Neural Network Toolbox Release Notes

Contents

Summary by Version

This table provides quick access to what's new in each version. For clarification, see About Release Notes.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Latest Version V5.1 (R2007b)	Yes Details	Yes Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current production documentation
V5.0.2 (R2007a)	No	No	Bug Reports	No
V5.0.1 (R2006b)	No	No	Bug Reports	No
V5.0 (R2006a)	Yes Details	Yes Summary	Bug Reports	No
V4.0.6 (R14SP3)	No	No	Bug Reports	No

About Release Notes

Use release notes when upgrading to a newer version to learn about new features and changes, and the potential impact on your existing files and practices. Release notes are also beneficial if you use or support multiple versions.

If you are not upgrading from the most recent previous version, review release notes for all interim versions, not just for the version you are installing. For example, when upgrading from V1.0 to V1.2, review the New Features and Changes, Version Compatibility Considerations, and Bug Reports for V1.1 and V1.2.

New Features and Changes

These include

- New functionality

- Changes to existing functionality
- Changes to system requirements (complete system requirements for the current version are at the MathWorks Web site)
- Any version compatibility considerations associated with each new feature or change

Version Compatibility Considerations

When a new feature or change introduces a reported incompatibility with the previous version, its description includes a **Compatibility Considerations** subsection that details the impact. For a list of all new features and changes that have reported compatibility impact, see the "Compatibility Summary for Neural Network Toolbox" on page 19.

Compatibility issues that are reported after the product has been released are added to Bug Reports at the MathWorks Web site. Because bug fixes can sometimes result in incompatibilities, also review fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

MathWorks Bug Reports is a user-searchable database of known problems, workarounds, and fixes. The MathWorks updates the Bug Reports database as new problems and resolutions are reported, so check it as needed for the latest information.

Access Bug Reports at the MathWorks Web site using your MathWorks Account. If you are not logged in to your MathWorks Account when you link to Bug Reports, you are prompted to log in or create an account. You then can view bug fixes and known problems for R14SP2 and more recent releases.

Related Documentation at Web Site

Printable Release Notes (PDF). You can print release notes from the PDF version, located at the MathWorks Web site. The PDF version does not support links to other documents or to the Web site, such as to Bug Reports. Use the browser-based version of release notes for access to all information.

Product Documentation. At the MathWorks Web site, you can access complete product documentation for the current version and some previous versions, as noted in the summary table.

Version 5.1 (R2007b) Neural Network Toolbox

This table summarizes what's new in Version 5.1 (R2007b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	Printable Release Notes: PDF Current production documentation

New features and changes introduced in this version are:

- Simplified Syntax for Network-Creation Functions
- Automated Data Preprocessing and Postprocessing During Network Creation
- Automated Data Division During Network Creation
- New Simulink Blocks for Data Preprocessing
- Properties for Targets Now Defined by Properties for Outputs

Simplified Syntax for Network-Creation Functions

The following network-creation functions have new input arguments to simplify the network creation process:

- `newcf`
- `newff`
- `newdtdnn`
- `newelm`
- `newfftd`
- `newlin`
- `newlrn`
- `newnarx`
- `newnarxsp`

For detailed information about each function, see the corresponding reference pages.

Changes to the syntax of network-creation functions have the following benefits:

- You can now specify input and target data values directly. In the previous release, you specified input ranges and the size of the output layer instead.
- The new syntax automates preprocessing, data division, and postprocessing of data.

For example, to create a two-layer feed-forward network with 20 neurons in its hidden layer for a given a matrix of input vectors `p` and target vectors `t`, you can now use `newff` with the following arguments:

```
net = newff(p,t,20);
```

This command also sets properties of the network such that the functions `sim` and `train` automatically preprocess inputs and targets, and postprocess outputs.

