

# A room-corrected Ambisonic listening rig made with free software

A fun usecase for Linux as a versatile and reliable tool for electro-acoustic hobbyists and professionals.

With a brief glimpse into Ambisonics in case you always wondered what it might be about.  
(But the methods described here also work for stereo or any other surround system.)

And most of the discussed software runs on Mac OS X, too.

# About me

Jörn Nettingsmeier, aged 33,

freelance audio engineer and IT admin,  
classically trained musician,  
qualified event technician,

been hanging out with the Linux audio crowd  
since 1998.

# About your next 20 mins:

My assumptions are:

- You are interested in **Linux** and/or in setting up an Ambisonic surround rig at home.
- You have between 4 and 24 speakers and sound I/O hardware available.
  - Your speakers and room are **not perfect**.
  - You are willing and able to afford software licensing costs of 0.00€ **and**
  - You are interested in **doing stuff yourself**.

# Why Ambisonics?

Free software is only fun with **free codecs**, **free algorithms** and **free exchange of information**.

All major Ambisonic patents have now expired, and there is a huge academic and hobbyist **community** of enthusiasts, plus a rich supply of freely available **code** and **documentation**.

And: ***it sounds good and scales well.***

# Why Linux?

Thanks to the **JACK** sound server, Linux has very flexible **inter-application routing**.

You can **mix and match** different specialised applications.

This makes Linux/JACK an ideal platform for experimental *and* production use with non-standard requirements and workflows.

And: **it's free as in „speech“ and „beer“**.

# Step 1a: arrange your speakers



sweet spot  
(listening seat)



For best spatial reproduction, choose a regular polygon or polyhedron.

(Yes, you can do 3D!)

Try to get the angles right. Distance errors can be corrected with delay.

# Step 1b: measure angle and distance



sweet spot  
(listening seat)

For each speaker, measure its actual angle (front is 0°, turn is counter-clockwise) and distance from sweet spot.



**laser range finder**  
**laser angle gauge**



# Step 2: Get speaker IRs



instrumentation  
microphone in  
sweet spot



For each speaker, measure its impulse response at the listening position (which will include room effects).

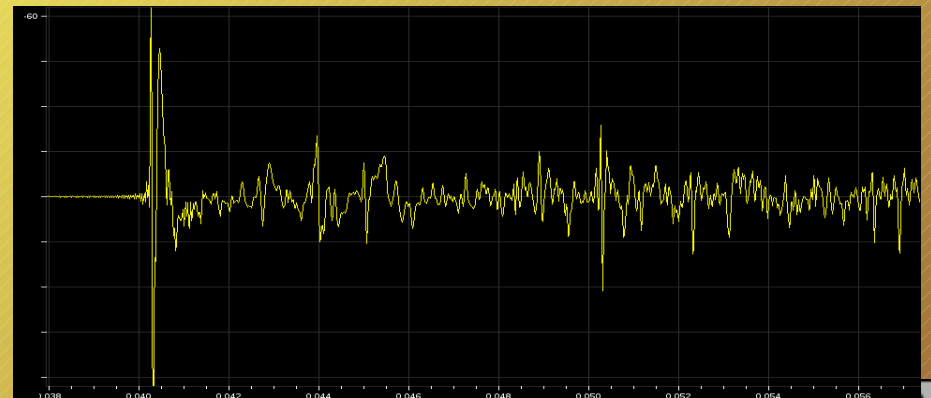
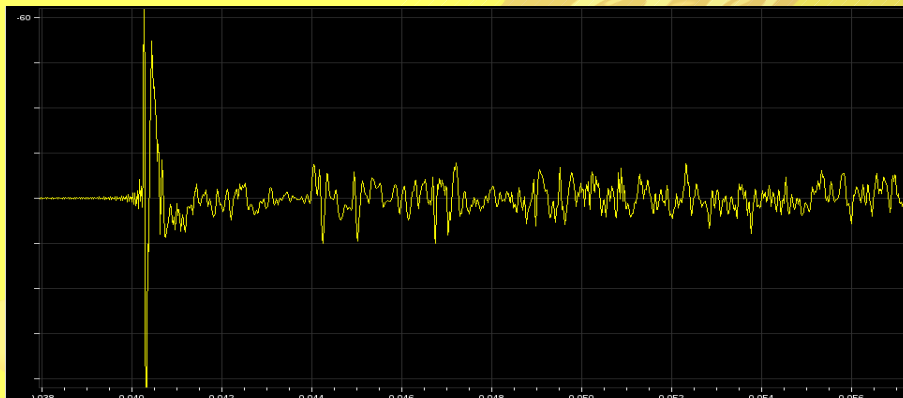
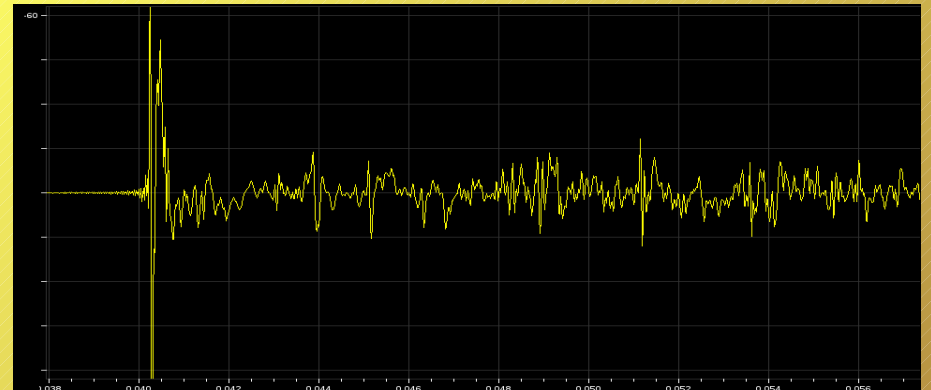
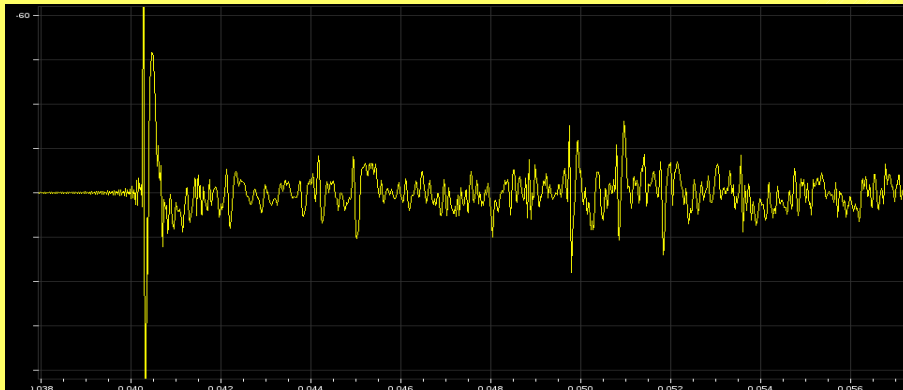
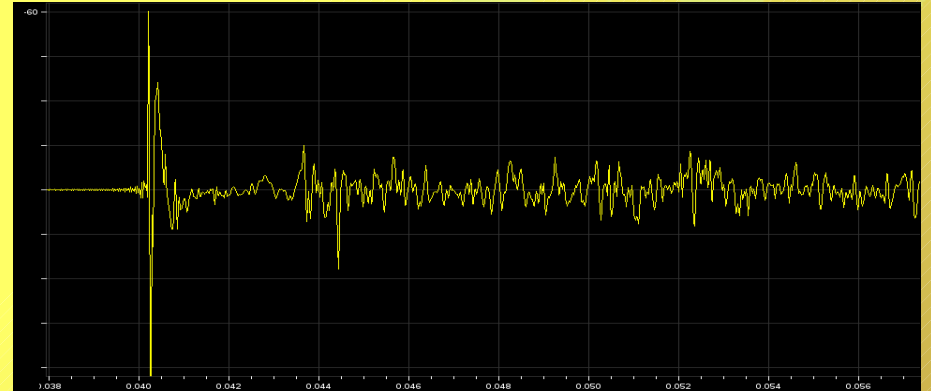
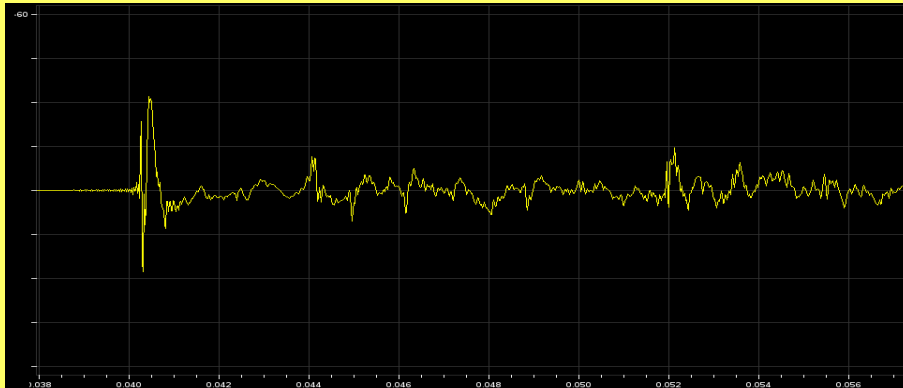
For best S/N ratio, use the swept-sine technique (Farina 2000).



