SoundMex Plugins Documentation

Realtime plugins for SoundMex

http://www.soundmex.de

User Manual

HörTech
Kompetenzzentrum für
Hörgeräte-Systemtechnik

License agreement

IMPORTANT- PLEASE READ CAREFULLY:

BY INSTALLING THE SOFTWARE (AS DEFINED BELOW), COPYING THE SOFTWARE AND/OR CLICKING ON THE "ACCEPT" BUTTON BELOW, YOU (EITHER ON BEHALF OF YOURSELF AS AN INDIVIDUAL OR ON BEHALF OF AN ENTITY AS ITS AUTHORIZED REPRESENTATIVE) AGREE TO ALL OF THE TERMS OF THIS END USER LICENSE AGREEMENT ("AGREEMENT") REGARDING YOUR USE OF THE SOFTWARE. IF YOU DO NOT AGREE WITH ALL OF THE TERMS OF THIS AGREEMENT, DO NOT INSTALL AND/OR USE THE SOFTWARE.

DEFINITIONS

The term "Software" includes all software distributed with this License including all documentation. The "Software" is licensed to you under the terms specified in the License Grant below.

HIGH RISK ACTIVITIES

The Software is not fault-tolerant and is not designed, manufactured or intended for use as on-line control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or other medical devices, or weapons systems, in which the failure of the Software could lead directly to death, personal injury, or severe physical or environmental damage ("High Risk Activities"). HörTech gGmbH and its suppliers specifically disclaim any express or implied warranty of fitness for High Risk Activities.

OWNERSHIP AND COPYRIGHT

This Software is owned by HörTech gGmbH or its suppliers and is protected by copyright law and international copyright treaty. Therefore you must treat this Software like any other copyrighted material. You acknowledge that no title to the intellectual property in the Software is transferred to you. Title, ownership, rights, and intellectual property rights in and to the Software shall remain in HörTech gGmbH.

LICENSE GRANT

Subject to the license terms, HörTech gGmbH hereby grants you a non-exclusive, non-transferable (except under the terms below) license to install and to use the Software under the terms of this license. Except as provided in this license agreement, you may not transfer, rent, lease, lend, copy, modify, translate, sublicense, time-share or electronically transmit the Software. You may only either make one copy of the Software solely for backup or archival purposes or transfer the Software to a single hard disk provided you keep the original solely for backup or archival purposes. you agree not to modify the Software or attempt to decipher, de-compile, disassemble or reverse engineer the Software, except to the extent applicable laws specifically prohibit such restriction.

LICENSE TRANSFER

You may transfer your license and the rights granted in the license to a third party only if a) the third party agrees to this license agreement, b) you completely uninstall and delete all copies of this Software, c) all parts of the Software and its distribution are transferred to the third party and d) the transfer includes the current version and all prior versions of the Software.

DISCLAIMER OF WARRANTY

THIS SOFTWARE IS SOLD "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OF MERCHANTABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF NONINFRINGEMENT OF THIRD PARTY RIGHTS, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. GOOD DATA PROCESSING PROCEDURE DICTATES THAT ANY PROGRAM BE THOROUGHLY TESTED WITH NON-CRITICAL DATA BEFORE RELYING ON IT. THE USER MUST ASSUME THE ENTIRE RISK OF USING THE PROGRAM. ANY LIABILITY OF THE SELLER WILL BE LIMITED EXCLUSIVELY TO PRODUCT REPLACEMENT OR REFUND OF PURCHASE PRICE. Under and restricted by the above terms, HörTech gGmbH warrants that the Software, as updated and when properly used, will perform substantially in accordance with its accompanying documentation, and the Software media will be free from defects in materials and workmanship. The limited warranty is void if the Software fails as a result of accident, abuse, misapplication or modification. LIMITATION OF LIABILITY You must assume the entire risk of using the Software. IN NO EVENT SHALL HörTech gGmbH BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND ARISING OUT OF THE USE OF THE HörTech gGmbH's SOFTWARE, EVEN IF HörTech gGmbH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL HörTech gGmbH's LIABILITY FOR ANY CLAIM, WHETHER IN CONTRACT, TORT, OR ANY OTHER THEORY OF LIABILITY, EXCEED THE LICENSE FEE PAID BY YOU. THIS LIMITATION SHALL APPLY TO CLAIMS OF PERSONAL INJURY TO THE EXTENT PERMITTED BY LAW.

