SoundscapeMeter

Bioacustic software: version 1.0 beta

USER MANUAL

Almo Farina, Emanuele Lattanzi, Luigi Piccioli, Nadia Pieretti
Department of Basic Sciences and Foundations
The Urbino University
Urbino, Italy

Written by Almo Farina, Luigi Piccioli and Nadia Pieretti.

The SoundscapeMeter plug-in was developed by Luigi Piccioli based on the algorithms designed by Almo Farina.

Copyright Page.

Almo Farina, Emanuele Lattanzi, Luigi Piccioli, Nadia Pieretti.
Department of Basic Sciences and Foundations
The Urbino University, Urbino, Italy.

Department of Basic Sciences and Foundations, The University of Urbino, Italy

CONTENTS

Department of Basic Sciences and Foundations, The University of Urbino, Italy

1. INTRODUCTION

1.1 About the SoundscapeMeter plug-in

The SoundscapeMeter is based on the algorithms designed by Almo Farina. It was developed by Luigi Piccioli with the support and under the supervision of Emanuele Lattanzi.
The **SoundscapeMeter** is a plug-in of **WaveSurfer© 1.8.5** (Sjölander & Beskow 2000).

Wavesurfer© is used to visualize and analyze sounds and as a way of applying the **SoundscapeMeter** plug-in. All of the information relating to the **Wavesurfer**© functions can be downloaded for free from:
http://www.speech.kth.se/wavesurfer/man.html

Wavesurfer can be downloaded for Linux, Windows and OSX system.

This manual only sets out the basic procedures for installing **WaveSurfer**. We strongly recommend that you read the **Wavesurfer**© manual and take the tutorial before using the **SoundscapeMeter** plug-in.

SoundscapeMeter plug-in has been tested in **Windows Vista, Windows 7**.

File format requested:
WaveSurfer© can read a number of sound file formats: WAV, AU, AIFF, MP3, CSL and SD.

1.2 The SoundscapeMeter at a glance

The **SoundscapeMeter** is a plug-in that adds new functions to the **WaveSurfer©** software. It is specifically utilized to process sound files, and has been tested in soundscapes with different acoustic qualities in both natural and human disturbed contexts. The **SoundscapeMeter** processes numerical files that are the result of the transformation of sound files via a **WaveSurfer© FFT** procedure. It also calculates indices that describe the relevant components of the acoustic structure of the environmental context.

The **SoundscapeMeter** plug-in is an efficient way to extract from sound files information about matters like intensity and structural complexity belonging to a frequency's categories. The plug-in uses the following indices: **Acoustic Complexity** (**ACIt**, **ACIf**), **ACI Evenness**, **Shannon Entropy** and **Levenshtein Distance**; see Section 5 for details.

It is possible to set different parameters with the **SoundscapeMeter**, such as: time interval, frequency range and noise filter. It is also possible to compute one file at a time, or several files contemporaneously and automatically. This latter option is particularly useful when series of sound files are available.
It is also possible to subdivide a file into several sub-samples. This is very useful when you have large sound files and need to reduce the computational time. What is more, each parameter can be fixed according to the goals of the investigator and the results that are anticipated.

2. GETTING STARTED

2.1 Install the SoundscapeMeter plug-in

a. Create a **SoundscapeMeter** folder on your desktop or in another preferred location.

b. Download **WaveSurfer©** 1.8.5 and install it in the **SoundscapeMeter** folder. **WaveSurfer©** is released as open source software under a BSD style licence, and can be downloaded from sourceforge.net/projects/wavesurfer (we suggest that you download version **1.8.5** to reduce the likelihood that you will experience any minor issues).

c. Install the **SoundscapeMeter** plug-in and download **SCM.plug** and **Scmlib.dll** from the site: DiSBeF.

d. Drag **SCM.plug** to your Personal User Folder in the directory (e.g. here, we use the acronym PUF to refer to this folder, but its name is usually that of the personal computer owner):

C:\Users\ PUF\.wavesurfer\1.8\plugins\SCM.plug

e. Drag the library **Scmlib.dll** to the **SoundscapeMeter** folder:

f. Click on the **WaveSurfer©** icon in the **SoundscapeMeter** folder and run the program. The **SoundscapeMeter** adds new functions to the **WaveSurfer©** platform. It does not modify the shape and appearance of the **WaveSurfer©** windows, but does expand some specific context menus.

g. Click on the **Help** menu (Fig. 1) and then on **About Plug-ins** to verify if the **SoundscapeMeter** plug-in has been correctly installed. The acronym, **SCM** (**SoundscapeMeter**), should appear at the top of the list of the installed plug-ins (Fig.2).

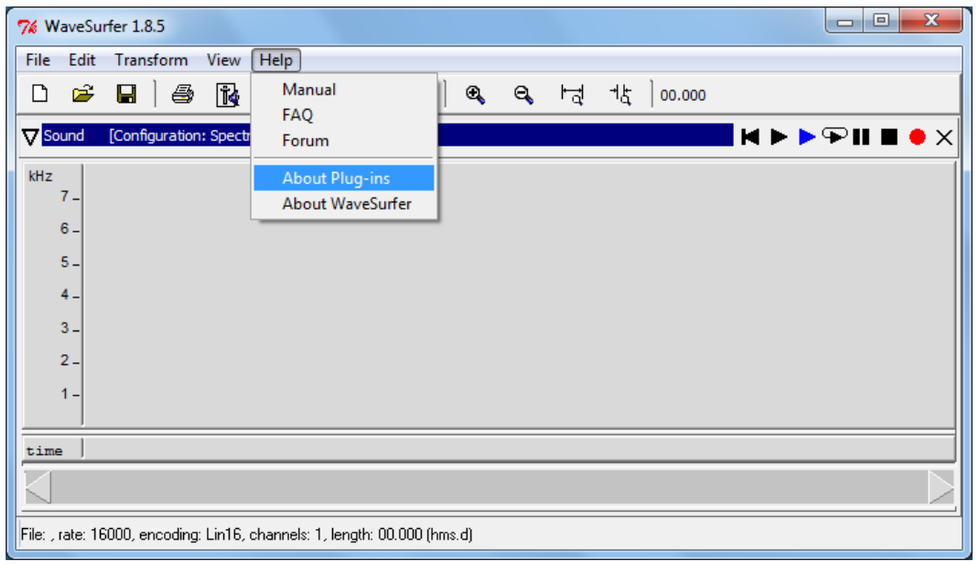

WaveSurfer 1.8.5
File
Edit
Transform
View
Help
Manual
FAQ
Forum
About Plug-ins
About WaveSurfer
00.000
Sound
[Configuration: Spectr
kHz
7
6
5
4
3
2
1
time
File: , rate: 16000, encoding: Lin16, channels: 1, length: 00.000 (hms.d)

Fig. 1

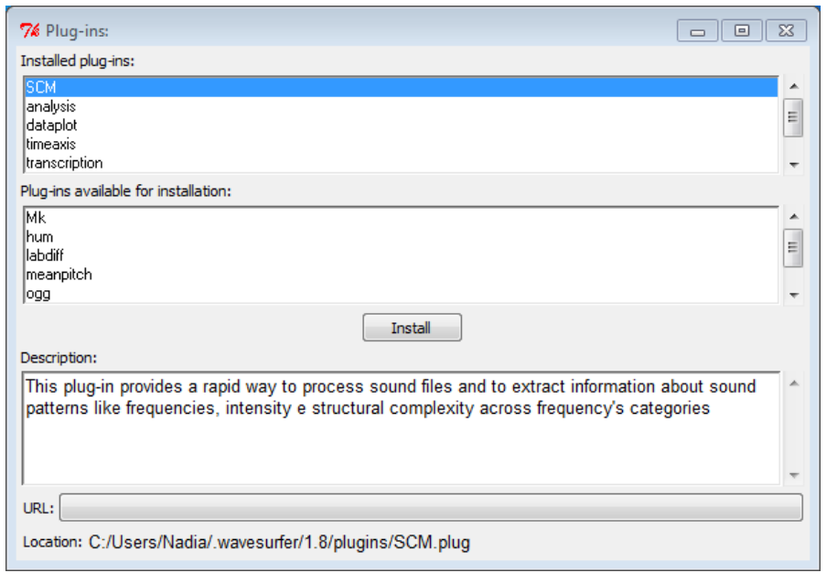

Plug-ins:
Installed plug-ins:
SCM
analysis
dataplot
timeaxis
transcription
Plug-ins available for installation:
Mk
hum
labdiff
meanpitch
ogg
Install
Description:
This plug-in provides a rapid way to process sound files and to extract information about sound patterns like frequencies, intensity e structural complexity across frequency's categories
URL:
Location: C:/Users/Nadia/.wavesurfer/1.8/plugins/SCM.plug

Fig. 2

2.2 Create a pane

The first step involves clicking the right mouse button inside the main **WaveSurfer**© window. This will open another window, with a list of commands, from which you should select <**Create a pane**> (Fig. 3). This will open a further menu from which you should select the ***Spectrogram*** option.