**Aliki**  
**a flat omni mike**



# Speaker IRs after deconvolution:



# Step 3: „invert“ your IRs

To get a correction filter, answer this simple question:

*„Which FIR kernel will yield a perfect pulse when convolved with the actual speaker/room IR?“*

Ok, jokes aside, computer to the rescue  
(and of course getting a usable, stable filter is a lot  
harder in practice).



# Step 3: „invert“ your IRs

**drc** is a room correction package with excellent documentation:

<http://drc-fir.sourceforge.net>

Use version 2.7.0

(3.0.0 has some unresolved numerical instabilities and chokes on some IRs, like mine...)

# Step 3: „invert“ your IRs

Begin with a mild correction setting (such as the ERB example configuration) for a large sweet spot.

Define your desired frequency response as a set of spline points in a text file.

If your speakers are port-loaded: don't push them below the port frequency!

```
0 -40.0
35 -40.0
40 0.00
50 0.00
100 0.00
150 0.00
19900 -3.95
20000 -4.00
20500 -4.00
22050 -40.0
```

# Step 3: „invert“ your IRs

Oh, I forgot: For those new to the UNIX way of doing things, drc is a **command line tool**.






# Step 3: „invert“ your IRs

```
$ for i in speaker_ir-*.pcm ; do drc --BCInFile $i -PSOutFile  
  filter-$i erb-48.0.drc ; done
```

```
DRC 2.7.0: Digital Room Correction  
Copyright (C) 2002-2008 Denis Sbragion
```

```
Input configuration file: erb-48.0.drc  
Parsing configuration file...  
<...>
```

```
$ for i in filter-speaker_ir-*; do sox -f -4 -r48000 -c1 -t raw $i  
  $i.wav; done
```



Invoke drc with the ERB configuration file (included), specify the input IR and the output file name for the filter kernels.



# Step 3: „invert“ your IRs

```
$ for i in speaker_ir-*.pcm ; do drc --BCInFile $i -PSOutFile  
filter-$i erb-48.0.drc ; done
```

```
DRC 2.7.0: Digital Room Correction  
Copyright (C) 2002-2008 Denis Sbragion
```

```
Input configuration file: erb-48.0.drc  
Parsing configuration file...  
<...>
```

```
$ for i in filter-speaker_ir-*; do sox -f -4 -r48000 -c1 -t raw $i  
$i.wav; done
```

Process all speaker  
IRs at once using a  
shell loop.  
(Nice, isn't it?)

# Step 3: „invert“ your IRs

```
$ for i in speaker_ir-*.pcm ; do drc --BCInFile $i -PSOutFile  
  filter-$i erb-48.0.drc ; done
```


```
DRC 2.7.0: Digital Room Correction  
Copyright (C) 2002-2008 Denis Sbragion
```

```
Input configuration file: erb-48.0.drc  
Parsing configuration file...
```

```
<...>
```

```
$ for i in filter-speaker_ir-*; do sox -f -4 -r48000 -c1 -t raw $i  
  $i.wav; done
```

Convert the raw  
filters to .wav files  
with **sox** (again in a  
loop to save typing).



# Step 3: „invert“ your IRs

```
$ for i in speaker_ir-*.pcm ; do drc --BCInFile $i -PSOutFile  
  filter-$i erb-48.0.drc ; done
```

```
DRC 2.7.0: Digital Room Correction  
Copyright (C) 2002-2008 Denis Sbragion
```

```
Input configuration file: erb-48.0.drc  
Parsing configuration file...  
<...>
```

```
$ for i in filter-speaker_ir-*; do sox -f -4 -r48000 -c1 -t raw $i  
  $i.wav; done
```

**Scared?  
Well, you can  
always...**



**...rent a nerd!**

Write to [linux-audio-user@lists.linuxaudio.org](mailto:linux-audio-user@lists.linuxaudio.org)!

# Step 4: Signal flow & software setup

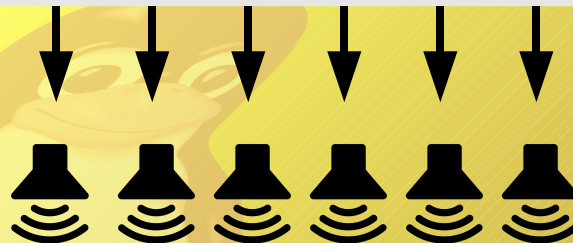
B-format audio

DRC filters

Ambisonic decoder

\* Realtime convolver

JACK  
Ardour  
AmbDec  
JConv



# Step 4: Signal flow & software setup

Ardour DAW

DRC filters

AmbDec

JACK



JConv



# Step 4: Signal flow & software setup

**Ardour DAW**





ambidrc - Mixer - Ardour <@kleineronke

Strips

Track	Mute	Solo	Level
master			-4.6
Mon			
Mon DRC			
B-Format			

ambidrc - Mixer - Ardour <@kleineronke

Mute Solo Mute Solo Mute Solo Mute Solo

Track	Mute	Solo	Level
master			-4.6
Mon			
Mon DRC			
B-Format			

ardour 2.6.1  
(built from revision 4119)

**Ardour is a complete DAW and also a very versatile mixer,**

\*ambidrc - Ardour <@kleineronke>

Session Transport Edit Region Track View JACK Window Options Help 48 kHz / 5.3 ms Buffers p:89% c:99% DSP: 23.1%

00:02:05:12 30 NDF Internal Time master Punch In Auto Play Auto Input SOLO

Slide Edit 00:00:00:00 No Grid Beats Mouse 00:00:05:00

Timecode 00:00:00:00 00:05:00:00 00:10:00:00

Meter 4/4  
Tempo 120.00  
Loop/Punch Ranges Loop  
CD Markers  
Location Markers start

B-Format m s p a g

AJH\_Stravinsky-Pulcinella-Suite

Regions Tracks/Busses Snapshots

AJH\_eight-  
Britten\_1st  
Britten\_2nd  
Britten\_4th  
Britten\_3rd  
Dvorak-Thr  
AJH\_Brahm  
AJH\_Stravi

CC BY SA

ARDOUR 2.6.1 (built from revision 4119)

with an arbitrary number of channels in tracks&buses!

The screenshot shows the Ardour 2.6.1 interface. On the left is the Ardour logo and version information. In the center, a yellow box contains the text "with an arbitrary number of channels in tracks&buses!". On the right, the mixer window is visible, showing several tracks with faders and meters. A window titled "jkmeter" is also open, displaying a multi-channel level meter with a scale from -40 to 20 dB.

\*ambi-drc - Ardour <@kleineronkel>

Session Transport Edit Region Track View JACK Window Options Help 48 kHz / 5.3 ms Buffers p:89% c:99% DSP: 23.1%

00:02:05:12 30 NDF Internal Time master Punch In Auto Play Auto Input SOLO Punch Out Auto Return Click AUDITION

Slide Edit 00:00:00:00 No Grid Beats Mouse 00:00:05:00

Timecode 00:00:00:00 00:05:00:00 00:10:00:00  
Meter 4/4  
Tempo 120.00  
Loop/Punch Ranges Loop  
CD Markers  
Location Markers start