1 Contents

2 Introduction

This documentation describes the configuration of and communication with the generic SoundMex 2 Plugins shipped with SoundMex. For a description how to activate plugins within a so called *plugin chain* please refer to the chapter 'The SoundMex realtime DSP-Plugin-Pipe' in the main SoundMex documentation.

The following example configurations are using a plugin chain configuration file 'plugin.ini' that is used in the example plugin.m in the examples directory of SoundMex:

```
[0]
FileName=..\\plugins\\visualize.dll
IniFile=$visualize.ini

[1]
FileName=..\\plugins\\olaeq.dll
EQFile=..\\plugins\\test.flt
OLAFFTLength=1024
OLAWindowLength=800
Active=0

[2]
FileName=..\\plugins\\visualize.dll
IniFile=$visualize.ini
```

After initializing SoundMex you can use the *sendplugin* command to communicate with a plugin. String commands can be sent to a single plugin, but the plugin itself must 'recognize' the command. In the example above, you can activate and deactivate the overlapped add equalizer plugin OLAEQ.DLL on the fly using the command

```
soundmex2('sendplugin', 'command', 'active=1', 'index',1); %activation
soundmex2('sendplugin', 'command', 'active=0', 'index',1); %deactivation
```

assuming the plugin chain defined above was activated for wave out device No. 0 (i.e. the file plugin0.ini from 'Activating a plugin chain' is used). For a list of recognized commands of the generic plugins please refer to the *SoundMex DSP-Plugins* documentation.

Important note: the indices of the plugins are not necessarily equal to the section names in your inifile: they start with 0 for the first plugin are strictly ascending, i.e. incremented by 1 for each plugin in the chain.

You can send a special command string to a plugin to get access to a MATLAB ® vector from a plugin: the command 'data:x' (x may be any MATLAB ® vector) will be translated to a string containing a pointer to the first double value of vector x as long int, the total number of double values contained in x and the name of the vector, e.g. the command

```
x = zeros(2, 44100);
```

```
soundmex2('sendplugin', 'command', 'data:x’);
```

will result in a string command passed to plugin with index 0 for waveout device 0

```
'data:12345678:88200:x’
```

(assuming that ‘12345678’ is the pointer to the first value). A plugin may use these values, please refer to the documentation of the particular plugin. **ATTENTION**: a plugin may access the data of the MATLAB ® workspace directly. So depending on the plugin, any change of the corresponding vector within MATLAB ® may cause access violations within the plugin!

3 The Visualization Plugin

The Visualization Plugin visualize.dll is an extension plugin to the regular visualization of playback or recording data in SoundMex. It can show level, spectrum, spectrogram and time signal of wave data within a plugin chain:

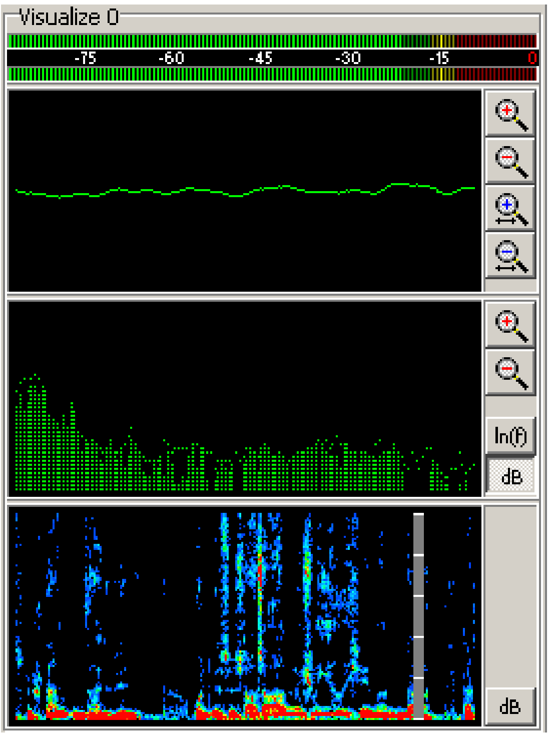


Figure 1

The plugin is used in the examples pluginеq.m and pluginceq.m. The configuration section for this plugin within a plugin chain configuration Inifile can contain the following values:

```
FileName=..\\plugins\\visualize.dll
IniFile=$visualize.ini
```

