The National Center for Voice and Speech Online Technical Memo No. 16, March 2013, version 1.0

EGG Wavegram Python Source Code Tutorial

Ben Larson[1] Christian T. Herbst[2] and Eric J. Hunter[1]

[1] National Center for Voice and Speech, The University of Utah, Salt Lake City, Utah, USA

[2] Laboratory of Bio-Acoustics, Dept. of Cognitive Biology, University Vienna, Austria

This memo serves as a tutorial to provide ready-to-run software for creating EGG Wavegrams based on the work of Christian Herbst. The contents of the memo provide: (1) Installation instructions for Python and the necessary libraries required for wavegram generation; (2) Instructions for the process of using the code to analyze wav files (the Python source code can be found at Christian Herbst's website: http://www.christian-herbst.org/index.php?page=wavegram); (3) An additional option to run the software without needing to install Python and necessary libraries. The instructions in this tutorial utilize Python v 2.6 and the tutorial is intended for Windows users, other OS users may use this tutorial as a guide then find their own respective downloads, or use emulation software like VMware or ParallelsDesktop. Links to needed modules and updates to this memo can be downloaded at http://www.ncvs.org/ncvs/library/tech .

Keywords: wavegram, tutorial, Python, Windows, EGG, Electroglottography, voice

1. Introduction

Electroglottography (or EGG) is a low-cost technology for measurement of the time-varying degree of contact between contacting vocal folds during phonation [1, 2]. An EGG signal is based on the electrical impedance in the larynx, which changes as the vocal folds consecutively come into contact and loose the contact. During use, two small electrodes are placed on the two sides of the thyroid lamina (of which the Adam's Apple is the most prominent feature). A small, high frequency current is passed between the two electrodes.

The EGG signal is commonly used to track the fundamental frequency of phonation. In this capacity, it is used often with stroboscopic equipment during laryngeal exams. Additionally, the EGG signal can be used in biofeedback for vocal pitch matching, stuttering, and vibrato tracking. EGG can also be used to estimate the open and close quotients during a vocal vibration cycle, a useful quantity when analyzing speech and singing [3].