B-Format m s p a g

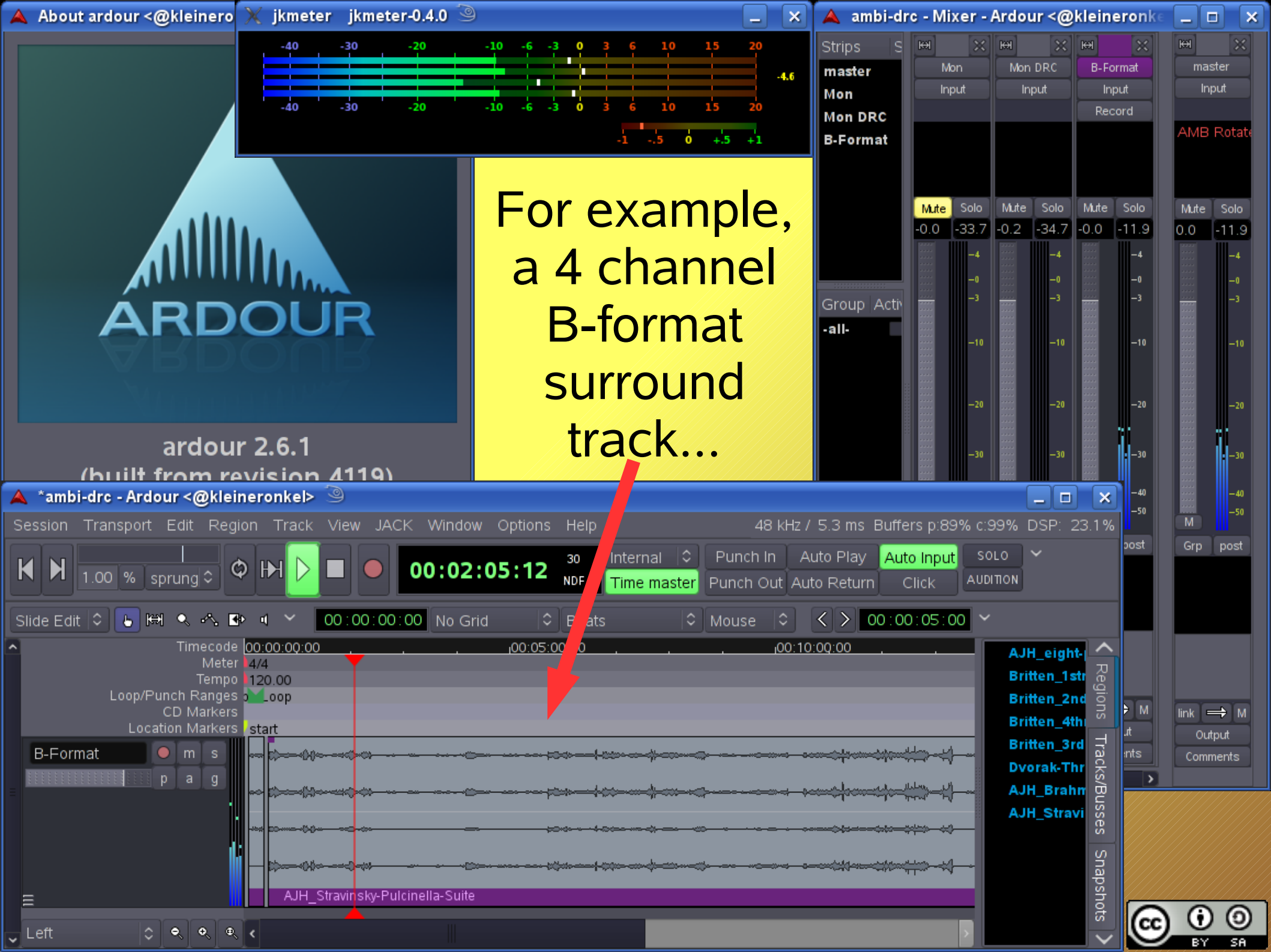
AJH\_Stravinsky-Pulcinella-Suite

Regions Tracks/Busses Snapshots

AJH\_eight-  
Britten\_1st  
Britten\_2nd  
Britten\_4th  
Britten\_3rd  
Dvorak-Thr  
AJH\_Brahm  
AJH\_Stravi

CC BY SA

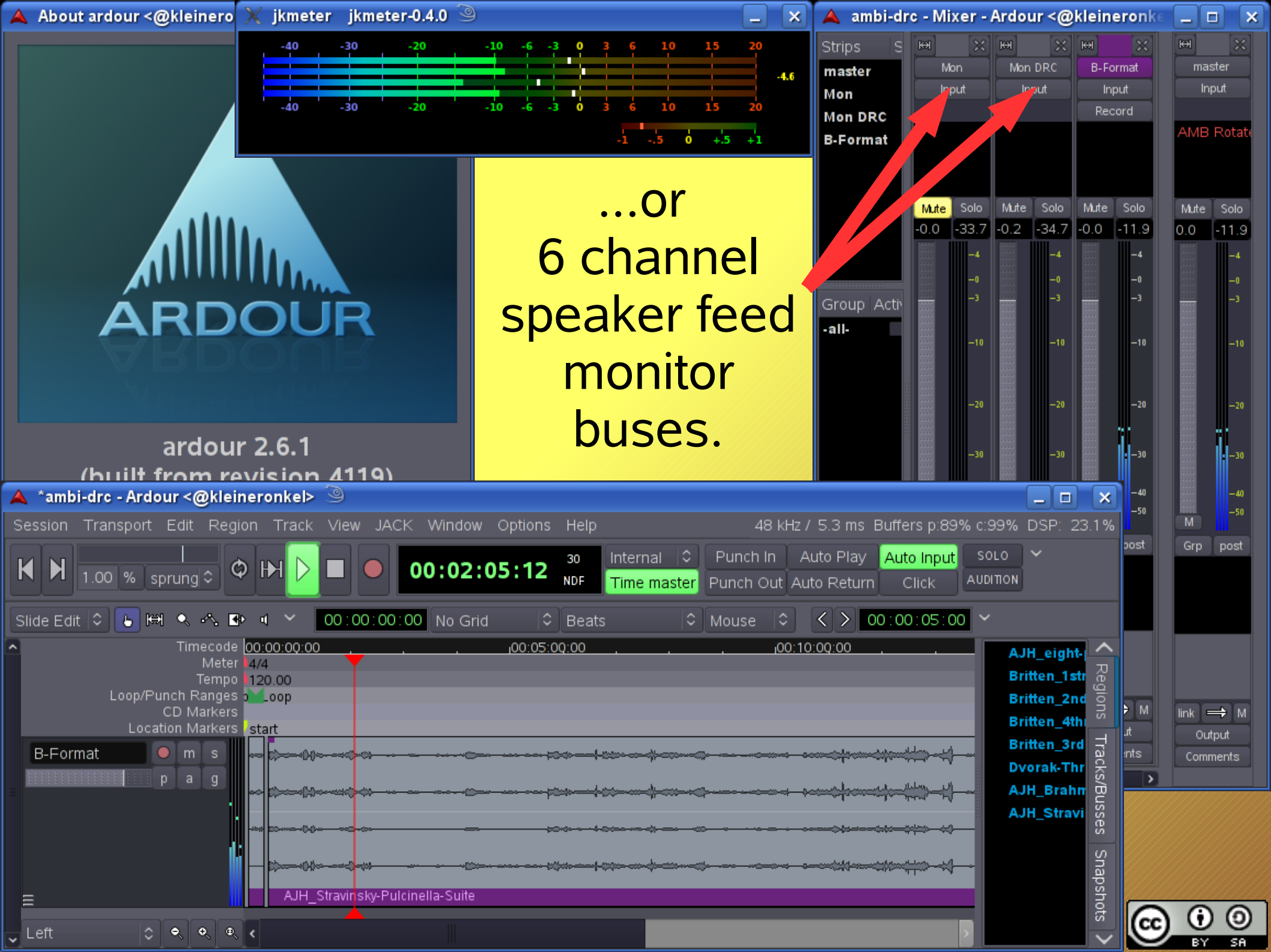
The screenshot shows the main Ardour 2.6.1 interface. At the top, the title bar reads "\*ambi-drc - Ardour <@kleineronkel>". Below it is a menu bar with options: Session, Transport, Edit, Region, Track, View, JACK, Window, Options, Help. The transport controls are visible, showing a timecode of 00:02:05:12 and a tempo of 120.00. The track view shows several tracks with waveforms, and a track named "AJH\_Stravinsky-Pulcinella-Suite" is highlighted. On the right, there are panels for "Regions", "Tracks/Busses", and "Snapshots". At the bottom right, there are Creative Commons BY-SA icons.



For example,  
a 4 channel  
B-format  
surround  
track...

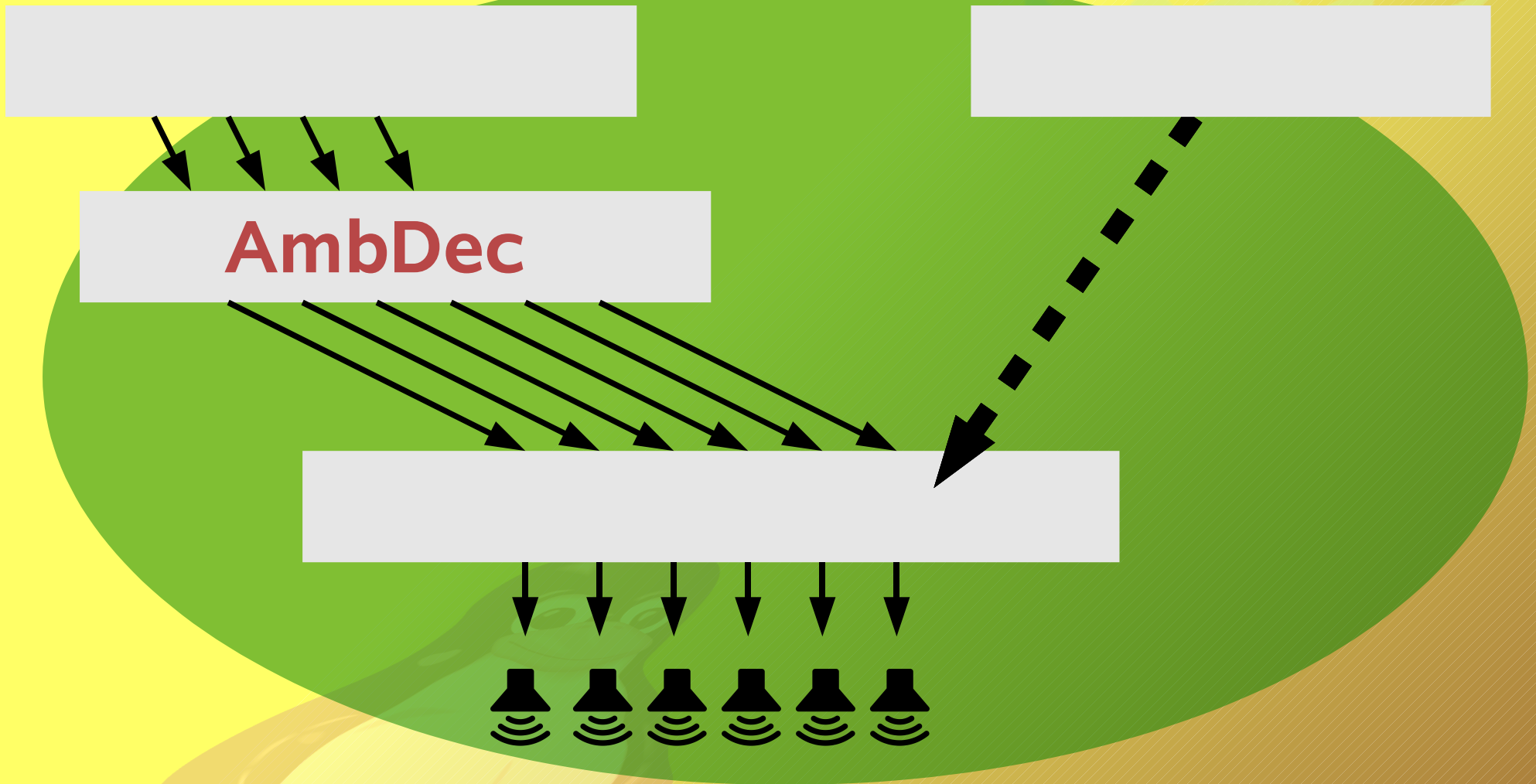
AJH\_Stravinsky-Pulcinella-Suite





...or  
6 channel  
speaker feed  
monitor  
buses.