Name	Description	Values
FileName	Full filename with path of the plugin's binary	
IniFile	Name of ini file for visualization plugin (see below)	filename. If filename starts with a '$' the file will be loaded from the directory of the plugin visualize.dll

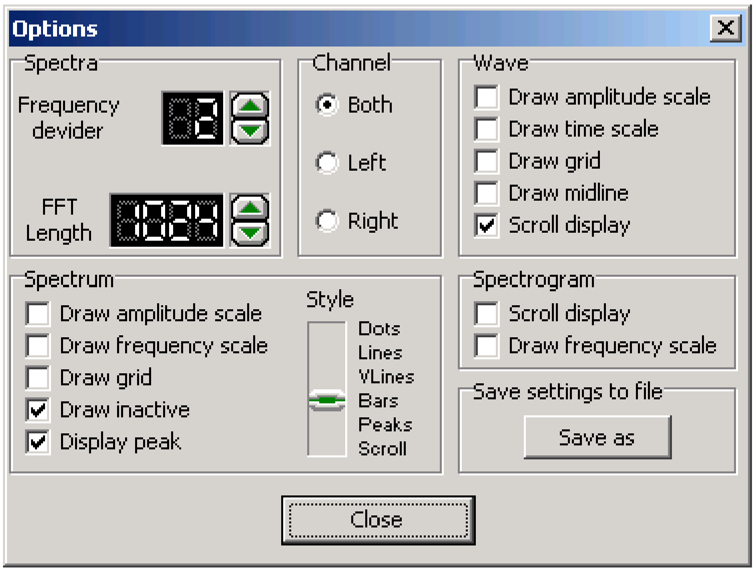
Table 1

A right mouse click on the spectrum, spectrogram or time signal shows a popup menu with the items ‘Undock’ and ‘Options’. Selecting ‘Undock’ or a double mouse click on spectrum, spectrogram or time signal will ‘release’ the window from the plugin chain and shows the visualization as single sizeable window. The size of each part of the visualization can be adjusted with the mouse. Closing the sizeable window (Alt + F4 or the cross button on the top right) will dock the window into the plugin chain again.

Use the buttons with the magnifying glasses to zoom horizontally or vertically into spectrum or waveform. If the waveform freezes, please click the horizontal ‘Zoom In’ (blue magnifying glass with the ‘+’) until it shows up again. Note: the magnifying glasses for the waveform have no function if you have selected ‘Scroll display’ in the options dialog (see below).

The ‘dB’ buttons toggle linear/logarithmic view of amplitudes/levels, the ‘ln(f)’ button toggles linear/logarithmic view of the spectrum frequency axis.

The properties of the visualization can be customized calling the options dialog by selecting the ‘Options’ command from the context menu (see the resulting view in Figure 3):



Options

Spectra
Frequency devider
FFT Length

Channel
● Both
○ Left
○ Right

Wave
□ Draw amplitude scale
□ Draw time scale
□ Draw grid
□ Draw midline
☑ Scroll display

Spectrum
□ Draw amplitude scale
□ Draw frequency scale
□ Draw grid
☑ Draw inactive
☑ Display peak

Style
Dots
Lines
VLines
Bars
Peaks
Scroll

Spectrogram
□ Scroll display
□ Draw frequency scale

Save settings to file
Save as

Close

Figure 2

The options are described in Table 2.

All changes are saved if an inifile was specified in the plugins configuration section. On clicking ‘Save as’ you are prompted for a filename to save the settings to. After using ‘Save as’ all further changes will automatically be written to the selected file.

All these settings are read on startup from the optional Inifile specified in the plugins configuration section in the plugin chain’s Inifile. The actual zoom and gain factors and actual the lin/log settings are saved as well. Some more options are written to and read from the inifile, see Table 3. So you can customize an inifile for your needs (see the example inifile ‘visualize.ini’ in the plugin directory of your SoundMex installation).

