Filter Design Toolbox Release Notes

The "Filter Design Toolbox 3.0 Release Notes" on page 1-1 describe the changes introduced in the Filter Design Toolbox 3.0. The following topics are discussed in these Release Notes:

The Filter Design Toolbox Release Notes also provide information about version 2.5, in case you are upgrading from a version that was released prior to Release 13 with Service Pack 1—"Filter Design Toolbox 2.5 Release Notes" on page 2-1.

Printing the Release Notes

If you would like to print the Release Notes, you can link to a PDF version.

Contents

1

Filter Design Toolbox 3.0 Release Notes

New Features

This section introduces the new features and enhancements added to the Filter Design Toolbox 3.0 since Version 2.5.

If you are upgrading from a release earlier than Release 13 with Service Pack 1, see "New Features" on page 2-2 in the Filter Design Toolbox 2.5 Release Notes.

Dfilt Objects for Quantized and Fixed-Point Filters

New objects called discrete-time filter objects, `dfilt` objects, replace the `qfilt` objects in the earlier releases. `dfilt` objects in this toolbox extend the `dfilt` objects found in the Signal Processing Toolbox by adding an `Arithmetic` property that defines the filter arithmetic. While `qfilt` objects continue to work, we recommend using the new `dfilt` objects moving forward. qfilt objects will stop working in the future.

In addition to adding/extending the `dfilt` objects for fixed-point filtering, the methods associated with filter analysis, like `grpdelay` or `zerophase`, are modified to work with dfilt objects as input arguments.

New Quantization Panel in FDATool

To support the new filter objects, the quantization panel in Filter Design and Analysis tool (FDATool) is different, with different options and settings to match the new `dfilt` objects.

New Multirate Filter Design Panel in FDATool

Now you can design multirate filters in FDATool. A new design panel, accessed from the side bar icons in FDATool, switches FDATool to multirate filter design mode.

Added Filter Design Object for Designing Filters

Another new object, referred to as a filter design object, `fdesign`, provides a different way to design filters. Using a function `fdesign.`*`type`*, where *`type`* is one of

- `bandpass`
- `bandstop`

- `decim`
- `halfband`
- `highpass`
- `interp`
- `lowpass`
- `nyquist`
- `src`

you design an object that specifies the type of filter response to implement. Then you use a design method, such as

- `butter`
- `cheby1`
- `cheby2`
- `ellip`
- `equiripple`
- `kaiserwin`
- `designmethods`—returns the design methods that apply to a given `fdesign` object.

to implement the filter. You can read more about these new methods in the reference pages for the toolbox.

New Methods for Fixed-Point Filters

We added a number of new methods for working with the new `dfilt` objects. More information about each method is provided in the online Help system. The new methods are:

- `coewrite`—write a .coe file
- `double`—cast a filter to a double arithmetic form
- `norm`—return the L2-norm of a digital filter
- `reorder`—rearrange the sections of an SOS filter
- `scalecheck`—check the scaling of an SOS filter
- `specifyall`—Fully specify fixed-point filter settings
- `setspecs`—set the specifications for an `fdesign` object.
- `noisepsd`—replaces the `nlm` function for estimating filter response.
- `noisepsdopts`—lets you create an object used by `noisepsd`.
- `scale`—controls and configures the scaling for a fixed-point `dfilt` object.

- `scaleopts`—lets you create an object to use with `scale`.
- `normalize`—normalizes the filter coefficients in a `dfilt` object.
- `cumsec`—lets you view the sections that compose a second-order sections filter. You can view the sections one at a time, or cumulatively.
- `denormalize`—reverses the effects of applying `normalize` to a `dfilt` object.
- `reffilter`—returns a `dfilt` object that has the reference (floating-point) filter coefficients associated with the fixed-point `dfilt` object.

New FIR Filter Design Function fircband

A new design function `fircband` designs equiripple FIR filters while constraining one or more bands of the filter to have a maximum ripple value.

New SOS Scaling and SOS Section Reordering Capability in FDATool

In the **Edit** menu of FDATool you will find a new option for working with SOS filters. The new option--**Reorder and Scale Second-Order Sections**--lets you change the scaling applied to the sections of an SOS filter, and move the sections to different orders to investigate how that changes your filter performance or needs.

To use the new option, use FDATool to create or import an SOS filter, then select **Edit->Reorder and Scale Second-Order Sections** from the menu bar. For more information refer to your Filter Design Toolbox documentation.