# Step 4: Signal flow & software setup





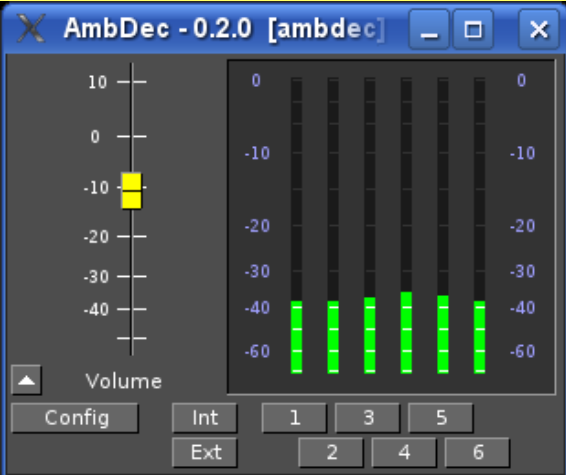
# AmbDec

an Ambisonic decoder for 2D and 3D

up to 24 speakers

psycho-acoustically correct dual-band operation

near-field compensation



The 'New configuration' dialog box has the following settings:

- Matrix scaling:  Normalised,  Furse-Malham
- Horizontal order:  First order,  Second order
- Vertical order:  Horizontal only,  First order,  Second order
- Frequency bands:  Single band,  Dual band
- Speakers:

Buttons: Cancel, Create

The 'Configuration' dialog box has the following settings:

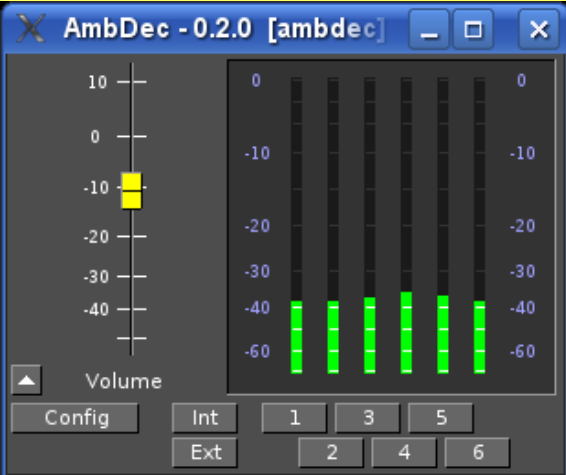
- Description: hexagon, max rV at LF, max rE at HF
- Matrix scaling:  Normalised,  Furse-Malham
- Input scaling:  Normalised,  Furse-Malham
- Speaker distance:  Delay comp.,  Gain comp.
- Near-field comp.:  None,  On inputs,  On outputs
- Crossover freq.:

ID	Speakers	LF Decoder			HF Decoder						
		Dist	Azim	Elev	G(order)	W	X	Y	G(order)	W	X
1	1.39	0.0	0.0	1.0000	0.3535	0.3535	1.4142	0.0000	1.0000	1.4142	0.0000
2	1.38	60.0	0.0	1.0000	0.3535	0.3535	0.7071	-1.2247	1.0000	0.7071	-1.2247
3	1.38	120.0	0.0	1.0000	0.3535	0.3535	-0.7071	-1.2247	1.0000	-0.7071	-1.2247
4	1.74	180.0	0.0	1.0000	0.3535	0.3535	-1.4142	0.0000	1.0000	-1.4142	0.0000
5	1.40	-120.0	0.0	1.0000	0.3535	0.3535	-0.7071	1.2247	1.0000	-0.7071	1.2247
6	1.42	60.0	0.0	1.0000	0.3535	0.3535	0.7071	1.2247	1.0000	0.7071	1.2247

Buttons: Speakers, Decoder

# AmbDec

Create a configuration for your speaker rig:



The 'New configuration' dialog box has four sections: Matrix scaling (Normalised, Furse-Malham), Horizontal order (First order, Second order), Vertical order (Horizontal only, First order, Second order), and Frequency bands (Single band, Dual band). The 'Speakers' field is set to 6. 'Cancel' and 'Create' buttons are at the bottom.

The 'Configuration' dialog box shows a description: 'hexagon, max rV at LF, max rE at HF'. It has sections for Matrix scaling, Input scaling, Speaker distance, Near-field comp., and Crossover freq. (500). Below is a table for the decoder settings.

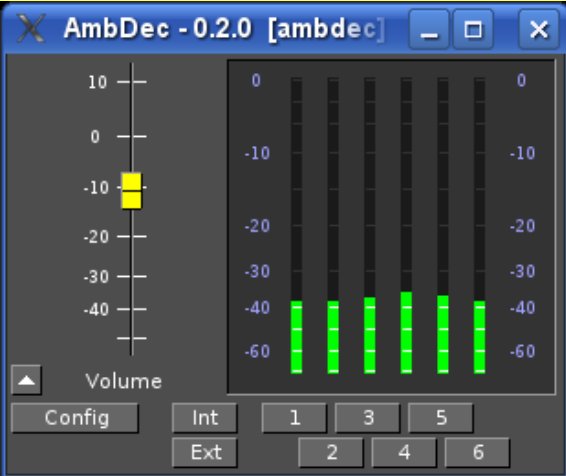
				LF Decoder			HF Decoder		
Speakers				G(order)	0.3535	0.3535	G(order)	0.3535	0.2500
ID	Dist	Azim	Elev	W	X	Y	W	X	Y
1	1.39	0.0	0.0	1.0000	1.4142	0.0000	1.0000	1.4142	0.0000
2	1.38	60.0	0.0	1.0000	0.7071	-1.2247	1.0000	0.7071	-1.2247
3	1.38	120.0	0.0	1.0000	-0.7071	-1.2247	1.0000	-0.7071	-1.2247
4	1.74	180.0	0.0	1.0000	-1.4142	0.0000	1.0000	-1.4142	0.0000
5	1.40	-120.0	0.0	1.0000	-0.7071	1.2247	1.0000	-0.7071	1.2247
6	1.42	60.0	0.0	1.0000	0.7071	1.2247	1.0000	0.7071	1.2247



# AmbDec

Create a configuration for your speaker rig:

Basic settings (order, number of speakers)

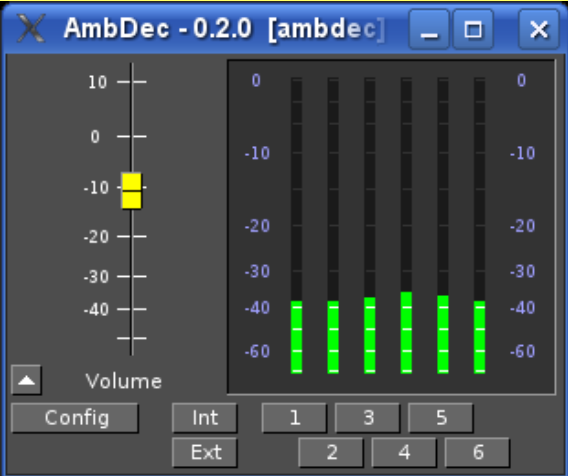


AmbDec - New configuration dialog box. It contains four sections: Matrix scaling (Normalised, Furse-Malham), Horizontal order (First order, Second order), Vertical order (Horizontal only, First order, Second order), and Frequency bands (Single band, Dual band). The Speakers field is set to 6. There are Cancel and Create buttons.

AmbDec - Configuration dialog box. It contains a Description field with the text "hexagon, max rV at LF, max rE at HF". It has several sections for settings: Matrix scaling (Normalised, Furse-Malham), Input scaling (Normalised, Furse-Malham), Speaker distance (Delay comp., Gain comp.), Near-field comp. (None, On inputs, On outputs), and Crossover freq. (500). Below these is a table with columns for Speakers, LF Decoder, and HF Decoder. The table has 6 rows of speaker data.