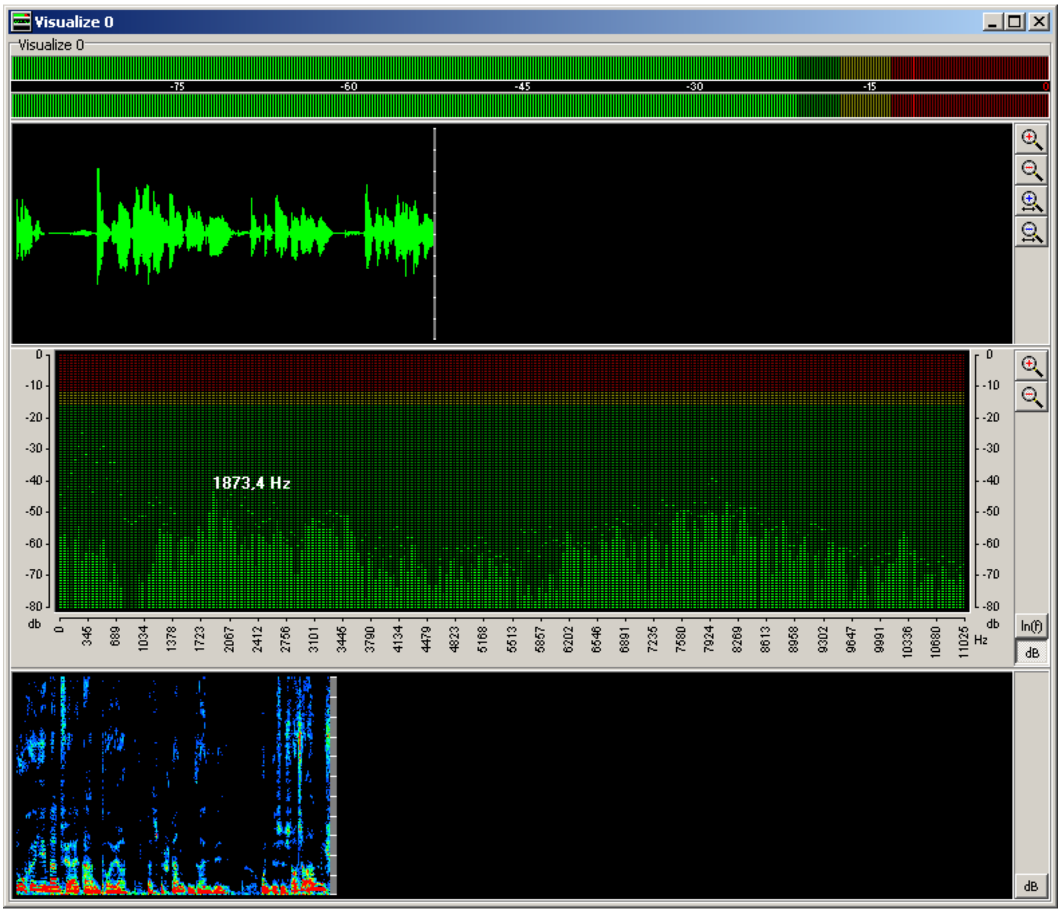


Figure 3

Group	Option	Description
Spectra	Frequency devider	Restricts maximum frequency shown in spectrum and spectrogram, calculated by SamplingFrequency/FrequencyDevider
Spectra	FFT Length	FFT Length used for spectrum and spectrogram. Possible values are restricted by the buffer length of the corresponding wave device. The value might be decreased automatically after starting the device (if buffer length of device is small)
Channel	Both, Left, Right	Select the wave channel to be shown as wave, spectrum and spectrogram
Wave	Draw amplitude scale	Shows relative amplitude scale (insensitive to vertical scale)
Wave	Draw time scale	Shows amplitude scale in ms
Wave	Draw grid	Shows ‘oscilloscope-like’ grid
Wave	Draw midline	Shows a ‘zero-line’
Wave	Scroll display	scrolls the visualization of the waveform in a large timescale
Spectrum	Draw amplitude scale	Shows amplitude scale of spectrum
Spectrum	Draw frequency scale	Shows frequency scale in Hz

Spectrum	Draw grid	Shows ‘oscilloscope-like’ grid
Spectrum	Draw inactive	Draws ‘active’ and ‘inactive’ values with different colors (only for styles ‘Bars’ and ‘Peaks’
Spectrum	Display peak	Shows numerical value of one or more spectral peaks
Spectrum	Style	Different visualization styles. Just try it out…
Spectrogram	Scroll display	When reaching the right border, the spectrogram is scrolled. If not checked, the spectrogram is cleared and restarted at the left border.
Spectrogram	Draw frequency scale	Shows frequency scale in kHz
Save…	Save as	Prompts for an inifile name to save actual settings to