New SOS View capability in FDATool and FVTool

On the **View** menu in FDATool and FVTool, you will find a new option—**SOS View**. When you have an SOS filter in either tool, **SOS View** lets you see the performance of the sections of the filter—individually, cumulatively, or overall. For more information, refer to “Viewing SOS Filter Sections” in the *Filter Design Toolbox User’s Guide*. Using the new viewing tool is the same in FDATool and FVTool.

Major Bug Fixes

The Filter Design Toolbox 3.0 includes several bug fixes made since Version 2.5. This section describes the particularly important Version 3.0 bug fixes. In addition, more bug fixes appear below.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

Improved ifir FIR filter design

Both the convergence and the accuracy of filters designed by `ifir` have been improved.

firgr (formerly gremez) Improved Convergence

The algorithm used in `firgr` now results in improved convergence in the designs. In addition, the **`minorder`** design option is faster.

Corrected firnyquist Decay Option

The decay option in `firnyquist` now provides a true slope parameter for the stopband. The decay parameter correctly sets the slope of decay for the stopband in db/(rad/sample).

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving to the Filter Design Toolbox 3.0 from Version 2.5.

If you are upgrading from a release earlier than Release 13 with Service Pack 1, see "Upgrading from an Earlier Release" on page 2-12 in the Filter Design Toolbox 2.5 Release Notes.

Fixed-Point Filters Require Fixed-Point Toolbox

To use the fixed-point filter capability in the Filter Design Toolbox 3.0, you must install the Fixed-Point Toolbox 1.0.

Qfilt Objects and Functions Obsolete

This toolbox no longer includes the `qfilt` objects. They are now obsolete. While they still work, they are superseded by the `dfilt` objects.

No FDATool Option To Turn Quantization On or Off

FDATool no longer requires you to turn quantization on to see your quantized filter. Setting the `Arithmetic` property for a filter to `Fixed-point` enables quantization. Use the new quantization panel to change from double-precision filters to fixed-point or single-precision filters.

Optimization/Scaling Dialog Removed from FDATool

A new scaling and optimization process removes the older optimization option from the quantization panel in FDATool.

Quantizer Objects Moved to the Fixed-Point Toolbox

Quantizers have been removed from this toolbox. They are part of the Fixed-Point Toolbox now. Your existing code continues to work.

QFFT Objects and Functions Removed from the Toolbox

We removed the quantized FFT objects, `qfft`, from the toolbox. We have not replaced the objects at this time. Your existing `qfft` code continues to work.

Unit Quantizer Objects and Functions Removed from the Toolbox

While your existing unit quantizer code continues to work, the unit quantizer object is no longer part of the toolbox. You can find it in the Fixed-Point Toolbox.

No dfilt limitcycle in the Toolbox

Discrete-time filters—`dfilt` objects—do not work with `limitcycle` so we have removed `limitcycle` from the toolbox. To estimate the magnitude response for a filter, import into or design the filter in FDATool and use the Magnitude Response Estimate option in the **Analysis** menu in FDATool.

Function coewrite Applies to dfilt.dffir Filter Objects

In version 3, `coewrite` accepts only fixed-point `dfilt.dffir` filter objects as input to create `.coe` files. Other `dfilt` objects do not work and generate an error in MATLAB. In programming terms, `coewrite` is a method of `dfilt.dffir` objects whose `Arithmetic` property is set to `fixed` only.

cicdecimate And cicinterpolate Now Obsolete

With this release, the functions `cicdecimate` and `cicinterpolate` are obsolete. Please use the new `mfilt` objects, such as `mfilt.cidecim` or `mfilt.cicinterp`.

Function gremez Renamed firgr

We renamed the filter design function `gremez` to `firgr`. The new name more closely identifies the design function as using a generalized Remez algorithm to design FIR filters, and better matches the current FIR filter design function naming scheme in the toolbox.

Multirate Filters Remove Trailing Zero-Valued Coefficients

When assigning a numerator to a multirate filter (`mfilt` objects), trailing zeros in the numerator are removed. We do this to avoid getting the wrong order for the polyphase components when using Nyquist filters. It also improves efficiency in the filtering.