Speakers				LF Decoder			HF Decoder		
ID	Dist	Azim	Elev	W	X	Y	W	X	Y
1	1.39	0.0	0.0	1.0000	1.4142	0.0000	1.0000	1.4142	0.0000
2	1.38	60.0	0.0	1.0000	0.7071	-1.2247	1.0000	0.7071	-1.2247
3	1.38	120.0	0.0	1.0000	-0.7071	-1.2247	1.0000	-0.7071	-1.2247
4	1.74	180.0	0.0	1.0000	-1.4142	0.0000	1.0000	-1.4142	0.0000
5	1.40	-120.0	0.0	1.0000	-0.7071	1.2247	1.0000	-0.7071	1.2247
6	1.42	60.0	0.0	1.0000	0.7071	1.2247	1.0000	0.7071	1.2247

# AmbDec



AmbDec - New configuration dialog box. Matrix scaling: Normalised, Furse-Malham. Horizontal order: First order, Second order. Vertical order: Horizontal only, First order, Second order. Frequency bands: Single band, Dual band. Speakers: 6. Buttons: Cancel, Create.

Create a configuration for your speaker rig:

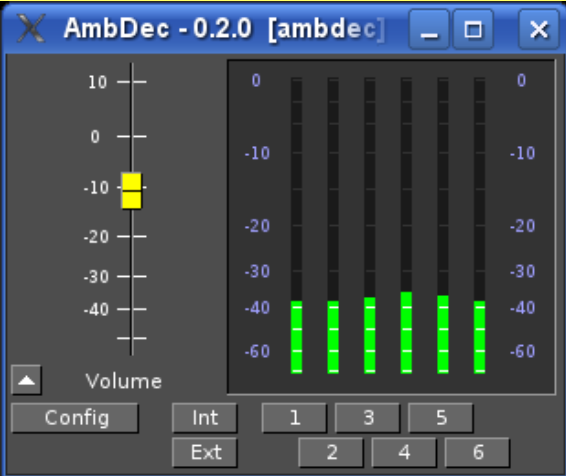
AmbDec - Configuration dialog box. Description: hexagon, max rV at LF, max rE at HF. Matrix scaling: Normalised, Furse-Malham. Input scaling: Normalised, Furse-Malham. Speaker distance: Delay comp., Gain comp. Near-field comp.: None, On inputs, On outputs. Crossover freq.: 500. Table with columns: Speakers, LF Decoder, HF Decoder. Buttons: Load, Save, New, Cancel, Apply, Speakers, Decoder.

ID	Dist	Azim	Elev	LF Decoder			HF Decoder		
				W	X	Y	W	X	Y
1	1.39	0.0	0.0	1.0000	1.4142	0.0000	1.0000	1.4142	0.0000
2	1.38	60.0	0.0	1.0000	0.7071	-1.2247	1.0000	0.7071	-1.2247
3	1.38	120.0	0.0	1.0000	-0.7071	-1.2247	1.0000	-0.7071	-1.2247
4	1.74	180.0	0.0	1.0000	-1.4142	0.0000	1.0000	-1.4142	0.0000
5	1.40	-120.0	0.0	1.0000	-0.7071	1.2247	1.0000	-0.7071	1.2247
6	1.42	60.0	0.0	1.0000	0.7071	1.2247	1.0000	0.7071	1.2247

Basic settings (order, number of speakers)

Decoding Matrix (fill in your speaker positions).

# AmbDec



The 'New configuration' dialog box has four sections: Matrix scaling (Normalised, Furse-Malham), Horizontal order (First order, Second order), Vertical order (Horizontal only, First order, Second order), and Frequency bands (Single band, Dual band). The 'Speakers' field is set to 6. A red arrow points from the 'Speakers' field to the 'Configuration' window below.

Create a configuration for your speaker rig:

The 'Configuration' dialog box shows various settings. The 'Description' field contains 'hexagon, max rV at LF, max rE at HF'. The 'Crossover freq.' is set to 500. The 'Matrix scaling' is set to Furse-Malham. The 'Input scaling' is set to Furse-Malham. The 'Speaker distance' has 'Delay comp.' and 'Gain comp.' checked. The 'Near-field comp.' has 'On inputs' and 'On outputs' checked. Below these settings is a table with columns for 'Speakers', 'LF Decoder', and 'HF Decoder'. A red circle highlights the 'Matrix coefficients' section of the table.

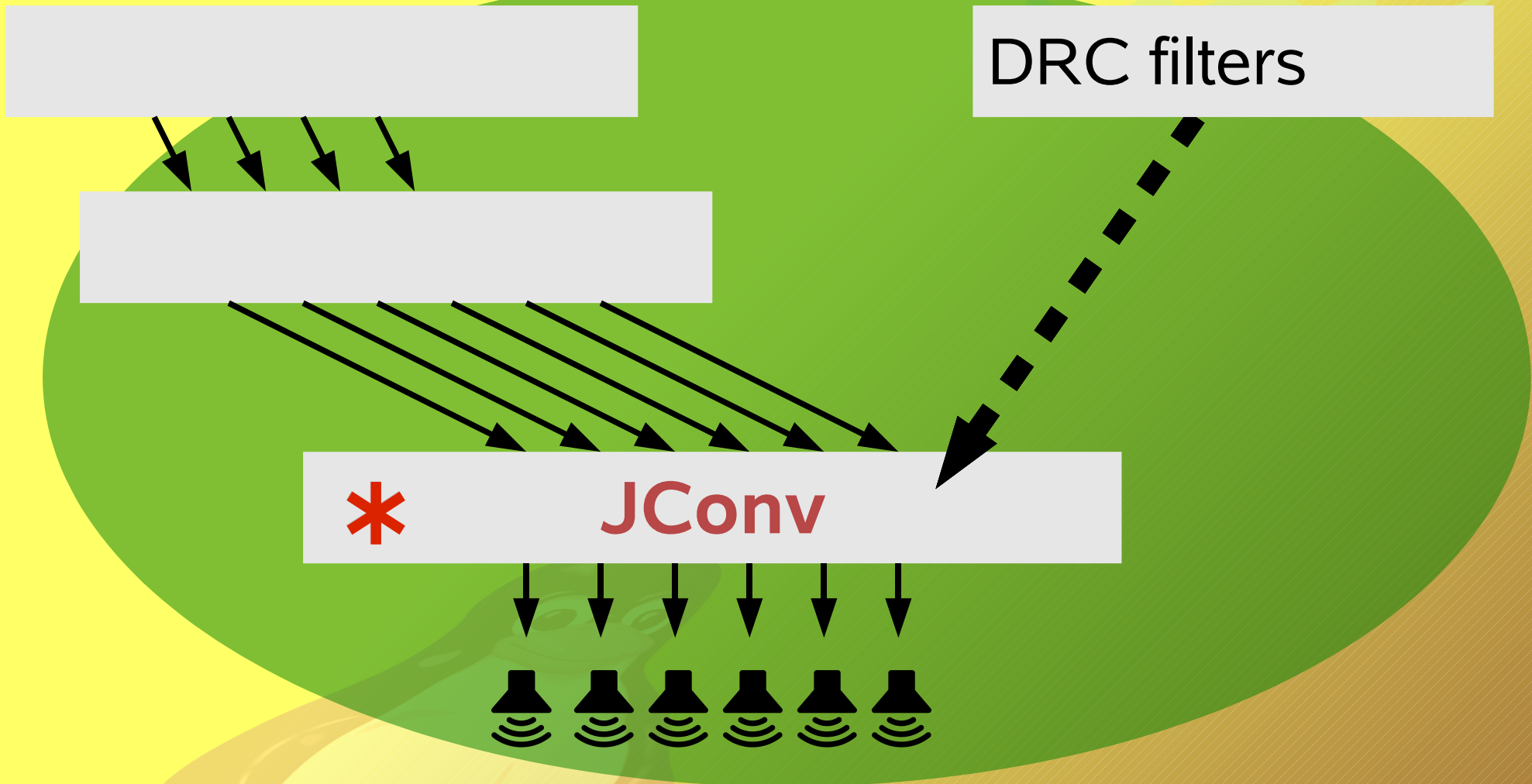
ID	Dist	Azim	Elev	W	Y	LF Decoder		HF Decoder		
						G(order)		G(order)		
1	1.39	0.0	0.0	1.0000	1.4142	0.3535	0.3535	1.0000	1.4142	0.2500
2	1.38	60.0	0.0	1.0000	0.7071	-1.2247	1.0000	0.7071	-1.2247	
3	1.38	120.0	0.0	1.0000	0.7071	-1.2247	1.0000	-0.7071	-1.2247	
4	1.74	180.0	0.0	1.0000	-1.4142	0.3535	1.0000	-1.4142	0.2500	
5	1.40	-120.0	0.0	1.0000	-0.7071	1.2247	1.0000	-0.7071	1.2247	
6	1.42	60.0	0.0	1.0000	0.7071	1.2247	1.0000	0.7071	1.2247	

Basic settings (order, number of speakers)

Decoding Matrix (fill in your speaker positions).