Table 2

The following additional values are read from the inifle:

Section	Field	Description	Value(s)
General	Left	Undocked left position on the screen in pixels	integer
General	Top	Undocked top position on the screen in pixels	integer
General	Right	Undocked right position on the screen in pixels	integer
General	Bottom	Undocked bottom position on the screen in pixels	integer
General	LevelHeight	Undocked height of the level visualization in pixels	integer
General	WaveHeight	Undocked height of the waveform visualization in pixels	integer
General	SpectrumHeight	Undocked height of the spectrum visualization in pixels	integer
Level	Hide	Show or hide level visualization section	0 or 1
Wave	Hide	Show or hide waveform visualization section	0 or 1
Spectrum	Hide	Show or hide spectrum visualization section	0 or 1
Spectrogram	Hide	Show or hide spectrogram visualization section	0 or 1

Table 3

The Visualization Plugin recognizes the following string commands at runtime (in the examples it is assumed, that the plugin is plugged into a plugin chain for waveout device 0 at position 1)

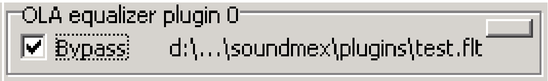
```
soundmex2('sendplugin', 'type', 'waveout', 'command', 'show=1', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'show=0', 'index', 1)
```

The ‘show=1’ command undocks the window, ‘show=0’ docks it again. In this way you can show a visualization in a customized size and position without showing the plugin chain itself.

You can 'plug' multiple instances of this plugin in one plugin chain, e.g. for visualization of data before and after manipulation using the Overlapped Add Equalizer Plugin (see below, and see the chapter 'The SoundMex realtime DSP-Plugin-Pipe' in the main SoundMex documentation).

4 The Overlapped Add Equalizer Plugin

The Overlapped Add Equalizer Plugin `olaeq.dll` is a realtime equalizer with variable FFT-length and window length (window shift is ½ window length) using zero padding.



☑ Bypass d:\...\soundmex\plugins\test.flt

Figure 4

The plugin is used in the example plugineq.m. The configuration section for this plugin within a plugin chain configuration inifile must contain the following values:

```
FileName=..\\plugins\\olaeq.dll
EQFile=..\\plugins\\test.flt
OLAFFTLength=1024
OLAWindowLength=800
Active=0
```

Name	Description	Values
FileName	Full filename with path of the plugins binary	
EQFile	Filename of the filter file (see below)	
OLAFFTLength	Shows/hides visualization of spectrum	between 64 and 4096 (2^n only!)
OLAWindowLength	Length of the window. (OLAFFTLength-OLAWindowLength)/2 zeroes will be padded at the beginning and the end of each frame before applying the windowing function	<= OLAFFTLength
Active	Enables/disables filter on startup	1 or 0