To avoid having to repeat this step in the future, you can open the sub-menu, ***Preferences***, in the ***File*** menu. Then, select the ***Misc*** menu followed by the ***Spectrogram*** in the ***Use configuration*** menu (Fig. 4).

This is the entire sequence:

Preferences | Misc | Use configuration | Spectrogram.

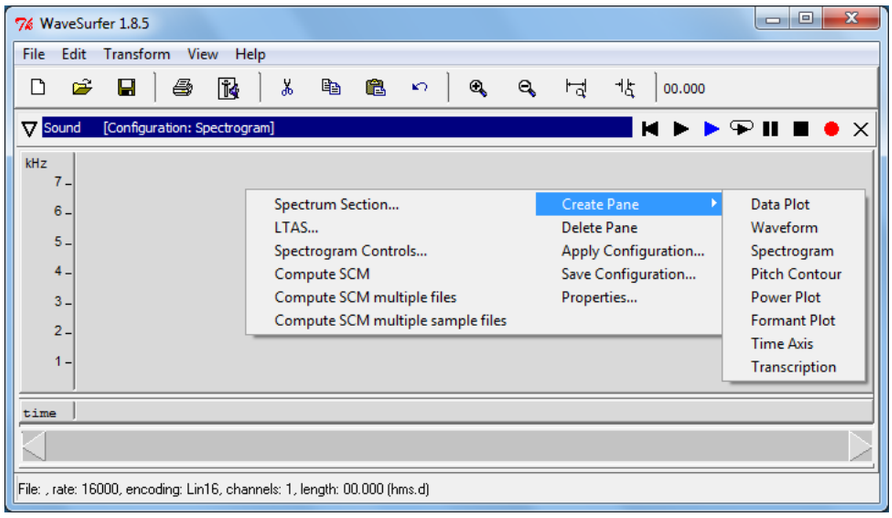


Fig. 3

2.3 Large files

Uploading very large sound files (f.i. more than 2GB) needs the default parameters to be changed. From the **File** menu, select **Preferences | Sound I/O | Sound storage** and then choose "**keep on disk**" (Fig. 5). Doing this will mean that the file will be accessed from the disk instead of being fully loaded into the computer's memory. This also applies when uploading several files during the "**Compute SCM multiple files**" procedure.

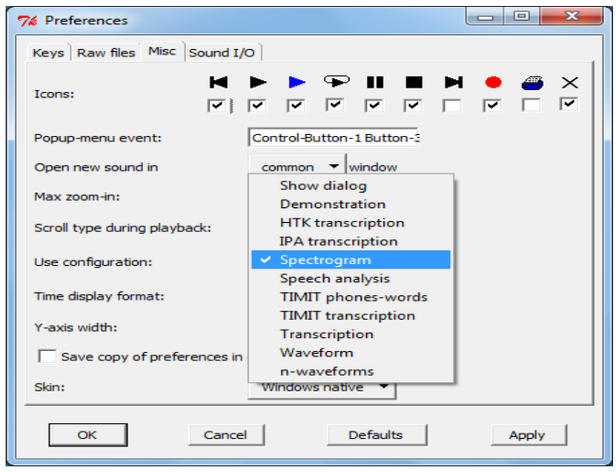

Preferences
Keys
Raw files
Misc
Sound I/O
Icons:
Popup-menu event:
Control-Button-1 Button-3
Open new sound in
common
window
Max zoom-in:
Scroll type during playback:
Use configuration:
Time display format:
Y-axis width:
Save copy of preferences in
Skin:
Windows native
Show dialog
Demonstration
HTK transcription
IPA transcription
Spectrogram
Speech analysis
TIMIT phones-words
TIMIT transcription
Transcription
Waveform
n-waveforms
OK
Cancel
Defaults
Apply

Fig. 4

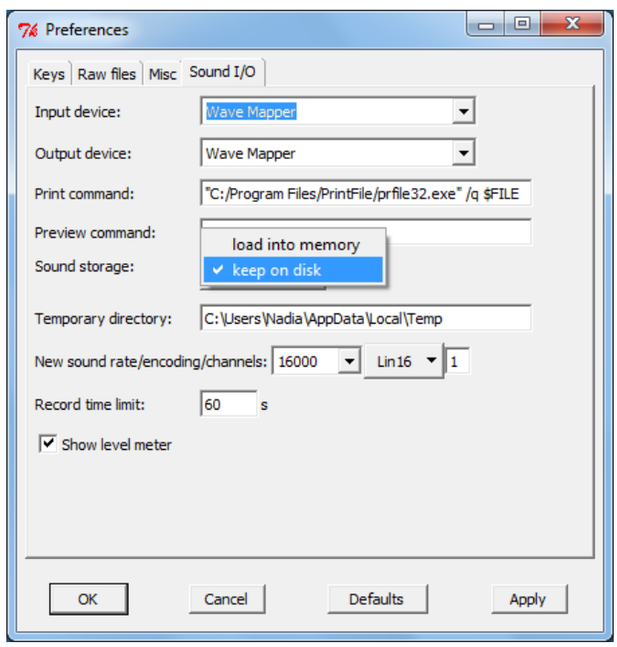

Preferences
Keys
Raw files
Misc
Sound I/O
Input device:
Wave Mapper
Output device:
Wave Mapper
Print command:
"C:/Program Files/PrintFile/prfile32.exe" /q $FILE
Preview command:
Sound storage:
load into memory
keep on disk
Temporary directory:
C:\Users\Nadia\AppData\Local\Temp
New sound rate/encoding/channels:
16000
Lin16
1
Record time limit:
60
s
Show level meter
OK
Cancel
Defaults
Apply

Fig. 5

3. THE PROCEDURES

The **SoundscapeMeter** has three different procedures (Fig. 6):

1. Compute one sound file at a time: <**Compute SCM**>.
2. Compute several files at once: <**Compute SCM multiple files**>.
3. Create several sub-sampled files from a unique file: <**Compute SCM multiple sample files**> (the computational processes of the **SoundscapeMeter** are applied to the sub-files).

The menu, **Properties**, fixes the parameters according to the different procedures.

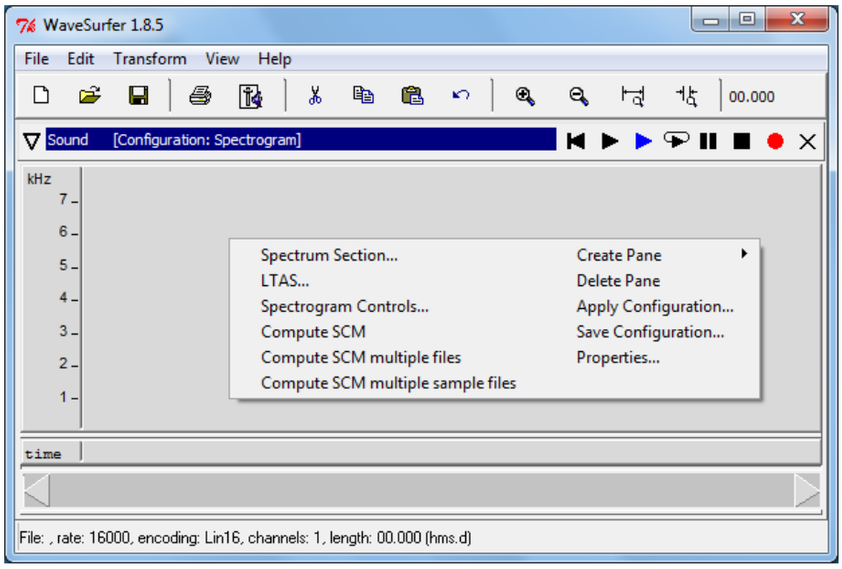


Fig. 6

3.1 Compute SCM (one sound file at a time)

3.1.1 Open a sound file ([**File**] <**open**>): this command is the first on the upper-left side of the **WaveSurfer**© mask. Select a file accordingly.