In version 2.5, the following was the multirate filter standard:

```
hm=mfilt.firinterp(5)

hm =

         FilterStructure: [1x38 char]
               Numerator: [1x121 double]
     InterpolationFactor: 5
    ResetBeforeFiltering: 'on'
                  States: [24x1 double]
     NumSamplesProcessed: 0

hm.numerator(end)

ans =

 -1.0429e-019
```

Notice that the `Numerator` property reports 121 coefficients and the final coefficient is very small (nearly zero). In version 3.0, deleting the trailing zeros generates the following result for the same filter, where the `Numerator` property is now 120 coefficients and the spurious coefficient has been removed:

```
hm=mfilt.firinterp(5)

hm =

        FilterStructure: 'Direct-Form FIR Polyphase Interpolator'
              Numerator: [1x120 double]
     InterpolationFactor: 5
    ResetBeforeFiltering: 'on'
                  States: [23x1 double]
     NumSamplesProcessed: 0
```

Known Software and Documentation Problems

This section includes a link to a description of known software and documentation problems in Version 3.0.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

Filter Design and Analysis Tool

To get help on a control in the dialog, use the Help browser. Select an option, such as **FDATool Help**, from the **Help** menu in FDATool. Or type `doc` at the MATLAB prompt to open the online help system.

Documentation Still Shows the Deleted Objects Qfft, Qfilt, Quantizer

For this release, the online documentation still shows some instances of the now obsolete objects `qfilt`, `qfft`, and `quantizers`. We will correct this for the next release. In addition, some references to the methods and functions associated with the objects remain as well.

Documentation Shows Instances of the Filter Property InheritSettings

Although you may notice references to a `dfilt` object property `InheritSettings`, the property is not included in `dfilt` objects at this time. This does not affect any discrete-time filters you create.

Certain FDATool Panes Do Not Have "What's This" Help

For this release, the context-sensitive help, or What's This help, for the quantization panel, multirate design panel, SOS View dialog, and Reordering and Scaling of Second-Order Sections dialog is not available. For help about these features, refer to your Filter Design Toolbox documentation online.

2

Filter Design Toolbox 2.5 Release Notes

New Features

This section introduces the new features and enhancements added to the Filter Design Toolbox 2.5 since Version 2.2 (Release 13).

If you are upgrading from a release earlier than Release 13, you should read "New Features" on page 3-2 in the Filter Design Toolbox 2.2 Release Notes.

New Object-Based Adaptive Filters

One of the most important additions to the toolbox is new filter objects for applying adaptive filters to data. Like qfilt objects, you construct the object using the appropriate constructor, and then apply your new object to your data. With more than 30 new adaptive filter types, we group them here by the fundamental algorithm each uses for adaptation.

Algorithms

For adaptive filter (adaptfilt) objects, the *`algorithm`* string determines which adaptive filter algorithm your adaptfilt object implements. Each available algorithm entry appears in one of the following tables along with a brief description of the algorithm. Click on the algorithm in the first column to get more information about the associated adaptive filter technique.

Least Mean Squares (LMS) Based FIR Adaptive Filters

adaptfilt.algorithm String	Description of the Adapting Algorithm Used to Generate Filter Coefficients During Adaptation
`adaptfilt.adjlms`	Use the adjoint LMS FIR adaptive filter algorithm
`adaptfilt.blms`	Use the block LMS FIR adaptive filter algorithm
`adaptfilt.blmsfft`	Use the FFT-based block LMS FIR adaptive filter algorithm
`adaptfilt.dlms`	Use the delayed LMS FIR adaptive filter algorithm
`adaptfilt.filtxlms`	Use the filtered-x LMS FIR adaptive filter algorithm
`adaptfilt.lms`	Use the LMS FIR adaptive filter algorithm
`adaptfilt.nlms`	Use the normalized LMS FIR adaptive filter algorithm
`adaptfilt.sd`	Use the sign-data LMS FIR adaptive filter algorithm
`adaptfilt.se`	Use the sign-error LMS FIR adaptive filter algorithm
`adaptfilt.ss`	Use the sign-sign LMS FIR adaptive filter algorithm

For further information about an adapting algorithm, refer to the reference page for the algorithm.

Recursive Least Squares (RLS) Based FIR Adaptive Filters

adaptfilt.algorithm String	Description of the Adapting Algorithm Used to Generate Filter Coefficients During Adaptation
`adaptfilt.ftf`	Use the fast transversal least squares adaptation algorithm
`adaptfilt.qrdrls`	Use the QR-decomposition RLS adaptation algorithm
`adaptfilt.hrls`	Use the householder RLS adaptation algorithm
`adaptfilt.hswrls`	Use the householder sliding window SWRLS adaptation algorithm
`adaptfilt.rls`	Use the recursive least squares (RLS) adaptation algorithm
`adaptfilt.swrls`	Use the sliding window (SW) RLS adaptation algorithm
`adaptfilt.swftf`	Use the sliding window FTF adaptation algorithm

For more complete information about an adapting algorithm, refer to the reference page for the algorithm.