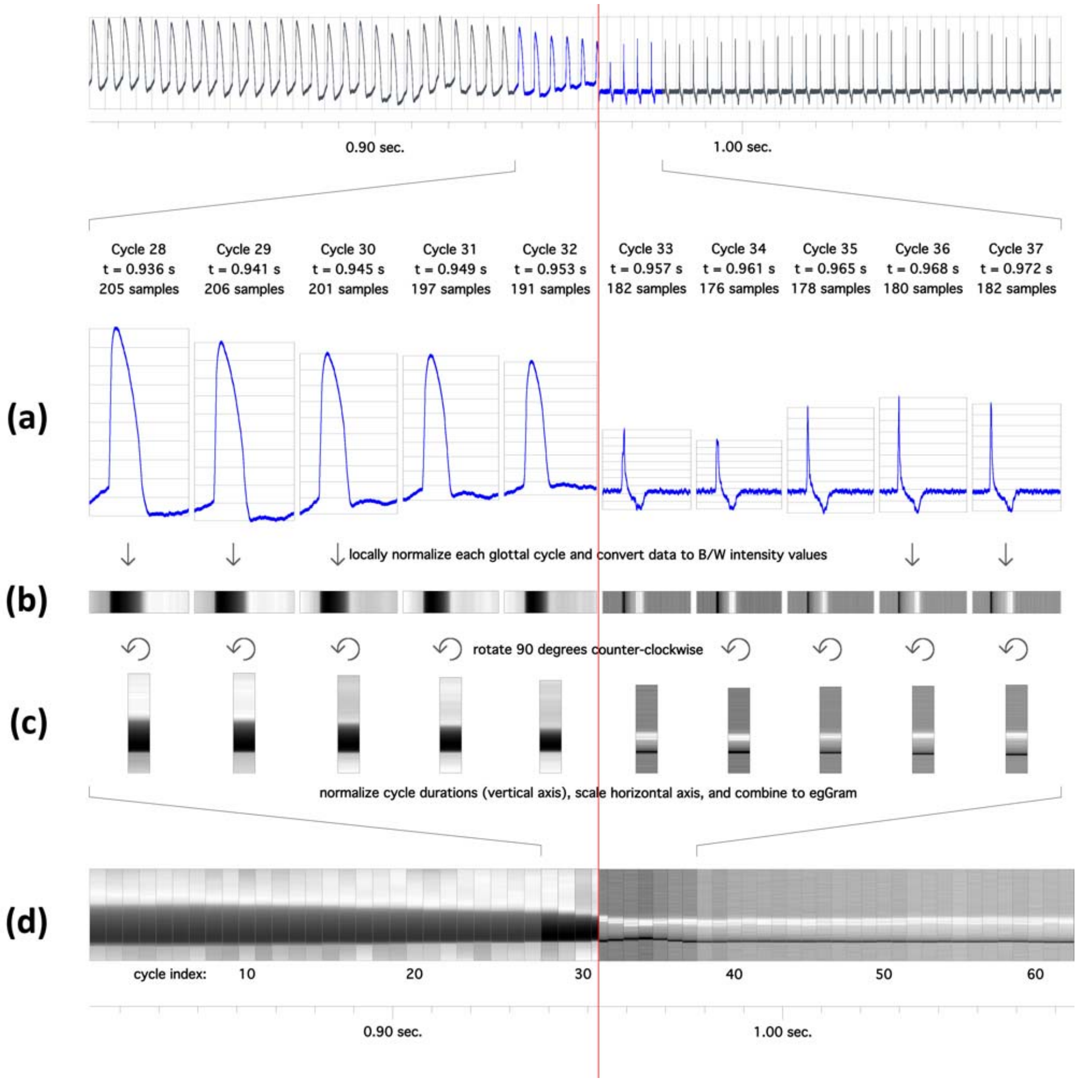


Figure 1: Basic processing steps to create an electroglottographic wavegram (see text). The two halves of the image illustrate the creation of wavegrams based on the EGG (left) and the dEGG (right) signal, respectively – taken from [4].

The wavegram is a visual representation of the EGG signal which provides an intuitive means for quickly assessing vocal fold contact phenomena and their variation over time (Figure6, 2nd row). Of particular interest are the locations of large change (i.e. the first mathematical derivative of the EGG signal: dEGG), indicating vocal fold contacting and de-contacting. In a wavegram, the EGG signal is decomposed into consecutive individual cycles, each of which is normalized in both duration and amplitude, and is displayed on the y-axis, going from bottom to top. Overall time is shown on the x-axis. In a dEGG wavegram, the first derivative of the EGG signal is used as the input signal. In such a display, the contacting and de-contacting phases for each glottal cycle are approximated by (a) one or more dark horizontal line(s) at the lower end of the graph (contacting phase), and (b) one or more light horizontal line(s) in the upper section of the graph (de-contacting phase) – see Figure 6, 2nd row, for an example. Much like in the sound spectrogram, information on vibratory behavior developing in time is compacted into one single graph, providing insight into changes of vocal fold dynamics. Vocal fold contacting and de-contacting appears here as a sequence of events rather than single incidents, taking place over a certain period of time, and changing with pitch, loudness and vocal fold oscillatory regime / vocal register [5]. The process of wavegram generation is illustrated in Figure 1.

This Technical Memo describes how to implement the source code used to create the wavegram. This component is for those who may want to use the source code in their own analysis or to modify it in some way (e.g., don't forget the copyright information below). For those who do not want to modify the source code but do want to create wavegrams from pre-recorded EGG files, the final section describes a ready to run version that doesn't require programing.

2. Installation of Python and necessary libraries

The EGG wavegram source code was written in Python. A Python interpreter and several different Python libraries are needed to run the wavegram code correctly. Links for reference to the library website and the immediate download of the required files are provided. The following download links will save the installer files to:

```
C:\Documents and Settings\_____\My Documents\Downloads
```

where you can open and install them. The links are intended for 32 bit windows systems and Python 2.6 as that is what the audiolab library uses. Make sure to install in the order indicated in this tutorial.

1. Download the Python compiler (version 2.6). Open and run the file once downloaded and follow the installation instructions.
   Download link: http://www.python.org/ftp/python/2.6/python-2.6.msi
   Website link: http://www.python.org/download/releases/2.6/
2. Download and install the numpy package version 1.6.2 for Python 2.6
   Download link: http://pypi.python.org/packages/2.6/n/numpy/numpy-1.6.2.win32-py2.6.exe#md5=941e4b1b65923addf8a7dc21ec7dbb7e
   Website link: http://pypi.python.org/pypi/numpy [1]
3. Download and install the Matplotlib package version 1.2.0 for Python 2.6
   Download link: https://github.com/downloads/matplotlib/matplotlib/matplotlib-1.2.0.win32-py2.6.exe
   Website link: https://github.com/matplotlib/matplotlib/downloads/

[1] For academic uses, the numpy package can alternatively be downloaded free of charge from http://www.enthought.com/products/open-scipy.php

4. Download and install scikits audiolab version 0.11.0 for Python 2.6
   Download link: http://pypi.python.org/packages/2.6/s/scikits.audiolab/scikits.audiolab-0.11.0.win32-py2.6.exe#md5=fb7820f27e48363641bbc13fc96bdd11
   Website link: http://pypi.python.org/pypi/scikits.audiolab/

You should now have all the software/packages to the wavegram source code. Note that as Python version updates, you may need to try new versions. The above have been tested on Windows XP SP2, Windows 7 (32bit), and Windows 7 (64bit). The original wavegram code has been developed on Mac OS X 10.5. The above steps should also work for Mac OS.

