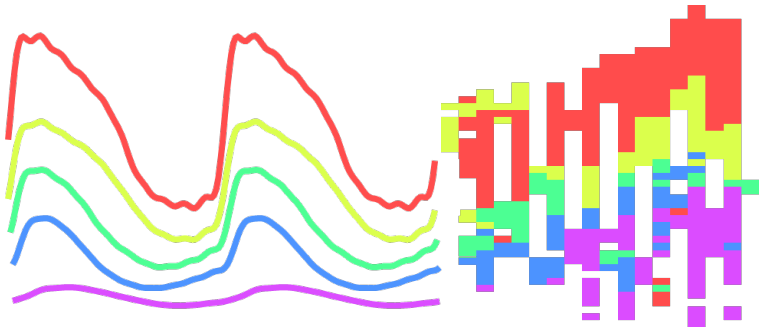


The FonaDyn Handbook

Version 2.0

Sten Ternström

with Dennis Johansson and Andreas Selamtzis

Division of Speech, Music and Hearing
School of Electrical Engineering and Computer Science
KTH Royal Institute of Technology
SE-100 44 Stockholm, Sweden

Disclaimer: FonaDyn and its supporting materials are provided to the public domain, free of charge and with no commitment to further support. Much effort has been invested in the development, and the code has no risky aspects that we are aware of. Still, all software is likely to have errors of one kind or another. We are of course interested in learning of any bugs or other problems that you might come across.

You may install, run or modify FonaDyn only if you agree that you will not hold the authors S.T., D.J. and A.S. responsible for any loss of data or function that may be incurred as a consequence of installing or using FonaDyn.

Licence: A link to the European Union Public Licence v1.2, can be found below. Any and all use of FonaDyn must comply with this licence.

EUPL v1.2: https://eupl.eu

Mandatory citation: *publications* of any research that you perform using FonaDyn or any derivative thereof must cite the following article. Likewise, *distribution* of any software that you develop using FonaDyn or any derivative thereof must observe the licensing conditions, and also cite the following article:

Ternström S, Johansson D, Selamtzis A (2018). FonaDyn - a system for real-time analysis of the electroglottogram, over the voice range. *SoftwareX number 7*, 2018, pp 74-80. DOI: 10.1016/j.softx.2018.03.002

The above is an Open Access article that can be accessed at this link:
http://linkinghub.elsevier.com/retrieve/pii/S235271101830030X ,
where also updates to FonaDyn can be downloaded when available.

FonaDyn version: 2.0.2

Contents

0 Overview

The FonaDyn program is a research tool for investigating the waveform of the electroglottographic signal (EGG), over the entire voice range. It gives real-time visual feedback of EGG metrics and phonation types, plotted as voice maps, in the paradigm of the voice range profile (VRP) [1]. 'Phonation types' may refer to modal/falsetto, or to any other types of phonatory differences, including gradings within a single voice register. FonaDyn can be used to pursue many kinds of research questions on phonation, voice source dynamics, source-filter interaction and effects of training and/or therapy. The only constraint is that the phonation must be reasonably periodic throughout. Of course, the researcher/educator/clinician needs to understand in some detail how FonaDyn works, and how to use it. That is the main goal of this handbook.

Time-domain and harmonic-domain parameters of the EGG waveform are computed for every EGG cycle. Different EGG waveshapes are automatically categorised and assigned to layers of different colours in the voice map. To the harmonic-domain data, FonaDyn applies a simple form of 'machine learning', by which different EGG pulse shapes are automatically classified into 'clusters'. The learning feature has several important advantages. The categories need not be known in advance, rather, FonaDyn helps you to find them. The classification does not initially rely on any prescribed thresholds, but rather emerges from the data. With a careful choice of the clustering parameters, it is possible to produce a classification or stratification of the EGG that is specific to the research question at hand.

FonaDyn also implements a novel way of detecting phonatory breaks and instabilities, using the so-called 'sample entropy' applied to cycle-by-cycle metrics. In moderate to loud phonation, this metric stays at zero, except when something sudden happens, such as a voice break.

FonaDyn is not a stand-alone application. It is run from within the SuperCollider system, which must be installed first. SuperCollider is a free open-source software development tool that runs on MacOS, Windows and Linux. It was originally developed by James McCartney for the computer music community.

For the hardware, you will need a reasonably modern computer, a high-end digital audio interface, a good microphone, an electroglottography system with an analog line output, and a quiet room in which to record. SuperCollider can use any Windows- or Mac-compatible audio interface, but not laboratory data acquisition boards. With some optional hardware, it is nevertheless possible to acquire also non-audio signals, in parallel and in synchrony with the voice and EGG signals.

This book has three parts:

- Part 1 describes how to get started with FonaDyn: the installation and the hardware requirements.
- Part 2 describes the theory and design choices underlying FonaDyn.
- Part 3 describes the user interface and the most common handling procedures.

1 PART ONE - Getting started

1.1 Hardware requirements

- A low-noise microphone at a fixed distance from the mouth. It is important to verify that no audible hum, hiss or other extraneous sound is present, by listening through the system, at high gain.
- A high-end digital audio interface that can be configured to have a microphone signal on the first input and a line-level signal on the second input. We recommend the RME line of USB/Firewire audio interfaces (www.rme-audio.com). Others may work just as well.
- An electroglottograph that has an analog output for the EGG signal. Note that this handbook does *not* cover the use of your EGG device as such. Please observe its instructions carefully, for how it must be powered, maintained and connected for use.
- A high-performance computer running Windows 7 or higher, or Apple Macintosh running macOS. The screen resolution needs to be at least 1200× 800 pixels. It is convenient to have two display screens, since other programs will often be used at the same time.
- For prompting of the subject, a pair of closed circumaural headphones.
- For parallel acquisition of non-audio signals, such as aerodynamic data, see section 3.2.7.

1.2 Software setup

IMPORTANT: If your computer is centrally managed by an IT department, there are several things that they have to know about, in order to install SuperCollider and FonaDyn in a useable way. These are listed in section 1.2.6.

If you have not already done so, you will need to install the driver and control software that came with your digital audio interface or sound card, and become reasonably familiar with it.

Then install SuperCollider on your system, as described below. Once you have installed SuperCollider itself, you should probably play around with it a bit, if only to learn how to run anything at all. You need to know at least how to evaluate a line of SuperCollider code, so that you can install and start FonaDyn. There is a good Help system built into SuperCollider. There is also a separate website at doc.sccode.org that you can read from anywhere, with the same content.

Installing FonaDyn will also incorporate a number of FonaDyn-specific files into the SuperCollider Help system. These will be of interest mostly if you are interested in *how* the FonaDyn software works, under the hood.

1.2.1 Versions of SuperCollider

SuperCollider is maintained by an active community of voluntary developers. New versions are released about once a year. There are versions for the major operating systems, and also in some cases separate 32-bit and 64-bit versions. From the user's perspective, there is little or no difference between them. The versions mentioned here were the current versions as of July 2019.

The SClang source code for FonaDyn is portable across all platforms. The FonaDyn plugins, however, come in different versions for each platform. FonaDyn version 2.0.2 is supplied with both 32-bit and 64-bit plugins. It has been developed on Windows and tested on MacOS and Linux.

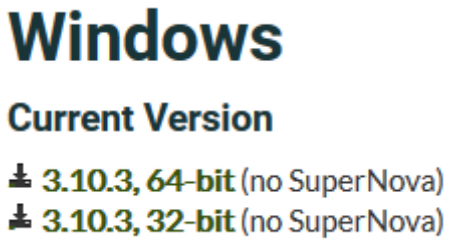
With version 3.9.0 of SuperCollider, the binary format for its plug-in executables was changed. This means that the FonaDyn-specific plug-ins in FonaDyn versions up to and including 1.5.0 will run only with SuperCollider versions up to 3.8.x. If you are updating from FonaDyn 1.5.0 then you must remove all existing components (backing them up in case you encounter problems and need to restore) and make a fresh installation. FonaDyn versions 2.0.0 and higher use only the new SuperCollider plug-in format.

1.2.2 Windows 7 and higher

Special note: all SuperCollider versions to date have the following known problem: on some Windows 10 computers that are under centralized management, the SCIDE editor may fail to start up properly. The symptom is that the Help docklet displays "Sending request..." indefinitely, and the interpreter does not become operative. This problem goes away, although no-one yet knows why, when the Windows policy "Audit Removable Storage" is deactivated. This can only be done by a system administrator.

Installing SuperCollider 3.10.3

1. Visit the SC download web page [2] and download one of the following two components. If you are running a 32-bit Windows, for instance on a breadboard computer, choose the 32-bit version. On the current versions of 64-bit Windows, either of these will work. ('SuperNova' is a multi-processor option that is not used by FonaDyn.)



2. Install SuperCollider 3.10.3, by running the downloaded .EXE file.

3. Back on the SC download web page, scroll down to **Plugins and Extensions** and click on **sc3-plugins**. On the new page, click on the button at **Download latest release**. Download the .ZIP archive that matches your choice at (1).

   sc3-plugins-3.10.0-Windows-x64-VS.zip (64-bit)

   sc3-plugins-3.10.0-Windows-x86-VS.zip (32-bit)

4. Open the ZIP and read the README file, to choose the right storage location. Then install the sc3-plugins, by extracting the other contents of the downloaded .ZIP file into the appropriate Extensions directory. We recommend that you install the sc3-plugins into the directory for all users. On recent versions of Windows, this will be the directory `C:\ProgramData\SuperCollider\Extensions`.

1.2.3 MacOS

Installing SuperCollider 3.10.3

1. Visit the SC web download page [2]. Download the following:

This will download a zip file that contains the Mac installer package and some README files. Read them.

2. For getting the sc3-plugins, scroll down to **Plugins and Extensions** and click on **sc3-plugins**. On the new page, click on the button at **Download latest release**. Download the file sc3-plugins-3.10.0-macOS.zip.

3. In the **sc3-plugins** .ZIP archive, read the README file, to choose the right location. Then install the sc3-plugins, by extracting the other contents of the downloaded .ZIP file into the appropriate Extensions directory. We recommend that you install the sc3-plugins into the directory for all users. On recent versions of macOS, this is `/Library/Application Support/SuperCollider/Extensions`.

1.2.4 Linux

Linux runs on many diverse platforms. At this writing, the FonaDyn 2 plugins have not yet been compiled for any Linux (older versions did run). Please contact the author at stern@kth.se if you are interested in using FonaDyn on some implementation of Linux, and/or in helping us make that build. You will then be provided with the plugin source codes, and with some guidance.

There is a Linux distribution for SuperCollider [2], but, as is customary with Linux, it is provided in source code form only. This means that you have to download the whole source and build SuperCollider itself, before you can install FonaDyn on Linux. Experience of making such builds will be necessary. You will need to find version 3.10.3 or higher in the Releases link at supercollider.github.io/download.

1.2.5 Installing FonaDyn into SuperCollider

1. Download **FonaDynInstall-*version*.zip** using the download link that you have obtained from the *SoftwareX* repository, or directly from us.
2. Unzip only the contained directories **FonaDyn** and **FonaDynTools** into an Extensions directory. We recommend that you put them into your own SuperCollider User Extensions directory.

   On recent versions of Windows, this will be

   `C:\Users\<username>\AppData\Local\SuperCollider\Extensions`.

   The directory `C:\Users\<username>\AppData` is often hidden, but if you type it into the address bar in the Windows Explorer, it will open.

   On MacOS, this will be the directory

   `/Users/<`**`username>`**`/Library/Application Support/SuperCollider/Extensions`.

3. The remaining contents of FonaDynInstall-2-0-2.zip are various useful files that are not SuperCollider code. Unzip and store them wherever you like, *except* in the Extensions directory or any of its subdirectories.
4. Evaluate the sclang code `Platform.userAppSupportDir` ; the path to this folder will appear in the Post window.
5. Unzip the file `startup.scd.example.txt` and adapt the settings to your system. Save it as `startup.scd` in the folder from (4).
6. Run SuperCollider. Wait for the "Post window" to display this line:

   ```
   *** Welcome to SuperCollider 3.10.3. *** For help press Ctrl-D.
   ```

7. In a new code window (**File** | **New** or Ctrl+N), type and then evaluate the line

   ```
   FonaDyn.install;
   ```

   This will perform a few checks, and copy a few additional files to their proper locations. You should now be able to start FonaDyn, by evaluating the line

   ```
   FonaDyn.run;
   ```

8. If the installation appears to have failed, or you change your mind, you can delete all the FonaDyn-specific files (no questions asked!) by evaluating the line

   ```
   FonaDyn.uninstall;
   ```

The FonaDyn directory contains source code and Help files that are specific to FonaDyn. You can read these help files using the built-in SuperCollider help system, under “Browse | FonaDyn”.

The FonaDynTools directory contains supporting code that was written for FonaDyn, and may be of interest also to developers of other SC programs. You can read the corresponding help files similarly, under “Browse | Tools”. The FonaDynTools directory also contains subfolders with the platform-specific compilations of the FonaDyn UGens.

1.2.6 Centrally managed installation

If your computer is centrally managed, it may have access policies in effect that will stop SuperCollider from doing certain things. Please show this section 1.2 of the handbook to your IT-support department, so that they can install SuperCollider with the necessary privileges, etc. Have them note especially the following:

SuperCollider is a software development environment in which it is necessary for users to be able to create/modify files in certain privileged locations and/or functions. At run-time, it does real-time processing of audio files, which needs to be fast, while being harmless from a malware point of view.

In Windows, the easiest thing is usually to install SuperCollider not under "Program Files" or "Program Files (x86)" but in some directory directly under C:, for example C:\SuperCollider\, and properly set Access Control Lists (ACL) to allow users to change files within this directory. This will allow any user to edit the important configuration file **startup.scd**, for example.

There are three executable modules: **scsynth.exe**, **sclang.exe** and **scide.exe**, which communicate with each other using a network protocol, and they must be allowed to do so. For performance reasons, it may also be necessary to include their processes in the list of processes that are excepted from anti-virus monitoring.

An active Windows 10 policy "Audit Removable Storage" is known to disrupt the communication between SCIDE and SCLANG. The reason for this is as yet unknown. The symptom is that the Help Docklet displays "Sending request…" indefinitely and the SCLANG interpreter never becomes operative. Therefore this policy must be deactivated on computers running SuperCollider.

Network security issue in SuperCollider versions up to 3.10.2: The process `scsynth` listens to *any* incoming traffic on port 57110, using the OSC (Open Sound Control) protocol. This can give too much access to an untrusted party that knows SuperCollider. Normally, only the internal processes `sclang` and `scide` should have access. Configure the machine's firewall to block outside traffic to port 57110. And/or, the following line can be added to **startup.scd**. It makes `scsynth` listen to the host machine only.

```
Server.program = Server.program + "-B 127.0.0.1";
```

This issue was fixed in 3.10.3.