Table 4

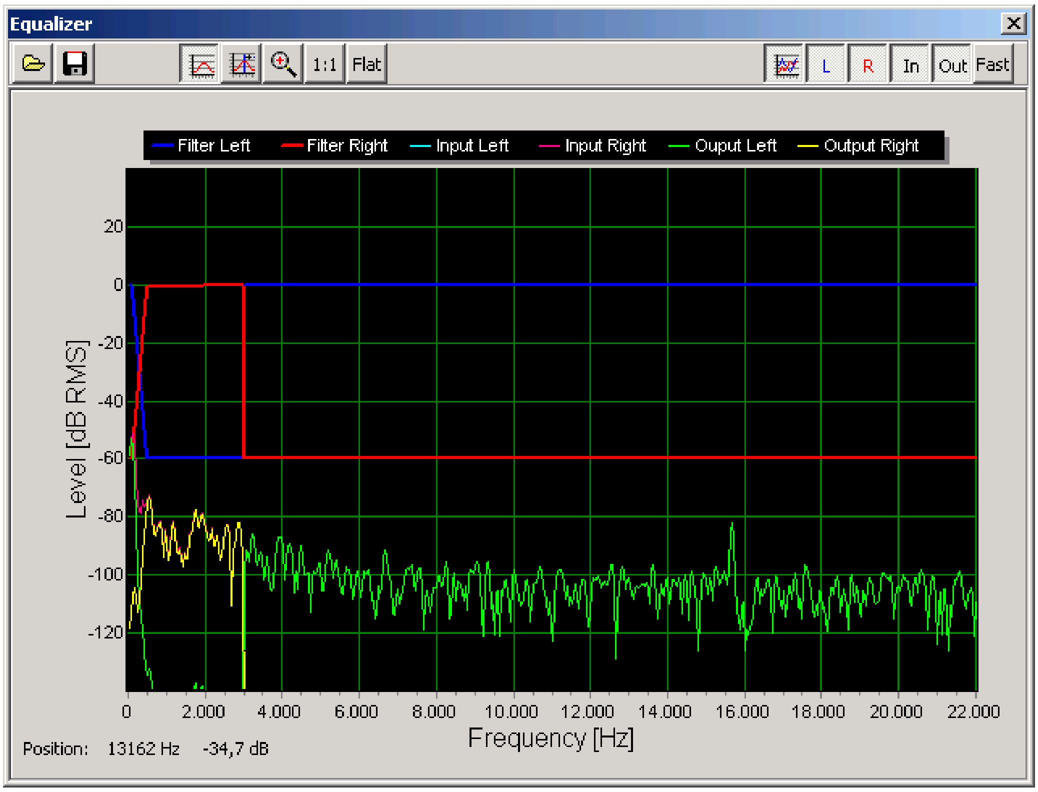
The equalizer recognizes the following string commands at runtime (in the examples it is assumed, that the plugin is plugged into a plugin chain for waveout device 0 at position 1)

```
soundmex2('sendplugin', 'type', 'waveout', 'command', 'show=1', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'show=0', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'active=1', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'active=0', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'filter?', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'filter=name', 'index', 1)
```

The first two commands show/hide the filter visualization and manipulation window (see below), commands three and four toggle the activity of the filter at runtime by code (see example plugin.m). The last two commands retrieve the filename of the actual filter or set a new filter filename respectively. A new filter can only be set when no wave output or input is active at the moment.

On runtime the filter activity can be toggled by unchecking/checking the 'Bypass' box on the plugin.

Pressing the small button on the top right corner of the plugin will show the filter visualization and manipulation window. On this window the filter shapes themselves as well as input and output spectra of the current signal (if any) can be visualized.



Position: 13162 Hz -34,7 dB

Figure 5

Button	Description
📂	Opens a file open dialog to load a filter file to the equalizer
💾	Saves actual filter to a filter file
📈	Enters 'Filter-Drawing-Mode': when this button is pressed, you can simply 'draw' the desired filter. Keeping left mouse button pressed will draw left channels filter, keeping right mouse button pressed will draw right channels filter.

📈	Enters 'Filter-Shifting-Mode': when this button is pressed, you can shift the filters up and down. Keeping left mouse button pressed will shift left channels filter, keeping right mouse button pressed will shift right channels filter.
🔍	Enters 'Zoom-Mode': when this button is pressed, you can zoom into the visualization by drawing a window with the mouse (left mouse button pressed) beginning at the top left corner. Pressing the right mouse button 'un-zooms'
1:1	Un-zooms the visualization
Flat	Flattens left and right filter.
📈	Toggles drawing of all realtime spectra
L	Toggles drawing of realtime spectra of the left (or mono) channel
R	Toggles drawing of realtime spectra of the right channel
In	Toggles drawing of the input spectra
Out	Toggles drawing of the output spectra
Fast	Toggles fast/slow update of the realtime spectra

Table 5

Filterfiles:

The filters are stored in windows inifiles containing two sections (see test.flt in the plugins directory of SoundMex):

```
[Settings]
Log=0

[Filter]
100=1;0.001
501=0.001;0.9
3000=0.001;1
3001=1;0.001
```

The 'Settings' section is optional. If `Log=1` is specified there, the gain values are interpreted logarithmic (in dB full scale).

The section 'Filter' contains the filter itself. It can contain an unlimited number of values of the form:

```
frequency=LeftValue;RightValue
```

If only one value is specified on the right it is applied for left and right channel the same way.