3.1.2 Click inside the main window with the right mouse button and a context menu will appear from which to select ***Properties*** (Fig. 7). Then, select the ***Spectrogram*** menu and choose the FFT window that is appropriate for your study. On the basis of our experience and previous research (see Annexes), we suggest 512 points. Then click on **Apply** (Fig. 8).

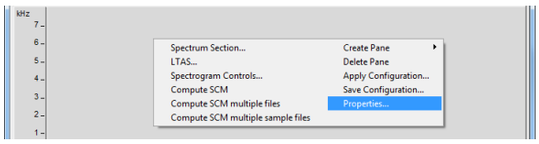


Fig. 7

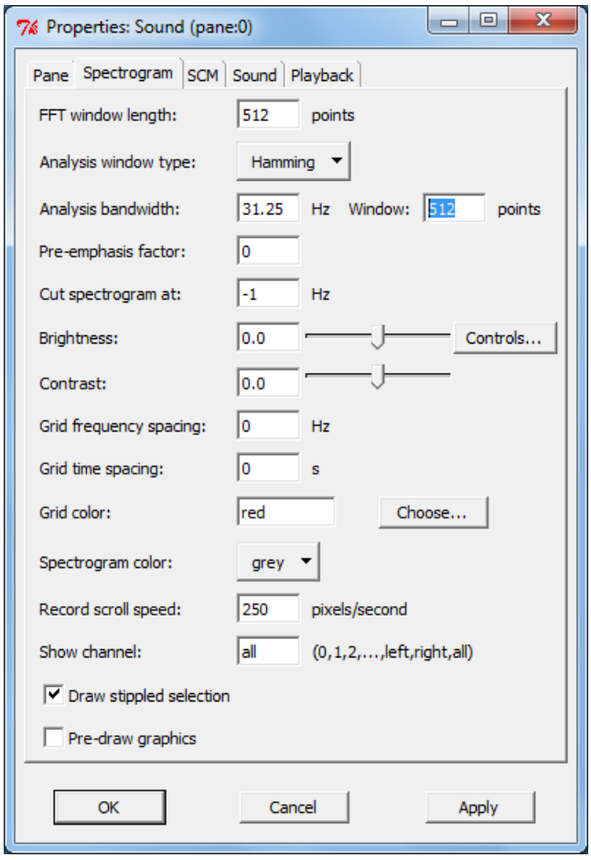


Fig. 8

3.1.3 Select the **SCM** context menu in the pane ***Properties*** (Fig. 9).
A sub-menu will enable you to set the following parameters:

Start Time: set the time in *mm:ss.ms* when the computation starts.
End Time: set the time in *mm:ss.ms* when the computation ends.
These two commands (Fig. 9 and Fig. 10) allow you to select a temporal window inside a sound file within which to perform the required computation.The entire file is processed when you select the default setting (**Start Time** 00:00.00; **End Time** -1).

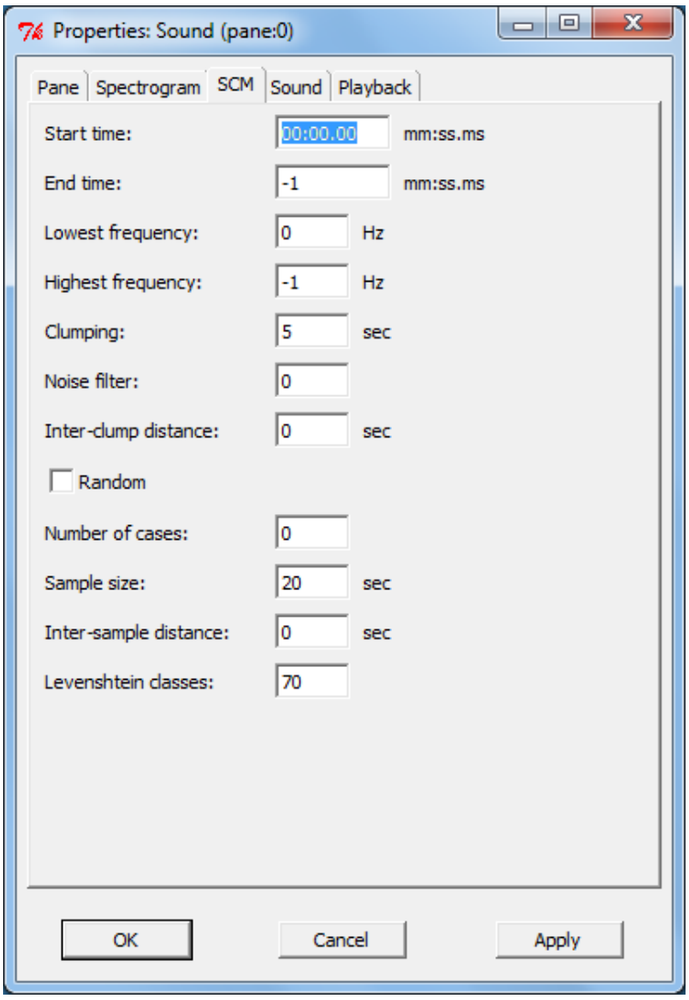


Fig. 9

Lowest Frequency: the lowest frequency (in Hz) considered for the computation.
Highest Frequency: the highest frequency (in Hz) considered for the computation (Fig. 10). The default setting (**Lowest frequency** 0, **Highest frequency** -1) processes the entire range of frequencies according to the sampling rate of the recording equipment (e.g. 22 kHz, 44 kHz, etc.).

Clumping: the temporal resolution (in seconds and fractions) at which the **ACI** algorithms are applied. This setting is needed for each computation. For instance, if you set the **Clumping** at one sec, with an FFT window length of 512 points, 86 pieces of data at a time will be clumped together.

Noise Filter: this filter excludes from the computation all of the data that has an amplitude equal to or less than the value selected here.
The filter cleans diffuse, feeble background noise (distributed in every frequency) from the sound file. The background noise largely depends on the environmental context in which the sound recorders operate. There are no precise rules for setting this filter, so experience may be your best guide. Moreover, the specific sensitivity of different microphones could mean that the **Noise Filter** requires an *ad hoc* setting.
For instance, using our experience, we calibrated the Noise Filter to 5000 for the sound files recorded by the Songmeter (Wildlife Acoustic) equipment, and 3000 for the Handy Recorders H4 (Zoom Corporation) (see Farina et al. 2011).

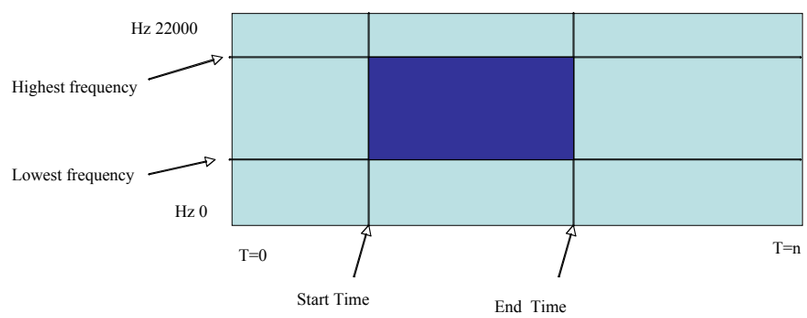


Fig. 10

Inter-clump distance: this is the time interval (in seconds and fractions) between a single ACI calculation and the subsequent one. For instance, for a file of five seconds, if you fix the **Clumping** to 0.5 seconds (Fig. 11) and an **Inter-clump distance** of one second, you will obtain five measurements calculated at 0.5 seconds each. If you fix the **Inter-clump distance** lower than the clumping value, this command would obviously not produce results.

Random: select this option when you need to choose portions of a sound file according to a randomized time. You must also set the **Clumping** and **Number of cases** parameters when using this command.

Number of cases: this is the number of random computations.
For instance, if you have a file of one minute and set **Clumping** to 10 seconds and the **Number of cases** to 2, you will get two sets of data selected randomly during the minute and calculated for a time interval of 10 seconds.

Levenshtein distance: this is the number of categories to which to apply the Levenshtein distance algorithm.

