Audio Measurements Workshop (part 2)

Fons Adriaensen
Huawei Europen Research Center, München

Linux Audio Conference 2015
JGU Mainz, Germany
1 Overview

• Theory.
  * No theory today. See slides of part 1.

• Quick intro to some tools
  * jaaa, jnoisemeter
  * python, numpy, scipy, pyaudiotools
  * matplotlib, pyqtgraphics

• Howto measure
  * Noise levels, S/N ratio
  * Frequency response
  * Harmonic distortion
  * Intermodulation distortion

• Practice as we go along.
Philosophical issues (again)

- Why measure things?
  * Verify your design and programming.
  * Have you been ripped off?
  * To know limits and create a level of confidence.
  * Curiosity.

- Always expect the unexpected. It happens. If your measurements are exactly as you imagined they would be, then
  * Congratulations!
  * It’s time to verify things and ask some questions.

- It’s very easy to overlook things and get completely wrong results.
  - Always double check and ask yourself 'can this be true', and if yes, how?
- Technical spectrum analyser.
- Accurately measure sine waves in noise and noise density.
4 Jnoisemeter

- Measure noise according to various standards.
- Filtered signal available at output.
• **python** : interpreted OO language, many extensions.
  
  – **numpy** : efficient calculations on arrays of any dimension.
  – **scipy** : scientific computation
  – **matplotlib, pyqtgraph** : 2d and 3d plotting.
  – **pyqt** : Qt bindings to python, much easier to use than plain Qt.

• **pyaudiotools** : collection of python classes for audio.
  
  – **AudioFile** : read/write audio files to/from numpy arrays.
  – **JackSignal** : Jack client, generate and capture arbitrary test signals from/to numpy arrays.
  – **Vresampler** : Resampling of numpy arrays.
  – Others not directly targeted at measurement.

• Combining all these makes it easy to create ad-hoc automated measuring tools of any complexity.
Other uses of pyaudiotools: quickly build complex audio systems

- All Jack modules are python classes.
- All parameters controlled by PyQt Gui.
- Complete system runs on Linux, OSX and Windows.
• Versatile python library for 2d and 3d plotting, actively developed.

• Some combined DSP/plotting code inherited from early releases, better ignore this. Current development concentrates on plotting only.

• Requires some getting used to, mainly because there are 3 APIs and these get mixed up in documentation and example code.
  – pylab: high level, stateful, MatLab style one-liners. DEPRECATED
  – pyplot: medium level, stateful, for interactive use.
  – object oriented: full control, required when combining with PyQt as pyplot uses a viewer which has its own event loop (and there can be only one).

• Advice: study the OO API.
• **Python library for scientific/engineering data visualisation.**
  – High quality presentation of complex data.
  – Complex interactive multiple-view displays.
  – Very efficient, 3D graphics use OpenGL.
  – Integrates perfectly with PyQt, adds its own set of specialised widgets.

• Adds some high level functionality:
  – LabView style flowchart of processing modules.
  – Parameter tree with graphic editor.

• Lots of 'not yet implemented' details, but actively developed.

• Advice: keep an eye on this.
9 Noise levels and S/N ratio

- One tool: jnoisemter.

- For a valid measurement you need:
  - defined BW or filter,
  - RMS or ITU response,
  - defined conditions: gain, source impedance, . . .

- Look at the spectrum as well:
  - is what what you measure really noise ?
  - is the spectrum what you expect (usually white) ?

- S/N ratio = working level / noise level.

- 'Dynamic range' = maximum level / noise level.
Frequency response

- The 'analog method': frequency sweep, measure output.
  - Need to coordinate sweep rate and detector speed.

- Single sample impulse and FFT.
  - Fast and accurate.
  - Also provides the impulse response.
  - Very low energy test signal, requires good S/N ratio.

- Log sweep, deconvolution and FFT.
  - High energy test signal, also works on noisy systems.
  - The same method can be used for speakers, room IRs etc.
• For a valid measurement, test conditions need to be defined.

• The analog method: sine wave test signal, filter out fundamental frequency and measure what remains (with jnoismeter).
  – The result is THD + noise.
  – Check the spectrum!

• Measure harmonics selectively using spectrum analyser or DSP code.
  – Usually 2nd and 3th harmonics dominate, but check higher ones.

• Analog electronics usually distorts most at higher levels.

• Exception: crossover distortion in power amplifiers.

• Digital electronics (AD/DA converters) can show high distortion at all levels, including very low ones.

• In digital domain, avoid 'round' frequencies.
Intermodulation distortion

• Test signal with two sine waves, F1, F2, measure level of
  – 2nd order IM: F1 - F2
  – 3th order IM: 2 * F1 - F2, 2 * F2 - F1
  – Higher order: k1 * F1 + k2 * F2, order = |k1| + |k2|

• SMPTE : F1 = 7 kHz, F2 = 60 Hz, amplitude 4 times that of F1, measurement relative to level of F1.

• DIN : F1 = 8 kHz, F2 = 250 Hz, amplitude 4 times that of F1, measurement relative to level of F1.

• IEC : F2 = F1 + 60 Hz, same amplitude, can be done at different frequencies. Measurement relative to sum of F1 and F2 level.

• In digital domain, avoid 'round' frequencies.