For its signal processing functions, SuperCollider uses a large number of small 'plugin' DLL's that have the filename extension .SCX (on Linux: ".SO") . On rare occasions it may happen that an anti-virus program will quarantine one of these, causing SuperCollider to complain that a particular module is not found. We have never known this actually to be malign, but of course your policies must be upheld. You will have to decide what to do about it.

For Windows, FonaDyn is supplied with plugins for both 64-bit and 32-bit versions of Supercollider. The appropriate set of plugins is selected automatically during installation using the routine FonaDyn.install, as described above.

For MacOS, there is in FonaDynInstall-*version*.zip/FonaDynTools/macos a library **libfftw3f.3.dylib** which might be present on the Mac already. If not, it needs

to be copied into /usr/local/lib. This copy operation is attempted by the FonDyn installation script (`FonaDyn.install`), but it may fail if the user is not authorized.

On MacOS, the FonaDyn installation replaces the sc3-plugin `PitchDetection.scx` with one that has been recompiled to call the abovementioned library, rather than the Mac's own FFT library – they give different results. Running `FonaDyn.uninstall` will restore the original `.scx` file.

1.2.7 Audio device configuration

FonaDyn requires one stereo pair in and one stereo pair out. If your system's default audio device has only two inputs and outputs, then these are what FonaDyn will use. If you are using an external multichannel audio interface (recommended), then your operating system will probably list more than one pair of inputs and one pair of outputs. FonaDyn will use the first pair of inputs and the first pair of outputs. It can be configured (in the program code) to record and play on any combination of multiple inputs and outputs, provided that they all sit on the same audio interface hardware (or on separate, but electrically synchronized devices). A list of currently available audio devices is printed in the Post window, whenever the server process SCSYNTH is booted.

If you will not be using SuperCollider for anything else, you can specify 2 ins and 2 outs in the SC startup file. For instructions on how to do this, see the SC documentation for the startup file. The startup file has its own entry in the **File** menu of SCIDE, so it is easy to find. Activating more ins and outs than are needed incurs an unnecessary CPU load.

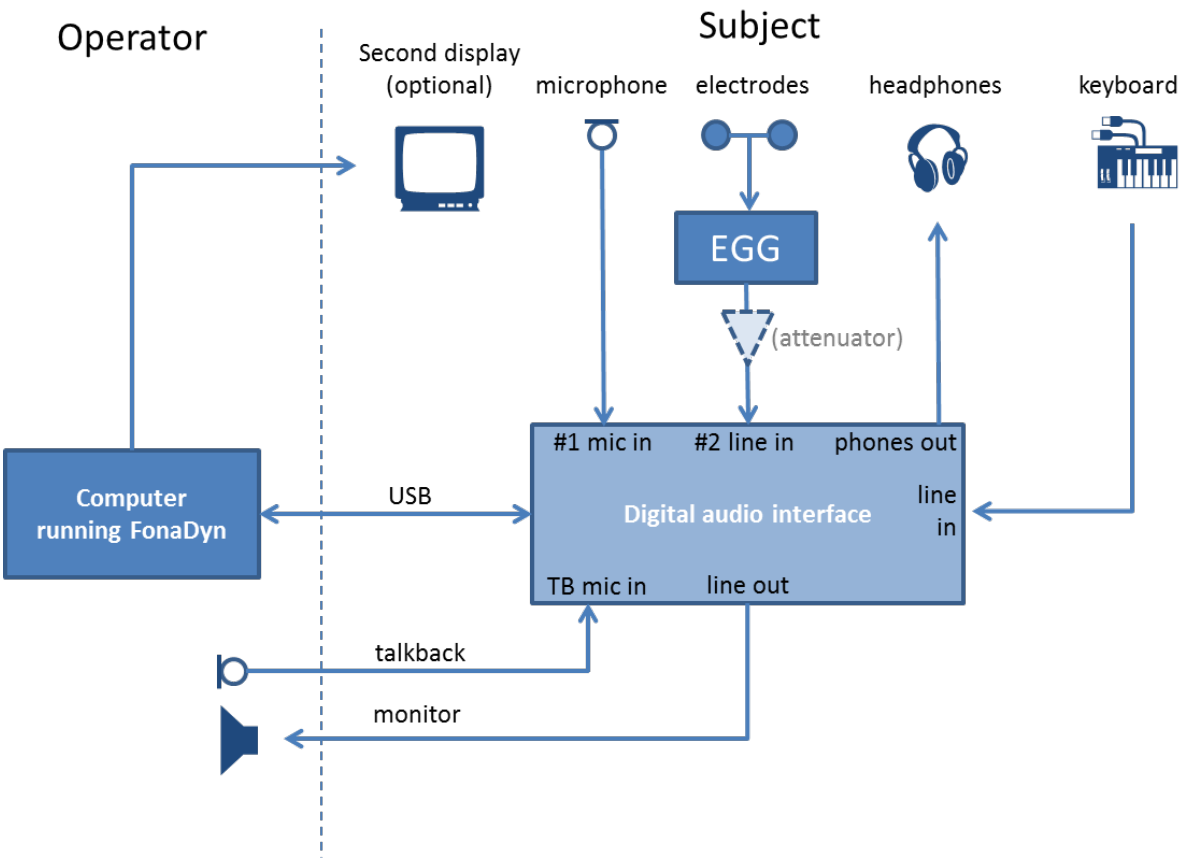


Figure 1. Typical recording setup.

A typical experimental setup is shown in Figure 1. The first (left) input channel must receive the signal from a microphone, via a microphone preamplifier. Many audio interfaces have built-in microphone preamps, on two or four inputs.

The second (right) input channel must receive the line-level signal from the electroglottograph. Unlike the microphone signal, the EGG signal level does not need to be calibrated. However, it should be adjusted so as to prevent clipping. Note that some EGG devices have a large voltage swing on their outputs, typically ±10 volts, which is too strong for most audio interfaces. If the EGG device's output level cannot be turned down enough, you may need to pass it via an attenuator to the audio interface. If your audio interface has adjustable input sensitivity, choose the *least* sensitive setting for the EGG signal.

2 PART TWO - Theory

2.1 Introduction

FonaDyn allows you to explore how the EGG waveform varies with the fundamental frequency (f_0) and the sound pressure level (SPL). It does this by computing various features of the EGG waveform and mapping these by colour onto the f_0-SPL plane, also called the 'voice field'.

The *time-domain* features (→2.5) are scalar numbers that are mapped straight to a colour scale. The resulting 'voice maps' unfold in real time as the subjects vocalizes, or as an existing recording is analyzed. The signal preconditioning and the computation of the time-domain features are described in reference [3].

The *harmonic-domain* features (→2.6) of the EGG waveform are high-dimensional vectors that cannot be assigned to colours as such. They are first clustered to ordinal cluster numbers, which then are mapped to colours. The clustering can be automatic, or, an experienced operator can cajole the clustering into categories that are of interest for a particular application. Normally such an analysis has these stages:

(1) record to file
(2) learn a set of EGG waveshapes from the recording
(3) use that set to classify other EGG waveshapes
(4) export the results for further treatment.

Here, stages (1) to (3) are done in FonaDyn. The "further treatment" in (4) is done with the tools of your choice, e.g. MS Excel or Matlab. To facilitate research work, practically all signals and results from FonaDyn can be exported to files in common formats (.csv, .wav, .aiff), for subsequent processing in other software.

2.2 Signal processing overview

The audio and EGG signals are processed in parallel, see Figure 2. From the microphone audio signal, four metrics are derived: the signal level, the fundamental frequency f_0 , the f_0 periodicity (or "clarity"), and the crest factor. The level and f_0 serve to specify the plot position in the voice field. The clarity metric is used as a periodicity gate: ranging from 0 to 1, it must be close to 1.0 for the EGG processing to proceed. The default threshold is 0.96 . The crest factor gives an approximate but useful indication of the high frequency content of the voice signal [5].

The gain of EGG signals is difficult to calibrate, and also typically varies a lot (due to varying larynx height, for instance). Therefore, all metrics computed by FonaDyn are normalized such that the EGG amplitude is disregarded. Only the EGG wave *shape* is considered to be of interest, and hence only a coarse adjustment of the EGG signal gain is needed.

The EGG signal is first segmented into cycles. From here on, the data rate is reduced, being proportional to the EGG cycle rate, rather than to the sampling rate. Each cycle is analyzed individually, both in the time domain, and in the harmonic domain. The harmonic analysis uses a Discrete Fourier Transform (DFT) for its N lowest Fourier components, where N is typically 10 or less.

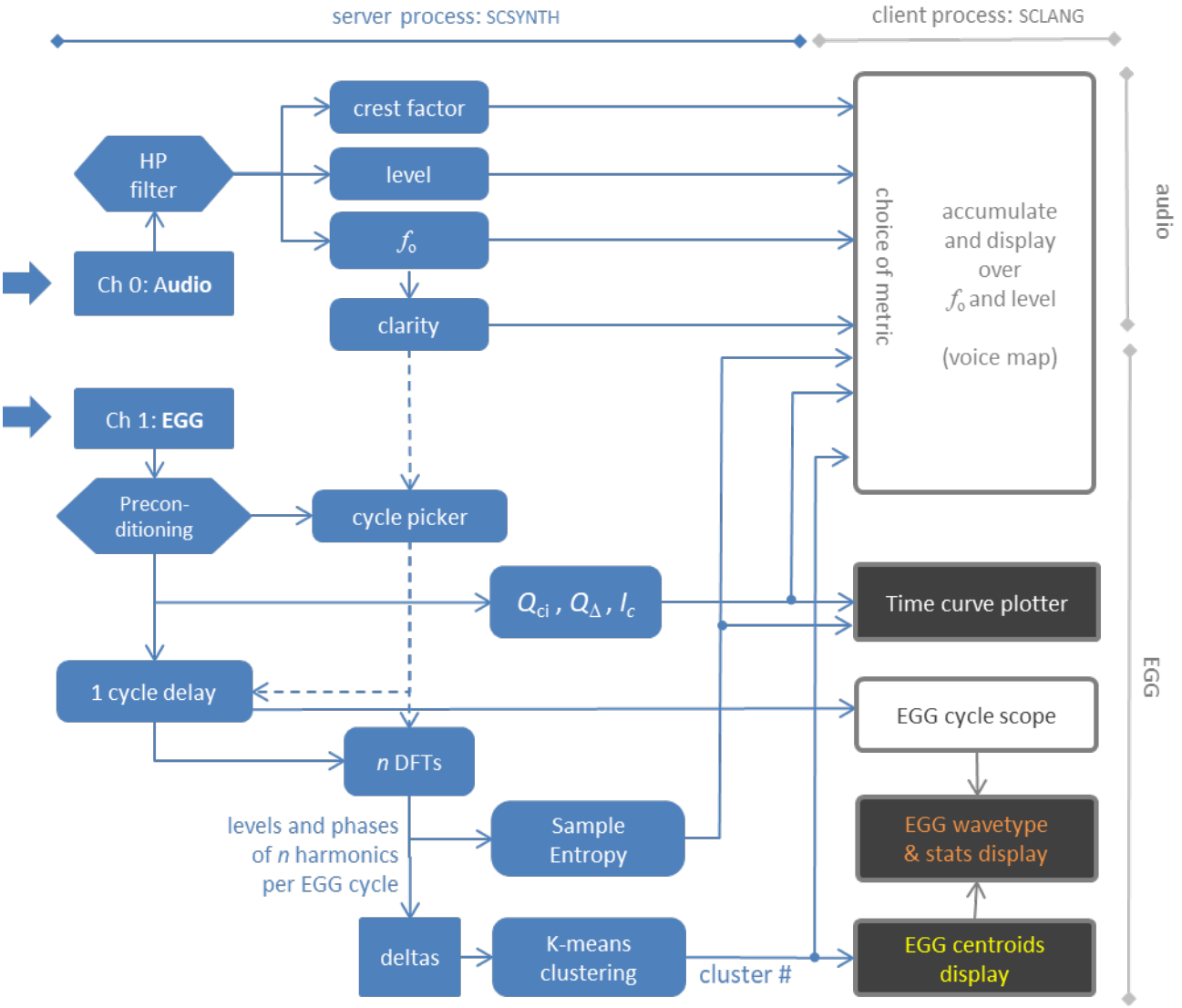


Figure 2. Overview of the signal processing in FonaDyn.

The underlying ideas for the harmonic-domain analysis were first proposed by Selamtzis and Ternström [4]. The input EGG signal is first segmented into individual cycles, each of which is analysed with the Discrete Fourier Transform. The resulting harmonic magnitudes and phases are then input to a so-called K-means clustering algorithm. This algorithm treats the data as point coordinates in a high-dimensional space, and allocates each point to the cluster whose centroid is closest in this space. That centroid is then updated to include the new point. This gives rise to a given number of categories, into which the cycles are classified. The clustering is 'learned' in real time, without any prior knowledge of EGG waveshapes. The clusters are colour-coded for display in the voice map, with a separate layer for each cluster. The number of categories, or clusters, must be chosen to be appropriate to the research question, usually 2...5 . Arriving at that choice usually requires some exploratory

passes over the data. 'Learned' cluster data can then be saved, reloaded and used to classify new signals.

Each Fourier component is a complex number with a real and an imaginary part, or, equivalently, with a magnitude and a phase. To minimize the influence of the variable gain of the EGG signal, *relative* magnitudes and phases are used instead. The first Fourier component, that is, the fundamental, is taken as the reference. For components 2…*N,* the *relative magnitude levels* and the *phase differences* to component 1 are taken as the input to a clustering algorithm. This means that the overall EGG amplitude does *not* affect the clustering. In addition to the *N* first Fourier components, the system estimates the level of that residual EGG signal power which is *not* accounted for by the first *N* Fourier components, and uses that as an extra clustering dimension.

In parallel with the clustering, the time series of per-cycle DFT data are analyzed also for their 'sample entropy', or SampEn for short. This is a metric of phonatory stability that is designed to detect abrupt changes, such as register breaks, while suppressing minor instabilities.

2.3 Audio processing

2.3.1 Input selection

FonaDyn takes a microphone audio signal on the first A/D channel and an EGG signal on the second A/D channel; or, takes its input from two-channel audio files. Valid input audio file formats are those supported by the 'libsndfile' library, in other words, almost any format. The FonaDyn sampling rate is 44100 Hz per channel. FonaDyn's own recordings of Voice+EGG are stored in 16-bit integer format .WAV files.

When recording from the live inputs, the 'raw' microphone and EGG signals are written unchanged to the output file, with no prior processing. This is to avoid repeated preprocessing when analyzing a recording rather than the live inputs. It also enables the recordings to be used by other analysis systems, with no constraints having been imposed on the signals. When analyzing a first-generation FonaDyn recording, the input signals are identical to the live inputs of the A/D-converters (→3.2.6).