3.1.4 Click OK when all of the parameters are fixed.

3.1.5 Click the right mouse button when the mouse is inside the spectrogram pane and then select <**Compute SCM**> (Fig. 12). The processing time largely depends on the size of the sound file. We recommend closing the previous sound file before processing a new one.

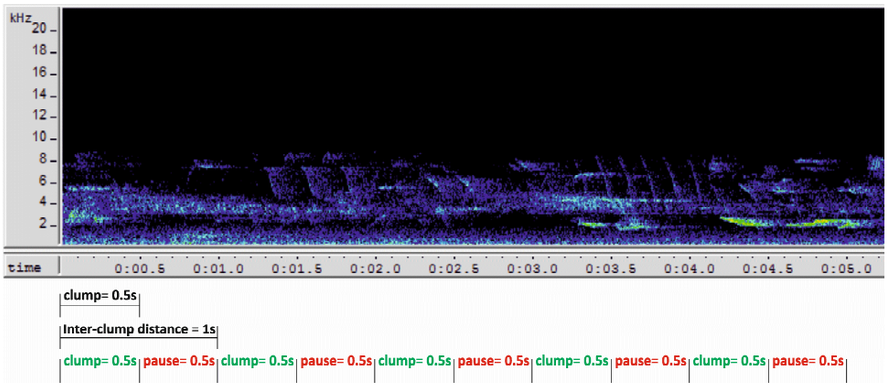


Fig. 11

3.2 <Compute SCM multiple files>
This option allows you to automatically open and process more than one file at a time. This is useful when you have to process several files that require the same settings for the parameters. For instance, SongMeter II (Wildlife Acoustic) or other automatic devices can be programmed to record several files in a day, and instead of running the computation on one a file at a time, it is possible to select a list of a number of files with just one click.

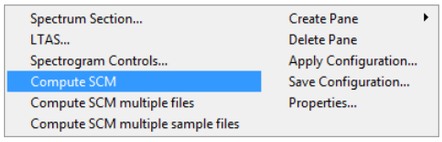


Fig. 12

3.2.1 To run the <Compute SCM multiple files > procedure, it is necessary to follow the steps in **3.1.2** and then to fix the ***Properties*** (see step **3.1.3**) before opening the sound file that is to be processed!

3.2.2 Click the right mouse button when the mouse is inside the spectrogram pane and then click on **<Compute SCM multiple files>** (Fig. 13). Another window within which to search for and select the list of files to be processed will appear. It is necessary to have a list ready in the same folder, since it is not possible to select files that are in different folders.

3.3 <Compute SCM multiple sample files>
This procedure is very useful if you want to fragment long files by creating sub-sampled files. To achieve this, you have to choose the temporal length (sample size) and the interval between each sub-sampled file (inter-sample distance).

3.3.1 Follow the steps in **3.1.2** and then select the **SCM** shortcut menu in the ***Properties*** window. In this sub-menu set the following parameters:

Sample size: this is the temporal dimension (in seconds or fractions) of the sub-sampled files.

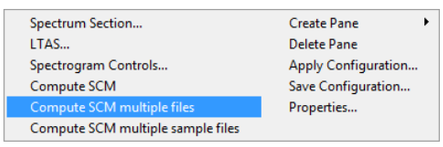


Fig. 13

Inter-sample distance: this is the distance (in seconds or fractions) between two sub-sampled files.
For instance, if you have a file of one minute in length and select a **Sample size** of 2 seconds and the **Inter-sample distance** as 8 seconds, you will be able to extract six sub-sampled files of two seconds each (Fig. 14).

The other parameters in this menu can be also set:

Start time
End time
] Calculated on the original file

Lowest frequency
Highest frequency
Clumping
Noise filter
Inter-clump distance
Random
] Calculated on the sub-sampled file

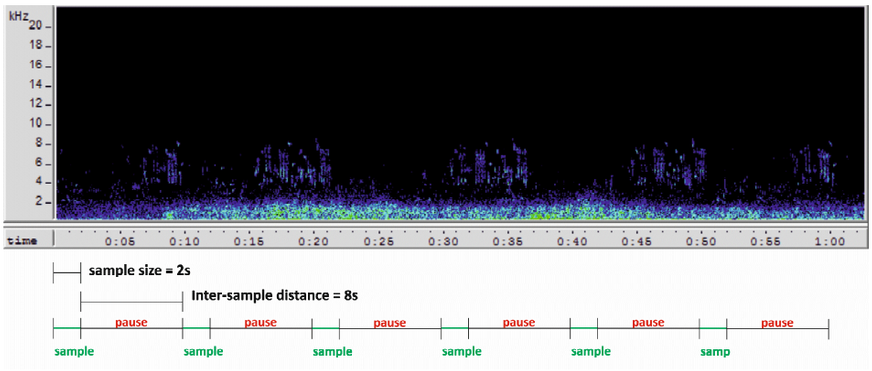


Fig. 14

3.3.2 Click the right mouse button when placed inside the spectrogram pane and the <**Compute SCM multiple sample files**> (Fig. 15). Another window on which to search and select the file to process will appear.

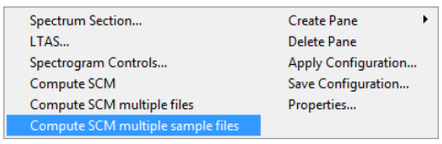


Fig. 15

4. THE OUTPUT

The **SoundscapeMeter** plug-in automatically creates a folder named **results** after the start of each procedure (**Compute SCM**, **Compute SCM multiple files**, **Compute SCM multiple sample files**) (Fig. 16).
This folder is automatically placed in the **SoundscapeMeter** folder that you created during the installation process and which contains the exe.file of **WaveSurfer©** and the **SoundscapeMeter** library, **Scmlib.dll**.
The **results** folder can be deleted at any time, since the program will create a new one for each new computation.
According to the procedure selected, the output will appear as follows:

4.1 When you process one sound file at a time (**Compute SCM** procedure), a folder with the same name as the processed sound file will appear in the results folder. There will be 16 files in the folder, which are automatically given the name of the sound file followed by the name of the metric used. For instance, if the name of the sound file is **test**, a sub-folder called test will appear inside the **results** folder with 16 files inside it that are automatically named as follows (Fig. 17):

1: **test_aci** (an array of ACIt values; the columns list the frequency bins and the rows the different clumpings of the sound files)(Fig. 18).

2: **test_AciEvenness** (a single value of the ACI evenness).

3. **test_aciIf** (an array of ACIf values; the columns list the frequency bins and the rows the different clumpings of the sound files).

4: **test_AciIfTot** (the sum of the ACIf values in each clump).

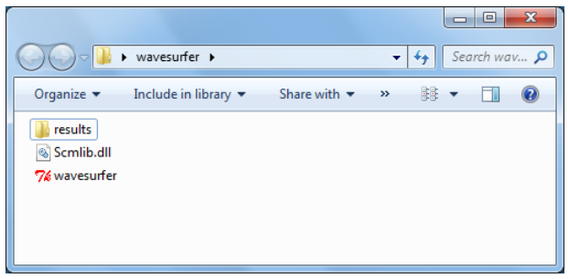


Fig. 16

5: **test_AciTot** (the sum of the ACIt values in each frequency).

6: **test_freq** (the occurrence of frequencies; that is, the number of times a frequency occurs in a specific clump).

7: **test_lev** (a vector created by the Levenshtein distance index and calculated on the basis of the ACIt categories).

8: **test_levDiff** (a vector created by the absolute difference between the adjacent lev data).

9: **test_levRapp** (a value obtained from the ratio between the Levenshtein distance and the levDiff).

10: **test_pow** (the value of intensity according to frequencies (raws) and clumps (columns).

11: **test_shannon** (the value of the Shannon entropy calculated for each clump along the range of the selected frequencies).

12: **test_spect** (the entire numerical matrix as a result of the FFT elaboration and upon which all of the **SoundscapeMeter** computations are based).

