H. Duifhuis

Manual for the Cochlea program

August 4, 2011

Springer

Preface

This manual and the underlying cochlea program are based on a Master's thesis presented at the Faculty of Mathematics and Natural Sciences of the University of Groningen in 2001, by Johan M. Kruseman.
Supervisors were Peter W.J. van Hengel and myself.

At the time the computer program was written in Fortran95. It was developed in a Windows 95 environment to provide a graphical user interface (GUI), if wanted. The choice of the Fortran compiler was based on its excellent parallel processing properties.

The GUI is available only in the Windows environment, but the Fortran code of the main program is applicable in a Linux environment. To achieve this, from the start by Johan Kruseman the setup has been to develop two programs, one associated with data input and output, and the other with the real computation of the nonlinear cochlea model. The frequently used input parameters and responses can be given and edited simply as general text. Thus, they are accessible to any alternative program.

Minor updates have been made to be able to run the program smoothly in newer (2011) environments.

Groningen, September 2011 *Hendrikus Duifhuis*

Contents

Chapter 1
The program manual

1.1 Introduction

The cochlea program simulates basilar membrane movement. The cochlea can be stimulated by one or two plain sine-waves and/or a pulse sequence or the user can specify an external file to be used as stimulus. With this stimulus and some user set parameters that define the behavior of the cochlea, the movement of the basilar membrane is calculated. The program can display the membrane movement changing in time on screen, and/or it can write the data to files.

1.2 Structure of the program

The program consists of two executables (`COCHLEA.EXE` and `COCHLEA2.EXE`) and a parameter file (`PARAMETERS.DAT`). `COCHLEA.EXE` is the main program. It shows a dialog box enabling the user to set the parameters. When the start-button is pressed, the parameters are saved to `PARAMETERS.DAT` and `COCHLEA2.EXE` is executed. When `COCHLEA2` has terminated, the dialog box pops up again and the user can start a new run.

`COCHLEA2` is the program that performs all the calculation and generates the output. It reads the parameters from the text file `PARAMETERS.DAT`. As mentioned above, `COCHLEA2.EXE` is initiated by `COCHLEA.EXE`. However, it may be useful to change the parameter file yourself and to run `COCHLEA2.EXE` directly (for instance in an automatic process, where you do not want a dialog box to change the settings). In that case be sure to maintain the layout and the order of the parameters (the parameter lists) in the parameter file.

1.3 Using the program

When COCHLEA is started, it first reads the current settings from PARAMETERS.DAT. Be sure that the file is in the same directory as the program. The user can change

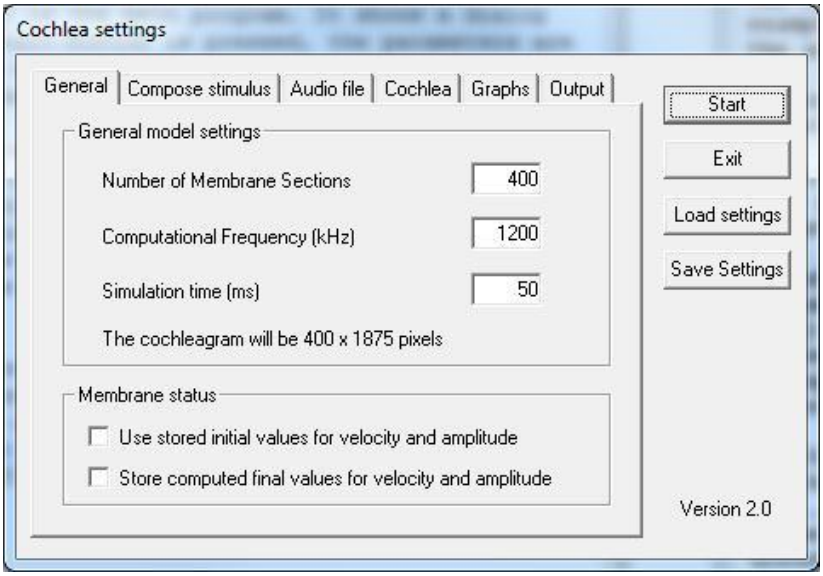

Fig. 1.1 General tab of the Cochlea parameters setting GUI. Each tab shows a set of parameters. If necessary, dependent parameters are linked or limited to valid values.