2.3.2 Audio preprocessing

A second order high-pass filter at 30 Hz is always applied to the microphone signal, to suppress low-frequency rumble. This filter is not linear-phase, and does not need to be.

2.3.3 Voice map metrics

From the microphone audio signal, four metrics are derived: the signal level in dB, the fundamental frequency f_0 , the f_0 periodicity (or ”clarity”), and the crest factor.

- The *level* of the audio signal serves to provide the vertical plot position in the voice maps. It is computed as follows.
  1. The conditioned audio input is squared.
  2. The squared signal is averaged over a 33 ms rectangular window. The reason for using a frame average (FIR) rather than an IIR low-pass filter is that the latter would incur sluggish onsets and drooping offsets.
  3. The 33 ms averages are median-filtered over 17 control rate periods (25 ms). This further reduces level lag at sudden signal onsets/offsets. The resulting total delay matches the parallel delays incurred in conditioning the EGG signal.
  4. The values are converted to decibels, on the assumption that 0 dB full scale corresponds to 120 dB SPL. You need to adjust the gain in your signal chain so that this is true, or, be prepared to correct your results in post-processing.

  Calibrating the signal level to true sound pressure level (SPL) is *very* important, for meaningful comparisons between across recordings; both your own and those of others (→3.2.4).

- The f_0 value serves to provide the horizontal plot position in the voice map. It is computed by a SuperCollider ‘UGen’ called `Tartini`, whose algorithm is based on autocorrelation via the FFT [5]. It updates the value of f_0 at intervals of about 5 ms. In FonaDyn, f_0 is measured in semitones, using MIDI note numbers with fractions. MIDI = 57.0 corresponds to f_0 = 220.00 Hz. No calibration of f_0 is needed.

- The *clarity* metric, also computed by the `Tartini` UGen, is used as a periodicity tester. Ranging from 0 to 1, it must be above a certain threshold for the EGG processing to proceed (the default threshold is 0.96). This is a fairly strict requirement, corresponding to stable phonation.

- The *crest factor* gives a simple but useful indication of the high frequency content of the voice signal [5]. In open vowels only, it is a close cousin of the maximum flow declination rate (MFDR). It is computed as the ratio of the peak amplitude to the RMS amplitude. A sine wave has a crest factor of $\sqrt{2} \approx 1.414$ (≈ 3 dB), while a very bright voice signal can reach a crest factor greater than $4 \approx 12$ dB.

2.4 EGG processing

2.4.1 Preprocessing

The absolute EGG peak amplitude is monitored during recording (→3.2.3). If it comes within 0.5 % of full scale, a clipping warning is flashed on the screen. Clipping would compromise the spectral analysis for the clustering.

The EGG signal is then

1. high-pass filtered at 100 Hz (-3 dB @ 80 Hz), using a 1024-point linear phase FIR filter. This suppresses the large near-DC content that is common in EGG signals;
2. median-filtered over 9 sample points; this reduces somewhat the low-level 'crackle' that is common on some EGG devices, especially when the battery cells are aging;
3. low-pass filtered at 10 kHz, using a "brick wall" 64-point linear phase FIR filter (-80 dB @ 12 kHz).

This EGG preprocessing introduces a 34 ms delay, which is accounted for internally. It also limits the range in f_0 : downwards, to 100 Hz, and upwards to 10 kHz divided by the number of DFT components in the analysis.

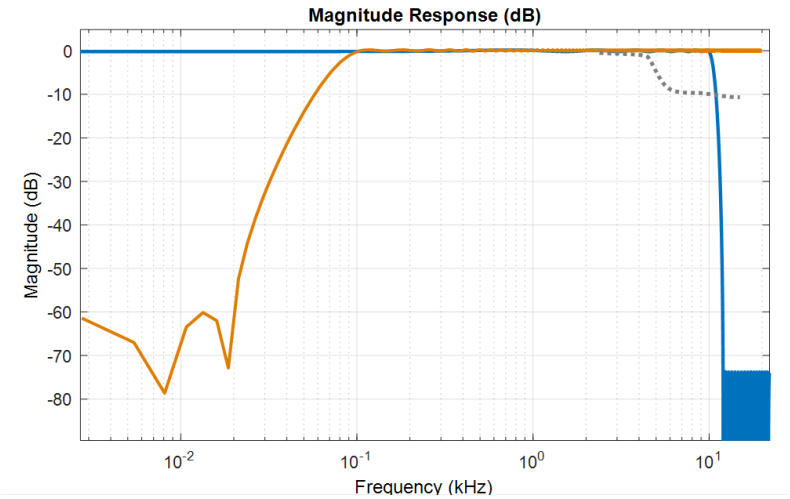


Figure 3.

Frequency responses of the EGG filters: high-pass (blue) and low-pass (yellow). The 9-point median filter (dotted) affects only low-level noise.

2.4.2 Period segmentation

FonaDyn implements two strategies for cycle separation: double-sided peak-following, after Dolanský [7]; and phase tracking. The double-sided **peak follower** is the simpler one; it is applied to the differentiated EGG signal (dEGG). The dEGG is computed as the differences between consecutive sample points. The positive peak-picker selects a point somewhere just prior to a local positive maximum of the dEGG signal. The 'circuit' then waits for a local negative maximum to occur, before allowing the next positive dEGG peak to trigger a new period. Using the dEGG rather than the

EGG makes the triggering point somewhat less dependent on f_0 , but also more sensitive to noise.

Sudden changes in dEGG amplitude (which are common) may lead to one or a few lost cycles. Sudden increases in dEGG amplitude may cause the preceding cycle time to be estimated as slightly shorter.

The default method [3], however, is based on the **phase portrait**. The EGG signal is time-integrated and then paired with the original to form a complex-valued signal of the integrated EGG. By using the integrated signal rather than the more common Hilbert transform (which is somewhat similar to the *derivative* of the EGG), we obtain better rejection of low-level noise, and the cycle trigger point becomes similar to that in the alternative peak-following method described above. The phase of this pseudo-analytic signal is now computed as the arctangent of the complex pair; and then that *phase* signal is subjected to the Dolanský method described above. The phase wrap-around from π to $-\pi$ typically occurs very near the dEGG peak, giving a convenient transient in the phase signal. This method for cycle segmentation generally performs better than peak-picking with noisy signals, as well as with fluctuating amplitudes, and it seems to give more robust segmentation overall. With unusual EGG signals, however, which may have multiple deep inflexion points per cycle, this method is sensitive to multiple loops in the phase portrait, and may be less reliable than the dEGG peak-following method.

Cycles longer than 20 ms (50 Hz) or shorter than 0.23 ms (4410 Hz) are rejected at this stage.

2.5 EGG time-domain metrics

A somewhat more detailed discussion of these metrics is given in reference [3]. These metrics are computed directly from the conditioned EGG signal.

2.5.1 The contact quotient Q_{ci}

The contact quotient is the portion of a whole EGG cycle for which the vocal folds can be said to be in contact; hence, it can range from zero to one. Numerous schemes have been proposed for calculating it [8]. Here, we are content to find a quantification that represents the relative *amount* of contacting over the cycle, without regard for the actual instants of opening or closing, nor for the possible complementarity with the glottal area waveform.

Consider an EGG pulse normalized to length 1, with the cycle amplitude normalized to the range 0...1 . The *area under* this normalized pulse can also range from 0 to 1. This area can be estimated for arbitrary pulse shapes, so it does not require a single peak, nor well-defined opening and closing events, nor a contacting threshold that would always be in some sense arbitrary. We will call this metric the Q_{ci} , for “quotient of contact by integration”. It correlates well with other quotient metrics. The figure below shows how it is derived.

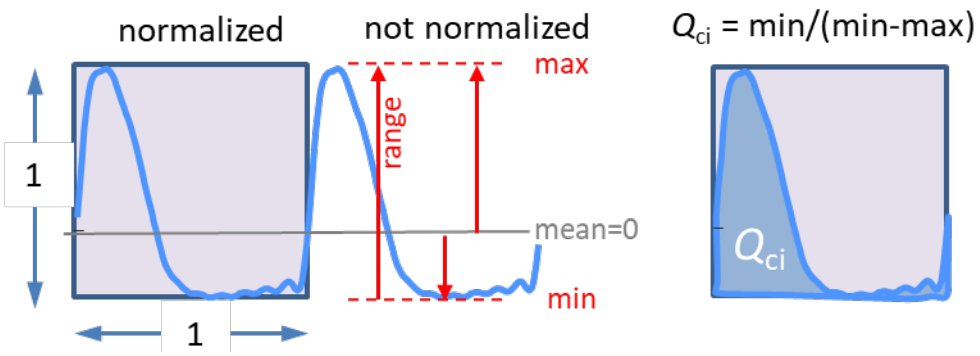
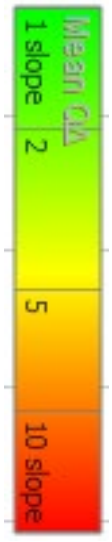
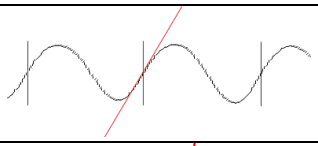
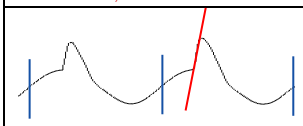
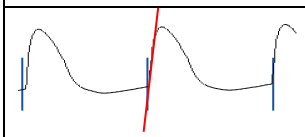
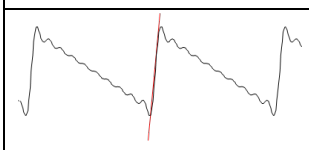


Figure 4. Computation of Q_{ci} . If the non-normalized EGG signal is DC free, with a mean of zero over the cycle, then the integral of the normalized pulse can be computed simply as min/(min - max) .

Note that this Q_{ci} metric has the same problem as other definitions of the contact quotient, in that when vocal fold contact ceases, the waveform becomes low in amplitude and nearly sinusoidal. Therefore, Q_{ci} will "erroneously" tend towards 0.5, which would correspond to rather a large amount of contact in normal phonation. You need to keep this in mind when considering the Q_{ci} in the voice map display. The next metric does not have this problem.

2.5.2 The Q_{Δ} metric

The Q_{Δ} metric gives the maximum positive slope of the cycle-normalized EGG signal. This is a metric of the relative rate of change of contact area. It is *not* the same thing as the speed with which the vocal folds are moving towards each other, although typically there will be a co-variation of these two speeds. The Q_{Δ} value is normalized such that a sine wave receives a slope of 1 . The wave shapes and their colour mapping in the voice map are shown in the table below.

Mean Q_{Δ}	EGG shape and slope	Interpretation
		1…1.5 Nearly sinusoidal: no vocal fold collision. This would be at very low EGG amplitudes.
		1.5…3 Incomplete contacting: just above the threshold of collision
		3…10: normal range of contacting rates in modal voice
		*N* = max rate of contacting using *N* harmonics (synthesized with phases=0 and amplitudes 1/k)

The slope is measured on the EGG waveform itself, rather than using the Fourier descriptors obtained for each cycle (see 2.6.1), and therefore it is somewhat sensitive to noise at low amplitudes; however, these slopes are averaged per cell in the voice maps, so this is not a major problem.

Because the EGG is high-pass filtered (DC free), the peak of the EGG derivative usually coincides closely in time with positive-going zero crossings in the EGG. The exception is for partial contacting (the yellow row above).

The Q_Δ metric is similar, but for a scale factor, to the *inverse* of the Normalized Amplitude Quotient (NAQ), introduced by Alku, that is sometimes used to characterize glottal *flow*.

2.5.3 The I_c metric

The 'index of contacting' I_c was introduced by Ternström [3]. It is a combination of Q_{ci} and Q_Δ , namely $I_c = Q_{ci} \times \log_{10}(Q_\Delta)$. The index of contacting has the property of being zero for a pure sine waveshape, as in very weak phonation, when there is vocal fold vibration without collision; and approximately one when Q_{ci} and Q_Δ are both high, as in strong phonation.

2.6 EGG harmonic-domain analysis

2.6.1 DFT analysis

A DFT analysis is performed of the lowest (and strongest) frequency components only of the EGG signal. For N components, the program computes N value pairs (magnitude and phase) cycle-synchronously over each EGG cycle. N can be chosen as 2...20; typically it will be 6...10. Because the DFT frame length adapts to the EGG cycle length, the DFT components are equivalent to the harmonics of the EGG signal. A small quantization error arises here, in that the true EGG period time is not exactly an integer multiple of the sampling interval. In practice, though, such errors are absorbed by the subsequent statistical clustering.

The Fast Fourier Transform is *not* used, because the frame length here must be exactly one EGG cycle (to the nearest sampling interval), and because N is small. Instead, the desired DFT cosine and sine terms for the most recently completed EGG cycle are computed directly in the time domain. This also results in a constant CPU load for short and long cycles.

2.6.2 EGG waveform DFT components

For each EGG cycle that has been judged as valid, the first Fourier component (the "fundamental") is taken as a reference, for both magnitude and phase. The magnitudes, or absolute values, of the complex Fourier components $k \in [\,2...N\,]$ are computed, and their ratios to the magnitude of Fourier component 1 (the fundamental) are expressed as level differences ΔL_k . Using only the *relative* harmonic levels eliminates the effect of a variable gain on the EGG signal.

Similarly, the phases of the Fourier components $k \in [2...N]$ are computed relative to the phase of the fundamental, i.e., $\Delta\varphi_k = \varphi_k - \varphi_1$. The reason for computing the phases relative to φ_1 is that the cycle detection algorithm finds a cycle triggering

point, whose location in the EGG cycle will depend somewhat on the current wave shape. By using the relative phases, this dependency is prevented from affecting the DFT results.

2.6.3 Residual energy

Since N is typically small, most of the high-frequency information in the EGG signal is lost. However, the overall spectrum slope of the EGG signal is typically quite uniform towards high harmonics, and thus the higher harmonics all tend to carry much the same information. Also, at low EGG signal levels, the higher harmonics often descend below the analog noise floor of the EGG hardware. In FonaDyn, the residual energy in all the harmonics > N (plus any noise) is estimated by summing the energies in the N lowest harmonics, subtracting that from the total energy of the EGG signal, and again expressing the difference as a level relative to that of the fundamental. For convenience, this residual H is treated as though it were an extra harmonic, number N+1 . For low EGG signal amplitudes, H will tend to represent the relative level of the noise floor, rather than the relative level of the omitted harmonics.

This analysis method emphasizes the overall vibratory pattern of the vocal folds. It will tend to disregard very brief or transient events that are manifest mostly at high frequencies, such as multiple contacting events that occur very closely in time (small fractions of a cycle).