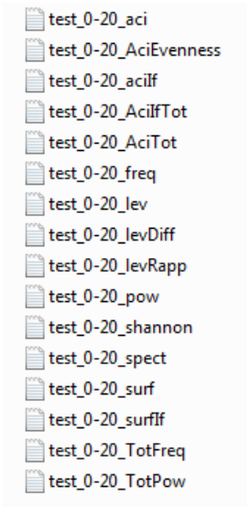
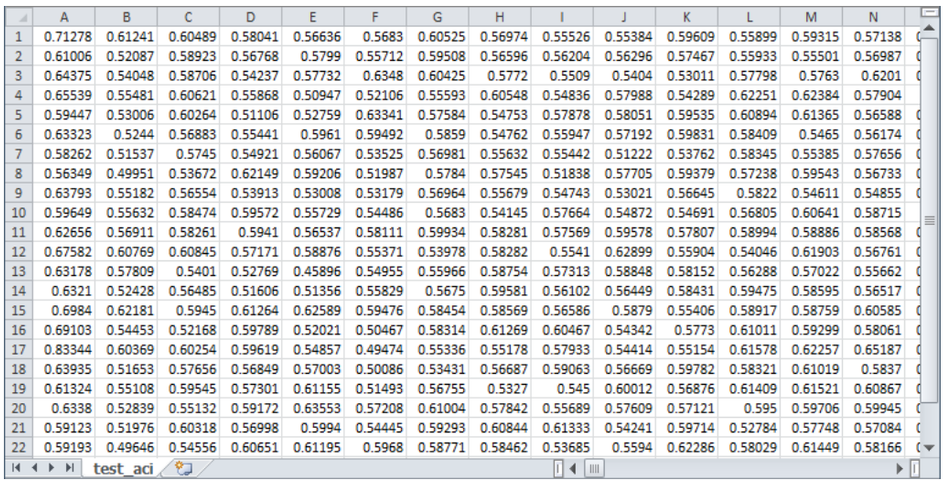


Fig. 17

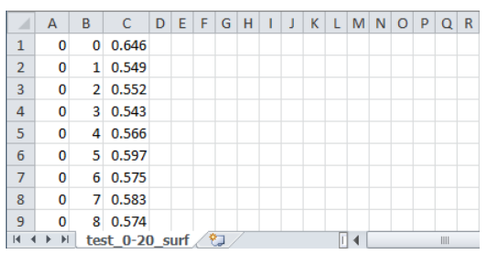


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	0.71278	0.61241	0.60489	0.58041	0.56636	0.5683	0.60525	0.56974	0.55526	0.55384	0.59609	0.55899	0.59315	0.57138
2	0.61006	0.52087	0.58923	0.56768	0.5799	0.55712	0.59508	0.56596	0.56204	0.56296	0.57467	0.55933	0.55501	0.56987
3	0.64375	0.54048	0.58706	0.54237	0.57732	0.6348	0.60425	0.5772	0.5509	0.5404	0.53011	0.57798	0.5763	0.6201
4	0.65539	0.55481	0.60621	0.55868	0.50947	0.52106	0.55593	0.60548	0.54836	0.57988	0.54289	0.62251	0.62384	0.57904
5	0.59447	0.53006	0.60264	0.51106	0.52759	0.63341	0.57584	0.54753	0.57878	0.58051	0.59535	0.60894	0.61365	0.56588
6	0.63323	0.5244	0.56883	0.55441	0.5961	0.59492	0.5859	0.54762	0.55947	0.57192	0.59831	0.58409	0.5465	0.56174
7	0.58262	0.51537	0.5745	0.54921	0.56067	0.53525	0.56981	0.55632	0.55442	0.51222	0.53762	0.58345	0.55385	0.57656
8	0.56349	0.49951	0.53672	0.62149	0.59206	0.51987	0.5784	0.57545	0.51838	0.57705	0.59379	0.57238	0.59543	0.56733
9	0.63793	0.55182	0.56554	0.53913	0.53008	0.53179	0.56964	0.55679	0.54743	0.53021	0.56645	0.5822	0.54611	0.54855
10	0.59649	0.55632	0.58474	0.59572	0.55729	0.54486	0.5683	0.54145	0.57664	0.54872	0.54691	0.56805	0.60641	0.58715
11	0.62656	0.56911	0.58261	0.5941	0.56537	0.58111	0.59934	0.58281	0.57569	0.59578	0.57807	0.58994	0.58886	0.58568
12	0.67582	0.60769	0.60845	0.57171	0.58876	0.55371	0.53978	0.58282	0.5541	0.62899	0.55904	0.54046	0.61903	0.56761
13	0.63178	0.57809	0.5401	0.52769	0.45896	0.54955	0.55966	0.58754	0.57313	0.58848	0.58152	0.56288	0.57022	0.55662
14	0.6321	0.52428	0.56485	0.51606	0.51356	0.55829	0.5675	0.59581	0.56102	0.56449	0.58431	0.59475	0.58595	0.56517
15	0.6984	0.62181	0.5945	0.61264	0.62589	0.59476	0.58454	0.58569	0.56586	0.5879	0.55406	0.58917	0.58759	0.60585
16	0.69103	0.54453	0.52168	0.59789	0.52021	0.50467	0.58314	0.61269	0.60467	0.54342	0.5773	0.61011	0.59299	0.58061
17	0.83344	0.60369	0.60254	0.59619	0.54857	0.49474	0.55336	0.55178	0.57933	0.54414	0.55154	0.61578	0.62257	0.65187
18	0.63935	0.51653	0.57656	0.56849	0.57003	0.50086	0.53431	0.56687	0.59063	0.56669	0.59782	0.58321	0.61019	0.5837
19	0.61324	0.55108	0.59545	0.57301	0.61155	0.51493	0.56755	0.5327	0.545	0.60012	0.56876	0.61409	0.61521	0.60867
20	0.6338	0.52839	0.55132	0.59172	0.63553	0.57208	0.61004	0.57842	0.55689	0.57609	0.57121	0.595	0.59706	0.59945
21	0.59123	0.51976	0.60318	0.56998	0.5994	0.54445	0.59293	0.60844	0.61333	0.54241	0.59714	0.52784	0.57748	0.57084
22	0.59193	0.49646	0.54556	0.60651	0.61195	0.5968	0.58771	0.58462	0.53685	0.5594	0.62286	0.58029	0.61449	0.58166

test_aci

Fig. 18 - The ACIt array where along the x axis are the frequency bins and along the y axis are the different temporal clumps

13: **test_surf** (a matrix, in an x,y,z format, to be exported into Surfer© software for a 3D representation of the ACIt values, where x = clump sequence, y = frequency sequence and z =ACI values' sequence)(Fig. 19).



	A	B	C
1	0	0	0.646
2	0	1	0.549
3	0	2	0.552
4	0	3	0.543
5	0	4	0.566
6	0	5	0.597
7	0	6	0.575
8	0	7	0.583
9	0	8	0.574

test_0-20_surf

Fig. 19

14: **test_surfIf** (a matrix, in an x,y,z format, to be exported into Surfer© software for a 3D representation of the ACIf values, where x = clump sequence, y = frequency sequence and z =ACI values' sequence).

15: **test_TotFreq** (a vector of the total number of occurences in each class of frequencies).

16: **test_TotPow** (a vector of the total value of the intensities in each class of frequencies).

For details about the significance of these metrics, see Section 5.

4.2 When you process several files at a time (**Compute SCM multiple files** procedure), a new folder is created inside the **results** folder for each file. For instance, if we process sound files called Test1, Test2 and Test3, three sub-folders will be created: Test1, Test2 and Test3. Inside each sub-folder you will find 16 files, as listed in 4.1. These files are automatically named according to the name of the related sound files (for instance, *results/Test1/Test1_Aci, Test1_AciEvenness*, ...) (Fig. 20).

In addition, 10 files are created in the **results** folder and summarize the value of the parameter of each sound file in the selected list:

AciIfTotAll (an array of ACIf values, where the columns are the frequency bins and the rows are the different files).
AciTotAll (an array of ACIt values, where the columns are the frequency bins and the rows are the different files).
EvennessAll (a vector in which the evenness of each file is listed).
levAll (an erray in which the Levenshtein distance lev of each file is listed).
levDiffAll (an erray in which the levDiff of each file is listed).
levRappAll (an erray in which the levRapp of each file is listed).
powTotAll (an array of the intensity values in each frequency bin (columns) and file (rows)).
ShannonAll (an array of the Shannon entropy in each frequency bin (columns) and file (rows)).

Summary (a summary table in which the following parameters are reported: PowTotAll, FreqTotAll, AciTotAll, ShannonAll, AciIfTotAll, LevAll, LevDiffAll, LevRappAll, EvennessAll resumed for each file).
TotFreqAll (an erray of the number of occurrences of each frequency bin in the overall matrix).

For details about the significance of these metrics (see Section 5).

