# pysoundanalyser Documentation

***Release ('0.2.41',)***

**Samuele Carcagno**

**May 28, 2020**

# CONTENTS

Contents:

CHAPTER

ONE

# WHAT IS PYSOUNDANALYSER?

pysoundanalyser is an application which provides a graphical user interface to analyse short (in the range of seconds) wav files. I developed and use it mainly to quickly load and visualize the waveforms and the spectral content of sounds generated for psychoacoustics research. pysoundanalyser can also generate some types of sounds used in psychoacoustics research (e.g. pure tones, amplitude modulated tones, frequency modulated tones, different colors of noise, etc...).

The repository of pysoundanalyser is hosted on GitHub. If you find bugs, please report them there.

CHAPTER

TWO

# INSTALLATION

pysoundanalyser has been successfully installed and used on Linux, Windows, and Mac platforms. Installation instructions for each operating system are provided below.

## 2.1 Installation on Linux

For Debian and Ubuntu LTS releases there are apt repositories that can be used to install and update pysoundanalyser. For other Linux distributions pysoundanalyser has to be installed from source (see Section *Installation from source*).

### 2.1.1 Installation on Debian

Binary packages for the Debian amd64 architecture are hosted on bintray. To install pysoundanalyser first install the apt-transport-https package if it is not already installed:

```
sudo apt-get install apt-transport-https
```

then add one of the following lines to /etc/apt/sources.list depending on your Debian version: For Jessie (stable):

```
deb https://dl.bintray.com/sam81/hearinglab jessie main
```

For Stretch (testing):

```
deb https://dl.bintray.com/sam81/hearinglab stretch main
```

Download the key with which the repository is signed and add it to the apt keyring:

```
wget -qO - https://bintray.com/user/downloadSubjectPublicKey?username=bintray | sudo␣
↪apt-key add -
```

Refresh the package database and install the package:

```
sudo apt-get update
sudo apt-get install pysoundanalyser
```

### 2.1.2 Installation on Ubuntu LTS Releases

Binary packages for Ubuntu Long Term Support (LTS) releases are hosted on Launchpad. To install pysoundanalyser run the following commands:

```
sudo add-apt-repository ppa:samuele-carcagno/hearinglab
sudo apt-get update
sudo apt-get install pysoundanalyser
```

## 2.2 Installation on Windows

A Windows installer is provided on the downloads page:

http://samcarcagno.altervista.org/pysoundanalyser/pysoundanalyser.html

## 2.3 Installation on the Mac

The easiest way to install `pysoundanalyser` on the Mac is to use Pyzo as a Python distribution. The steps are similar as those for the installation with Pyzo on Windows (see below). Please, note that if you install with Pyzo you will need to use the PySide version of `pysoundanalyser`.

## 2.4 Installation from source

`pysoundanalyser` depends on Python and a handful of Python modules. There are two ways to obtain these dependencies. One is to install Python and all the dependencies "manually" (that is one by one). The other (and easier) way is to install a Python distribution that comes with a bundle of pre-installed modules. These include:

- Pyzo: http://www.pyzo.org/
- Anaconda: **'https://www.continuum.io/downloads https://www.continuum.io/downloads>'_**
- WinPython (Windows only): **'http://winpython.github.io/ http://winpython.github.io/>'_**

Step by step instructions to install `pysoundanalyser` on Windows with Pyzo are provided below. Although these instructions are specific to Windows the installation steps are similar on Mac OS X and Linux systems. If you get stuck at some point don't hesitate to get in touch <sam.carcagno@gmail.com>.

### 2.4.1 Install with Pyzo on Windows

Download the latest version on Pyzo and unpack it to a folder of your choice. Download the Windows source package (PySide) of *pysoundanalyser*. Open the DOS command prompt and change directory to the folder where you unpacked `Pyzo`, which should contain the *python.exe* executable. For example:

```
cd C:\Users\audiolab\Desktop\pyzo2013c
```

once you're in the Pyzo folder, you can instruct the Pyzo Python interpreter to run *pysoundanalyser* by calling python.exe followed by the path where the *pysoundanalyser* main file is located. For example:

```
python.exe "C:\Users\audiolab\Desktop\pysoundanalyser-pyside-0.2.81\pysoundanalyser.
↪pyw"
```

Currently there is not an application launcher. There is, however, a file called `pysoundanalyser-launcher.bat` inside the *scripts* folder of the source distribution of *pysoundanalyser* that after some modifications can be used as a launcher. The content of the file is the following:

```
C:\Python32\python "C:\Python32\site-packages\pysoundanalyser.pyw"
```

The first statement `C:\Python32\python` is the path to the Python executable. The second statement is the path to the main file of the `pysoundanalyser` app. You simply need to replace those two statements to reflect the Python installation on your system. Following the example above, you would change the contents of the file to:

```
C:\Users\audiolab\Desktop\pyzo2013c\python.exe
↪"C:\Users\audiolab\Desktop\pysoundanalyser-pyside-0.2.81\pysoundanalyser.pyw"
```

You can place the `.bat` launcher wherever you want, for example on your `Desktop` folder. Simply double click on it, and `pysoundanalyser` should start.