2.6.4 Phase of the fundamental

Since the phase of the residual energy H is undefined, that variable slot is used instead for the phase of the fundamental. This value is needed in order to reconstruct the EGG waveforms from the cluster centroid values. The phase of the fundamental is taken relative to the cycle trigger point found by the cycle detection. We do not want this phase to affect the clustering, though, so it is first down-weighted by 0.001.

2.6.5 Representation of phase differences

Since the phase angles φ_k are expressed in the range $[-\pi, \pi)$, a phase whose value crosses $\pm\pi$ will cause a 2π jump. This must be avoided, since it could give rise to falsely disjunct data clusters. One could take the absolute phase difference, folded over π , thereby avoiding jumps. The clustering would then improve, but some information is lost, and it would not be possible to reconstruct EGG waveforms from the cluster centroid values. Therefore, each relative phase $\Delta\varphi_k = \varphi_k - \varphi_1$ is instead represented by the value pair $(\cos(\Delta\varphi_k), \sin(\Delta\varphi_k))$. This eliminates the discontinuities, at the cost of an extra clustering dimension per Fourier coefficient.

From here on, the EGG harmonic-domain processing splits into two paths: DFT component clustering, and DFT component sample entropy estimation. The clustering is described first.

2.7 Clustering

2.7.1 Clustering dimensions

As described above, for each harmonic $k \in [\, 2 \ldots N \,]$, FonaDyn computes the relative level, and the cosine and sine of the relative phase. For each EGG cycle, this results in $3\times(N\text{-}1)$ values. Using these values, plus the estimated level of the residual energy and the value pair $[\, \cos(\varphi_1), \sin(\varphi_1) \,]$, the statistical clustering is performed in a space with $3\times N$ dimensions. This is done using a real-time implementation of the algorithm 'online Hartigan *k*-means' [8]. Compared to other methods for clustering, the *k*-means method has these advantages: (1) it computes quickly even in many dimensions, and (2) the *number* of points already in the clusters does not affect the classification, only the centroid updates. The latter means that, in a data set with thousands of EGG cycles, a small minority of cycles of an unusual shape can still give rise to a cluster of their own, especially if they occur early in the recording. A typical example is the weak sinusoidal cycles at onset and offset of voicing.

For the clustering to be effective, all dimensions should have values extending over roughly the same numerical range. For this reason, the level difference values as fed to the clusterer are expressed in Bels rather than decibels. This makes for a better match to the (cos, sin) values, which are always in the range [-1…1].

Note that *none* of f_0, SPL and total EGG amplitude are input as parameters to the clustering algorithm. Also, the positions in the clustering space of the cluster centroids are *not* related to the locations of the coloured regions in the voice map. Rather, each centroid represents a particular EGG pulse shape. This means that affinity to a cluster is *not* given by proximity in the f_0/SPL-plane. However, different EGG pulse shapes do tend to occupy different regions in the voice map, typically with some overlap. It is these regions that make up FonaDyn's 'cluster maps' of the EGG signal.

2.7.2 Resynthesis

In order to facilitate the user's interpretation of the clustering, the cluster centroid values of levels, cos and sin are used also to resynthesize and display the approximated cycle waveform (section 3.1.2), by addition of cosines. A potential problem here is that the cluster centroid values for sin and cos, since they are not strictly paired, no longer necessarily fulfil the trigonometric identity $\cos^2(\varphi) + \sin^2(\varphi) = 1$. Another potential problem is that the probability distributions of sin and cos are very far from rectangular. In practice, though, tests with synthetic waveforms (triangle, sawtooth, square) have shown that such reconstruction works well enough.

2.8 Sample entropy of cycle data

The 'sample entropy' of a signal [10][11] is an interesting metric that has found numerous biomedical applications. For brevity, it is often called SampEn. The SampEn is low for a regular, self-similar signal and high for a signal that is transient, erratic, or noisy. While others [12] with some success have taken the SampEn of high-rate, isochronous signals such as sampled EGG or audio, we have found that the

SampEn metric is particularly effective for phonation data that are low-rate, cycle-synchronous. SampEn has a threshold or 'tolerance' parameter that keeps it at zero while phonation is stable, even with changing pitch, but when 'something unusual' happens, the SampEn peaks. This 'something' could be a voice break, such as those occurring between chest and falsetto voice [2], or other instabilities in phonation.

In FonaDyn, the spectral analysis produces a vector of values that is updated on every phonatory cycle, so we obtain new values of the level and the phase, for all harmonics. These sequences of numbers each provide the input to one of several SampEn estimators, running in parallel for all levels and phases of the first few harmonics. The final SampEn value is simply the sum of the SampEns for all levels and phases.

The algorithm used for estimating the SampEn is given in [13]. The number of harmonics is selectable; typically it will be smaller than for the clustering analysis. A local SampEn is computed over a short sliding window of w glottal cycles, where the integer w can be chosen by the user ('Window'). The window is advanced one cycle at a time, so a new value is obtained on every cycle. Another parameter is the sub-sequence length ('Length'). The 'Tolerance' setting is analogous to a noise threshold. See also section 3.5.6.

For computing the sample entropy of the phases, we have potentially the same wrap-around problem as described in section 2.6.5. Jumps of $\pm 2\pi$ would give large but meaningless peaks in the SampEn. Here, this is avoided by substituting the phase with its absolute value. This gives a continuous signal, for all phases; and the resulting ambiguity is of little consequence for the SampEn estimation.

2.9 Limitations

The analysis considers only the $N \leq 20$ lowest harmonics of the EGG signal, and typically 5 or 10 are used. This means that some high-frequency aspects, such as multiple contacting events in quick succession, might not be resolved.

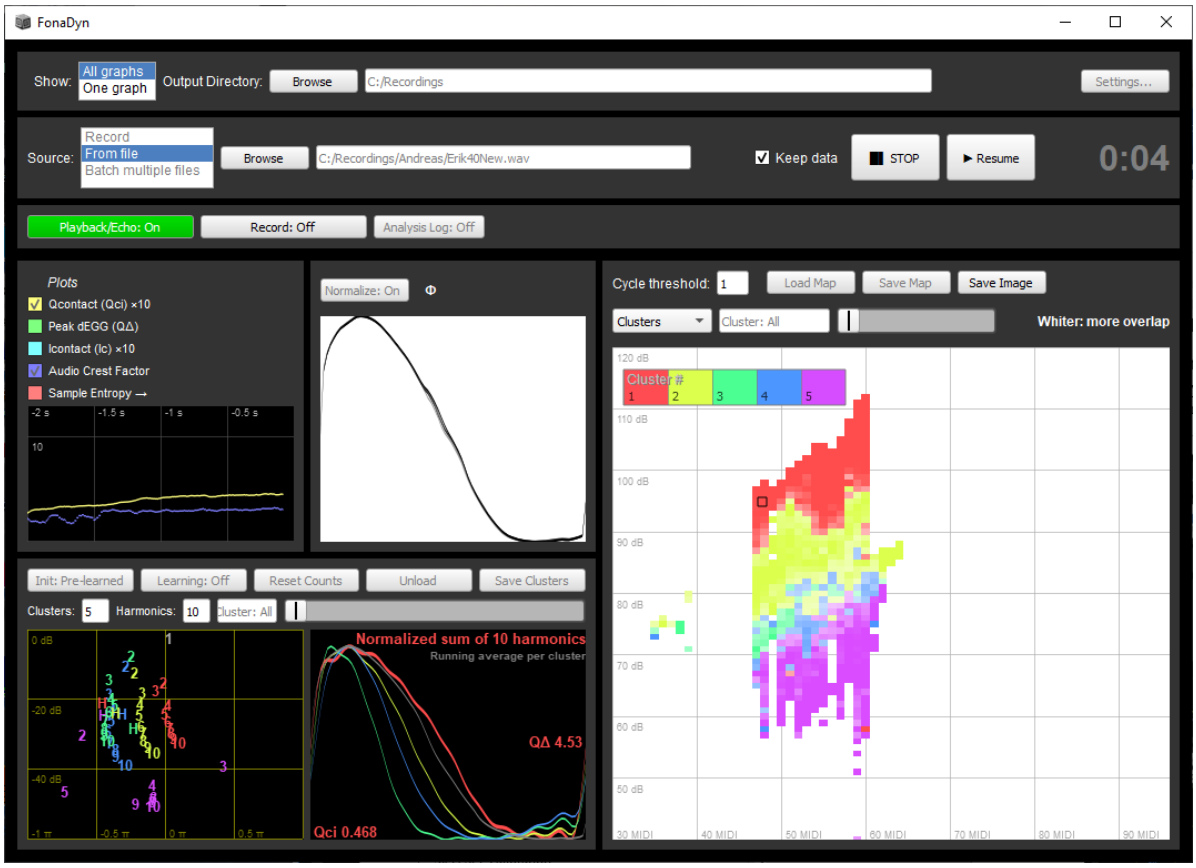
The preconditioning of the EGG signal means that the f_0 should be 80 Hz or greater, and not higher than $10000/N$ Hz. The more these bounds are exceeded, the more the clustering will start to depend on f_0.

3 PART THREE - Using FonaDyn

3.1 Window layout

3.1.1 Main screen

The FonaDyn main screen is a bit like the control panel of a machine. It has no pull-down menus; almost all controls are visible (Figure 5).



Panels 1-3, from top: General settings, Input selection and Start/Stop/Pause, Outputs selection		
Selected time plots panel Values during the last 2 s	Incoming EGG waveform oscilloscope	Voice map panel ↑ SPL / f_0 →
Cluster panel: centroids	EGG pulse shapes, or statistics	

Figure 5. The main screen of FonaDyn.

The top row holds general settings. The second row selects the signal for input, controls START/STOP/PAUSE, and displays the time since START. The buttons in the third row control the output options. The four subpanels with graphs show the analysis results, in real time, as described below.

You can use Tab or Shift+Tab to move the keyboard focus from one control to the next. *Numbers* can be edited with the keyboard, by dragging the mouse cursor vertically, or with the up-arrow and down-arrow keys. Ctrl and Shift increase the step size. There are no *text* fields that can be edited. All file names are entered by browsing to files.

3.1.2 The Clusters panel

The Clusters panel has five buttons along its top:

Button	States	Action
Init	Relearn	On START, clear the current cluster data
	Pre-learned	On START, keep the currently loaded cluster data, for continued learning, or for classification
Learning	On	Continue learning, updating the centroids
	Off	Classify incoming data, without updating the centroids
Reset Counts	<push>	Set the cycle counts of all clusters to zero, now. All centroids are initialized to the current DFT values – but they soon diverge. Useful for initializing, and for clearing spurious undesired centroids.
Load Clusters	Load	Load a cluster data file, for initialization or classification
	Unload	Clear all cluster data
Save Clusters	<push>	Save the cluster data to a file (`*_clusters.csv`)

If the filename you enter for **Save** does not end in `.csv`, FonaDyn will append `_clusters.csv` to the name. If the filename you enter *does* end in `.csv`, then FonaDyn will not change it.

A *cluster centroid* is a set of values that describe the average location of all the cycle points in the clustering space. In the centroids display, each *colour* corresponds to one centroid (and, equivalently, to one cluster). In Figure 6, one centroid (yellow) is shown. Here, we have specified 10 harmonics, so one centroid has 3×10 dimensions: 10 relative levels, and 10 sines plus 10 cosines for relative phases. The thin gray line shows a running average of the most recent EGG pulses that were assigned to this cluster.

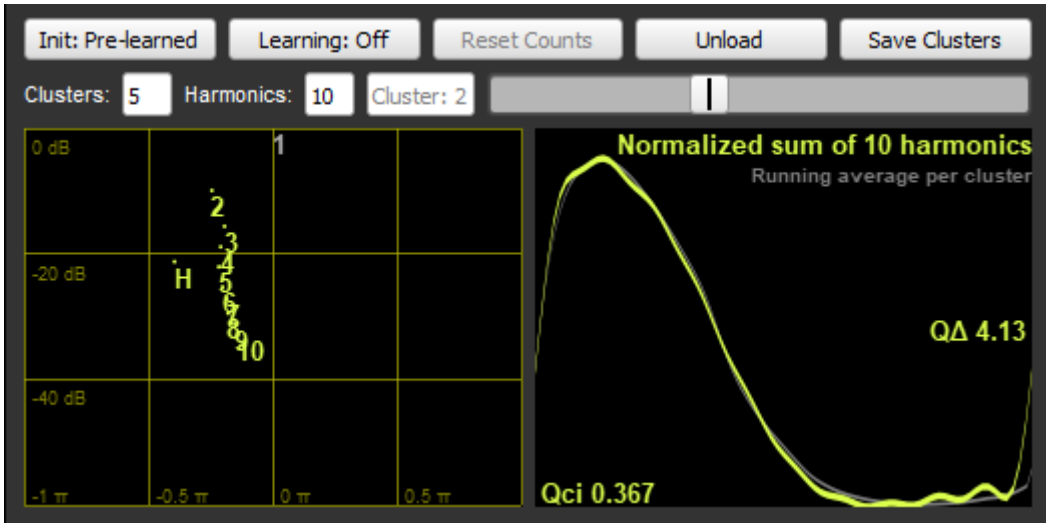
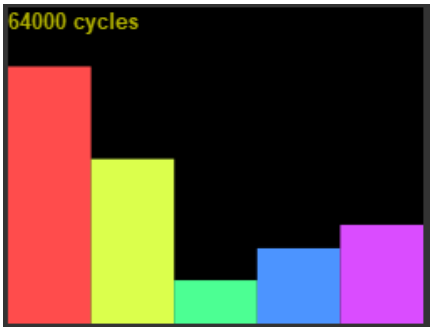


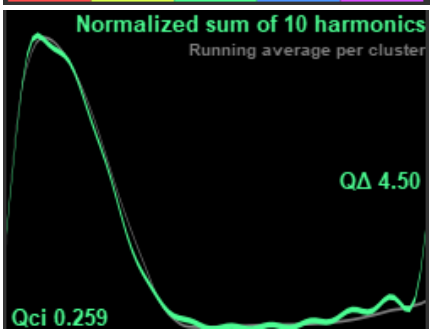
Figure 6. The graphs in the Clusters panel.

The grid to the left shows the levels and phases of the EGG harmonics, relative to the fundamental. By definition, then, the fundamental or first harmonic has the level 0 dB and the phase 0 radians. The gray '1' at top center (0 dB, zero phase) is thus implicit and shown for visual reference only. The data points are at the top-left corner of the glyphs. In Figure 6, the second harmonic '2' has a relative level of about -10 dB and a relative phase of about -0.25π radians. Finally, the 'H' shows the relative level of the residual power of all Higher harmonics, and also the pHase of the fundamental, relative to the cycle trigger point. If you were to string out the digits in sequence from left to right, keeping their heights, you would get the typical power spectrum of the EGG pulses in the given cluster.