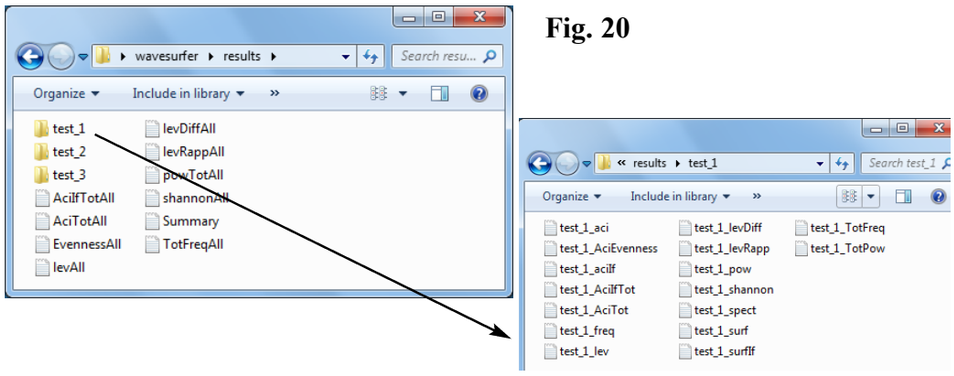


Fig. 20

4.3 When a file is divided into several sub-samples (**Compute SCM multiple sample files** procedure), a new folder for each one will appear in the **results** folder. Each is described by a specific name composed of:
"name of the file_start time-end time--number of sub-sample" (e.g. ***results/Test_0-20***, ...). Inside each folder, you will find 16 files according to the parameters listed in Section 4.1. These are named automatically (e.g. ***results/Test_0-20\Test_0-20_Aci, ... Test_0-20_AciEvenness***, ...) (Fig. 21).

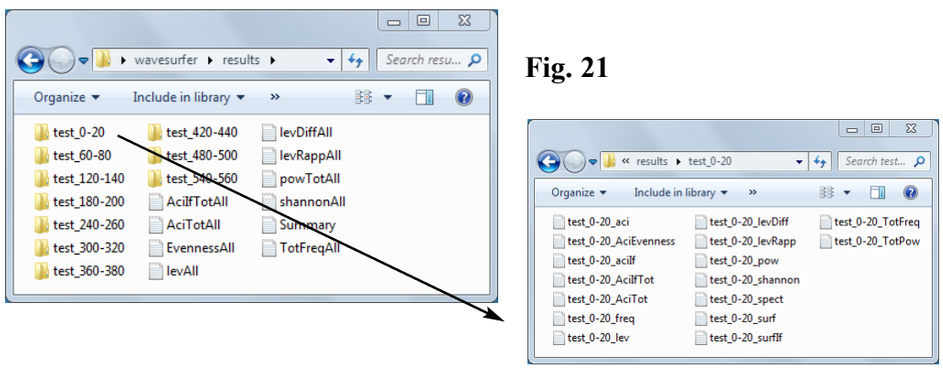


Fig. 21

In addition, in the results folder, 10 files are created that summarize the value of the parameters of the composing files:

AciIfTotAll (an array of ACIf values, where the columns are the frequency bins and the rows the different files).
AciTotAll (an array of ACIt values, where the columns are the frequency bins and the rows the different files).
EvennessAll (a vector in which the evenness of each file is listed).
levAll (an erray in which the Levenshtein distance lev of each file is listed).
levDiffAll (an erray in which the levDiff of each file is listed)
levRappAll (an erray in which the levRapp of each file is listed
powTotAll (an array of the intensity values in each frequency bin (columns) and file (rows)).
ShannonAll (an array of the Shann entropy in each frequency bin (columns) and file (rows)).
Summary (a summary table in which the following parameters are reported: PowTotAll, FreqTotAll, AciTotAll, ShannonAll, AciIfTotAll, LevAll, LevDiffAll, LevRappAll, EvennessAll resumed for each file).
TotFreqAll (an array of the number of occurrences of each frequency bin in the overall matrix).

When you apply this procedure, an additional new folder, called **tmp,** is created in the **SoundscapeMeter** folder adjacent to the **results** folder. The **tmp** contains the sub-sampled sound files to which the computation is applied (Fig. 22).

NB: If sound files with identical names are used, we recommend moving the **results** and **tmp** to a specific folder after every computation to avoid automatic overwriting.

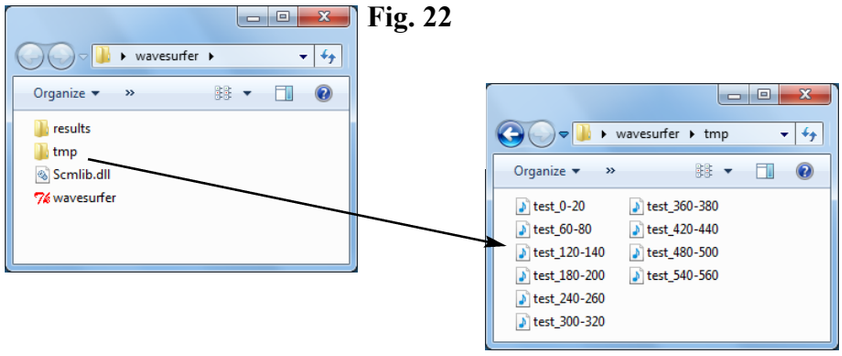

wavesurfer
Organize
results
tmp
Scmlib.dll
wavesurfer
wavesurfer tmp
Organize
test_0-20
test_60-80
test_120-140
test_180-200
test_240-260
test_300-320
test_360-380
test_420-440
test_480-500
test_540-560

Fig. 22

5. THE SoundscapeMeter ALGORITHMS

5.1 Acoustic Complexity Index (ACI_t, ACI_{if}) (**aci** and **aciIf** in the output)

Recently the Acoustic Complexity Index has been splitted in two distinct Indeces: **ACIt** that calculates the complexity occurring along a single clump inside a frequency bin, and $\mathbf{ACI_{if}}$ that calculates the complexity occurring inside a single clump between adjacent frequency bins

The **ACIt**, which was extended by Farina and Morri (2008) and Pieretti et al. (2011), processes the intensities registered in audio-files and produces a direct and rapid quantification of the soundscape.

The hypothesis upon which the ACI formula is based is the observation that, at the right temporal scale, many biophonies, such as bird song, are characterized by an intrinsic variability of intensities, while some types of geophonies (wind, flowing water, rain) and anthrophonies (such as a car passing or airplane transit) have very constant intensity values.

The **ACIt** calculates the absolute difference of intensity (I_k and $I_{(k+1)}$) between two adjacent temporal intervals (Δt_k) in a single frequency bin (Δf_j) according the following algorithm:

$$ACI_t = \frac{\sum_{k=1}^{n-1} |I_k - I_{k+1}|}{\sum_{k=1}^{n} I_k}$$

where n = number of I_k value in the selected clumping interval. The Δt_k and Δf_j depend on the FFT size (See for more details Pieretti et al. 2011). In the application of the **SoundscapeMeter** plug-in the n value is setted by the **Clumping** command.

The **ACI_{if}** (along each temporal step) calculates the difference in intensity between a frequency bin and the successive bin along the same clumping interval adopted for calculating the **ACIt**.
The algorithm of this index has the same structure of the **ACIt** but with a significant difference that the denominator is composed by all measures of intensity used in the comparison:

$$ACI_f = \sum_{j=1}^{m-1} \frac{\sum_{k=1}^{n} \left| I_{kj} - I_{k(j+1)} \right|}{\sum_{k=1}^{n} I_{kj} + \sum_{k=1}^{n} I_{k(j+1)}}$$

where m= number of Δf_j.

5.2 ACI_t Evenness (**AciEvenness** in the output)

This index measures the distribution of the ACI_t along each frequency class

$$ACI_t Evenness = \frac{1}{\sum p_i^2}$$

where p_i is the relative abundance of the **ACI_t** index along the different **i** frequency classes.

5.3 Acoustic Diversity Index (ADI) (**Shannon** in the output)

This measure of spectral entropy was first used by Sueur et al. (2008) [index Hf] and measures the complexity of intensity values along the different frequency bins using the Shannon diversity algorithm

$$ADI = -\sum pi \ln pi$$

where pi is the relative importance of intensity values along a frequency bin.

5.4 The Levenshtein distance (LD) (**lev** in the output)

The Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into another, with the allowable edit operations being: insertion, deletion, or the substitution of a single character (See Annex 1).

