on generating MATLAB code for:

- Building a fuzzy system, see Generate MATLAB Code for Building Fuzzy Systems.
- Tuning a fuzzy system, see Generate MATLAB Code for Tuning Fuzzy Systems.

Fuzzy Logic Designer: View data propagation through FIS tree for specified input values

Using Fuzzy Logic Designer, you can now evaluate the behavior of a FIS tree by viewing the propagation of inference results through the tree structure for specified input values. For more information, see FIS Tree Data Flow.



Fuzzy Logic Designer: Programmatically open app for fuzzy system stored in MAT file

You can now programmatically open Fuzzy Logic Designer and specify the name of a MAT file that contains one FIS or FIS tree object. The app opens and imports the specified fuzzy system.

Fuzzy Logic Designer: Export tuning options and tunable settings

You can now export FIS tuning options and tunable parameter settings to the MATLAB workspace from Fuzzy Logic Designer. You can then use the exported tuning configuration objects to perform command-line tuning.

For more information, see Export Tuning Options and Settings from Fuzzy Logic Designer.

Fuzzy Logic Controller Block: Specify FIS using MAT file

When simulating a FIS using the Fuzzy Logic Controller block, you can now specify the FIS to evaluate using a MAT file in the current working folder or on the MATLAB path. The MAT file must contain only one FIS object.

New Examples: Design fuzzy systems for medical diagnostics and decision making

The following new examples show how to design fuzzy systems for medical diagnostics and decision making.

- Decision Making Using Fuzzy Discrete Event Systems
- Classify Breast Tumors from Ultrasound Images Using Fuzzy Inference System

R2023b

Fuzzy Logic Designer: Interactively design, tune, and analyze FIS trees

You can now interactively design, tune, and analyze FIS trees using the Fuzzy Logic Designer app. For more information, see the following examples.

- Build FIS Tree Using Fuzzy Logic Designer
- Tune FIS Tree Using Fuzzy Logic Designer

Fuzzy C-Means Clustering: Specify initial cluster centers

When clustering data using the `fcm` function, you can now specify initial estimates of the cluster centers. Previously, the `fcm` function randomly initialized the cluster centers.

To specify the cluster centers, create an `fcmOptions` object and set the `ClusterCenters` option.

⚠Fuzzy C-Means Clustering: Specify multiple values for number of clusters

When clustering data using FCM clustering, you can now specify multiple values for the number of cluster centers, *C*. The `fcm` function returns cluster centers for the optimal number of clusters, which it determines using a validity index.

To specify the number of clusters, create an `fcmOptions` object and set the `NumClusters` option.

For more information, see Fuzzy Clustering.

⚠Compatibility Considerations

By default, the `fcm` function now computes clusters for multiple values of *C*. For more information, see “fcm function computes clusters for multiple cluster counts by default” on page 5-2.

Gath-Geva FCM Clustering: Detect nonspherical clusters with variable densities

When clustering data using the `fcm` function, you can now use the Gath-Geva algorithm, which allows you to detect nonspherical clusters in the presence of variable cluster densities and unequal numbers of data points in each cluster. This algorithm uses an exponential distance metric based on fuzzy maximum likelihood estimation.

For more information, see Fuzzy Clustering.

⚠Functionality being removed or changed

fcm function computes clusters for multiple cluster counts by default
Behavior change

When the `NumClusters` property of an `fcmOptions` object is `"auto"`, the `fcm` function now computes clusters for multiple cluster counts (2 through 11). Previously, the default number of clusters was 2.

FIS tree connections automatically update when FIS and variable names change
Behavior change

When you change the following names within a FIS tree object, the FIS tree connections now automatically update to use the new names.

- Name of a component FIS object
- Name of an input or output variable within a component FIS object

Previously, any connections containing the modified FIS or variable name were deleted from the FIS tree. You would then have to add the deleted connections back into the FIS tree.

Tunable parameter selections stored with each FIS design in Fuzzy Logic Designer
Behavior change

When tuning FIS designs in Fuzzy Logic Designer, tunable parameter selections are now stored separately for each design. Previously, the app maintained a single set of tunable parameters for each app session.