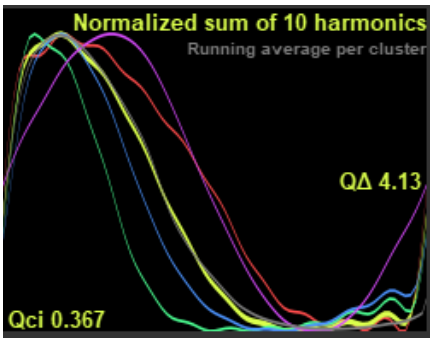
The graph to the right has three different display modes, as shown in Figure 7. By clicking on this graph, you can toggle between a bar graph, a single clustered waveform or all clustered waveforms. The horizontal scroll bar, too, selects one or all clusters.



a) Bar graph mode, with EGG cycle counts, per cluster. The number at top left is the scale of the vertical axis, which rescales automatically. Click on one of the bars to display (b). Click near the top of the display to display (c).



b) EGG wave resynthesis display, with one waveshape selected. Time and amplitude are cycle-normalized. Vocal fold contact area increases upwards. Gray: a running average of recent pulses in this cluster. Green line: EEG pulse re-synthesized from one of the cluster centroids. The values of Q_{ci} and Q_{Δ} shown here are computed from this resynthesized waveform. Click on the graph to display (a).



c) EGG wave resynthesis display, with all waveshapes selected. Time and amplitude are cycle-normalized. Here, for example, the sinusoidal purple signal (no vocal fold contact) is actually much weaker than the others, while the red one is the strongest. Click on the graph to display (a).

Figure 7. The display modes of the Cluster panel.

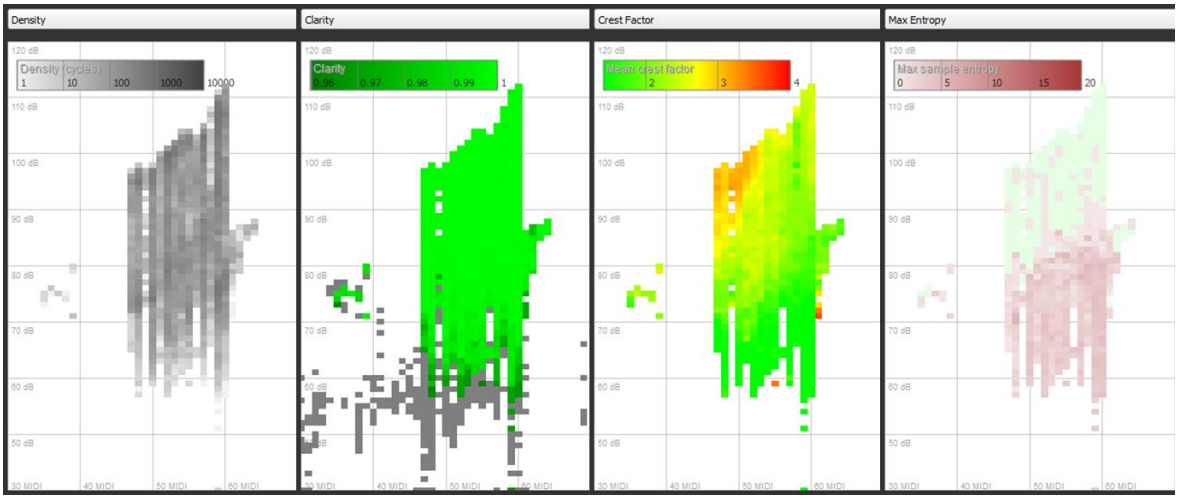
The rippling which can be seen in the resynthesized (colour) curves is an artefact of truncating the spectrum after a few harmonics. It is known as Gibb's phenomenon, and it is not a property of the EGG signal.

Tip: you can press Alt+c to hide or show the Clusters panel.

3.1.3 The Voice Map panel

The Voice Map panel can be switched to show one of several acoustic and EGG metrics (Figure 8, Figure 9), or the prevalence of each or all clusters (Figure 10). The horizontal axis is the f_0 in semitones (57 MIDI = 220 Hz); or, right-click to display the f_0 axis in Hz. The vertical axis is the sound level in dB. This should be calibrated to correspond to SPL @ 0.3 m microphone distance (→3.2.4). Metrics (a), (b) and (c) are derived from the audio signal, while the rest are derived from the EGG signal.

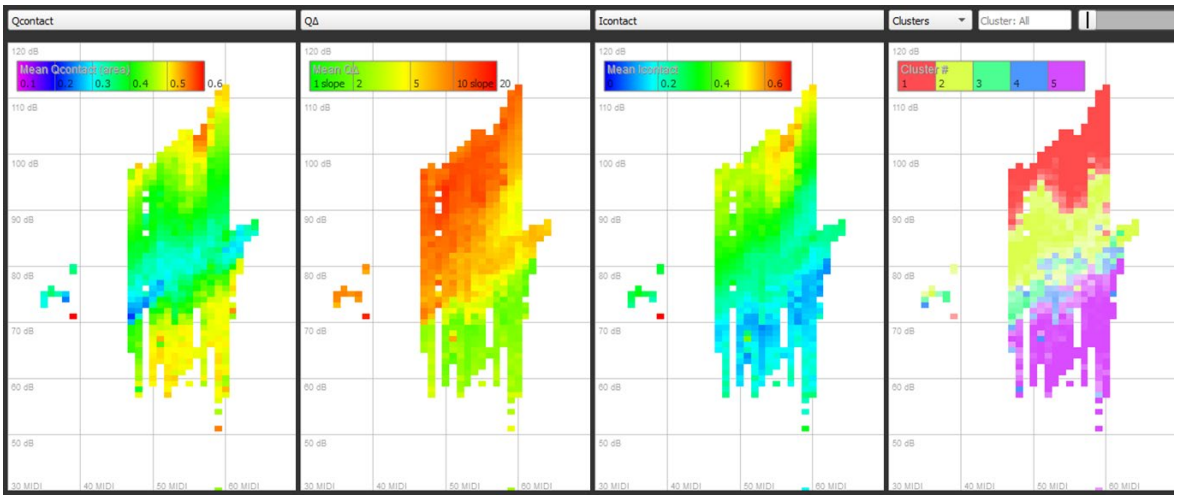
For the examples in *Figure 10*, an amateur male singer repeated soft-loud-soft /a/ vowels on several constant pitches over more than an octave. This recording took about 6 minutes. (a) 'Density', where darkest gray means >10000 EGG cycles. (b) 'Clarity' showing accepted cycles (green) and rejected cycles (gray). (c) The mean of the crest factor (peak-to-RMS ratio) of the audio signal, where red means 4 (corresponding to 12 dB). (d) Maximum SampEn; more brown means less stable phonation, pale green means perfectly stable phonation.



(a) Density (b) Clarity (c) Crest factor (d) Sample Entropy

Figure 8. The first four layers of the Voice Map display.

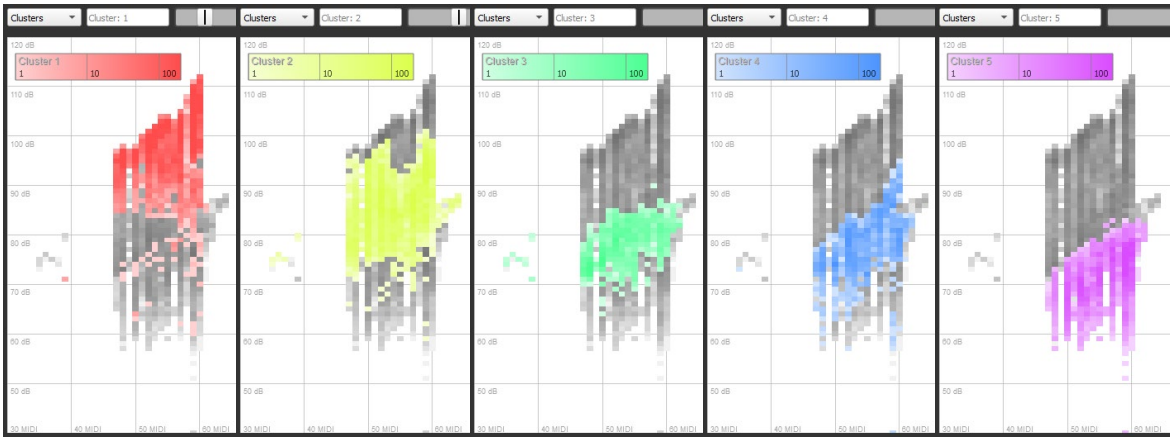
Figure 9 shows the layers for the three time-domain metrics, averaged per cell, plus the clusters map. (e) The contact quotient by integration, Q_{ci}; (f) the normalized peak dEGG Q_Δ; (g) the index of contacting I_c ; and (h) the dominant EGG waveshape cluster by colour. The 'dominant' cluster is the one with the most cycles in a given cell. Less colour saturation (whiter colour) signifies more overlap. Interestingly, we see that the vocal folds can vibrate without contacting at SPLs as high as 70-80 dB; as shown by the blue area in (g) or the purple area in (h). For this example, the wave shapes that correspond to each colour in (h) can be seen in Figure 7c.



(e) Quotient of contact (f) norm. peak dEGG (g) Index of contacting (h) Dominant clusters

Figure 9. Layers 5…8 of the Voice Map display

Figure 9h can be further layered by the individual clusters, as seen in Figure 10: (i)-(m) Actual extents of individual cluster waveshapes. The Density plot from (a) is in the background. (i) 'red' cluster waveshapes, strongest phonation; (j) 'yellow' cluster waveshapes; here, firm phonation with full vocal fold contact; (k), (l) 'green' and 'blue' cluster waveshapes; here, fairly soft phonation with brief contact; (m) 'purple' cluster: softest phonation with no vocal fold contact, i.e., a nearly sinusoidal EGG.



(i) (j) (k) (l) (m)

Figure 10. The Voice Map display of the cluster extents (layers 9…9+N)

All the data shown in the above three figures can be saved to a single .CSV text file for further analysis in other software (→3.4). Press **Save Map** to open a standard File Save dialog box. If the filename you enter there does not end in `.csv`, then FonaDyn will append `_VRP.csv` to the name. If the filename you enter *does* end in `.csv`, then FonaDyn will not change it.

You can load a previously saved map by pressing **Load Map**. This is useful for inspecting maps from earlier recordings. It also lets you accumulate more data into an existing map: just check the **Keep data** box before starting. To accumulate more

data, you must also continue to use the same cluster data as were used to make the loaded map, or the clustering/classification results will be meaningless.

When you press **Save Image**, a partial screen dump is generated and shown in a preview window of its own. When you *close* this window, a Save File dialog appears. Also, a list of available image file formats is printed in the Post window of the SCIDE. Choose one, and *type it in as the extension* to the image filename that you specify. Or, you can leave the preview window open, for easy visual comparison with your *next* voice map.

Tip: you can press Alt+v at any time to hide or show the Voice Map panel.

3.1.4 The Plots panel

The Plots panel can show one or more time series. The display scrolls, showing the most recent two seconds by default. You can change the duration of the plot (1 to 10 seconds) by dragging the grid sideways with the mouse, even when running. Durations longer than 2 seconds need to be set before FonaDyn is started, so that enough memory can be reserved.

You can choose from the three time-domain EGG metrics (→2.5), the audio crest factor, and the sample entropy of the harmonic-domain data. Every fleck in the plot represents data from one EGG cycle. Whenever the 'clarity' metric of the audio signal drops under the threshold, there will be gaps in these curves.

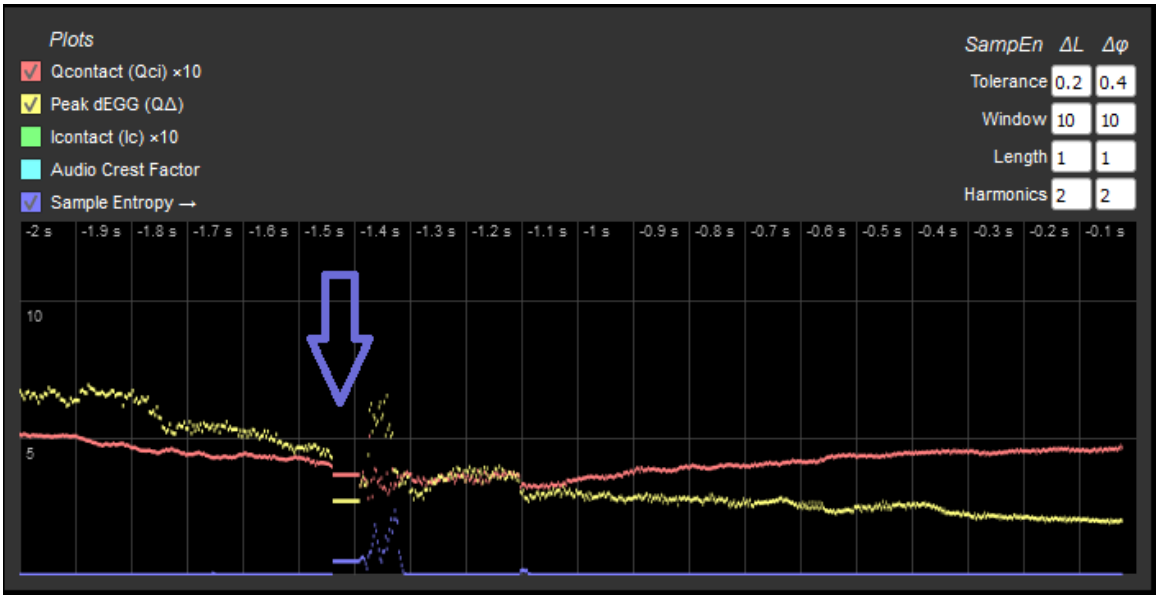


Figure 11. The Plots panel, selected using Show: One Graph. Here the Sample Entropy (the indigo curve) was based on ΔL_2, ΔL_3, $\Delta\varphi_2$ and $\Delta\varphi_3$. The figure shows a short excerpt from an upward glissando by a male subject. The arrow (added here) indicates a voice break from modal into falsetto voice. To change the displayed duration, drag the mouse sideways.

In the Plots panel, the Sample Entropy controls at top right appear only when the Sample Entropy box is checked. The SampEn metric is computed from the cycle-time series of the EGG harmonic magnitudes and phases. By default, only the first two harmonics (i.e. 2 and 3, relative to the fundamental) are used. The value shown by the scrolling indigo curve is the sum of those four SampEn values. The default values for Tolerance, Window length and sequence Length (the last two in EGG cycles) have been found by trial and error to work fairly well, but they may need to be adjusted in different scenarios.