From these values the 'real' filter is computed by linear interpolation on the frequency axis and logarithmic interpolation on the gain axis.

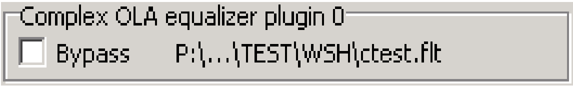
ATTENTION: in linear filters values above 1 will amplify the signal and may result in digital overdrive. The same holds for values above 0 for logarithmic filters.

When using the filter visualization and manipulation window for saving filters, frequency values will be written for every FFT-bin (depending on FFT-length and actual sampling frequency)

5 The Complex Overlapped Add Equalizer Plugin v2

Important note: the older version of this plugin 'complexeq.dll' shipped with former versions of SoundMex2 is discontinued (that version did not calculate values from a complex multiplication).

The Complex Overlapped Add Equalizer Plugin `complexeq.dll` is a realtime equalizer with variable FFT-length and window length (window shift is ½ window length) using zero padding. Compared to the Overlapped Add Equalizer Plugin from paragraph 4, this plugin supports complex filter coefficients. However, it is designed as simple filter without any graphical interface.



Complex OLA equalizer plugin 0

□ Bypass P:\...\TEST\WSH\ctest.flt

Figure 6

The plugin is used in the example pluginceq.m. The configuration section for this plugin within a plugin chain configuration inifile must contain the following values:

```
FileName=..\\plugins\\cplxeq.dll
EQFile=..\\plugins\\ctest.flt
OLAFFTLength=1024
OLAWindowLength=800
Active=0
```

Name	Description	Values
FileName	Full filename with path of the plugins binary	
EQFile	Filename of the filter file (see below)	
OLAFFTLength	FFTLength of filter	between 64 and 4096 (2^n only!)
OLAWindowLength	Length of the window. (OLAFFTLength-OLAWindowLength)/2 zeroes will be padded at the beginning and the end of each frame before applying the windowing function	<= OLAFFTLength
Active	Enables/disables filter on startup	1 or 0

Table 6

The equalizer recognizes the following string commands at runtime (in the examples it is assumed, that the plugin is plugged into a plugin chain for wave out device no 0 at position 1)

```
soundmex2('sendplugin', 'type', 'waveout', 'command', 'active=1', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'active=0', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'filter?', 'index', 1)
soundmex2('sendplugin', 'type', 'waveout', 'command', 'filter=name', 'index', 1)
```

The first two command toggle the activity of the filter at runtime by code (see example plugin.m). The last two commands retrieve the filename of the actual filter or set a new filter filename respectively. A new filter can only be set when no wave output or input is active at the moment.

On runtime the filter activity can be toggled by unchecking/checking the 'Bypass' box on the plugin.

Filterfiles:

The filters are stored in windows inifiles containing one section (see ctest.flt in the plugins directory of SoundMex):

```
[Filter]
100=1
300=0.001+0.002i;1+0.1i
8000=0.001+0.001i;1+0.4i
10000=1
```

It can contain an unlimited number of values of the form:

```
frequency=LeftValue;RightValue
```

If only one value is specified on the right it is applied for left and right channel the same way. Complex value must have the format re+imi, for example

```
0.5+3i              % re 0.5, im 3, left and right identical
0.8-2i;0.2+0.5i     % left: re 0.8, im -2, right: re 0.2, im 0.5
0.7-2i;5            % left: re 0.7, im -2, right: re 5, im 0.0
```

From these values of the filter section the filter to be applied is computed by linear interpolation.

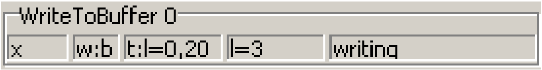
ATTENTION: the frequencies must be sorted ascending!

ATTENTION: filter values above 1 will amplify the signal and may result in digital overdrive.

Important note: the processed complex FFT values are calculated by a full compley multiplication, of actual spectrum and filter values.

6 The WriteToBuffer Plugin

The WriteToBuffer plugin is a plugin that can write input or output data buffers of a device directly to a MATLAB ® vector. It may be started immediately or threshold driven, and may write a predefined length or in loop mode. The plugin usage is demonstrated in the example pluginwrite.m.