Affine Projection (AP) FIR Adaptive Filters

adaptfilt.algorithm String	Description of the Adapting Algorithm Used to Generate Filter Coefficients During Adaptation
`adaptfilt.ap`	Use the affine projection (AP) algorithm that uses direct matrix inversion
`adaptfilt.apru`	Use the affine projection (AP) algorithm that uses recursive matrix updating
`adaptfilt.bap`	Use the block affine projection (AP) adaptation algorithm

To find more information about an adapting algorithm, refer to the reference page for the algorithm.

FIR Adaptive Filters in the Frequency Domain (FD)

adaptfilt.algorithm String	Description of the Adapting Algorithm Used to Generate Filter Coefficients During Adaptation
`adaptfilt.fdaf`	Use the frequency domain (FD) adaptation algorithm
`adaptfilt.pbfdaf`	Use the partition block version of the FDAF algorithm
`adaptfilt.pbufdaf`	Use the partition block unconstrained version of the FDAF algorithm
`adaptfilt.tdafdct`	Use the transform domain adaptation algorithm using DCT
`adaptfilt.tdafdft`	Use the transform domain adaptation algorithm using DFT
`adaptfilt.ufdaf`	Use the unconstrained FDAF algorithm for adaptation

For more information about an adapting algorithm, refer to the reference page for the algorithm.

Lattice Based (L) FIR Adaptive Filters

adaptfilt.algorithm String	Description of the Adapting Algorithm Used to Generate Filter Coefficients During Adaptation
`adaptfilt.gal`	Use the gradient adaptive lattice filter adaptation algorithm
`adaptfilt.lsl`	Use the least squares lattice adaptation algorithm
`adaptfilt.qrdlsl`	Use the QR decomposition LSL adaptation algorithm

For more information about an adapting algorithm, refer to the reference page for the algorithm.

New Object-Based Multirate Filters

Along with new adaptive filter object, Version 2.5 adds many multirate filter objects as well. In addition to the existing `cicdecimate` and `cicinterpolate` functions already available, you now have the following multirate filtering objects at hand, as shown in this table.

Multirate Filter Object	Filter Object Description
`mfilt.cicdecim`	Construct a cascaded integrator-comb decimation filter object
`mfilt.cicdecimzerolat`	Construct a cascaded integrator-comb decimation filter object that does not display latency
`mfilt.cicinterp`	Construct a cascaded integrator-comb interpolation filter object
`mfilt.cicinterpzerolat`	Construct a cascaded integrator-comb interpolation filter object that does not display latency

Multirate Filter Object	Filter Object Description
`mfilt.fftfirinterp`	Construct an overlap-add FIR polyphase interpolation filter object
`mfilt.firdecim`	Construct a direct-form FIR polyphase decimation filter object
`mfilt.firfracdecim`	Construct a direct-form FIR polyphase fractional decimation filter object
`mfilt.firfracinterp`	Construct a direct-form FIR polyphase fractional interpolation filter object
`mfilt.firinterp`	Construct a direct-form FIR polyphase interpolation filter object
`mfilt.firsrc`	Construct a direct-form FIR polyphase sample rate conversion filter object
`mfilt.firtdecim`	Construct a direct-form, transposed FIR polyphase decimation filter object
`mfilt.holdinterp`	Construct an FIR interpolation filter object that uses “hold” interpolation between input samples
`mfilt.linearinterp`	Construct an FIR linear interpolation filter object that applies linear interpolation between input samples

Updated and New Methods for Adaptive Filter and Multirate Filter Objects

To enable you to work with the new filter objects in the toolbox, Version 2.5 includes the following new methods that support adaptive filter and multirate

filter objects. In addition, existing filter analysis methods such as `freqz` work with the new objects just as they do for qfilt and dfilt objects.