**Matrix coefficients from literature**

# Step 4: Signal flow & software setup



# JConv is a real-time convolution engine (and another command-line tool). :-D

It runs in the background and takes a simple  
configuration file:

```
/cd /home/nettings/drc-filters

#           in  out  partition  maxsize
# -----
/convolver/new      6   6      1024     256000

#
#           in out  gain  delay  offset  length  chan  file
# -----
/impulse/read      1   1   1     0     0     0     1  filter-speaker_ir-1.pcm.wav
/impulse/read      2   2   1     0     0     0     1  filter-speaker_ir-2.pcm.wav
/impulse/read      3   3   1     0     0     0     1  filter-speaker_ir-3.pcm.wav
/impulse/read      4   4   1     0     0     0     1  filter-speaker_ir-4.pcm.wav
/impulse/read      5   5   1     0     0     0     1  filter-speaker_ir-5.pcm.wav
/impulse/read      6   6   1     0     0     0     1  filter-speaker_ir-6.pcm.wav
```

# JConv is a real-time convolution engine (and another command-line tool). :-D

It runs in the background and takes a simple  
configuration file:

```
/cd /home/nettings/drc-filters
```

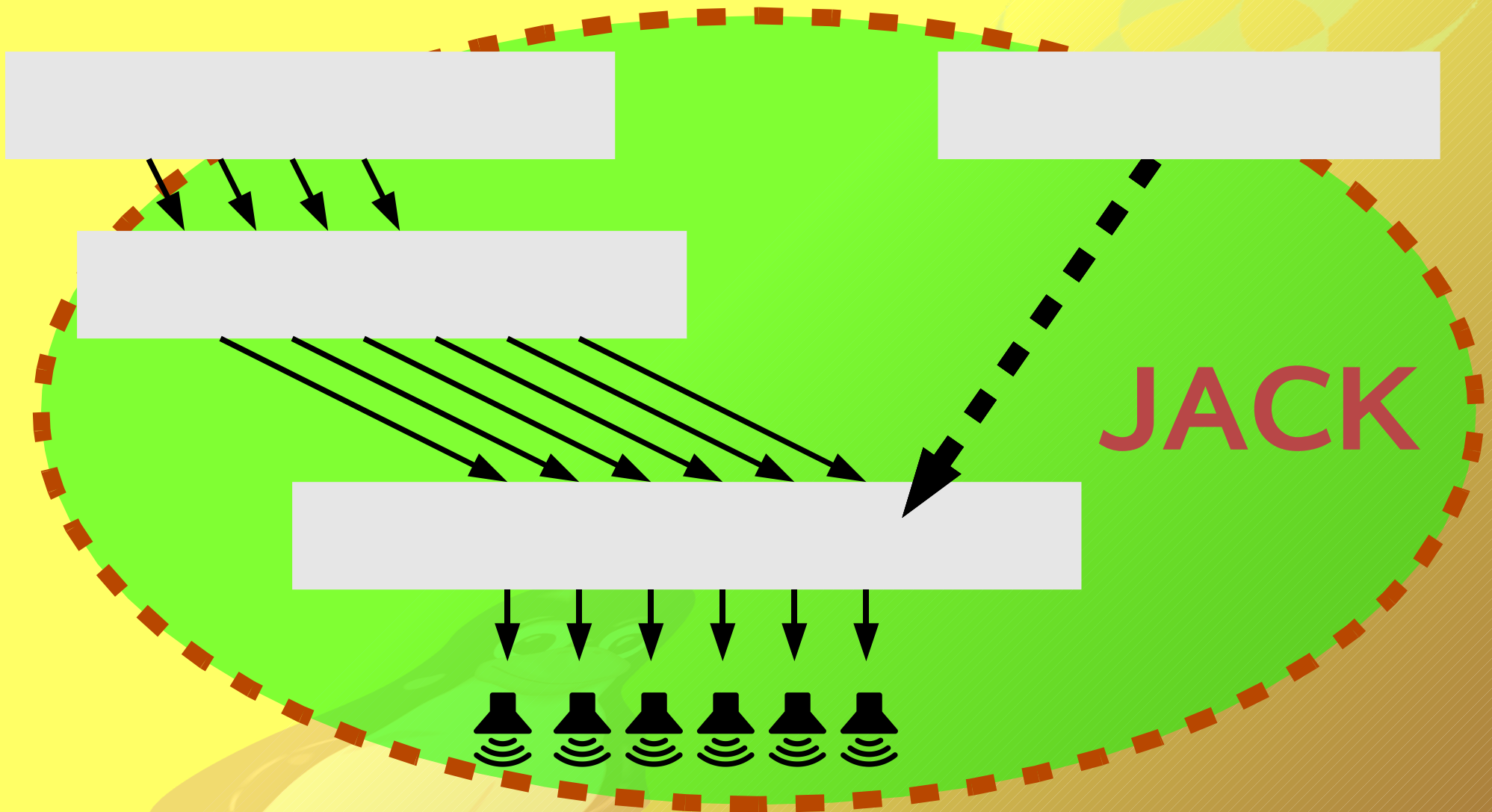
```
#           in  out  partition  maxsize
# -----
/convolver/new      6   6      1024      256000
```

```
#
#           in out  gain  delay  offset  length  chan  file
# -----
/impulse/read      1   1   1     0     0     0     1  filter-speaker_ir-1.pcm.wav
/impulse/read      2   2   1     0     0     0     1  filter-speaker_ir-2.pcm.wav
/impulse/read      3   3   1     0     0     0     1  filter-speaker_ir-3.pcm.wav
/impulse/read      4   4   1     0     0     0     1  filter-speaker_ir-4.pcm.wav
/impulse/read      5   5   1     0     0     0     1  filter-speaker_ir-5.pcm.wav
/impulse/read      6   6   1     0     0     0     1  filter-speaker_ir-6.pcm.wav
```

The filter kernels we  
created earlier:

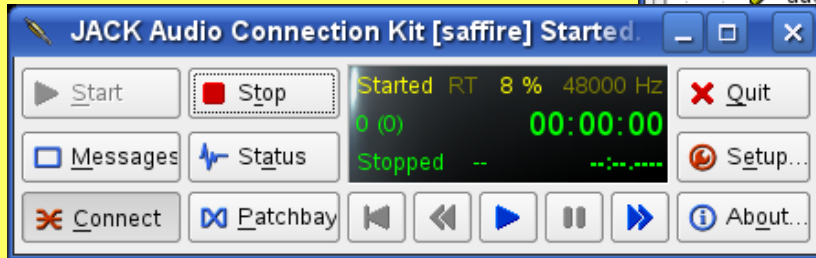


# Step 4: Signal flow & software setup



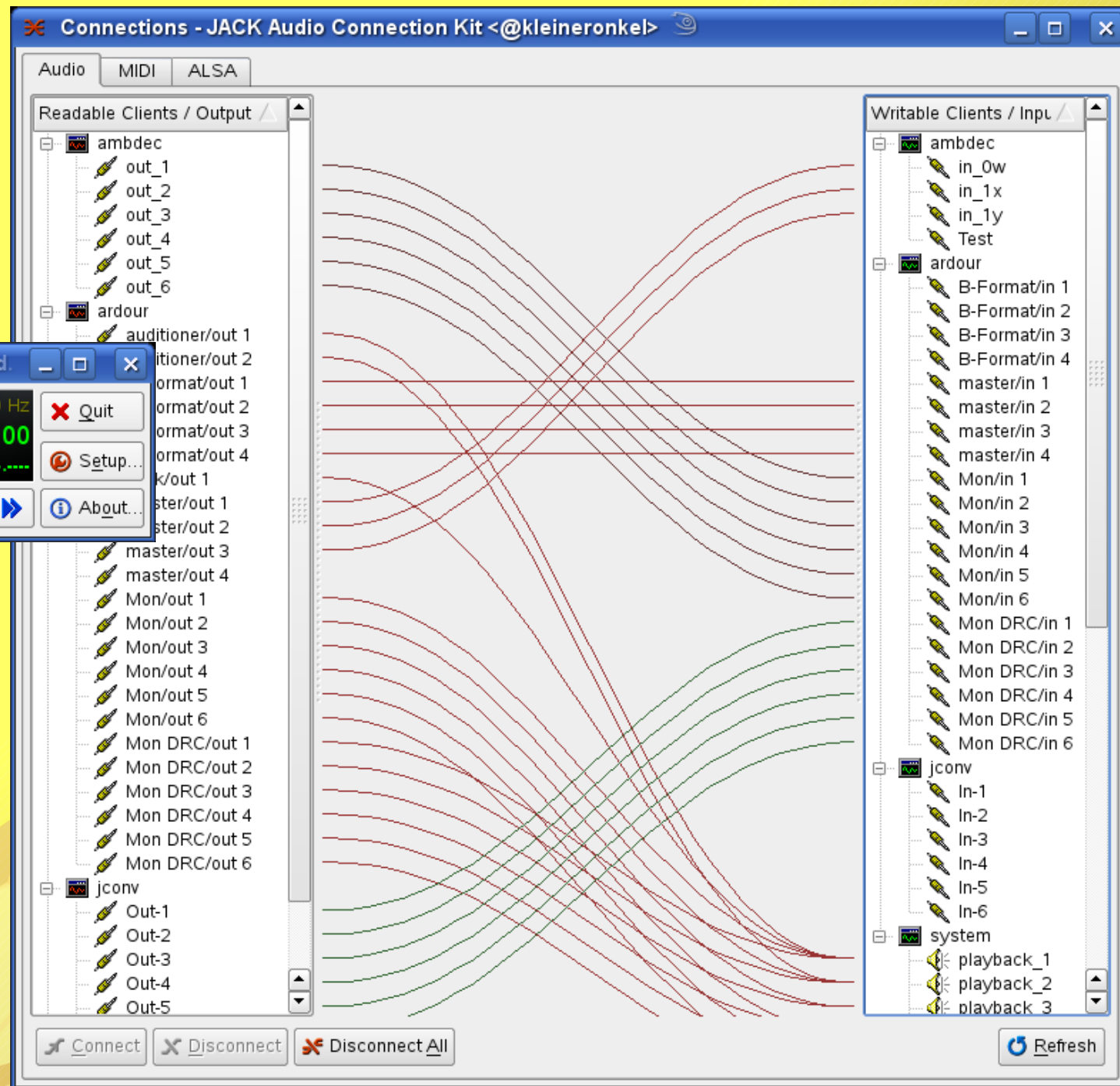


Putting it all together with JACK and qjackctl:



Imagine this with 24 speakers!