3. Process of using code to analyze wav files

We now need the code that computes the wavegram from a WAV file. The “module” and “test” source codes need to be copied and created to a file that Python can access.

1. Using either notepad or IDLE (start > programs > Python2.6 > IDLE) we can create the needed file. Start 2 new blank documents with either program (with IDLE you will need to start 2 new separate windows). Then:
   a. Copy all of the “module” code from Christian Herbst’s website. Paste it into your new blank document. Save the file as `wavegram.py` in this folder:
      `C:\Python26\Lib\site-packages\scikits`
      It is important that you type the “.py” extension after the filename so Python can use it.
   b. Copy all the “test” code from Christian Herbst’s website into the other blank document. Save the file as `test.py` in the same folder:
      `C:\Python26\Lib\site-packages\scikits`
      Remember it is important that you type the “.py” extension after the filename so Python can use it.
   c. Go to the `scikits` folder: `C:\Python26\Lib\site-packages\scikits`
      Make sure the `test.py` and `wavegram.py` files you saved earlier are here. This directory is also where you will need to copy and paste the WAV files you want to analyze. Your `scikits` folder should look similar to Figure 2.

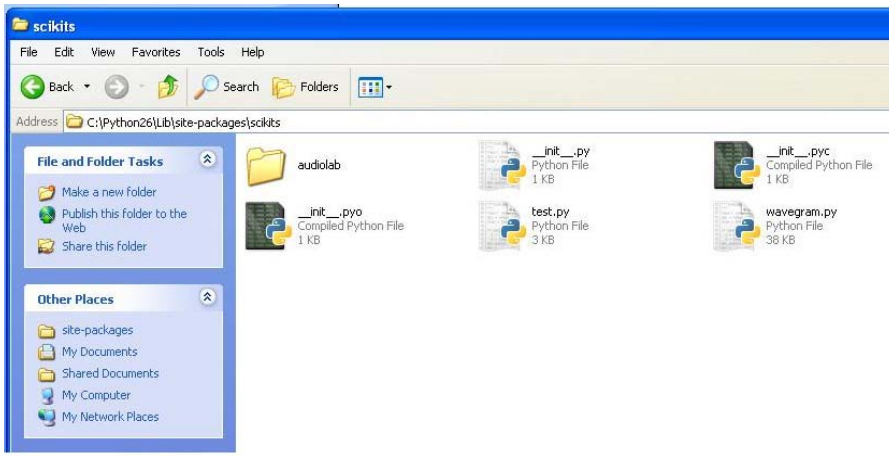
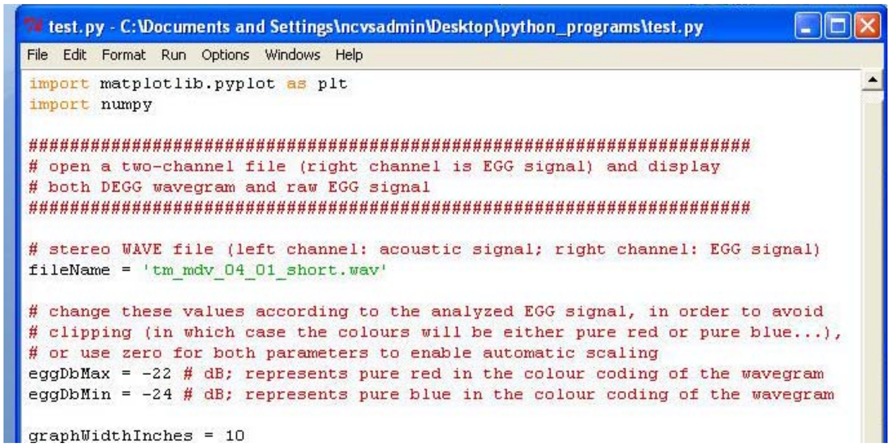

scikits
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Python26\Lib\site-packages\scikits
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder
Other Places
site-packages
My Documents
Shared Documents
My Computer
My Network Places
audiolab
__init__.py Python File 1 KB
__init__.pyc Compiled Python File 1 KB
__init__.pyo Compiled Python File 1 KB
test.py Python File 3 KB
wavegram.py Python File 38 KB

Figure 2. The `scikits` folder

2. Copy and paste any `wav` files that you wish to analyze into this folder.
3. Right click the `test.py` file and “edit with IDLE”. Two windows will open. We want to apply changes to the `test.py` file (close the Python Shell window or minimize it). You will see something like Figure 3.