WriteToBuffer 0

x | w:b | t:l=0,20 | l=3 | writing

Figure 7

The five text fields contain the following information

Field	Description
1	Name of vector to write to in MATLAB ® workspace.
2	channel to write to (w:b both, w:l left, w:r right channel)
3	Threshold channel (w:b both, w:l left, w:r right channel) and value (here 0.2)
4	loopcount within vector (if looping is enabled, here three loops written)
5	actual state

Table 7

The MATLAB ® vector, where the plugin should write the data to must be set **after** initialization using the ‘sendplugin’-command (assuming a waveout plugin for device 0 with index 1):

```
x = zeros(2, 2*44100); % allocate the memory for the plugin
soundmex2('sendplugin', 'type', 'waveout', 'command', 'data:x', 'index', 1)
```

ATTENTION: never change the vector that was passed to the plugin within the MATLAB ® workspace! This will change the address in memory and an error will occur within the plugin! Only **read** from this vector, but this may fail too, if the plugin is writing at the same address where you try to read! It is highly recommended to set the status of the plugin to inactive (active=0, see below) before accessing the data. and switch it to active again after copying the data, or after simply changing the vector to write to by a new call to the ‘data’ command.

The configuration section for this plugin within a plugin chain configuration inifile may contain the following values, only ‘FileName’ is mandatory:

```
FileName=..\\plugins\\WriteToBuffer.dll
WriteChannel=both
ThresholdChannel=left
Threshold=0.2
Active=1
Loop=0
```

Name	Description	Values	Default
FileName	Full filename with path of the plugins binary		
WriteChannel	The channel(s) that will be written to the MATLAB ® vector	left, right or both	both
ThresholdChannel	The channel that will be monitored for the threshold to be exceeded (if threshold is enabled at all)	left, right or both	both
Threshold	Threshold that has to be exceed on channel(s) 'ThresholdChannel' to start writing data	float value betwwen 0 and 1, 0 disables threshold, writing starts immediately	0
Active	Enables/disables plugin on startup	0 or 1	1
Loop	Enables/disables looping on startup. If looping is enabled, the plugin writes in cycles to the vector, otherwise it stops wrting if the vector is filled.	0 or 1	0
Offset	Starts writing to the buffer after 'Offset' samples are played. If Threshold is enabled, this value is ignored.	Integer value >= 0	0

Table 8

All values except 'FileName' can be set or queried (with a question mark after the values name) using the 'sendplugin' of SoundMex command, however only 'active' can be set while the plugin is running (i.e. the corresponding device is dunning; in the examples it is assumed, that the plugin is plugged into a plugin chain for waveout device no 0 at position 1):

```
soundmex2('sendplugin', 'command', 'NAME=VALUE', 'index', 1) % set a value
soundmex2('sendplugin', 'command', 'NAME?', 'index', 1)     % query a value
```

NAME and VALUE are one of the names and values respectively form Table 8 (except 'FileName'). a name followed by a question mark returns the actual value of the corresponding name.

The following additional query commands are implemented

```
soundmex2('sendplugin', 'command', 'loops?', 'index', '1')
soundmex2('sendplugin', 'command', 'pos?', 'index', '1')
soundmex2('sendplugin', 'command', 'state?', 'index', '1')
```

The ‘loops?’ command returns the actual writing loop count, i.e. how often the buffer iwas filled completely (if looping is enabled at all) The ‘pos?’-command returns the actual position of the writing pointer within the MATLAB ® vector. NOTE: this position is zero based and is counted relative to the very first value (position 0). The position increases with rows and columns subsequently. For example a 2x3 matrix has the follwing position indices:

1	4
2	5
3	6

The ‘status?’ command returns one of the following status strings:

Status	Description
none	the vector to write to is not set. The plugin will be set to this status at startup and after the corresponding device is stopped.
error	an error occurred duringthe writing process. This is most likely the case if the vector to write to was changed during runtime.
set	the vector to write to is set, device is not running.
writing	the plugin is actually writing to the vector
done	the plugin has finihed writing to the vector (only occurs in non-looping mode)
inactive	the plugin is deactivated via inifile or string ommand.
wait thrs	the plugin is set and running, but not writing yet. It waits for the threshold to be exceeded on the selected channel(s).

NOTE: all sendplugin commands of this plugin are case insensitive!