R2023a

Fuzzy Logic Designer App: Interactively tune rules and parameters of fuzzy inference systems

You can now interactively tune the rules and parameters of a fuzzy inference system using the Fuzzy Logic Designer app. For an example, see Tune Fuzzy Inference System Using Fuzzy Logic Designer.

Fuzzy Logic Designer App: Interactively evaluate performance of fuzzy inference system designs using testing data

You can now interactively evaluate the performance of fuzzy inference system designs for given input/output testing data using the following documents in the Fuzzy Logic Designer app.

- **System Validation** — Compare the outputs from each FIS design with the corresponding output value from the testing data. Using this document, you can assess the performance of multiple FIS designs.
- **Error Distribution** — For a given FIS design, view the output error for different combinations of inputs. Using this document, you can determine which input combinations produce poor results. You can then adjust your rules and membership functions accordingly.

For more information, see Analyze Fuzzy System Using Fuzzy Logic Designer.

Fuzzy Logic Designer App: Automatically distribute membership functions across variable range

When defining membership functions for input and output variables, you can now evenly distribute existing membership function across the variable range. For more information on defining membership functions, see Define Membership Functions Using Fuzzy Logic Designer.

Gustafson-Kessel FCM Clustering: Find clusters with different geometrical shapes

When clustering data using the `fcm` function, you can now use the Gustafson-Kessel algorithm, which allows you to detect clusters with different geometrical shapes within the same data set. This algorithm uses a Mahalanobis distance metric instead of the Euclidean distance metric used in classical FCM clustering. For more information, see Fuzzy Clustering.

For an example that uses both Gustafson-Kessel and classical FCM clustering, see Brain Tumor Segmentation Using Fuzzy C-Means Clustering.

getTunableSettings Function: Obtain tunable settings for component FIS in FIS tree

When tuning a FIS tree, you can now obtain the tunable settings for individual FIS objects within the FIS tree. For more information, see `getTunableSettings`.

⚠Functionality being removed or changed

Specify options for fcm function using fcmOptions object
Behavior change

To specify options for clustering data using the `fcm` function, you now use an `fcmOptions` object.

Previously, you specified the number of clusters using an input argument and specified other options in a vector format.

```
Nc = 3;
exp = 2.5;
maxIter = 200;
minImprove = 1e-4;
verbose = false;
options = [exp maxIter minImprove verbose];
[centers,U] = fcm(data,Nc,options);
```

Now, you specify these clustering options using an `fcmOptions` object.

```
options = fcmOptions(...
    NumClusters=Nc,...
    Exponent=exp,...
    MaxNumIteration=maxIter,...
    MinImprovement=minImprove,...
    Verbose=verbose);
[centers,U] = fcm(data,options);
```

Neuro-Fuzzy Designer will be removed in a future release
Still runs

The Neuro-Fuzzy Designer app will be removed in a future release.

You can now tune a single-output type-1 Sugeno system using the ANFIS method in Fuzzy Logic Designer instead. For an example, see Train Adaptive Neuro-Fuzzy Inference Systems.

R2022b

⚠Redesigned Fuzzy Logic Designer App: Design fuzzy inference systems using improved interactive workflows

The redesigned Fuzzy Logic Designer app streamlines workflows for interactively building fuzzy inference systems. Using the updated app, you can:

- Design both Mamdani and Sugeno fuzzy inference systems
- Design fuzzy inference systems with either type-1 or type-2 membership functions

For an example that builds a fuzzy inference system using the updated app, see Build Fuzzy Systems Using Fuzzy Logic Designer.

⚠Compatibility Considerations

The `fuzzyLogicDesigner` function now opens the updated app.

Code Generation: Generate C and C++ code to evaluate FIS trees

The `evalfis` function now supports code generation for evaluating FIS trees using MATLAB Coder™.

To generate code for evaluating a FIS tree, you must first convert your `fistree` object into a homogeneous structure using `getFISCodeGenerationData`.

For more information on generating code for fuzzy systems, see Generate Code for Fuzzy System Using MATLAB Coder.