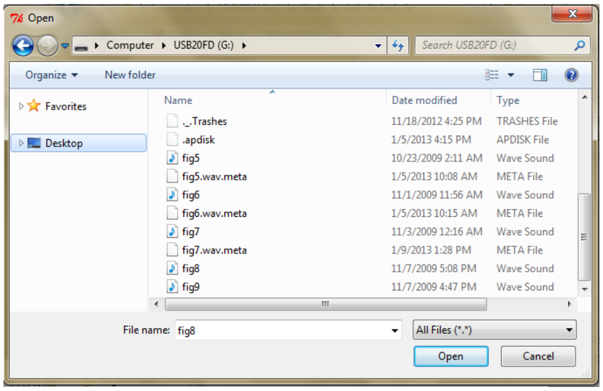

test.py - C:\Documents and Settings\ncvsadmin\Desktop\python_programs\test.py
File Edit Format Run Options Windows Help

```
import matplotlib.pyplot as plt
import numpy

######################################################################
# open a two-channel file (right channel is EGG signal) and display
# both DEGG wavegram and raw EGG signal
######################################################################

# stereo WAVE file (left channel: acoustic signal; right channel: EGG signal)
fileName = 'tm_mdv_04_01_short.wav'

# change these values according to the analyzed EGG signal, in order to avoid
# clipping (in which case the colours will be either pure red or pure blue...),
# or use zero for both parameters to enable automatic scaling
eggDbMax = -22 # dB; represents pure red in the colour coding of the wavegram
eggDbMin = -24 # dB; represents pure blue in the colour coding of the wavegram

graphWidthInches = 10
```

Figure 3. The Python Shell window folder.

4. Locate `fileName = 'tm_mdv_04_01_short.wav'` (line 11) and change the text in single quotes to the `wav` file you want to analyze. Remember to add the .`wav` extension (i.e. `fileName = 'myWavFile.wav'`).
5. When finished make sure to save. Now you can either press F5 to run this file, click Run>Run Module from the file menu, or exit out and just double click on the `test.py` file.
   Note: each time you want to analyze a different wav file you will need to change the file name here and save before running `test.py`.

6. Wait. The wavegram can take up to 1 minute (or more) to generate (depending on duration/sampling frequency of the analyzed file and the processing power of your computer). Save the wavegram or just exit all the figure windows and Python Shell window to end the program.



Name	Date modified	Type
._.Trashes	11/18/2012 4:25 PM	TRASHES File
.apdisk	1/5/2013 4:15 PM	APDISK File
fig5	10/23/2009 2:11 AM	Wave Sound
fig5.wav.meta	1/5/2013 10:08 AM	META File
fig6	11/1/2009 11:56 AM	Wave Sound
fig6.wav.meta	1/5/2013 10:15 AM	META File
fig7	11/3/2009 12:16 AM	Wave Sound
fig7.wav.meta	1/9/2013 1:28 PM	META File
fig8	11/7/2009 5:08 PM	Wave Sound
fig9	11/7/2009 4:47 PM	Wave Sound

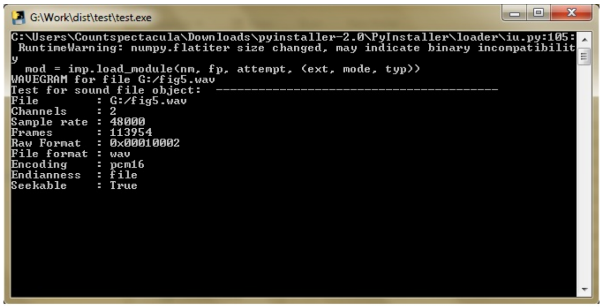
File name: fig8

Figure 4. The file selection dialog window.

4. Using software without installing Python

This option allows you to run the Python script and create a wavegram without any installation of the Python compiler or Python libraries.

