

# Diffrogram v3.35

The reference Matlab utility for DF measurements in audio.  
Manual with examples and verification vectors.

## Overview

The function compares two audio files which usually represent reference and output signals of some device under test. It operates piece-wise with a specified time window and computes the array of difference levels and powers:

*df(:,1)* - waveform DF [DFwf, dB]  
*df(:,2)* - magnitude DF [DFmg, dB]  
*df(:,3)* - phase DF [DFph, dB]  
*df(:,4)* - RMS power [PP, dB]

Each *df* vector is accompanied by the diffrogram – the image which has Time along horizontal dimension and Frequency - along vertical. It shows degradation of an output signal with time and frequency. Color of each pixel corresponds to DF level of an output signal in particular time frame and frequency sub-band. Brightness of its color corresponds to the energy of this portion of the output signal. DF/color mapping is defined algorithmically ["colormap" option]. On the diffrograms DF values below -150 dB are “clipped” to -150 dB; DF value *-Inf* dB (perfect match between reference and output waveforms) is coded with Grey. File name of the diffrogram image includes Median and [Max Min] DF values of corresponding *df* vector. First and last values for a signal are not counted as they are often erroneous due to edge effect.

For correct calculation of DF levels an output signal must be precicely time/pitch aligned to the reference signal. If required, this operation of time warping can be performed by the diffrogram function too. Resulting “warped” output signal is returned by the function.

## Syntax and description

*df* = *diffrogram33(fref, fout, Tw, options)* reads the audio files named *fref* and *fout* and returns three output arguments:

- an array *df* of DFwf, DFmg and DFph values in dB,
- three corresponding diffrogram images and
- a warped output signal.

The audio files can be of any sample rate and bit depth. All DF levels are computed for the region of interest 20Hz - 20kHz regardless of the sample rate.

*Tw* - width of the rectangular time window in milliseconds used for computing DF levels; if *Tw* is greater than the length of a reference signal, the diffrogram33 function computes scalar DFwf, DFmg and DFph levels for the whole signal.

*options* (a string, case insensitive, any order) must include at least one of the following arguments:

*Left* | *Right* | *Mono* - channel mode. Default: *Mono*

*NoWarp* - skip time warping of the output signal if it is known to be perfectly time/pitch aligned already.

*SyncMargin:integer* - time frame in milliseconds at the beginning and the end of a reference signal that is used for cutting the output signal accordingly. The default value is 3000 ms [or less if the signal is shorter]. Increase the value if the signals contain too quiet passages at the beginning and the end. Periodic signals of high frequency may require a shorter *SyncMargin* value: 1 ... 10. The output signal for the function can be prepared (cut out) approximately, with some extra margin of about 1/10th of *SyncMargin* value. If zero is specified the automatic/smart cutting is not performed and this operation must be done beforehand with the accuracy of a few samples.

*ColorMap:integer* - returns the color map of the size specified in pixels; all other input arguments are ignored. Default: 1500.

*WAV* - returns time warped output signal as a .wav file of 32 bit depth and sample rate of the reference signal.

*Octave1/6* | *Octave1/12* - output of diffrogram images with corresponding frequency bands. Default: *Octave1/12*

*WarpMargin:integer* - number of extra samples around the time window for more robust and accurate time warping. Default value 5 is suitable for most cases but can be reduced for high frequency Sine signals if the time warping produces errors - unreasonably low DF levels for some time frames. This is a development option.

Calling *diffrogram33(fref, fout, Tw, options)* without output argument returns diffrogram images and time warped output signal (if warping was applied and *WAV* specified).

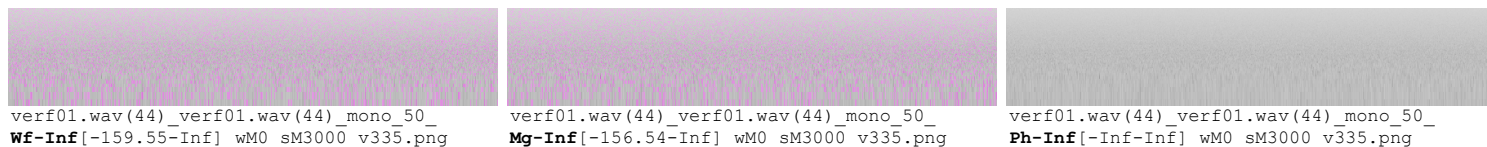
## Usage

These examples can be also used for verification of DF computation in third-party tools. Test vectors *verf01.wav*, *verf02.wav*, *verf03.wav*, *verf06.flac* and *verf07.flac* are included in this utility's package along with this document and *diffrogram33.m* Matlab code.