In the previous release, you had to use the following three commands to create the same network:

```
pr = minmax(p);
s2 = size(t,1);
net = newff(pr,[20 s2]);
```

Compatibility Considerations

Your existing code still works but might produce a warning that you are using obsolete syntax.

Automated Data Preprocessing and Postprocessing During Network Creation

Automated data preprocessing and postprocessing occur during network creation in the Network/Data Manager GUI, Neural Network Fitting Tool GUI, and at the command line.

At the command line, the new syntax for using network-creation functions, described in “Simplified Syntax for Network-Creation Functions”, automates preprocessing, postprocessing, and data-division operations.

For example, the following code returns a network that automatically preprocesses the inputs and targets and postprocesses the outputs:

```
net = newff(p,t,20);
net = train(net,p,t);
y = sim(net,p);
```

To create the same network in a previous release, you used the following longer code:

```
[p1,ps1] = removeconstantrows(p);
[p2,ps2] = mapminmax(p1);
[t1,ts1] = mapminmax(t);
pr = minmax(p2);
s2 = size(t1,1);
net = newff(pr,[20 s2]);
net = train(net,p2,t1);
y1 = sim(net,p2)
y = mapminmax('reverse',y1,ts1);
```

Default Processing Settings

The default input `processFcns` functions returned with a new network are, as follows:

```
net.inputs{1}.processFcns = ...
              {'fixunknowns','removeconstantrows', 'mapminmax'}
```

These three processing functions perform the following operations, respectively:

- `fixunknowns`—Encode unknown or missing values (represented by `NaN`) using numerical values that the network can accept.
- `removeconstantrows`—Remove rows that have constant values across all samples.
- `mapminmax`—Map the minimum and maximum values of each row to the interval `[-1 1]`.

The elements of `processParams` are set to the default values of the `fixunknowns`, `removeconstantrows, and mapminmax` functions.

The default output `processFcns` functions returned with a new network include the following:

```
net.outputs{2}.processFcns = {'removeconstantrows','mapminmax'}
```

These defaults process outputs by removing rows with constant values across all samples and mapping the values to the interval `[-1 1]`.

`sim` and `train` automatically process inputs and targets using the input and output processing functions, respectively. `sim` and `train` also reverse-process network outputs as specified by the output processing functions.

For more information about processing input, target, and output data, see Backpropagation in the Neural Network Toolbox User's Guide documentation.

Changing Default Input Processing Functions

You can change the default processing functions either by specifying optional processing function arguments with the network-creation function, or by changing the value of `processFcns` after creating your network.

You can also modify the default parameters for each processing function by changing the elements of the `processParams` properties.

After you create a network object (`net`), you can use the following input properties to view and modify the automatic processing settings:

- `net.inputs{1}.exampleInput`—Matrix of example input vectors
- `net.inputs{1}.processFcns`—Cell array of processing function names
- `net.inputs{1}.processParams`—Cell array of processing parameters

The following input properties are automatically set and you cannot change them:

- `net.inputs{1}.processSettings`—Cell array of processing settings
- `net.inputs{1}.processedRange`—Ranges of example input vectors after processing
- `net.inputs{1}.processedSize`—Number of input elements after processing

Changing Default Output Processing Functions

After you create a network object (`net`), you can use the following output properties to view and modify the automatic processing settings:

- `net.outputs{2}.exampleOutput`—Matrix of example output vectors

- `net.outputs{2}.processFcns`—Cell array of processing function names
- `net.outputs{2}.processParams`—Cell array of processing parameters

Note These output properties require a network that has the output layer as the second layer.

The following new output properties are automatically set and you cannot change them:

- `net.outputs{2}.processSettings`—Cell array of processing settings
- `net.outputs{2}.processedRange`—Ranges of example output vectors after processing
- `net.outputs{2}.processedSize`—Number of input elements after processing

Automated Data Division During Network Creation

When training with supervised training functions, such as the Levenberg-Marquardt backpropagation (the default for feed-forward networks), you can supply three sets of input and target data. The first data set trains the network, the second data set stops training when generalization begins to suffer, and the third data set provides an independent measure of network performance.