Method Name	Supported Objects	Description
`block`	Multirate (some)	Generate a DSP Blockset block that duplicates the filter object. Works only when DSP Blockset is installed. Some multirate objects cannot be modelled as blocks since the blockset does not provide blocks for all the multirate filters in the toolbox.
`coefficients`	Multirate	Return the multirate filter objects coefficients.
`euclidfactors`	Multirate	Use Euclid's theorem to determine the integer factors for an `mfilt` object.
`msepred`	Adaptive filter	Return the predicted mean-square error for the `adaptfilt` object.
`msesim`	Adaptive filter	Return the measured mean-square error for the `adaptfilt` object via simulation.
`maxstep`	Adaptive filter	Return the maximum step size for an `adaptfilt` object.
`nstates`	Multirate filter	Return the number of states in an `mfilt` object.
`polyphase`	Multirate filter	Return the polyphase matrix for an `mfilt` object.

To see the full listing of analysis methods that apply to the new adaptfilt and mfilt objects, type `help adaptfilt` and `help mfilt` at the MATLAB prompt.

The Filter Visualization Tool (FVTool) supports these new objects so you can use the full power of FVTool to analyze the objects you create.

New Demos for Adaptive Filters and Multirate Filters

To help you learn about the new objects in the toolbox, Filter Design Toolbox 2.5 includes many new demos to introduce the new adaptive and multirate objects. To see the new demos, select **Demos** from the **Help** menu in MATLAB. In the Help browser, look under **Toolboxes** for the **Filter Design** entry.

Import XILINX Coefficient (.COE) Files with FDATool

Version 2.2 of the toolbox introduced the ability to read and write XILINX coefficients files from the MATLAB command line with `coeread` and `coewrite`. Now you can import XILINX coefficients (.COE) files into FDATool to create quantized filters directly using the imported filter coefficients.

To use the new import feature:

1. Select **File->Import Filter From XILINX Coefficient (.COE) File** in FDATool.

2. In the **Import Filter From XILINX Coefficient (.COE) File** dialog, find and select the `.coe` file to import.

3. Click **Open** to dismiss the dialog and start the import process.

   FDATool imports the coefficient file and creates a quantized, single-section, direct-form FIR filter.

Major Bug Fixes

The Filter Design Toolbox 2.5 includes several bug fixes made since Version 2.2. This section describes the particularly important Version 2.5 bug fixes.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving to the Filter Design Toolbox 2.5 from Version 2.2.

Previous Adaptive Filter Functions Now Obsolete

The following adaptive filter functions no longer appear in the toolbox. When you use one of these functions, you get an error suggesting that you replace your function with one of the adaptive filter objects that are new in Version 2.5.

- `adaptkalman`
- `adaptlms`
- `adaptrls`
- `adaptsd`
- `adaptse`
- `adaptss`

References to these functions have been removed from the documentation, although they will continue to work until a future release. There is no replacement object for the `adaptkalman` function.

Uninstall Previous Versions Before Installing Version 2.5

Because Version 2.5 of the toolbox relocates some files to improve performance and consistency, you must uninstall all earlier toolbox versions of the Filter Design Toolbox before you install Version 2.5. In addition, you must remove earlier versions of the Signal Processing Toolbox and install the latest version, Signal Processing Toolbox 6.1. Refer to the release notice for the Signal Processing Toolbox 6.1 for information about installing the Signal Processing Toolbox 6.1.

Caution Uninstalling the toolbox removes all the files in the toolbox directory `$MATLAB/toolbox/filterdesign`, including all files you may have stored there. Before performing the uninstall operation, make backup copies in another directory of all the files to save.

Uninstalling Previous Versions of the Filter Design Toolbox

To uninstall previous versions of the Filter Design Toolbox, perform these steps.

1. Close MATLAB. You cannot uninstall toolboxes while MATLAB is running.
2. Do one of the following depending on whether you are removing the toolbox from a Microsoft Windows® platform or a UNIX platform.
   - On Microsoft Windows platforms—use the MATLAB Uninstaller. Select **Programs->MATLAB 6.5->Uninstall MATLAB 6.5** from the **Start** Menu. Select **Filter Design Toolbox** from the **Uninstall Product List** and click **OK**.
   - On UNIX platforms, use the following command to remove the toolbox

     ```
     rm -rf $MATLAB/toolbox/filterdesign
     ```

     where `$MATLAB` in the full path name of the root directory where you installed MATLAB.

Changed Calattice and Calatticepc Dfilt Objects for Version 2.5

In this release of the toolbox, we changed the `dfilt.calattice` and `dfilt.calatticepc` filter objects. Older versions will no longer work. With the new objects, the `Allpass1` and `Allpass2` properties now store a vector of filter coefficients rather than a `dfilt.latticeallpass` object as they did in earlier versions.