**#1.** The same white noise as input and output signal; without time warping.

```
diffrogram33('verf01.wav', 'verf01.wav', 50, 'Mono NoWarp');
```

Resulting diffrograms:



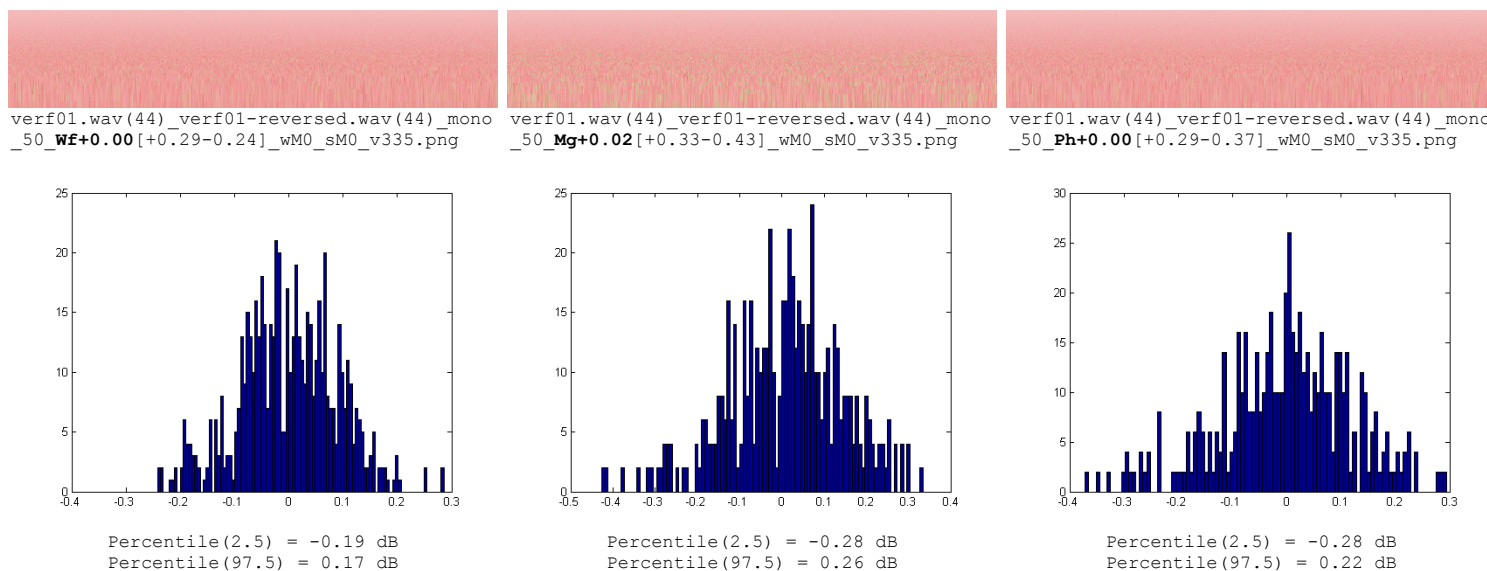
The diffrograms show the accuracy of computation DF levels for bit-identical signals without time warping.

**#2.** Uncorrelated white noise as output signal; without time warping; without initial approximate time alignment of the output signal.

```
% Prepare the time reversed (uncorrelated) signal
sig = audioread('verf01.wav');
sig = flipud(sig);
audiowrite('verf01-reversed.wav', sig, 44100, 'BitsPerSample', 32)
```

```
df = diffrogram33('verf01.wav', 'verf01-reversed.wav', 50, 'Mono NoWarp SyncMargin:0');
```

Resulting diffrograms with distributions of DF levels:

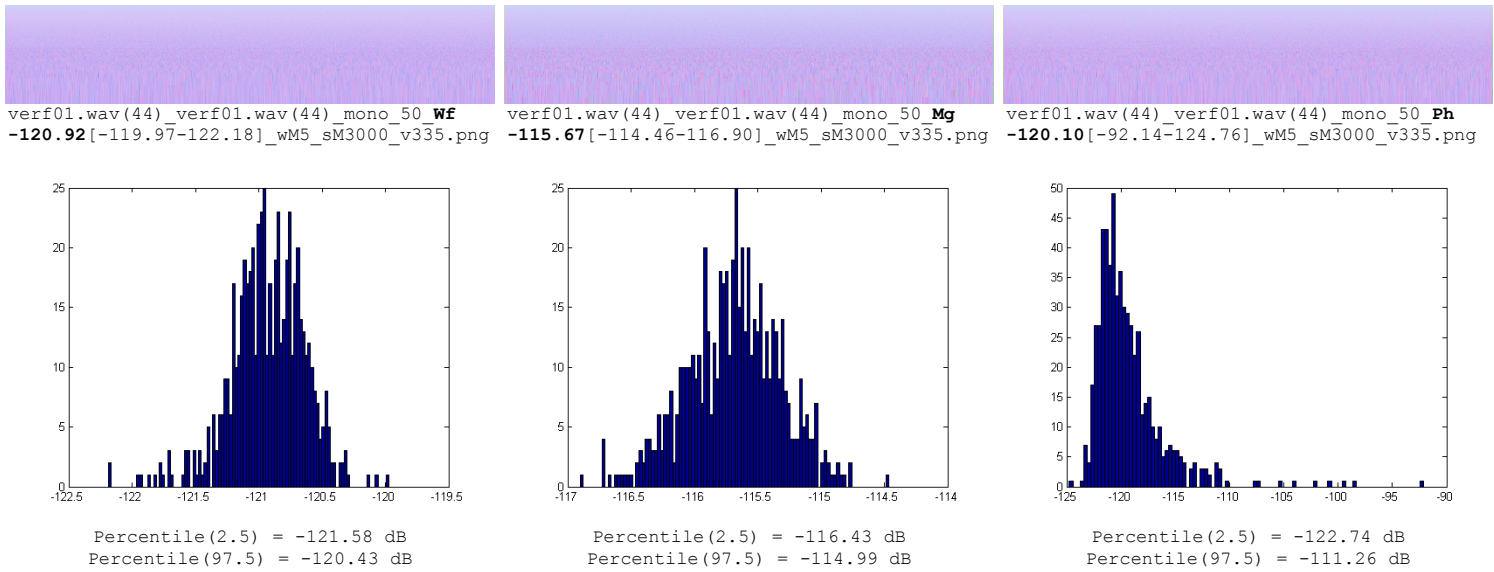


Examples #1 and #2 show the limiting cases for DF computation.

#3. The same white noise as input and output signal; with time warping.

```
diffrogram33('verf01.wav', 'verf01.wav', 50, 'Mono');
```

Resulting diffrograms with distributions of DF levels:



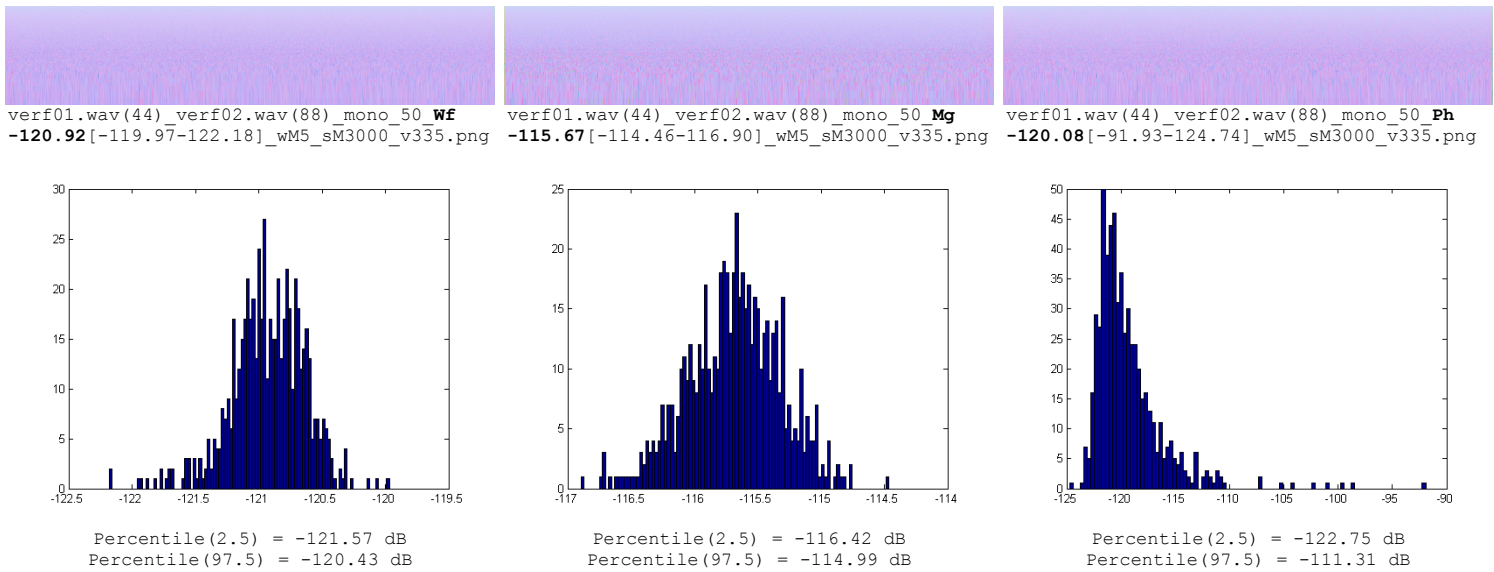
The diffrograms shows the accuracy of time warping and the lowest (best) measurement limits of DF computation for non-aligned signals.

#4. The same white noise at higher sample rate as the output signal; with time warping.

The signals *verf01.wav* and *verf02.wav* represent the same white noise because they are sampled from the same population of band-limited white noise @176400 Hz sample rate. Results of the computation must be identical to example #3.

```
diffrogram33('verf01.wav', 'verf02.wav', 50, 'Mono');
```

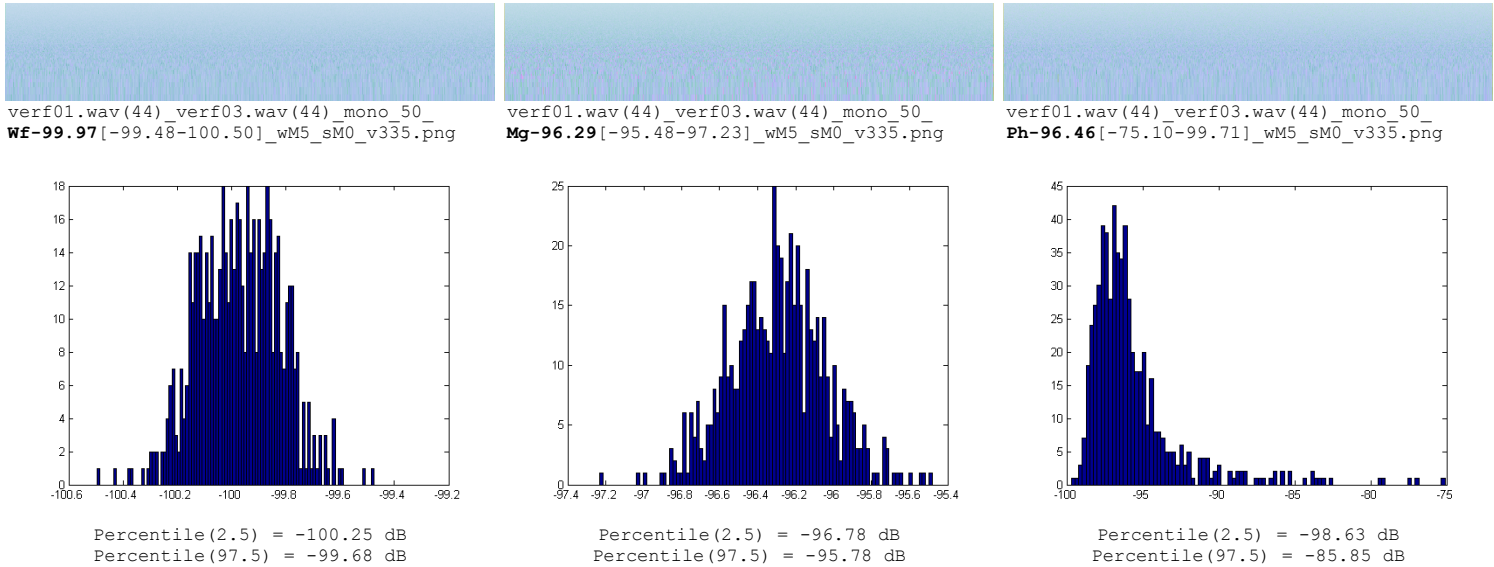
Resulting diffrograms with distributions of DF levels:



#5. The same white noise, time/pitch shifted with added noise, as the output signal; with time warping.

```
df = diffrogram33('verf01.wav', 'verf03.wav', 50, 'Mono SyncMargin:0');
```

Resulting diffrograms with distributions of DF levels:

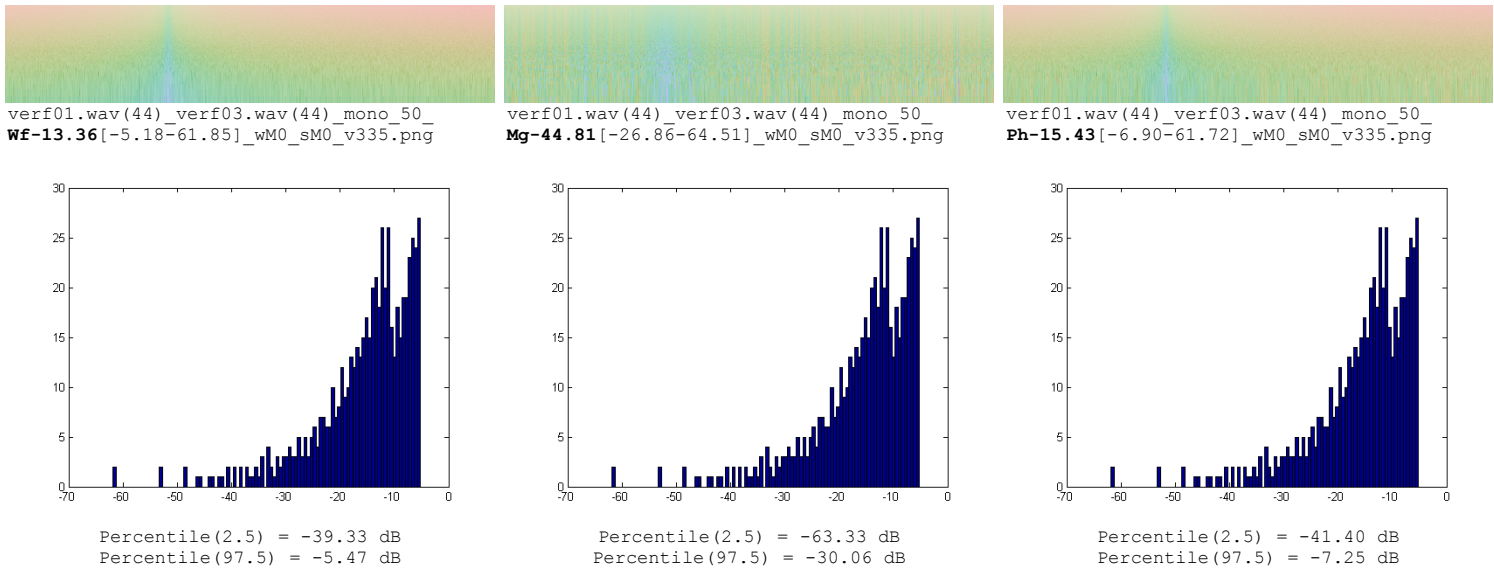


By design DFwf level of `verf03.wav` is precisely equal to -100 db and it's measured correctly by the utility despite some time/pitch deviation. Without precise time warping (example #6) DF measurements are completely erroneous.

#6. The same white noise, time/pitch shifted with added noise, as the output signal; without time warping.

```
df = diffrogram33('verf01.wav', 'verf03.wav', 50, 'Mono NoWarp SyncMargin:0');
```

Resulting diffrograms with distributions of DF levels:



Even the slightest time misalignment of input and output signals results in substantial increse (worsening) of DF levels and is visible on diffrograms. After the time warping the level of added noise in the output signal can be precisely measured (example #5).