⚠Functionality being removed or changed

fuzzyLogicDesigner function opens new Fuzzy Logic Designer app
Behavior change

The `fuzzyLogicDesigner` function now opens the updated Fuzzy Logic Designer app.

mfedit function will be removed
Warns

`mfedit` will be removed in a future release.

To interactively view the rules for a fuzzy inference system, open the Fuzzy Logic Designer app using the `fuzzyLogicDesigner(fis)` command. Then, select the **Membership Function (MF) Editor** document.

ruleedit function will be removed
Warns

`ruleedit` will be removed in a future release.

To interactively view the rules for a fuzzy inference system, open the Fuzzy Logic Designer app using the `fuzzyLogicDesigner(fis)` command. Then, select the **Rule Editor** document.

ruleview function will be removed
Warns

`ruleview` will be removed in a future release.

To interactively view the rules for a fuzzy inference system, open the Fuzzy Logic Designer app using the `fuzzyLogicDesigner(fis)` command. Then, on the **Design** tab, in the **Simulation** section, click **Rule Inference**.

surfview function will be removed
Warns

`surfview` will be removed in a future release.

To interactively view the surface plot for a fuzzy inference system, open the Fuzzy Logic Designer app using the `fuzzyLogicDesigner(fis)` command. Then, on the **Design** tab, in the **Simulation** section, click **Control Surface**.

fcmdemo application has been removed
Errors

The `fcmdemo` application has been removed. For an example that interactively demonstrates fuzzy c-means clustering, see Fuzzy C-Means Clustering.

R2022a

Version: 2.9

New Features

Bug Fixes

Linear Saturation Membership Functions: Implement fuzzy variables with maximum membership at the extremes of the universe of discourse

You can now create fuzzy systems that use the following piecewise-linear saturation membership functions.

- `linzmf` — Linear version of the `zmf` membership function
- `linsmf` — Linear version of the `smf` membership function

You can use these membership functions when:

- Creating type-1 fuzzy systems at the command line. For more information, see `mamfis`, `sugfis`, and `fismf`.
- Creating type-2 fuzzy systems at the command line. For more information, see `mamfistype2`, `sugfistype2`, and `fismftype2`.
- Interactively creating fuzzy systems using Fuzzy Logic Designer.
- Tuning fuzzy system parameters using `tunefis` or Neuro-Fuzzy Designer.
- Generating code using MATLAB Coder or Simulink Coder.
- Deploying fuzzy systems using MATLAB Compiler™.

New Examples: Design fuzzy inference systems to explain the behavior of black-box models

The following new examples show how to tune a fuzzy inference system to explain the behavior of black-box models.

- Explain Black-Box Model Using Fuzzy Support System
- Explainable Fuzzy Support System for Black-Box Model of Robot Obstacle Avoidance

R2021b

⚠plotfis Function: Visualize FIS tree structure

You can now visualize the component FIS objects and connections within a `fistree` object using the `plotfis` function.

⚠Compatibility Considerations

Previously, `plotfis` displayed a summary of the FIS tree properties in the MATLAB Command Window. This information is no longer available in the Command Window when using `plotfis`.

To access information about the FIS tree components, inputs, outputs, and connections, access the properties of the `fistree` object using dot notation.

FIS Trees: Specify names for fistree objects

To distinguish between FIS trees, you can now specify names for `fistree` objects using the new `Name` property.

New Example: Design fuzzy logic controller for artificial pancreas

The Design Controller for Artificial Pancreas Using Fuzzy Logic example shows how to design and tune a FIS tree that controls insulin infusion for type-1 diabetes.

R2021a

Version: 2.8.1

Bug Fixes

R2020b

Version: 2.8

New Features

Bug Fixes

ANFIS Training: Algorithm implementation converted to MATLAB code

The ANFIS training algorithm is now implemented using MATLAB code. Previously, the training algorithm was implemented as a C MEX file application.

The new ANFIS implementation displays the training error and step size increases in the MATLAB Command Window after each training epoch. Previously, the training data for all epochs was displayed in the Command Window at the end of training.

The new implementation can also reduce training time for some training configurations and platforms.