Increasing the Tolerance suppresses the influence of small variations. Increasing the Window makes a smoother curve and reduces the temporal resolution. Increasing the Length can make the SampEn peaks more localized.

You can save the entire time history of the these and all other metrics into a multichannel Log file, cycle by cycle, or at an isochronous frame rate. For making and saving customized time-series plots of any metric in FonaDyn, create an Analysis Log file, and then use Matlab or some other software to customize your graphs from that. The Matlab function `FonaDynPlotLogFile` demonstrates how to do this.

Tip: you can press Alt+p at any time to hide or show the Plots panel.

3.1.5 The 'Moving EGG' panel

This panel displays the incoming EGG cycles in real time. The period length is normalized to the width of the display frame. By default, the peak-to-peak amplitude is normalized to the height of the display frame. To see the actual EGG amplitude, relative to full scale, choose **Normalize: Off**. This is useful for checking the amplitude of the conditioned input signal.

The display draws the n=5 most recent cycles with a fading gray scale. The cycle curves are rendered as k=80 straight line segments. To modify this rendering, press **Settings...** (→3.1.6) and check the box **Show additional diagnostic features**. This displays extra control fields. Change n by entering a different value for **Count**. Change k by entering a different value for **Samples**. Increasing Count or Samples may give a nicer image, but also increases the processing load.

The rightmost symbol at the top indicates the type of cycle segmentation (→2.4.2), with Φ for the phase tracker, and Λ for the peak follower. You can change this option, too, in the **Settings...** dialog box.

Tip: you can press Alt+m to hide or show the Moving EGG panel.

3.1.6 The Settings… box

To reduce screen clutter, some diverse settings that are infrequently used have been placed in a separate dialog box.

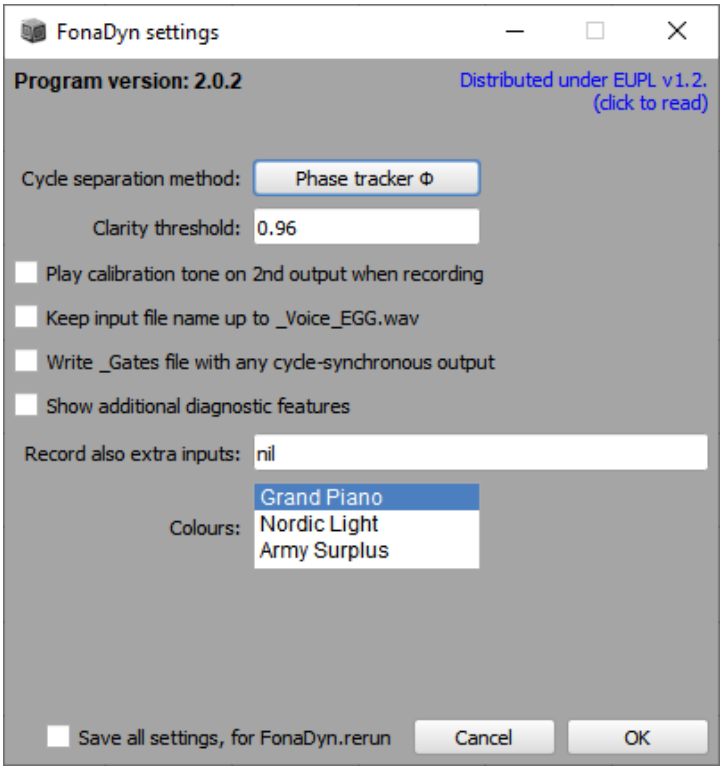


Figure 12. The Settings… box.

Click the blue text to open the license text from the Internet.

Cycle separation method selects Phase tracker or Peak follower →2.4.2.

Clarity threshold allows you to adjust the threshold for cycle regularity (→2.3.3). This refers to the regularity of the *audio* signal, not the EGG signal. With soft or breathy voices you may want to lower this. You can use the Clarity layer in the voice map to see which phonations are rejected from analysis.

Play calibration tone →3.2.4.

Keep input file name →3.4.

Write _Gates file →3.4.1.

Record also extra inputs →3.2.7.

Colours selects one of the available colour schemes. Since a black background often works poorly in print, you may prefer the 'Nordic Light' colour scheme for screen-dump illustrations that will be printed.

Save all settings If checked, *all* settings, not just the ones in this dialog box, are saved when FonaDyn is closed. To resume with those saved settings in the next session, start it using `FonaDyn.rerun` instead of `FonaDyn.run` (→3.4.3).

3.2 Recording

3.2.1 Recording environment

Use a quiet room. FonaDyn uses the 'clarity' metric as a criterion for acceptance, which means that the EGG analysis gate will be opened whenever the *audio* signal from the microphone is *sufficiently periodic*, even if it is very soft. The rationale for this design is that the signal-to-noise ratio of the audio channel is typically much better than that of an EGG device; and while a weak EGG signal is often electrically rather noisy, its lower harmonics can still be analyzed. It follows that any tonal sound in the recording room may open the EGG analysis gate, even something other than the subject's voice, including other voices, a piano, or machinery such as fans or buzzing light fixtures, etc.

The recording environment should follow the established recommendations [14] for VRP acquisition, with regard to ambient noise and room absorption. If the computer or any other source of tonal or soft fan noise must be present in the recording room, then the microphone should be of a cardioid type and be pointed exactly away from the noise source. If the subject needs prompting pitches, auditory stimuli, or a background audio track, these should be presented over closed circumaural headphones.

Headphones can also be useful for restoring some hearing-of-self in anechoic or very dampened recording rooms. Some audio interfaces even have a built-in reverb effects unit, which can help, if applied in moderation. Just be sure that the reverb is not routed into the signal that is being recorded.

3.2.2 Normal setup

By default, the two output channels of your audio interface play a copy of what the microphone is picking up. In this way, you can monitor the incoming audio signal on headphones, and check that it is free of noise and hum. You may wish to connect a loudspeaker instead, for listening to earlier recordings together with other people.

The button **Playback/Echo** turns the audio output On or Off. If you have a loudspeaker connected, in the same room as the subject, it can be useful to choose Off prior to recording, to prevent feedback. This must be done before pressing ▸ START. Or, monitor with headphones.

Processing and display are started when you press ▸ START, but nothing is saved to disk, by default. This is useful for checking levels, and for working interactively with real-time feedback.

3.2.3 Checking the EGG level

Connect the EGG device and strap on the electrodes to the subject's neck. Turn on the EGG device and ask the subject to phonate. In the oscilloscope panel, choose **Normalize: Off**. Press ▸ START, and wait for that button to display ■ STOP. Adjust the output level of the EGG device such that you get a clearly visible signal. Try also adjusting the electrode position for maximum signal. Ideally, the displayed curve should exercise the major part of the vertical axis, for the strongest signals.

FonaDyn monitors the EGG signal amplitude during recording. If the usable range is exceeded, the words **EGG CLIPPING** are flashed for about a second. This typically happens not so much for strong phonation but more often if the larynx moves a lot. It is usually the low-frequency drifting that pushes the EGG signal out-of-range of the A/D converter. If so, reduce the output level of your EGG device. If necessary, insert a signal attenuator between the EGG device and the input to your audio interface (sound card).

Once the EGG amplitude appears to be satisfactory, press ■ STOP, and wait until that button again displays ▸ START. Restore **Normalize:** to **On**.

3.2.4 SPL calibration

In order for the SPL axis in the voice map to be correct, the gain of the microphone amplifier has to be calibrated prior to recording. The procedure is different depending on if you are using a head-mounted boom microphone at close range (A), or a stand-mounted microphone at 30 cm in front of the subject's mouth (B). To do this, you will need a hand-held sound level meter, and, for (B), an active loudspeaker (i.e., with a built-in amplifier).

Turn on the sound level meter, and select a flat frequency response or the C-weighting charactistic. Do not use the A-weighting characteristic. If available, select "slow" response rather than "fast".

3.2.4.1 (A) Head-mounted boom microphone

Because head-mounted microphones are very close to the mouth, their pick-up is also very sensitive to distance. You will need to use the subject's own voice as a sound source, and you will need to make a new calibration every time the boom microphone is adjusted or hung anew on the subject.

- Have the subject wear the microphone boom and adjust the capsule so that it is about 7 cm from the center of the mouth.
- Place the sound level meter so that its tip is 30 cm in front of the subject's mouth, and orient the meter so that both the subject and you can read it while you are sitting at the computer.
- Switch the sound level meter to Slow and to C-weighting.
- Select **Source: Record**.
- Select **Playback/Echo: Ready**.
- Press ▸ **START**.
- Ask the subject to sustain a vowel at a steady level around 80 dB. This may take some prior practice.
- On your audio interface, or in its control panel app, adjust the input gain for the mic signal such that the moving cursor on FonaDyn's voice map display sits at the same SPL as the value displayed on the level meter (to within the nearest decibel). The microphone gain is now calibrated. If you are recording the calibration, which is wise, also read the level value out aloud into the microphone. Then press ■ STOP, and make a note of the file name of the new calibration file.

Tip: to read the level meter more easily at the same time as the cursor on the voice map display, you can set up a webcam aimed at the level meter and use a webcam app such as AMCap to display the image of the level meter on top of the voice map.

- Make a note of the calibrated gain setting. Also, the control software for your audio interface might have an option for saving its settings.

3.2.4.2 (B) Stand-mounted microphone

For convenience, FonaDyn provides a built-in calibration tone which gives better accuracy in the calibration. If you want to use it, route the *second* output channel of your audio interface to a loudspeaker. Place that loudspeaker about one meter from the microphone that will be recording the subject. The exact distance is not critical. If you prefer, you can use any other constant tone generator, set to 200-300 Hz.

- Place the microphone on its stand, at 30 cm in front of the subject's mouth. A fixed head rest can be useful to help the subject stay at that distance. An error in distance of 10% incurs about 1 dB of error in the level reading.
- Switch the sound level meter to Slow and to C-weighting.
- Click the button **Settings...** . In the box that appears, check the option "**Play calibration tone on 2nd output when recording**", and choose OK.
- Select **Source: Record**.
- Select **Playback/Echo: Ready**.
- Press ▸**START**. FonaDyn will now output a 250 Hz tone to the loudspeaker. Hold the tip of the sound level meter close to the microphone (within two centimeters or less). Adjust the volume of the loudspeaker so that the level meter shows about 80 dB. The actual value is not important.
- On your audio interface, or on its software control panel, adjust the input gain for the mic signal such that the moving cursor on FonaDyn's voice map display sits at the same SPL as the value displayed on the level meter (to within the nearest decibel). The microphone gain is now calibrated. If you are recording the calibration, also read the level value out aloud into the microphone. Then press ■STOP, and make a note of the file name of the new calibration file.
- Make a note of the calibrated gain setting. Also, the software for your audio interface might have an option for saving its settings.
- In the **Settings...** box, turn off the calibration tone.

3.2.5 Recording live signals

- In the top panel of FonaDyn, check that the **Output Directory** is the one where you want new recordings to be stored. If it is not, use the adjacent **Browse** button to select another directory.
- In the **Source** list, select **Record**.
- Select **Record: Ready**. An empty text field appears, where the name of the new file will be shown. You cannot choose the file name yourself, even though this field is editable. FonaDyn creates a file name based on the current date and time (see also section 3.2.8). The timestamp makes the filename unique.
- Press ▸START. After a moment, the Record button becomes bright red to indicate that **Recording** is in progress.
- Have the subject perform the phonatory production procedure.
- Press ■STOP. After a moment (possibly quite a few seconds), the Record button returns to the dull red **Ready** state**.**
- FonaDyn has now created the file and given it a name based on the current date and time. You can copy the resulting file name of the `_Voice_EGG.wav` file from the panel, and paste it into your log of the experiment, with a comment on what was recorded. You can of course also rename the file afterwards, if you wish; but let the new name end in `_Voice_EGG.wav`, if possible, because it can simplify the post-analysis with FonaDyn.
- When done, select **Record: Off**. It is easy to forget it in the **Ready** state, in which case new files will consume your disk space whenever you press ▸START.

3.2.6 Re-recording during analysis

Even when you select a file for analysis rather than the live inputs, it is possible to re-record the input file. This would seem rather pointless, but for two things. First, the re-recorded file will receive a new time-stamped file name, with the same time stamp as any other output files that you may be generating in the same run. This may make it easier to track different analyses of the same file. Second, the re-recorded file contains not the raw input EGG and voice signals, but rather the preconditioned signals, which allows you to inspect the signals as they are after conditioning, if you wish. The EGG signal will be about 34 ms delayed relative to the audio track. To remind you that you are re-recording, the Record button turns **orange** rather than **red**.

3.2.7 Recording additional signals in parallel

FonaDyn itself normally records a stereo WAV file only, with voice and EGG. If you want to record additional *audio-rate* signals in synchrony with voice and EGG, a way of doing that is to record using instead some digital audio workstation (DAW) software such as Reaper, ProTools, Logic or other multitracking programs. You can then export the voice+EGG signals to a stereo WAV file that FonaDyn can analyze, and export the other tracks to synchronized files in the format of your choice.

Sometimes, though, one may wish to record also *slow physiological signals* such as subglottal pressure, larynx height, or breathing-related signals. These cannot be

recorded by audio interfaces, because the latter block DC, and AC signals below about 20 Hz. However, some audio interfaces offer so-called ADAT optical connections, each of which adds another 8 inputs or outputs. Enthusiasts in the music community for analog synthesizers have developed DC-coupled ADAT-linked converters for slow control voltages [15]. FonaDyn can acquire such signals, downsampling them to a fixed frame rate of 100 Hz.[1] No anti-aliasing is performed, so the signals should be band-limited to <50 Hz.

Be aware that such 'consumer' devices are not generally certified for safe use with body-contact transducers. Although malfunction is rare, a precaution would be to power them from batteries rather than from the mains.

To activate this feature, first activate the required number of hardware inputs in the SuperCollider startup file, and restart SCLANG. In FonaDyn, open the **Settings** dialog. In the text field **Record also extra inputs**, type a list of the inputs to which you have connected slow signals. The list must consist of comma-separated integers, and must be enclosed in square brackets, like so: **[10,11,12,13,14]** . The inputs must refer to active hardware inputs, where the first two (usually **[0, 1]**) are reserved for the microphone and the EGG. The inputs do not have to be contiguous nor in any particular order. The order that you specify is the order in which the signal tracks will appear in the output file.

The output file will be a multichannel WAV file of 16-bit integer data, with as many interleaved channels as you have specified in the list. The signals will be synchronized with those in the _Voice_EGG file. The filename will have the same timestamp as the _Voice_EGG file, followed by "`_Extra.wav`". That file will report its sampling rate as being 44100 Hz, but this is not correct; its actual sampling rate per channel is 100 Hz.

To turn off the recording of these extra channels, clear the text field **Record also extra inputs**.