Thankfully, virtual cables are cheap :-D



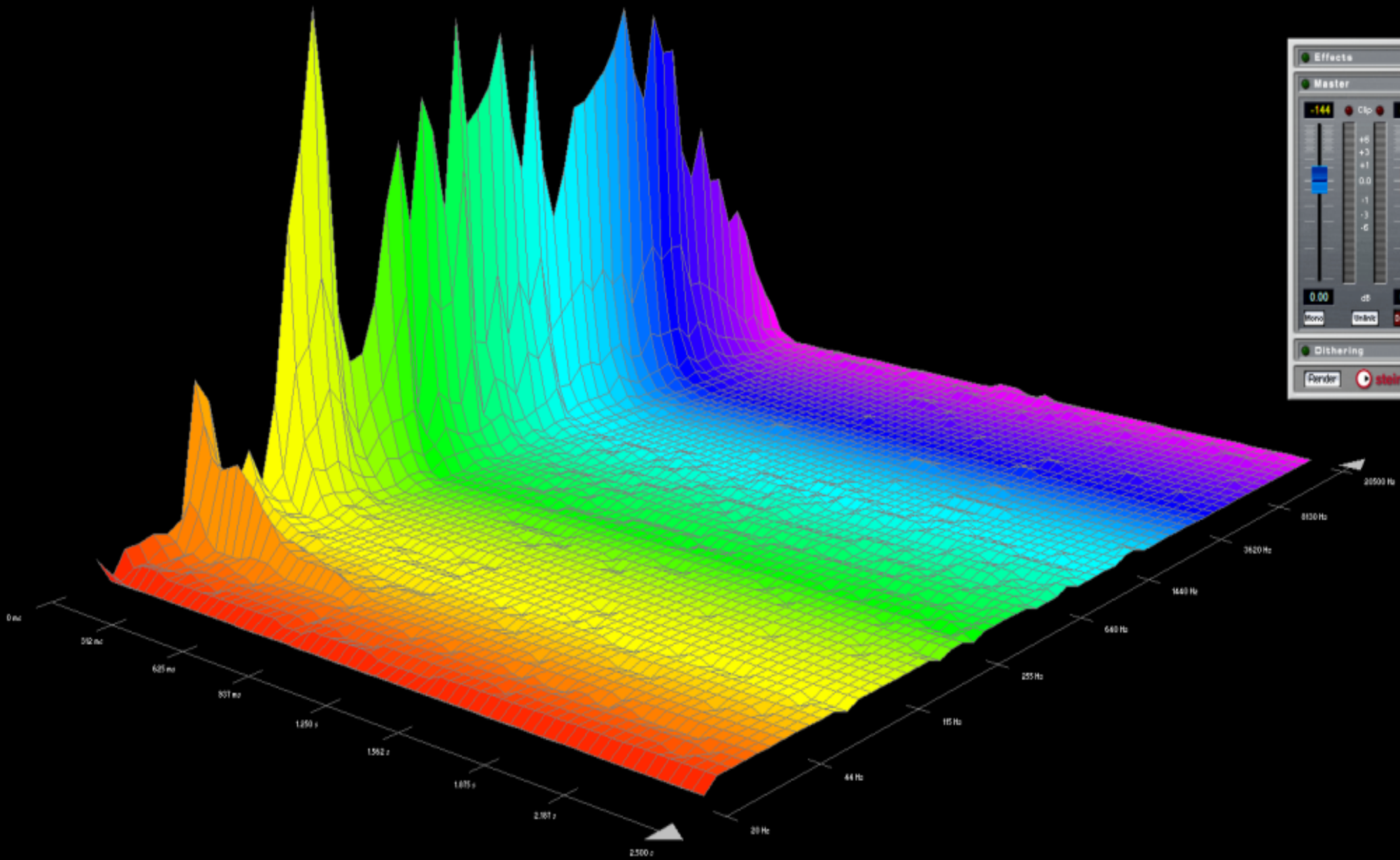
# So what does it sound like?

To evaluate the effects of the application of DRC, here are a few „before & after“ comparisons....:



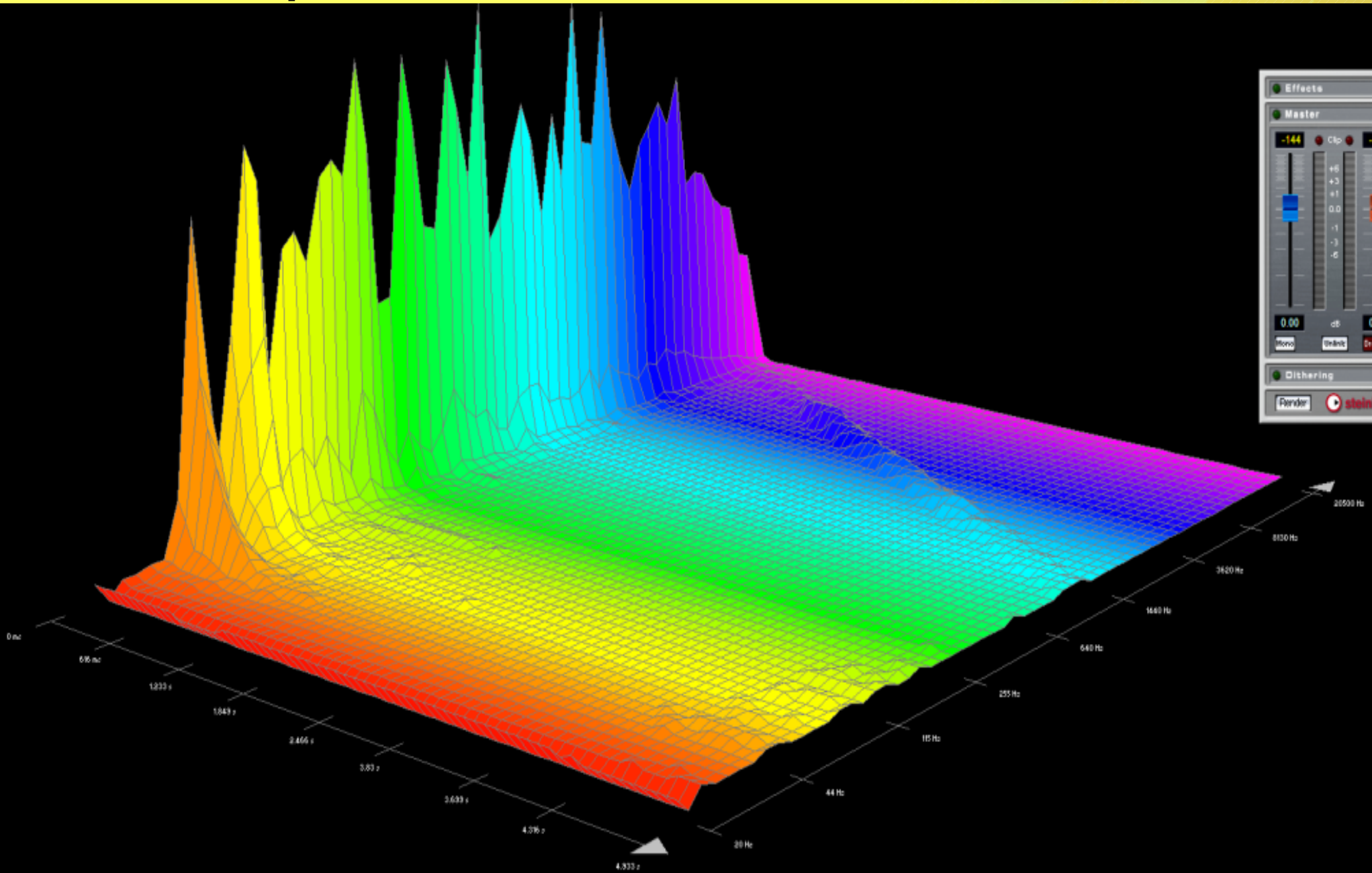
**JAPA,  
WaveLab (er, well...)**

# Speaker No. 6 before DRC:

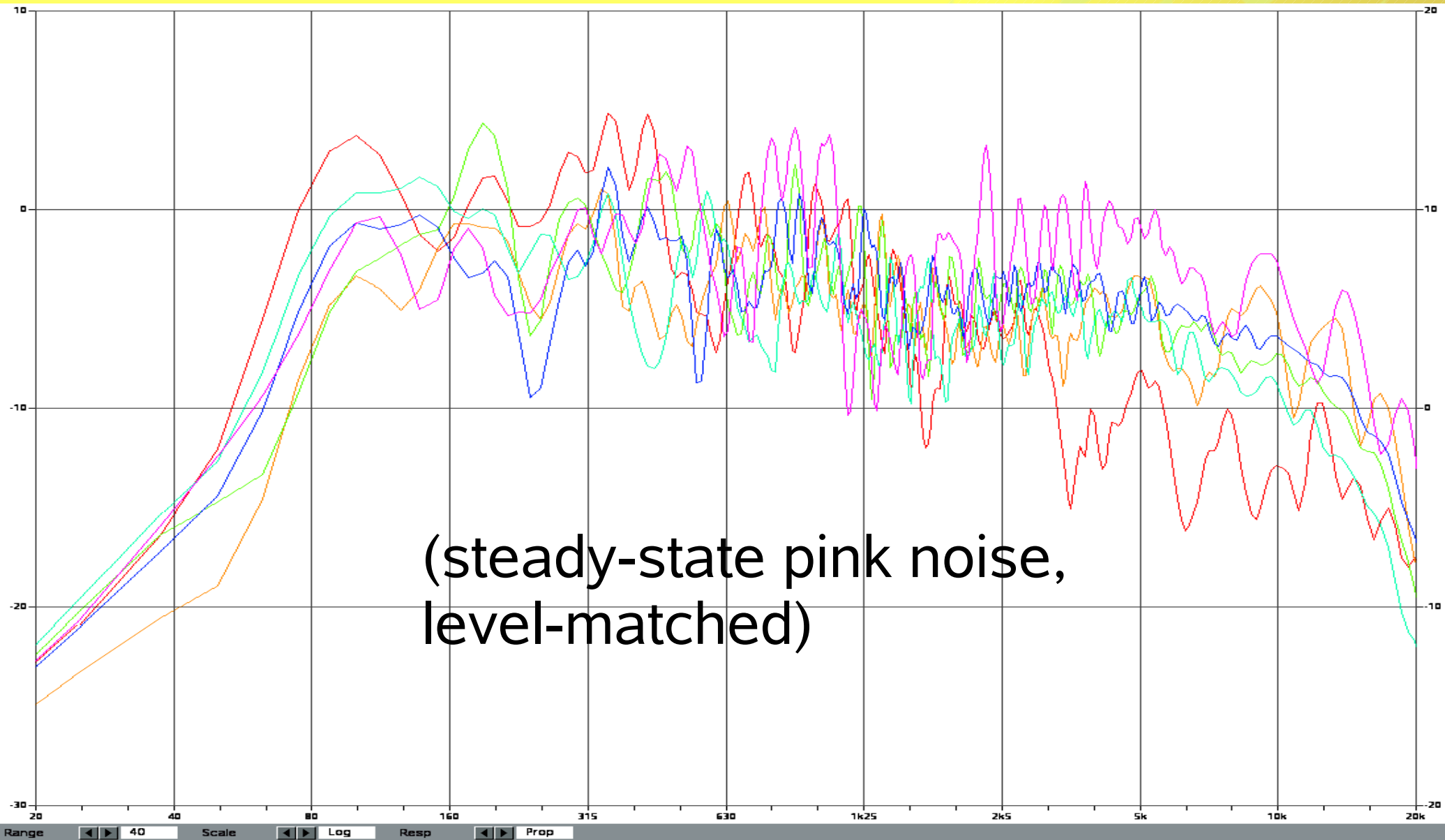




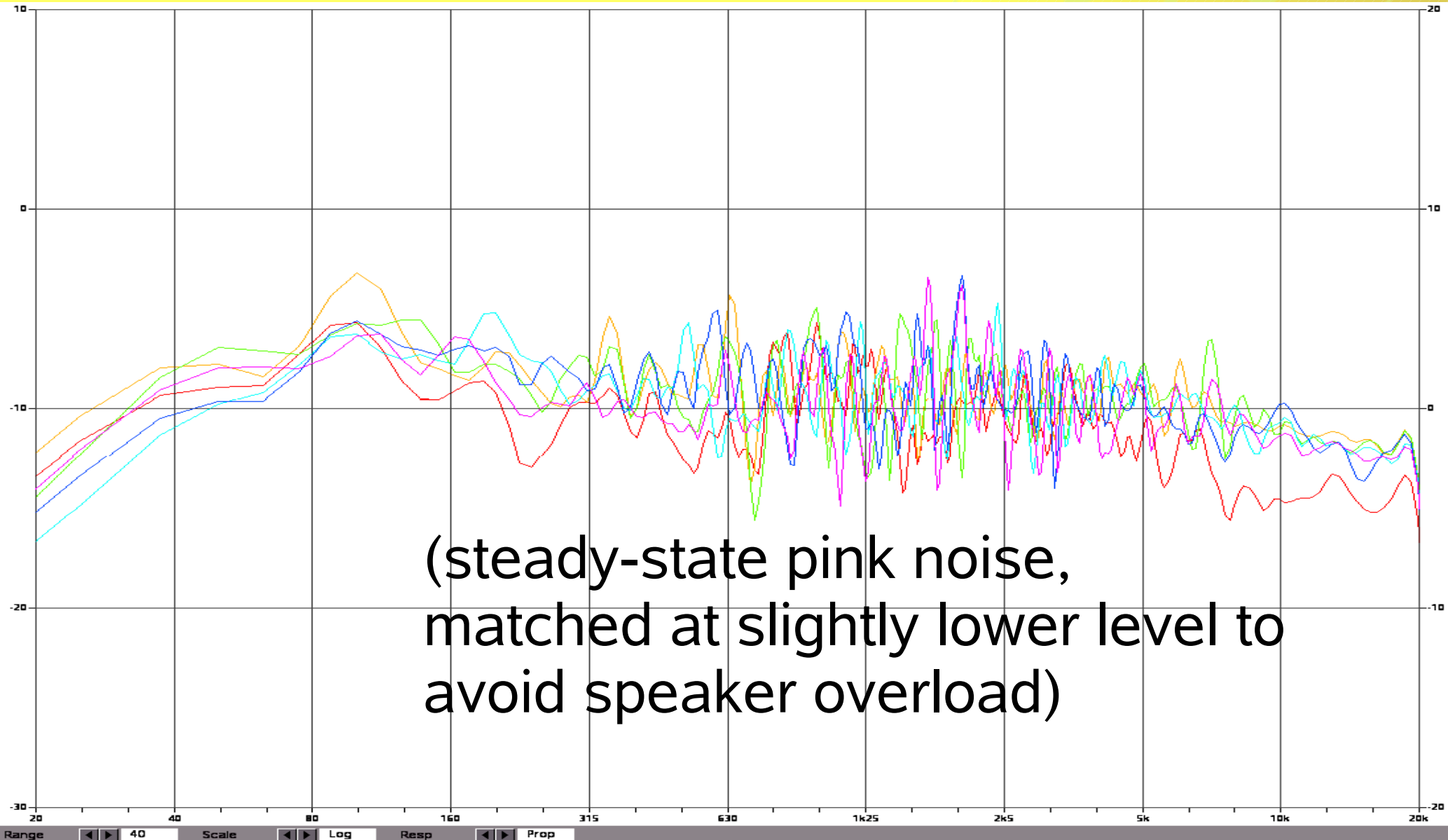
# Speaker No. 6 after DRC:



# Frequency response before DRC:



# Frequency response after DRC:



(steady-state pink noise,  
matched at slightly lower level to  
avoid speaker overload)

# Evaluation of the DRC:

Listening tests show **more pleasant timbre**, especially with stringed instruments, more **„sparkle“ in the treble band**, a **„tighter“ bass** and improved precision of **localisation**.

(The speakers used for this test are low-end active studio monitors with a cost of ca. 200€ per channel.)

# Thanks to:

the VdT and especially Sebastian Gabler for the invitation,

Fons Adriaensen (author of Aliko, AmbDec and JConv),

Denis Sbragion (author of drc),

the JACK development team,

the Ardour development team,

Eric Benjamin, Richard Lee and other patrons of the sursound mailing list,

and of course

<http://www.have-a-great-time-in-south-australia.com> for the shark images.



# Thanks for your attention!

# Your Questions?



# References:

## **Aliki, AmbDec, JConv, JAPA:**

<http://kokkinizita.net>

**drc:** <http://drc-fir.sourceforge.net>

**JACK:** <http://jackaudio.org>

**Ardour:** <http://ardour.org>

**qjackctl:** <http://qjackctl.sourceforge.net>

**sox:** <http://sox.sourceforge.net>

Farina 2000: *Angelo Farina, Simultaneous measurement of impulse response and distortion with a swept-sine technique, AES Preprint, <http://pcfarina.eng.unipr.it/Public/Papers/134-AES00.PDF>, 2000*

*The corresponding paper to this presentation can be downloaded from [http://spunk.dnsalias.org/public\\_stuff/linux\\_audio/tmt08/](http://spunk.dnsalias.org/public_stuff/linux_audio/tmt08/)*