MIR démo

MIRtoolbox

General Principles

List of all the functions available in *MIRtoolbox*:

```
help mirtoolbox
```

Help for one specific function:

```
help miraudio
```

Display the waveform of audio file 'ragtime.wav' and store the result in a variable a:

```
a = miraudio('ragtime.wav')
```

Add ';' to avoid the display of the results:

```
a = miraudio('ragtime.wav');
```

Optional operations such as centering and resampling:

```
a = miraudio('ragtime.wav', 'Center', 'Sampling', 11025)
```

or in several steps:

```
a = miraudio('ragtime.wav', 'Center')
a = miraudio(a, 'Sampling', 11025)
```

Play the resulting audio:

```
mirplay(a)
```

Analyze all files of the current folder (for instance, the folder located at the address: MIRToolboxDemos/train_set) :

```
miraudio('Folder')
```

Compute for instance the spectrum of a:

```
mirspectrum(a)
```

Obtain the output in numeric format:

```
mirgetdata(a)
```

Get additional data stored with your variable *a*, such as the sampling rate:

```
get(a,'Sampling')
```

Basic Operators

Trim the silence at the beginning and the end of the audio:

```
a = miraudio(a,'Trim')
```

Extract the first second only:

```
miraudio(a,'Extract',0,1)
```

Extract the amplitude envelope:

```
e = mirenvelope(a)
```

Compute the spectrum (Fourier transform):

```
s = mirspectrum('Amin3.wav')
```

Select frequencies below 3000 Hz:

```
s = mirspectrum(s,'Max',3000)
```

Display the energy in decibel scale:

```
s = mirspectrum(s,'dB')
```

Decompose the energy in Mel bands:

```
s = mirspectrum(a,'Mel')
```

Detect peaks:

```
mirpeaks(s)
```

Compute the autocorrelation function:

```
ac = mirautocor('Amin3.wav')
```

Display the function in frequency domain:

```
ac = mirautocor('Amin3.wav','Freq')
```

Decompose the audio into frames of length 0.1 s. and half-overlapped:

```
f = mirframe('ragtime.wav',.1,.5)
```

Compute the spectrum for each frame (i.e., spectrogram):

```
s = mirspectrum(f)
```

This can be written in one line:

```
s = mirspectrum('ragtime.wav', 'Frame', .1, .5)
```

Detect the peaks:

```
mirpeaks(s)
```

Compute the resulting spectral flux:

```
mirflux(s)
```

The "flux" can be computed for any frame-decomposed representation, for instance:

```
mirflux(mirautocor(a,'Frame'))
```

Decompose the audio signal into 5 channels:

```
fb = mirfilterbank(a, 'NbChannels', 5)
```

Compute the envelope in each channel:

```
e = mirenvelope(fb)
```

Compute the autocorrelation function for each envelope in each channel:

```
ae  = mirautocor(e)
```

Compute the autocorrelation summary:

```
sa = mirsum(ae)
```

Feature Extractors

Dynamics

Root-mean-square energy curve:

```
r1 = mirrms('movie1.wav', 'Frame')
r2 = mirrms('movie2.wav', 'Frame')
```

Low-energy rate:

```
mirlowenergy(r1)
mirlowenergy(r2)
```

Rhythm

Tempo estimation:

```
[t,ac] = mirtempo('ragtime.wav')
```

Temporal evolution of tempo estimation:

```
[t,ac] = mirtempo('czardas.wav', 'Frame')
```

Timbre

Attack Onset estimation:

```
mironsets('ragtime.wav')
```

Attack slope:

```
mirattackslope('ragtime.wav')
```

Brightness

```
mirbrightness('ragtime.wav', 'Frame')
```

Roughness:

```
mirroughness('ragtime.wav', 'Frame')
```

Pitch and Tonality

Multi-pitch extraction:

```
[p,a] = mirpitch('ragtime.wav', 'Frame','Multi')
```

Wrapped chromagram:

```
mirchromagram('ragtime.wav')
```

Tonality estimation:

```
mirkey('ragtime.wav')
k = mirkey('ragtime.wav','Frame')
```

Mode estimation:

```
mirmode('ragtime.wav')
m = mirmode('ragtime.wav','Frame')
```