### 2.4.2 "Manual" Installation from Source

`pysoundanalyser` depends on the installation of a handful of other Python modules:

- Python (version 3) http://www.python.org/
- numpy http://sourceforge.net/projects/numpy/files/
- scipy http://sourceforge.net/projects/scipy/files/
- matplotlib http://matplotlib.org/

additionally it is necessary to install one of the modules providing Python bindings to the Qt widgets toolkit. There are three parallel versions of `pysoundanalyser` that support the major modules providing Python bindings to Qt (PyQt5, PyQt4, and PySide). You need to install only one of these modules, and use the corresponding version of `pysoundanalyser`

- PyQt5 https://riverbankcomputing.com/software/pyqt/download5
- PyQt4 http://www.riverbankcomputing.co.uk/software/pyqt/download
- PySide https://pypi.python.org/pypi/PySide/

these programs need to be installed manually. Once these programs are installed you can proceed with the installtion of `pysoundanalyser`:

```
python3 setup.py install
```

you can then invoke `pysoundanalyser` from a terminal by typing the command

```
pysoundanalyser.pyw
```

### 2.4.3 Install Python and the Dependencies manually on Windows

Please, note that you will need Python version 3 or above to run *pysoundanalyser*.

To install the dependencies, download them from their respective websites. Make sure that you pick versions compatible with your architecture (64 or 32 bits), and compatible with you Python version.

After installing the dependencies, it is recommended to add the directory where the Python executable resides to the system `PATH`. In this way you can call `python` from a `DOS` shell by simply typing its name, rather than typing the full path to the Python executable.

By default `python` is installed in `C:`. The name of the Python directory depends on its version number, for example, if you installed Python version 3.2, the python directory will be `C:\Python32`. To add this directory to the system path go to `My Computer` and click `Properties`, then click `Advanced System Settings`. In the `System`

Properties window click Environment Variables. There you will find an entry called Path. Select it and click Edit. Be careful not to remove any of the entries that are already written there because it could corrupt your system. Simply append the name of the full path of the folder where Python is installed, at the end of the other entries.

To run pysoundanalyser, unpack the pysoundanalyser .zip file containing the source code. Open a DOS shell, cd to the directory where you unzipped pysoundanalyser and launch it with the following command:

```
python pysoundanalyser.pyw
```

Currently there is not an application launcher. There is, however, a file called pysoundanalyser-launcher.bat inside the *scripts* folder of the source distribution of *pysoundanalyser* that after some modifications can be used as a launcher. The content of the file is the following:

```
C:\Python32\python "C:\Python32\site-packages\pysoundanalyser.pyw"
```

The first statement C:\Python32\python is the path to the Python executable. The second statement is the path to the main file of the pysoundanalyser app. You simply need to replace those two statements to reflect the Python installation on your system. You can place the .bat launcher wherever you want, for example on your Desktop folder. Simply double click on it, and pysoundanalyser should start.

CHAPTER

# THREE

# USER INTERFACE

- **Load Sound** The `Load Sound` button allows you to load a wav file into the program. Currently only 16 and 32 bit wav files with one or two channels are supported. Note that the entire wav file is loaded in memory, this is fine and fast for wav files of short durations (tens of seconds), but longer sound files are going to consume huge amounts of RAM and may even halt your computer. If you need to work on long sound files, please use other software, like audacity.
  - **Save As** The `Save As` button allows you to save PSA sound objects as wav files. Currently it is only possible to save sounds as 16 or 32 bit wav files with one or two channels. If you choose a mono (1-channel) format, and multiple PSA sound objects have been selected for saving, they will be summed together before saving. If you choose a stereo (2-channels) format, "right" and "left" PSA sound objects will be saved to their respective channels in the wav file; if multiple "right" or "left" PSA sound objects have been selected, they will be summed before saving.
- **Clone Sound**
- **Concatenate**
- **Cut** The `Cut` button allows you to remove segments of a sound waveform. The starting and ending points of the segments to be cut off can be specified in seconds, milliseconds, or sample numbers. When working with sample numbers, note that PSA indexing works exactly like numpy indexing. Examples:

```
>>> import numpy as np #import numpy
>>> sig = np.arange(10) #generate a 10-elements array
>>> x[0:5] #Select the first five samples, indexing starts from 0
array([0, 1, 2, 3, 4])
>>> sig[7:10] #select last 3 samples
array([7, 8, 9])
>>> sig[1:3] #select the second and third sample
array([1, 2])
>>> sig[1:2] #select the second sample
array([1])
>>> sig[1:1] #note that if the start and end point are the same nothing is␣
↪selected
array([], dtype=int64)
```

- **Play** The `Play` button allows playback of the currently selected sound.
- **Plot Waveform** The `Plot Wavform` button allows you to plot the waveform of the currently selected sound.
- **Spectrum** Plot the spectrum of the currently selected sound.
- **Spectrogram** Plot the spectrogram of the currently selected sound.
- **Autocorrelation** Plot the autocorrelation function of the currently selected sound.
- **Autocorrelogram** Plot the autocorrelogram of the currently selected sound.

- **Level Difference** Show the difference in level between two selected sounds.
- **Scale** Change the level of the currently selected sound.
- **Resample** Resample the currently selected sound.
- **Rename** Rename the currently selected sound.
- **Remove** Remove the currently selected sound from the workspace.
- **Remove All** Remove all sounds from the workspace.

CHAPTER

FOUR

# INDICES AND TABLES

- genindex
- modindex
- search