3.2.8 Notes on using FonaDyn with RME audio interfaces

The digital audio interfaces from RME are very versatile and ideal for multichannel acquisition. The models Fireface 400, UC and UCX, as well as the Babyface all have their microphone preamps on inputs 1 and 2. This is what FonaDyn expects. The larger models Fireface 802, UFX and UFX+, however, all have their microphone preamps on inputs 9-12; while those on the model 800 are on inputs 7-10. This requires a little extra configuration, which can be done in RME's TotalMix software. It is described in the separate document *Using FonaDyn with RME audio interfaces*.

3.2.9 Using FonaDyn without an EGG device

If you are interested only in the voice maps as such, and do not have an EGG device at hand, you still need to feed an "EGG-like" signal to the second input. One way is just to branch a line-level copy of the microphone signal to the second input,

[1] There is no user interface for the frequency 100 Hz, but it can be changed by editing on line 252 of the file FonaDyn/classes/SynthDefs/VRPSDIO.sc.

preferably with a lot of low-pass filtering. The cluster displays will probably make little sense, and the voice map display may seem a little jerky. The displays of Density, Clarity and Crest factor will be correct, though. Or, if you are curious, you can test what FonaDyn makes of an accelerometer signal or a photoglottographic signal, instead of the EGG. We haven't yet had time to do that.

3.3 File locations

The **Browse** buttons open a file system dialog for opening or saving files. When FonaDyn is started, the suggested directory will be the default directory for recordings as shown in the top line. Subsequent uses of **Load** or **Save** of files will start in the directory you last used, for *that kind of file*. Note also that the Open File dialog usually has at the top a drop-down list of recent locations, which can speed up your navigation of the file system.

You can also specify a persistent default directory for recordings, which is useful, because the standard location is rarely the one you want. To do so, add the following line in the SC startup file, with your desired location in double quotes:

```
thisProcess.platform.recordingsDir_("C:/Recordings");
```

In SC code such as this, always use *forward* slashes (/) in pathnames, even on Windows. Once you have restarted SuperCollider, FonaDyn will display this path in the field **Output Directory**, and recordings as well as log files will be saved in that directory.

3.4 File names and file formats

FonaDyn can export and import several types of data file that can be used for mathematical processing and display with Matlab and other software. Some Matlab examples are provided in the 'FonaDyn Extras' folder. The signal file formats are documented also in the online documentation for the class `VRPViewMainMenuOutput`.

When running, FonaDyn processes either live inputs or an existing recording. By default, the output files that contain *signals* are automatically given a file name beginning with a time stamp YYMMDD_HHMMSS_ (shown as * below). The time is that at the start of *recording*; or, if analyzing a file, from the starting time of the *analysis* (not the time-stamp of the analyzed recording – that would risk confusion if the same file were analyzed repeatedly with different settings). All signal output file types are optional, and all can be written to simultaneously. To the time stamp is appended a string suffix indicating the type of the output file.

If instead you prefer the output files to receive the same base name as the input file, go to the **Settings...** box and check that option. The output files will then inherit the original time stamp (or other name), which makes it easier to see which files are derived from which input signals. This will work only for input files whose names end in "`_Voice_EGG.wav`".

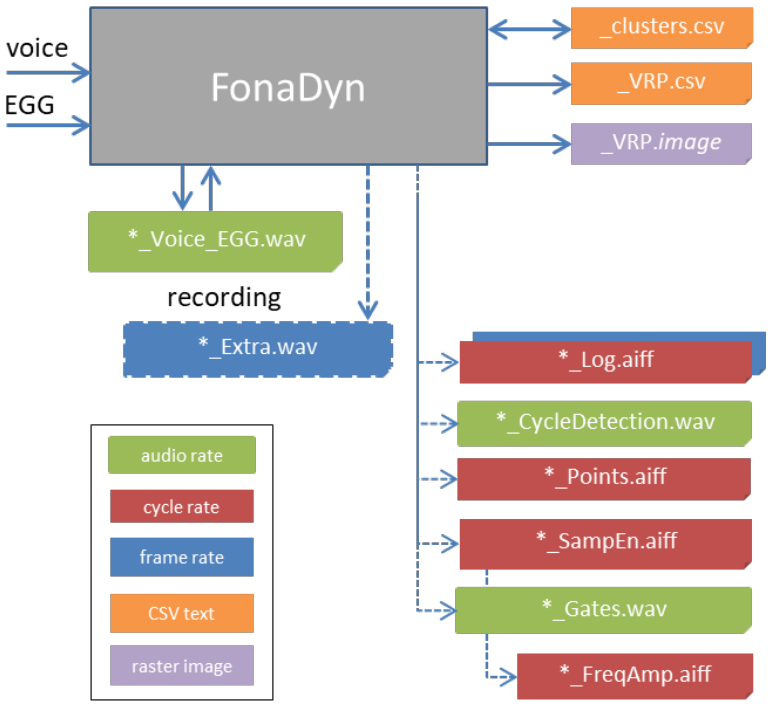


Figure 13. Summary of file types handled by FonaDyn.

Recordings are made into 2-channel files at 44.1 kHz per channel and 16 bits resolution, named `*_Voice_EGG.wav` .

Analysis Log. This is a multichannel file called `*_Log.aiff` . It contains one frame of data for every EGG cycle. Each frame has these tracks: time (s), f_0 (semitones), signal level (dBFS), clarity (0...1), crest factor (dB), SampEn, I_c, Q_{ci}, Q_Δ, cluster number, and the levels (Bels) and phases (radians) of all analyzed harmonics of the EGG signal. The levels are relative to 0 dB full scale. The phases are absolute, i.e., relative not to the fundamental but to the cycle detection trigger point. The frame rate (or "sampling rate") in this file is selectable as cycle-synchronous, with new data for every EGG cycle, or one of 50, 100 or 200 Hz. Frames of accepted cycles only (above the clarity threshold) are written to this file. That is why the time track is helpful; without it, the absolute times of each cycle would be harder to reconstruct (see below). The data are stored as 32-bit float values. These values are *not* scaled down to ±1.0, as is the convention for normal audio files. Therefore, the signals in many of the tracks will appear to be out-of-range, if a `*_Log.aiff` file is opened in a multitrack audio editor.

If a file of this type is opened in other software, its sampling rate will be appear to be 44100 Hz, but this is not correct. The effective 'sampling rate' is the f_0 of the analyzed signal.

Physiological signals can be saved synchronously into multichannel WAV files at 100 Hz per channel and 16 bits resolution, named `*_Extra.wav` . (→3.2.7).

3.4.1 Diagnostic output files

Buttons to create these files are shown only if you have checked **Show additional diagnostic features** in the **Settings**. More detailed info can be found in the online help for the class `VRPViewMainMenuOutput`.

Cycle Detection Log. This is a two-channel, 16-bit file (`*_CycleDetection.wav`), containing the conditioned EGG signal, and the corresponding cycle trigger pulses. This file affords post-inspection of whether or not the cycle triggering was accurate.

Output Points. This file type (`*_Points.aiff`) contains twice as many tracks as there are harmonics. It contains the cycle-synchronous level and phase *differences*, for accepted cycles only. This is almost the data that is input to the clustering algorithm; see section 2.7 for details. Not scaled. The last delta-level track contains the relative level of the residual high frequency energy. The last delta-phase track contains *twice* the absolute phase of the fundamental (for internal use).

Sample Entropy. This is a single-channel, cycle-synchronous file `*_SampEn.aiff`. Not scaled.

Frequency and Amplitude. This file type does not have any on/off button, but such a file is always written when either the SampEn measurement is written, or the Points are written. The reason is that you often want to see these metrics together. The file is a cycle-synchronous AIFF file with floats as samples. These files are called `*_FreqAmp.aiff`.

Gates Log. This file is called `*_Gates.wav`. It contains five tracks of audio-rate 16-bit samples, with the raw EGG, the conditioned EGG (delayed by 34.1 ms), and three trigger tracks. It shows exactly where all conditioned EGG pulses were segmented and also which of the pulses that were regular enough to be retained for further analysis. This enables close scrutiny of the data input to the DFT analysis, as well as the extraction (e.g., in Matlab) of cycle-by-cycle signal data with absolute times. Because it becomes big, this file is written only if you have checked that option in the **Settings...** box, and only if there is some other simultaneous cycle-rate output being written.

3.4.2 Result files

Cluster data and voice map data are saved as **text files**. Their file names are *not* given an automatic time stamp; rather, you must choose a full name yourself. You will probably want to encode the relevant settings into the file name in your own way. The same applies for image files.

Cluster data. When an analysis is completed, the centroids of the resulting clusters can be saved and then reloaded. Typically, you would do this to continue learning with more input files, or to classify other signals using the same cluster data, or to use the cluster data in analyses outside of FonaDyn. Figure 14 shows an example of a `*_clusters.csv` file opened in a spreadsheet, with added comments. Matlab examples are provided that show how to resynthesize EGG cycle shapes from this data.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	4277	-1.1125	-1.887	-2.354	-3.8626	0.4308	0.1371	0.1646	0.0003	-0.7807	-0.6472	-0.4933	-0.0009
2	8305	-0.6012	-1.3393	-1.8837	-2.3313	0.6181	0.1997	0.2316	0.0006	-0.7734	-0.9389	-0.858	-0.0008
3	8931	-2.905	-3.8167	-4.3397	-3.625	-0.2025	0.1015	0.1032	5E-05	-0.3321	0.3709	-0.007	-0.001
4	19605	-1.0258	-1.5655	-1.9074	-1.9629	0.7232	0.8096	0.7966	0.0003	-0.6728	-0.5271	-0.5659	-0.0009
5	22124	-1.2888	-1.5077	-1.8943	-1.8424	0.9706	0.9553	0.9772	2E-05	-0.0479	-0.2532	0.0565	-0.001
6													
7	*Column*	*Contents*											
8	A1:A5	Cycle counts in clusters 1..5											
9		*Centroids of*											
10	B:D	relative levels of harmonics 2..4 (in Bels)											
11	E	relative level of higher harmonics (in Bels)											
12	F:H	cosines of relative phases of harmonics 2..4											
13	I	cosine of the phase of the fundamental * 0.001											
14	J:L	sines of the relative phases of harmonics 2..4											
15	M	sine of the phase of the fundamental * 0.001											

*Figure 14. Example of cluster data file, opened in a spreadsheet. This example is for 5 clusters and 4 harmonics. The rows 7-15 and the colouring are explanatory only – they do not appear in the *_clusters.csv file.*

Voice map data. When an analysis is completed, the data underlying the voice map can be saved to a CSV file (press **Save Map**). An example of a `*_VRP.csv` file is shown in Figure 15. Each line corresponds to an occupied cell in the voice map graph. Matlab examples are provided that show how to plot voice map charts from this data.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	MIDI	dB	Total	Clarity	Crest	Entropy	dEGGmax	Qcontact	Icontact	maxClust	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
294	76	96	121	0.9994	1.731	0.46353	2.443441	0.341705	0.1291	4	46	6	8	61	0
295	77	96	159	0.9996	1.863	0.23659	2.3941	0.303576	0.1136	4	11	9	26	111	2
296	66	97	27	0.9901	2.299	0	4.883369	0.391201	0.26929	3	2	0	25	0	0
297	67	97	253	0.9979	2.233	0	4.763272	0.401452	0.27198	3	0	0	242	0	11
298	68	97	135	0.9975	1.751	0	4.051132	0.293234	0.17734	2	0	115	1	19	0
299	69	97	167	0.9976	1.917	0	4.286901	0.291203	0.18236	2	32	124	2	6	3
300	70	97	314	0.998	2.05	0	4.320231	0.343382	0.21809	5	3	26	22	2	261

Figure 15. An example of a _VRP.csv file from version 2.0, opened in a spreadsheet. Each row holds data for one cell in the voice map. Columns A and B give the cell coordinates, column C the total number of cycles in the cell, D the most recent value of the clarity metric, E the average level of the crest factor of the audio signal, F the maximum value seen of the SampEn metric, G, H and I the means of the Q_Δ (a.k.a. dEGGmax), Q_{ci} and I_c metrics, I the number of the cluster with the largest number of cycles, J…N the cycle counts per cluster. Only non-empty cells are included.

The cycle counts in this file are absolute, that is, they are not scaled by f_0 . One second of phonation results in 100 cycles at 100 Hz, but 400 cycles at 400 Hz. You can obtain the approximate phonated time by dividing these cycle counts by the corresponding f_0 , where

$$f_o = 220 \cdot 2^{\left(\frac{MIDI-57}{12}\right)}$$

In these voice map files, f_0 is quantized to whole semitones (≈6% increments), so the maximum error in the duration for one cell would be ±3% . (In the files `*_Log.aiff`, MIDI(f_0) is given with a decimal fraction to a high resolution.)

As the **column separator** in CSV files, FonaDyn uses the semicolon (;). If this is inconvenient, you can change it in the source code file `VRPMain.sc`, although such a change is not recommended. It is not possible in FonaDyn to change the **decimal character**, which is always the period (.) .

In Windows, the Region settings in the Control Panel allow you to choose the decimal character, which is then applied system-wide by all apps that heed this setting (FonaDyn does not). In addition, Microsoft Excel allows you to specify its own decimal character, independently of the Control Panel setting. Not exactly simple, but workable.

3.4.3 Settings file

When FonaDyn is closed, all settings in the user interface are saved in a file with the name `<Platform.userAppSupportDir>/FonaDynSettings.SCarchive`. Only one set of settings is saved. If **Settings | Save all settings on exit** is checked, this file is overwritten each time the session is ended. If you start FonaDyn with `FonaDyn.run`, the default settings are used. If you start FonaDyn with `FonaDyn.rerun`, the settings most recently saved from a previous session are used.

If you want to save a set of settings for posterity, copy the above file to another location and give it another name. To reinstate those settings, copy the file back.

The settings file is in editable text, although its syntax is a bit obscure. With some knowledge of SClang, you can edit it, to manipulate FonaDyn's settings externally.

3.5 Analysis

This section applies to the analysis of both live and prerecorded signals.

3.5.1 Parameters for clustering

Ideally, choosing the *number of clusters* should be as easy as choosing the number of phonation types that you want FonaDyn to identify. To discriminate between modal and falsetto singing, for example, two clusters could suffice, in principle. But there is no way of knowing beforehand if the EGG waveform really does change in a way that is so cleanly separable. For example, you may find that you get better discrimination with three clusters, of which two are needed to catch all the falsetto-mode cycles [3].

Therefore, a good strategy may be first to specify more clusters than you actually need, and let FonaDyn try. The most common wave-shapes will tend to produce a few well-populated clusters, while the other clusters will contain much fewer cycles. For research work, it helps to edit your recordings first, so as to eliminate pauses. If your signal contains pauses or other non-voice events, these may give rise to one or two 'trash' clusters, holding weird pulse shapes. You can save and edit the cluster data files to remove such trash clusters. An example of this method is given in [16].