Automated data division occurs during network creation in the Network/Data Manager GUI, Neural Network Fitting Tool GUI, and at the command line.

At the command line, to create and train a network with early stopping that uses 20% of samples for validation and 20% for testing, you can use the following code:

```
net = newff(p,t,20);
net = train(net,p,t);
```

Previously, you entered the following code to accomplish the same result:

```
pr = minmax(p);
s2 = size(t,1);
net = newff(pr,[20 s2]);
```

```
[trainV,validateV,testV] = dividevec(p,t,0.2,0.2);
[net,tr] = train(net,trainV.P,trainV.T,[],[],validateV,testV);
```

For more information about data division, see Backpropagation in the Neural Network Toolbox User's Guide documentation.

New Data Division Functions

The following are new data division functions:

- `dividerand`—Divide vectors using random indices.
- `divideblock`—Divide vectors in three blocks of indices.
- `divideint`—Divide vectors with interleaved indices.
- `divideind`—Divide vectors according to supplied indices.

Default Data Division Settings

Network creation functions return the following default data division properties:

- `net.divideFcn = 'dividerand'`
- `net.divedeParam.trainRatio = 0.6;`
- `net.divideParam.valRatio = 0.2;`
- `net.divideParam.testRatio = 0.2;`

Calling `train` on the network object net divided the set of input and target vectors into three sets, such that 60% of the vectors are used for training, 20% for validation, and 20% for independent testing.

Changing Default Data Division Settings

You can override default data division settings by either supplying the optional data division argument for a network-creation function, or by changing the corresponding property values after creating the network.

After creating a network, you can view and modify the data division behavior using the following new network properties:

- `net.divideFcn` - Name of the division function
- `net.divideParam` - Parameters for the division function

New Simulink Blocks for Data Preprocessing

New blocks for data processing and reverse processing are available. For more information, see the description of processing blocks.

The function `gensim` now generates neural networks in Simulink that use the new processing blocks.

Properties for Targets Now Defined by Properties for Outputs

The properties for targets are now defined by the properties for outputs. Use the following properties to get and set the output and target properties of your network:

- `net.numOutputs`—The number of outputs and targets
- `net.outputConnect`—Indicates which layers have outputs and targets
- `net.outputs`—Cell array of output subobjects defining each output and its target

Compatibility Considerations

Several properties are now obsolete, as described in the following table. Use the new properties instead.

Recommended Property	Obsolete Property
`net.numOutputs`	`net.numTargets`
`net.outputConnect`	`net.targetConnect`
`net.outputs`	`net.targets`

Version 5.0.2 (R2007a) Neural Network Toolbox

This table summarizes what's new in Version 5.0.2 (R2007a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports	No

Version 5.0.1 (R2006b) Neural Network Toolbox

This table summarizes what's new in Version 5.0.1 (R2006b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports	No

Version 5.0 (R2006a) Neural Network Toolbox

This table summarizes what's new in Version 5.0 (R2006a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	No

New features and changes introduced in this version are

- Dynamic Neural Networks
- Wizard for Fitting Data
- Data Preprocessing and Postprocessing
- Derivative Functions Are Obsolete

Dynamic Neural Networks

Version 5.0 now supports these types of dynamic neural networks:

Time-Delay Neural Network

Both focused and distributed time-delay neural networks are now supported. Continue to use the `newfftd` function to create focused time-delay neural networks. To create distributed time-delay neural networks, use the `newdtdnn` function.

Nonlinear Autoregressive Network (NARX)

To create parallel NARX configurations, use the `newnarx` function. To create series-parallel NARX networks, use the `newnarxsp` function. The `sp2narx` function lets you convert NARX networks from series-parallel to parallel configuration, which is useful for training.