## #7 Full-band white noise compared to its version down-sampled and quantized to 16bit

```
% prepare the signals
PeakdB = -1; % [dB]
% full-band white noise, 30s, mono, 192k/32bit
noise192 = randn(30*192000,1);
noise192 = noise192 .* ((10^(PeakdB/20))/max(abs(noise192)));
audiowrite('verf04.wav', noise192, 192000, 'BitsPerSample',32)
% downsampled and quantized version of the noise
noise48 = resample(noise192,1,4,24000); % downsample to 1/4
noise48 = noise48 .* ((10^(PeakdB/20))/max(abs(noise48)));
noise48 = int16(round(noise48.*(2^15))); % quantize to 16bit
audiowrite('verf05.wav', noise48, 48000, 'BitsPerSample',16)

df = diffrogram33('verf05.wav', 'verf04.wav', 50, 'Mono WarpMargin:5 SyncMargin:0');
```

Resulting diffrograms with distributions of DF levels:



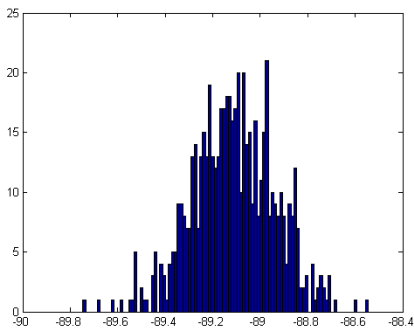
verf05.wav(48)\_verf04.wav(192)\_mono\_50\_  
**Wf**-89.11[-88.55-89.75]\_wM5\_sM0\_v335.png



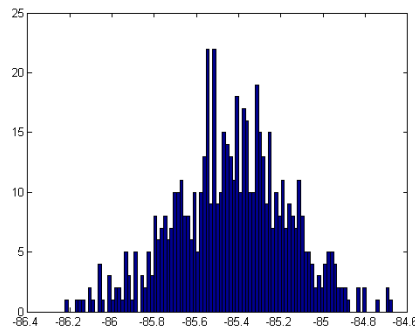
verf05.wav(48)\_verf04.wav(192)\_mono\_50\_  
**Mg**-85.43[-84.67-86.22]\_wM5\_sM0\_v335.png



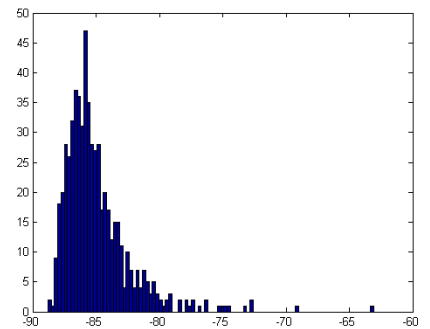
verf05.wav(48)\_verf04.wav(192)\_mono\_50\_  
**Ph**-85.64[-63.07-88.80]\_wM5\_sM0\_v335.png



Percentile(2.5) = -89.47 dB  
Percentile(97.5) = -88.77 dB



Percentile(2.5) = -86.00 dB  
Percentile(97.5) = -84.93 dB



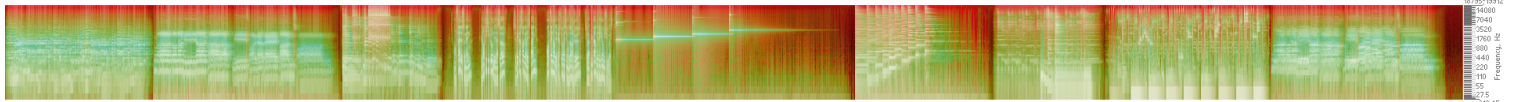
Percentile(2.5) = -87.96 dB  
Percentile(97.5) = -77.71 dB

DFwf of these two vectors should be equal to -89dB with some small variance (+/- 0.75 dB) due to stochastic nature of these signals generation.