This index has been applied to test the level of similarity of the **ACIt** between two adjacent clumps. To calculate this distance, the **ACIt** values are transformed into n categories. The utilization of 70 categories is recommended, but you can increase or reduce this number according to your specific needs.

5.5 Levenshtein difference (LDF) (**levDiff** in the output)

This (intermediate) index is the result of the absolute difference between the **LD** at time t and **LD** at time t+1 and it measures the variability along a period of time. **LDF** is applied in the **Levenshtein ratio (LR)** algorithm.

5.6 The Levenshtein ratio (LR) (**levRapp** in the output)

This index is the result of the ratio between

$$LR = \frac{\sum LDF}{\sum (LD_t + LD_{t+1})}$$

and measures the total amount of variability of the $\mathbf{ACI_t}$ **LD** of a sound file.

REFERENCES

Farina A., Morri D. 2008. Source-sink e eco-field: ipotesi ed evidenze sperimentali. Atti X congresso nazionale della SIE-IALE. Ecologia e Governance del paesaggio: esperienze e prospettive. Bari, 365-372.

Pieretti N., Farina A., Morri D. 2010. A new methodology to infer the singing activity of an avian community: The Acoustic Complexity Index (ACI). *Ecological Indicators* doi:10.1016/j:ecolind.2010.11.005.

Farina A., Lattanzi E., Malavasi E. R., Pieretti N., Piccioli L. 2011. Avian soundscapes and cognitive landscapes: theory, application and ecological perspectives. *Landscape Ecology* 26: 1257-1267.

Sjölander K, Beskow J. 2000. WaveSurfer - an open source speech tool. Proceedings of the ICSLP 2000, IV:464-467.

Sueur J., Pavoine S., Hamerlynck O., Duvail S. 2008. Rapid acoustic survey for biodiversity appraisal. *PloS ONE,* 3(12): e40

ANNEXES

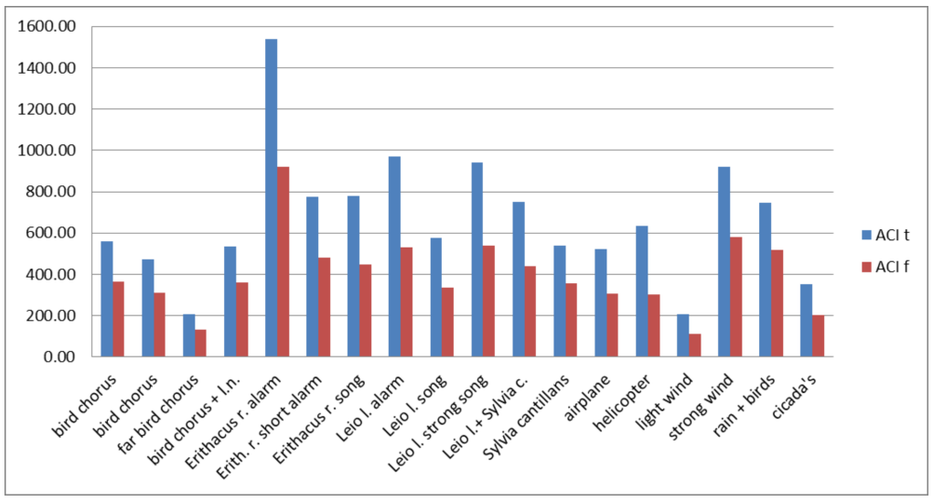
ANNEX 1 - Routine to calculate the Levenshtein distance

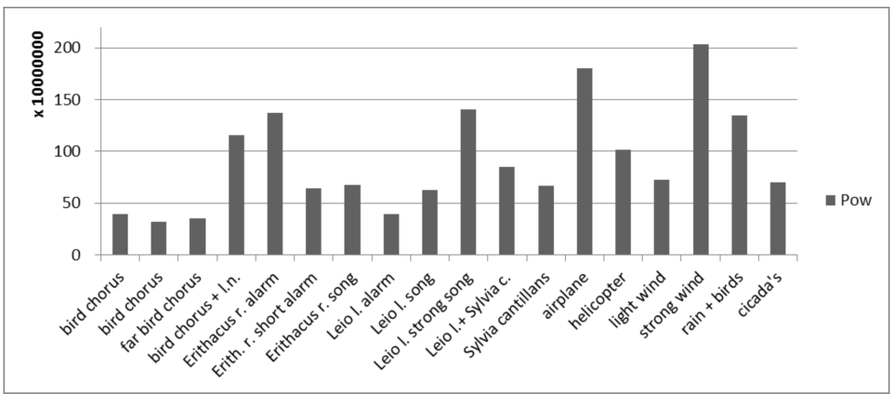
```
int LevenshteinDistance(char s[1..m], char t[1..n])
{
  // for all i and j, d[i,j] will hold the Levenshtein distance between
  // the first i characters of s and the first j characters of t;
  // note that d has (m+1)x(n+1) values
  declare int d[0..m, 0..n]

  for i from 0 to m
    d[i, 0] := i // the distance of any first string to an empty second string
  for j from 0 to n
    d[0, j] := j // the distance of any second string to an empty first string

  for j from 1 to n
  {
    for i from 1 to m
    {
      if s[i] = t[j] then
        d[i, j] := d[i-1, j-1]       // no operation required
      else
        d[i, j] := minimum
                   (
                     d[i-1, j] + 1,  // a deletion
                     d[i, j-1] + 1,  // an insertion
                     d[i-1, j-1] + 1 // a substitution
                   )
    }
  }

  return d[m,n]
}
```

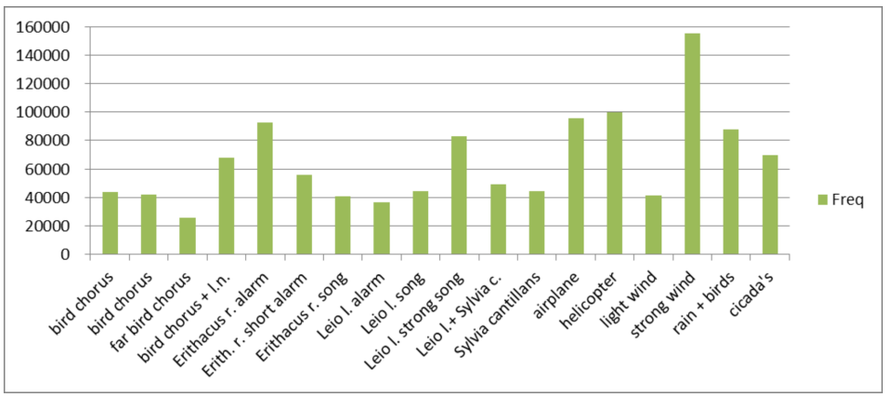
ANNEX 2

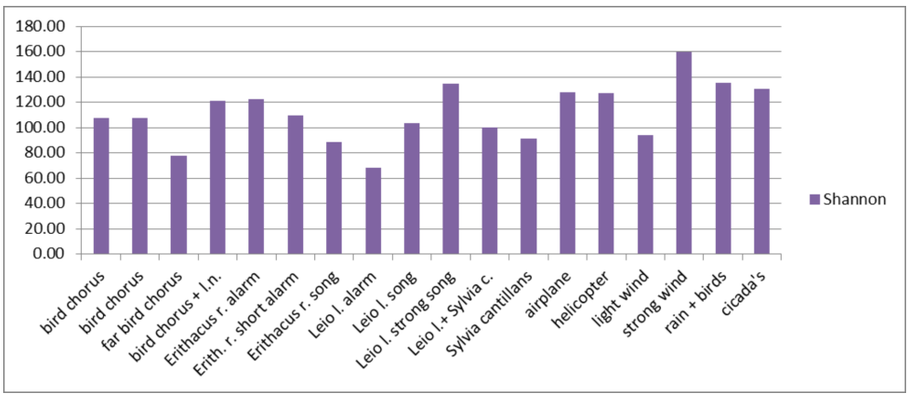
Example of metrics calculated on selected files of 1 minute set at an FFT of 512, noise filter at 3000 and clumping at one second.