Layer Recurrent Network (LRN)

Use the `newlrn` function to create LRN networks. LRN networks are useful for solving some of the more difficult problems in filtering and modeling applications.

Custom Networks

The training functions in Neural Network Toolbox are enhanced to let you train arbitrary custom dynamic networks that model complex dynamic systems. For more information about working with these networks, see the Neural Network Toolbox documentation.

Wizard for Fitting Data

The new Neural Network Fitting Tool is now available to fit your data using a neural network. The Neural Network Fitting Tool is designed as a wizard and walks you through the data-fitting process step by step.

To open the Neural Network Fitting Tool, type the following at the MATLAB prompt:

```
nftool
```

Data Preprocessing and Postprocessing

Version 5.0 provides the following new data preprocessing and postprocessing functionality:

dividevec Automatically Splits Data

The `dividevec` function facilitates dividing your data into three distinct sets to be used for training, cross validation, and testing, respectively. Previously, you had to split the data manually.

fixunknowns Encodes Missing Data

The `fixunknowns` function encodes missing values in your data so that they can be processed in a meaningful and consistent way during network training. To reverse this preprocessing operation and return the data to its original state, call `fixunknowns` again with `'reverse'` as the first argument.

removeconstantrows Handles Constant Values

`removeconstantrows` is a new helper function that processes matrices by removing rows with constant values.

mapminmax, mapstd, and processpca Are New

The `mapminmax`, `mapstd`, and `processpca` functions are new and perform data preprocessing and postprocessing operations.

Compatibility Considerations. Several functions are now obsolete, as described in the following table. Use the new functions instead.

New Function	Obsolete Functions

`mapminmax`	`premnmx`
	`postmnmx`
	`tramnmx`

`mapstd`	`prestd`
	`poststd`
	`trastd`

`processpca`	`prepca` `trapca`

Each new function is more efficient than its obsolete predecessors because it accomplishes both preprocessing and postprocessing of the data. For example, previously you used `premnmx` to process a matrix, and then `postmnmx` to return the data to its original state. In this release, you accomplish both operations using `mapminmax`; to return the data to its original state, you call `mapminmax` again with `'reverse'` as the first argument:

```
mapminmax('reverse',Y,PS)
```

Derivative Functions Are Obsolete

The following derivative functions are now obsolete:

```
ddotprod
dhardlim
dhardlms
dlogsig
```

```
dmae
dmse
dmsereg
dnetprod
dnetsum
dposlin
dpurelin
dradbas
dsatlin
dsatlins
dsse
dtansig
dtribas
```

Each derivative function is named by prefixing a `d` to the corresponding function name. For example, `sse` calculates the network performance function and `dsse` calculated the derivative of the network performance function.

Compatibility Considerations

To calculate a derivative in this version, you must pass a derivative argument to the function. For example, to calculate the derivative of a hyperbolic tangent sigmoid transfer function `A` with respect to `N`, use this syntax:

```
A = tansig(N,FP)
dA_dN = tansig('dn',N,A,FP)
```

Here, the argument `'dn'` requests the derivative to be calculated.

Version 4.0.6 (R14SP3) Neural Network Toolbox

This table summarizes what's new in Version 4.0.6 (R14SP3):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports	No

Compatibility Summary for Neural Network Toolbox

This table summarizes new features and changes that might cause incompatibilities when you upgrade from an earlier version, or when you use files on multiple versions. Details are provided with the description of the new feature or change.

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version V5.1 (R2007b)	See the **Compatibility Considerations** subheading for this new feature or change: • Simplified Syntax for Network-Creation Functions • Properties for Targets Now Defined by Properties for Outputs
V5.0.2 (R2007a)	None
V5.0.1 (R2006b)	None
V5.0 (R2006a)	See the **Compatibility Considerations** subheading for this new feature or change: • mapminmax, mapstd, and processpca Are New • Derivative Functions Are Obsolete
V4.0.6 (R14SP3)	None