The *number of harmonics* will depend on your research question. If you want to *see* a close representation of the actual EGG pulse shapes, 10 harmonics are usually needed. If you think that the feature you are investigating is a low-frequency phenomenon, or if you are mostly interested in the SampEn metric, then fewer harmonics will usually be enough.

3.5.2 Clustering in multiple passes

Because FonaDyn is a real-time program, it can not know anything at the START about what the data set as a whole is going to look like. This applies to recordings as well as to live signals. FonaDyn therefore has to adapt the clusters to the data as it goes along. This means that, on the first pass, the cluster colours early in the recording will not correspond to quite the same wave-shapes as late in the recording. Over time, the clusters will become stable, but initially, it helps if the range of variation in the early part of the signal is representative of that in the rest. Therefore it is a good idea to try to include excerpts from the full range of conditions as early as possible in a recording, and also to re-run a second pass over the same signal, using the cluster data from the first pass for initialization, as explained below.

3.5.3 Initializing the clusters

The outcome of the clustering is very sensitive to the first few cycles that are detected. There is usually some quiet before the subject starts phonating, which may cause some spurious 'cycles' to be detected in the background noise. Then, some of the initial centroids will be too far apart in their $3N$-dimensional space to describe EGG waveforms, and most subsequent real EGG cycles will tend to be 'captured' by the one or two 'random' centroids that happen to best describe EGG signals. We are trying to find an automated solution to this.

For the time being, there are two ways of dealing with it. The first is to wait until the subject is producing stable phonation and then press the button **Reset counts**. From the EGG waveform at that moment, this will generate equal 'seed' centroids that soon drift apart from one another. The end result will depend slightly on exactly *when* **Reset counts** is pressed. The final outcome will however become quite stable if you record the signals and make a second or even a third pass over the data, using the setting **Init: Pre-learned** (without pressing **Reset counts** or **Unload** in between).

The other workaround, when analyzing pre-recorded files, is to edit out leading pauses from the signal, such that it starts immediately at medium voice amplitude. In this way, the analysis will start out 'seeded' with similar waveshapes in all clusters, which soon diverge as the signal changes.

3.5.4 Rearranging clusters

Another issue of initialization is that the algorithm's allocation of cluster numbers to different EGG waveshapes is necessarily arbitrary. Since each cluster number is assigned a different display colour, this means that the cluster *colours* are unlikely to be the same every time, when a recording protocol is repeated in the learning phase. Once the learning can be turned off, as for classifying signals, this is no longer a problem.

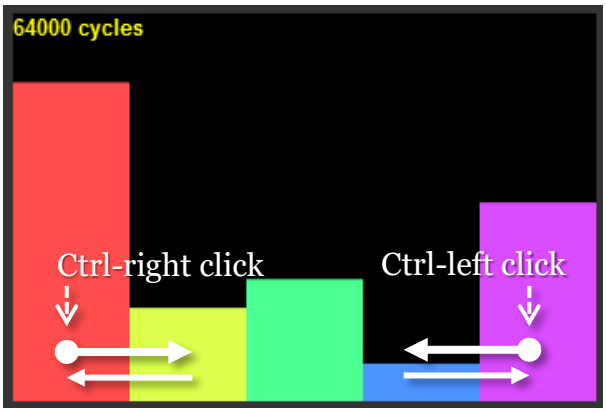


Figure 16.
The sequence of colours is given by the number of clusters. You can rearrange the cluster mapping to colours by swapping the positions of two clusters at a time. Since this is a swap, repeating a click in the same place will undo the change.

To make your figures consistent, you may wish to re-arrange the cluster order. This can be done only when FonaDyn has stopped. *Ctrl*-click left or right on the cycle-count columns (Figure 16). Two adjacent clusters are then swapped, as shown here. Repeat this until you have achieved the desired order. The first and last clusters are also treated as adjacent; the order 'wraps around.' The cluster colors in the current voice map are also updated. Cluster data are saved to files of type `*_clusters.csv`.

Another way to rearrange clusters is to edit the .csv file directly. Use a spreadsheet program to rearrange the data clusters into the order that suits your application. A tip: first, colour the text on each *row* to a sequence of colours similar to those shown in FonaDyn's bar graph (Figure 16). It is then somewhat easier to rearrange manually those rows into the order that you want (for instance, from weak to strong phonation). With this method, you can also *remove* a cluster that you don't

want, simply by deleting that line. When done, save the `_clusters.csv` file, and reload it into FonaDyn.

3.5.5 Detecting specific EGG wave shapes

Editing `_clusters.csv` files also comes in handy for constructing cluster files that capture rare but interesting pulse shapes. If you want FonaDyn to look for a specific EGG wave shape, proceed as follows. Using an audio editor, or some numerical procedure, create a `*_Voice_EGG.wav` file that contains mostly pulses of that waveshape. Train FonaDyn on that waveshape, so that it looks right when resynthesized, and then save the `*_clusters.csv` file. Use a spreadsheet application or flat text editor to copy the row for the corresponding centroid, and append that row into another existing clusters file. Save the new spreadsheet as a new `*_clusters.csv` file. Repeat this for several different waveshapes, if you like. Then load your new clusters file for classifying signals, with **Init: Pre-learned** and **Learning: off**. Adjust the voice map display to show only the cluster of your rare waveshape. When it occurs in the input signal, it will appear on the voice map. FonaDyn can manage up to 20 clusters (wave shapes) in one file.

3.5.6 Analyzing from multiple takes or multiple files

By default, FonaDyn clears the current cluster data and the voice map data when you press START. If you want instead to continue accumulating recordings into the current voice map, check the box **Keep data** that is to the left of the START button. The **Keep data** option could be appropriate for a long acquisition protocol, during which a subject needs to pause, or for analysing a batch of multiple input files for which the results should be combined into one voice map. If you are analyzing a batch of files, you will want the data to be cleared on START, but not for the following files. If so, then defer checking **Keep data** until after you have pressed START.

In most cases, you will also want to choose the clustering setting **Init: Pre-learned**. Choose **Learning: On** to continue the same clustering operation across input takes or files. Choose **Learning: Off** to classify all the input files using one fixed clustering (which must first be **Load**ed).

The **Load Map** button lets you load a previously saved voice map for inspection. Then, check the box **Keep data** before starting, and you can continue accumulating voice recordings into the existing map. The cluster data must be consistent with the existing map, or the results will be meaningless; use **Save Cluster/Load Cluster** to ensure it.

3.5.7 SampEn parameters

The SampEn analysis is still somewhat experimental, so we have given you access to all the settings, to try out. It has three parameters: Tolerance, Window and Length, which can be set to different values for the harmonic levels and the phases (→2.8).

3.6 Known problems

FonaDyn version 2.0.2 has the following known issues.

1) **CPU load**. The higher the fundamental frequency, the more EGG cycles have to be handled every second. In the high soprano range, FonaDyn may struggle noticeably to keep up, unless the computer is quite fast. The display may update less smoothly, in fits and starts.
2) Occasionally, with file input and/or output, FonaDyn will **fail to start or stop** properly. This can be due to a format error in the input file (did you choose the right file?), but also to inter-process handshaking errors, or network delays. Working with all files on a local hard disk is usually best. In the stopped state, by design, the START button displays "START". In the running state, it displays "STOP".
3) After a successful but longish run, it may take up to a minute for FonaDyn to go from "Stopping…" to the really stopped state – just be patient. If the START button refuses to re-appear, close the main window, and invoke `FonaDyn.run` once more, perhaps also with action (4) below. Before you do that, it may be possible to save cluster data and voice map data, even when FonaDyn is in a waiting state.
4) It may happen that the **server process is orphaned**, that is, SCSYNTH loses its connection with SCLANG. The solution is go to the SCIDE window and invoke the command **Kill All Servers** from the Server menu (or evaluate `Server.killAll`), and then start again.
5) Specifying the highest numbers of clusters and harmonics (up to 20, 20) may cause SCSYNTH to complain that the connection diagram becomes too complex. (To appreciate this, you might look at the file FonaDyn-code-diagrams.pdf.) If you really want that many, this can be allowed by saving into the SC startup file the line

   ```
   Server.local.options.numWireBufs = 256;  // the default is 64
   ```

   Now, restart SuperCollider, and you will probably be able to run up to the maximum. The rainbow graphs are pretty, but remember that with 20 harmonics, you are limited to f_0 values of less than 10000/20 = 500 Hz (section 2.4.1).
6) On some Windows 10 computers under centralized management, SuperCollider has the following known problem: the SCIDE may fail to start up properly on some computers. Its Help docklet displays "Sending request…" indefinitely, and the language interpreter does not become operative. The solution is to deactivate the policy "Audit removable storage", although no-one knows why. Only your system administrator can deactivate policies.
7) On some Windows computers, the first time you run FonaDyn, the server process SCSYNTH in SuperCollider can take a long time to start (one or two minutes). You have to wait until, in SCIDE, the numbers at the bottom right turn green, which means that the server is up and running. On subsequent starts, there is no delay.

Acknowledgments

FonaDyn is written in SuperCollider [2], an interactive real-time audio and music processing environment, first created by James McCarthy. SuperCollider is cross-platform (Windows, Mac, Linux, iPhone, and more) and open-source freeware. The foundation version of FonaDyn was written in 2015 by Dennis Johansson for his M.Sc. degree project in Computer Science [13]. Isak Nilsson, in the course of *his* M.Sc. degree project [16], ran tests on MacOS, recompiled the MacOS *PitchDetection* UGens for FFTW, and contributed supporting Matlab code. FonaDyn continues to be developed by Ternström and co-workers. The work was partially funded by the Swedish Research Council (Vetenskapsrådet), project 2010-4565.

References

[1] Ternström S, Pabon P, Södersten M (2016). The Voice Range Profile: its function, applications, pitfalls and potential. *Acta Acustica united with Acustica*, 102(2), 268-283.

[2] SuperCollider website: http://supercollider.github.io/

[3] Ternström S (2019). Normalized time-domain parameters for electroglottographic waveforms. J Acoust Soc Am.;146(1):EL65-EL70. doi:10.1121/1.5117174

[4] Selamtzis A, Ternström S (2014). Analysis of vibratory states in phonation using spectral features of the electroglottographic signal. *J. Acoust. Soc. Am.*, 136(5), 2773-2783.

[5] McLeod P, Wyvill G (2005). A Smarter Way to Find Pitch. *Proc Int'l Computer Music Conf*; ICMC 2005, 138-141. [An implementation of the above, called "Tartini", is included with the 'SC3-plugins' library of signal function blocks.] Permalink: http://hdl.handle.net/2027/spo.bbp2372.2005.107.

[6] Pabon JPH (1991): Objective acoustic voice-quality parameters in the computer phonetogram. *J. Voice* 5(3) 203-216.

[7] Dolanský LO (1955). An Instantaneous Pitch-Period Indicator. *J. Acoust. Soc. Am.* 27, 67-72 (1955); http://dx.doi.org/10.1121/1.1907499

[8] Herbst C, Ternström S. A comparison of different methods to measure the EGG contact quotient. *Logopedics Phoniatrics Vocology*, 2006;31(3):126-138. doi:10.1080/14015430500376580.

[9] McFee B (2012). *More like this: machine learning approaches to music similarity*. PhD thesis, University of California at San Diego, 186 p (algorithm B.1, p. 152), http://bmcfee.github.io/papers/bmcfee_dissertation.pdf. [An implementation of the above, called "KMeansRT", is included with the 'SC3-plugins' library of signal function blocks. FonaDyn supplements this with "KMeansRTv2", which provides the option of continued learning or classifying with a pre-learned vector of centroids.]

[10] Richman JS, Randall Moorman J (2000). Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology*, 278 (6), 2039-2049.

[11] Yu-Hsiang Pan, Yung-Hung Wang, Sheng-Fu Liang, Kuo-Tien Lee (2011). Fast computation of sample entropy and approximate entropy in biomedicine. *Computer Methods and Programs in Biomedicine*, 104 (3), 382-396.

[12] Fabris C, De Colle W, Sparacino G (2013). Voice disorders assessed by (cross-) Sample Entropy of electroglottogram and microphone signals. *Biomedical Signal Processing and Control*, 8 (6), 920-926, ISSN 1746-8094, http://dx.doi.org/10.1016/j.bspc.2013.08.010. (http://www.sciencedirect.com/science/article/pii/S1746809413001237)

[13] Johansson D (2015). *Real-time analysis, in SuperCollider, of spectral features of electroglottographic signals*. M.Sc. degree thesis in computer science, KTH Royal Institute of Technology, Stockholm, Sweden. Available online at this link (October 2016).

[14] Schutte HK, Seidner W (1983). Recommendation by the Union of European Phoniatricians (UEP): Standardizing Voice Area Measurement/ /Phonetography. *Folia Phoniatrica* 1983;35:286–288, (DOI:10.1159/ /000265703)

[15] http://www.expert-sleepers.co.uk/ You will need the modules ES-3, ES-6, optionally the ES-7, two TosLink optical cables, an extra power supply module and a box in which to mount the modules.

[16] Nilsson I (2016). *Electroglottography in real-time feedback for healthy singing*. M.Sc. degree thesis in computer science and communication, KTH Royal Institute of Technology, Stockholm, Sweden. Available online at this link (December 2016).

Some other relevant sources, not mentioned in the text

[17] Selamtzis A (2014). *Electroglottographic analysis of phonatory dynamics and states*. Licentiate thesis in Speech and Music Communication, Stockholm: KTH Royal Institute of Technology, 2014. , vii, 31 p. Available online at http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-145692

[18] Selamtzis A, Ternström S (2017). Investigation of the relationship between the electroglottogram waveform, fundamental frequency and sound pressure level using clustering. *J. Voice,* 31 (4), July 2017, 393-400, available online at http://dx.doi.org/10.1016/j.jvoice.2016.11.003.

[19] Roubeau B, Henrich N, Castellengo M (2009). Laryngeal Vibratory Mechanisms: The Notion of Vocal Register Revisited. *J. Voice*, 23 (4), July 2009, 425–438.

[20] *Matlab* © The MathWorks, Inc. www.mathworks.com

[21] Herbst C (2004). MovingEGG. Available online at this link. (Accessed August 2019).

[22] Švec JG, Granqvist S (2010). Guidelines for selecting microphones for human voice production research. *American Journal of Speech-Language Pathology*, 19, 356–368, November 2010.

[23] Ternström S, D'Amario S, Selamtzis A (2018). Effects of the lung volume on the electroglottographic waveform in trained female singers. *J. Voice* (2018), e-pub ahead of print, Open Access. https://doi.org/10.1016/j.jvoice.2018.09.006