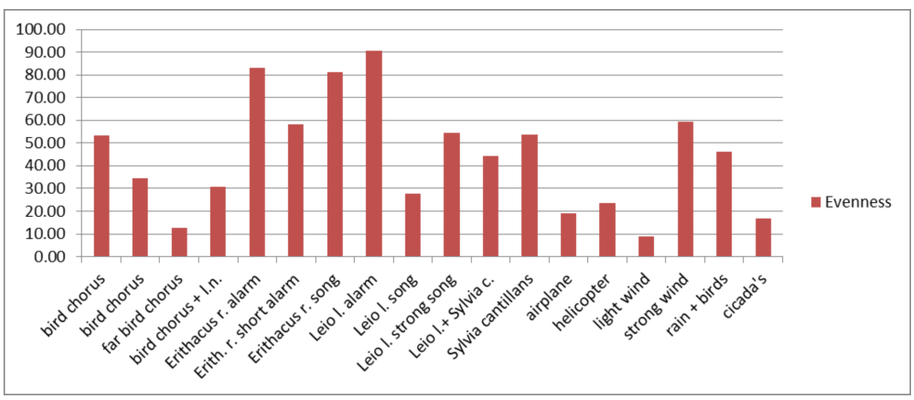
File name	Description	Label	ACI_t	ACI_f	Pow	Freq	Shannon	Evenness	Lev	LevDiff	LevRapp
20110510_050000	bird chorus	bird chorus	558.38	364.38	396374504	43820	107.28	53.15	2549	778	0.31
20110522_043300	bird chorus	bird chorus	471.30	311.04	318184419	41647	107.84	34.51	2435	252	0.10
20110529_123700	far bird chorus	far bird chorus	208.17	132.39	350364652	25565	77.62	12.54	1111	265	0.24
20110515_073400	bird chorus + light noise	bird chorus + l.n.	535.49	361.84	1160671578	68066	120.84	30.67	2318	465	0.20
20110112_073900	*Erithacus rubecula* alarm	*Erithacus r.* alarm	1538.37	920.08	1375634537	92723	122.30	83.25	5648	905	0.16
20110112_075000	*Erithacus rubecula* short alarm + *Leiothrix lutea*	*Erith. r.* short alarm	773.62	480.26	638882327	55545	109.53	58.40	3170	515	0.16
20110503_073400	*Erithacus rubecula* song	*Erithacus r.* song	780.78	448.02	676249995	40671	88.35	81.31	2904	1549	0.53
20110116_151000	*Leiothrix lutea* alarm	*Leio l.* alarm	971.29	532.25	394707522	36574	67.94	90.79	3953	1377	0.35
20110505_143200	*Leiothrix lutea* song	*Leio l.* song	576.75	335.27	629929563	44491	103.61	27.68	1611	188	0.12
20110723_072900	*Leiothrix lutea* strong song	*Leio l.* strong song	940.65	538.10	1405576670	83014	134.92	54.36	3173	1413	0.45
20110530_090800	*Leiothrix lutea* + *Sylvia cantillans*	*Leio l.* + *Sylvia c.*	748.64	438.92	847063862	49399	99.75	44.16	2777	493	0.18
20110509_082900	*Sylvia cantillans*	*Sylvia cantillans*	539.97	356.77	664605944	44075	91.41	53.66	2409	479	0.20
20110127_145900	airplane	airplane	520.25	306.98	1802387727	95743	127.55	19.03	1923	252	0.13
20110114_151000	helicopter	helicopter	634.85	301.80	1014315090	99976	126.96	23.75	1544	112	0.07
20110109_115200	light wind	light wind	208.80	112.39	724704600	41336	93.95	8.76	705	91	0.13
20110112_060000	strong wind	strong wind	918.69	579.76	2032994385	155220	159.51	59.30	4372	728	0.17
20110509_051100	rain + birds	rain + birds	746.46	518.01	1349927103	87739	135.47	46.31	3061	744	0.24
20110809_155500	cicada's song	cicada's	351.18	202.51	699660443	69468	130.33	16.93	1461	113	0.08

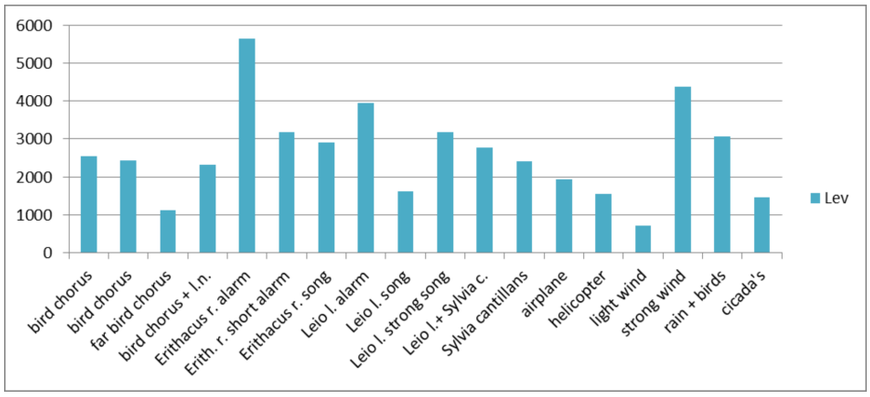

1600.00
1400.00
1200.00
1000.00
800.00
600.00
400.00
200.00
0.00
ACI t
ACI f
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's

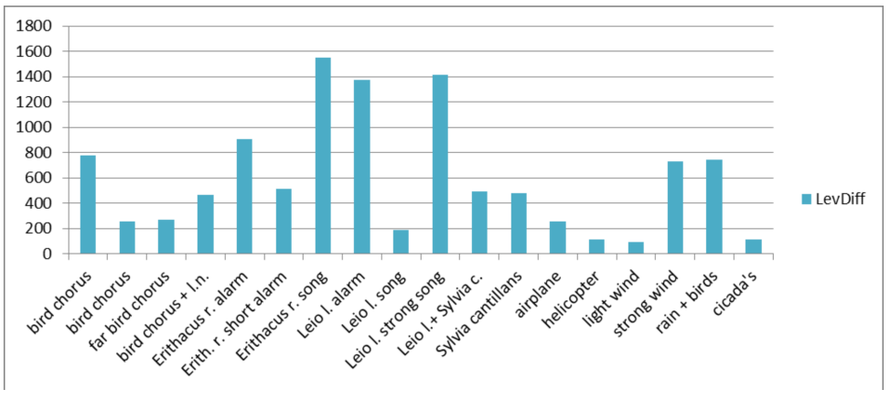

x 10000000
200
150
100
50
0
Pow
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's

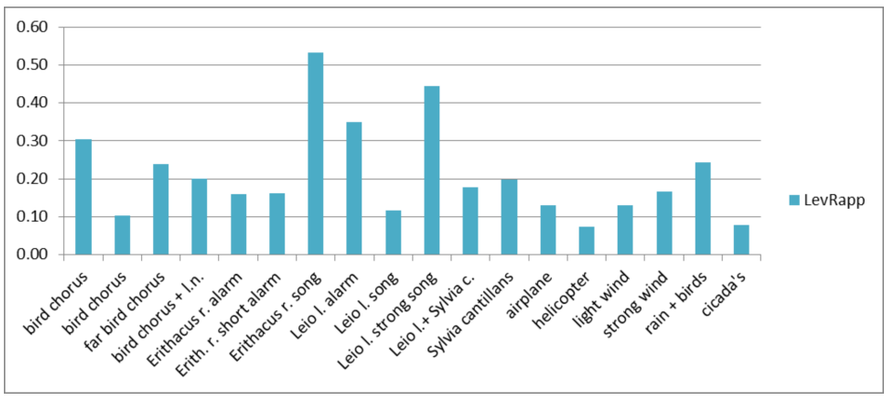

160000
140000
120000
100000
80000
60000
40000
20000
0
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
Freq


180.00
160.00
140.00
120.00
100.00
80.00
60.00
40.00
20.00
0.00
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
Shannon


100.00
90.00
80.00
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
Evenness


6000
5000
4000
3000
2000
1000
0
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
Lev


1800
1600
1400
1200
1000
800
600
400
200
0
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
LevDiff


0.60
0.50
0.40
0.30
0.20
0.10
0.00
bird chorus
bird chorus
far bird chorus
bird chorus + l.n.
Erithacus r. alarm
Erith. r. short alarm
Erithacus r. song
Leio l. alarm
Leio l. song
Leio l. strong song
Leio l.+ Sylvia c.
Sylvia cantillans
airplane
helicopter
light wind
strong wind
rain + birds
cicada's
LevRapp