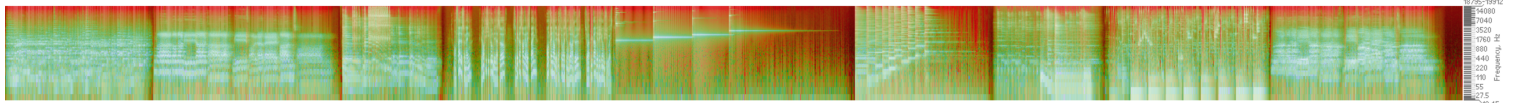
## #8 The real-life example of testing Apple dongle (A1749) with the nine sound samples used for listening tests on SoundExpert [\[soundexpert.org/sound-samples\]](https://soundexpert.org/sound-samples). The OUT signal was recorded with PCM4222 ADC.

```
df = diffrogram33('verf06.flac', 'verf07.flac', 50, 'Mono');
```

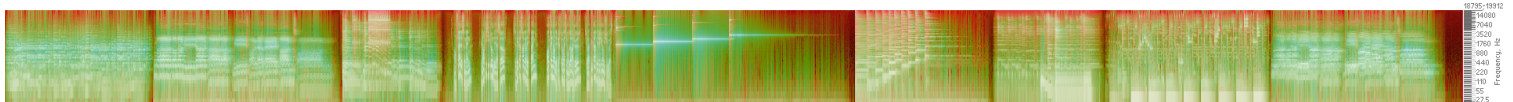
Resulting diffrograms with distributions of DF levels:



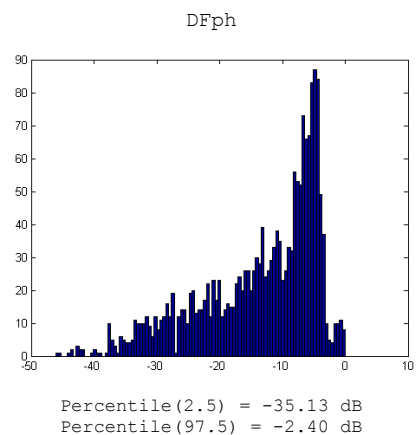
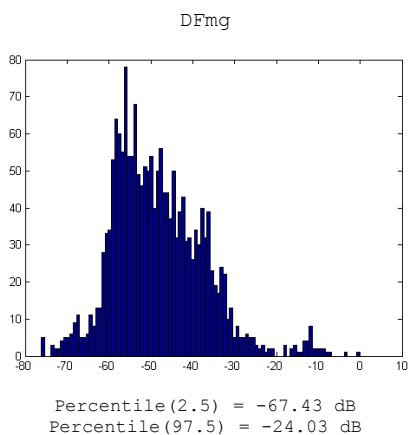
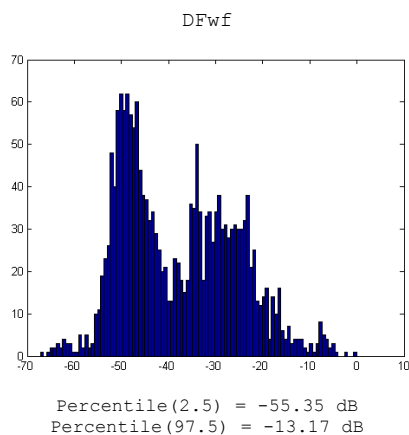
verf06.flac(44)\_verf07.flac(48)\_mono\_50\_  
**Wf**-39.22[+0.00-67.18]\_wM5\_sM3000\_v335.png



verf06.flac(44)\_verf07.flac(48)\_mono\_50\_  
**Mg**-49.73[+0.00-76.16]\_wM5\_sM3000\_v335.png



verf06.flac(44)\_verf07.flac(48)\_mono\_50\_  
**Ph**-10.79[+0.00-46.10]\_wM5\_sM3000\_v335.png



While the magnitude performance of A1749 is pretty good, the phase performance is substantially worse. The latter is the main factor of sound degradation in the device.

\* \* \*

These signals are sufficient and recommended for testing other implementations of df computing.

The diffrogram utility v3.35 measures DFwf levels down to -100dB with the error less than 0.1dB for any signals.

More details about diffrograms can be found in the article "Diffrogram: visualization of signal differences in audio research"  
[\[soundexpert.org/articles/-/blogs/visualization-of-distortion#part3\]](https://soundexpert.org/articles/-/blogs/visualization-of-distortion#part3)