1. Download then unzip the file `Wavegram.zip` which accompanies this website. You can unzip the files to any location on your computer.



```
C:\Users\Countspectacula\Downloads\pyinstaller-2.0\PyInstaller\loader\iu.py:105:
 RuntimeWarning: numpy.flatiter size changed, may indicate binary incompatibilit
y
  mod = imp.load_module(nm, fp, attempt, (ext, mode, typ))
WAVEGRAM for file G:/fig5.wav
Test for sound file object:  ----------------------------------------
File        : G:/fig5.wav
Channels    : 2
Sample rate : 48000
Frames      : 113954
Raw Format  : 0x00010002
File format : wav
Encoding    : pcm16
Endianness  : file
Seekable    : True
```

Figure 5. The Python Shell window folder

2. Inside the folder is a shortcut called `Wavegram`. Also are two sample EGG files, a `meta` file and a folder called `scripts`. The `scripts` folder contains the executable `Wavegram.exe` and accompanying libraries. These do not need to be modified. Executing the shortcut will start the application.

   You will be prompted to choose a wave file from your computer (Figure 4). Select a wave file and click “open”. A command prompt will appear as the wavegram is generated. The messages it displays will be unimportant. Possibly something similar to Figure 5.

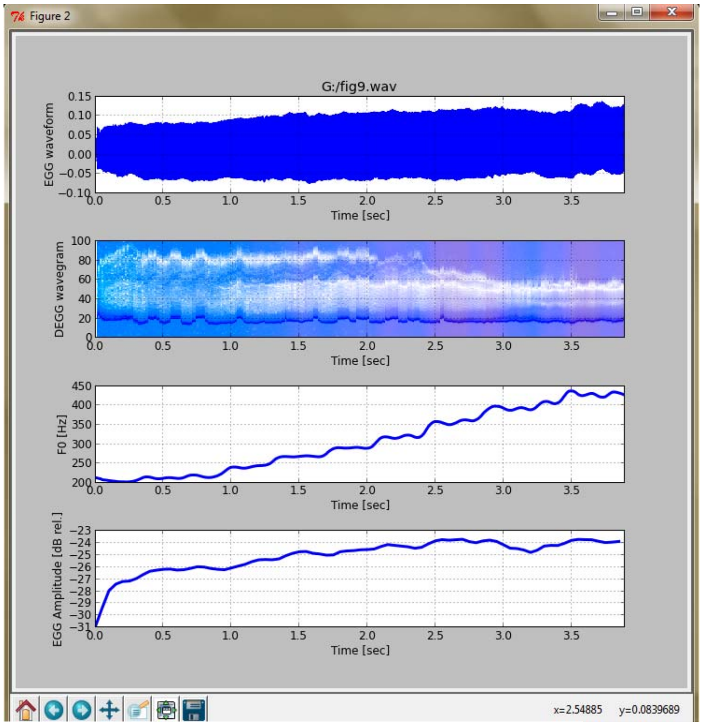


Figure 6. Example Wavegram from fig9.wav.

After a few moments (sometimes 30+ seconds) the wavegram (Figure 6) will finish generating and opens an interactive window that allows the same operations as if you had installed Python as in option 1-2 of this memo. The wavegram in Figure 5 is from the sample file, Fig5.wav[2] (which accompanies the zip file).

Every time a wav file is analysed, a `meta` file is created. If you keep the `meta` file, the analysis is faster the next time you analyse the corresponding wave file. An example of this file is in the example folder with this document.

5. Literature

[1] R. J. Baken, "Electroglottography," *Journal of Voice,* vol. 6, pp. 98-110, 1992.

[2] R. J. Baken and R. F. Orlikoff, *Clinical Measurement of Speech and Voice (2nd Edition)* vol. 2nd Edition: Singular Publishing, Thompson Learning, 2000.

[3] C. Herbst and S. Ternström, "A Comparison of Different Methods to Measure the EGG Contact Quotient," *Log. Phon. Vocol.,* vol. 31, pp. 126-138, 2006.

[4] C. T. Herbst, "Investigation of glottal configurations in singing," Ph.D., Department of Biiophysics, Palacky University, Olomouc, Czech Republic, 2011.

[5] C. T. Herbst*, et al.*, "Electroglottographic wavegrams: a technique for visualizing vocal fold dynamics noninvasively," *J Acoust Soc Am,* vol. 128, pp. 3070-8, Nov 2010.

Copyright

Revisions

1.0 Ben Larson: Main document with edits from Eric Hunter and Christian Herbst (March. 2013)

[2] The .wav files which accompany the wavegram source code are the examples used in [5]. These names of these files are derived from the figure numbers from the original publication, showing the wavegrams of the respective `.wav` files.