the settings in the dialog box and start a run. The dialog box contains six tab pages: General, Compose stimulus, Audio file, Cochlea, Graphs and Output. At the right hand side there are four buttons: Start, Exit, Load settings and Save settings.

Start	The Start-button saves the parameters to PARAMETERS.DAT and executes COCHLEA2.EXE. After that, the dialog box will return.
Exit	The Exit-button terminates the program. Settings are not saved.
Load settings	This button allows the user to load a previously saved parameter set.
Save settings	Used to save the current parameter set to file. The extension is cpf (cochlea parameter file). The saved settings can be retrieved later using the button Load settings.

1.3.1 Rescaling

When you have selected the parameters and pressed the Start-button, the program will start the simulation. If switched on the legend shows how many per cents of the simulation have been completed. After the simulation has finished a rescale-dialog box pops up.

In this box you can change the scaling parameters and let the program plot the graphs again. The minimum and maximum value to plot are automatically set to the smallest and largest value of the last run. So, using these default settings, the membrane will range along the total height of the membrane graph and the cochleogram color range equals the total range of the color bar. For one or both, you can select to use the current value, which is the value set in the main dialog box. As a last option, one can type a new value. In this way it is possible to look in detail to a specific area. When for example the automatic scaling is from -10 to 10, setting the range from 0 to 2 will give a better look on low positive values. High positive values are plotted white and negative values are plotted black, but for the small area from 0 to 2, the entire color range is available.

In the Select Graphs section in the Rescale dialog box, you can turn on or off one of the graphs. With plot options you can determine whether the displacement of the basilar membrane has to be plotted or the velocity (scaling values are adjusted accordingly).

Note that the program does not redo the calculation, only a rescaling of the previously calculated points is performed. Therefore, when you do not have a good idea of what scaling parameters you should use, it may simplify matters to first execute the calculation without plotting (switching the graphs off in the graphs tab page of the main dialog box) and to use the rescaling box to start the plotting with reasonable parameters.

1.3.2 Comparing images of different runs

To compare images of different runs, it is necessary to store the results of interest.

1.4 Setting the parameters

This section is divided into six sections, each of which treats a tab page of the dialog box.

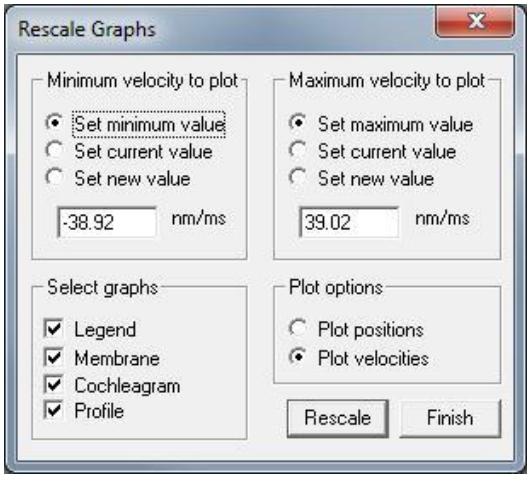

Fig. 1.2 The upper blocks provide current maximum and minimum values of the velocity plot. The lower left block allows a modification of the display, and the lower right allows the choice between velocity and deflection responses.

1.4.1 General TAB

The general tab page contains a group box to change some general settings and a group box for the membrane status. The following values can be set in the upper section:

The 'Number of membrane sections' field

The membrane is divided into sections so the model can be solved numerically. Using more sections will improve the accuracy of the computations, but will also increase computation time. When results look suspicious, try to increase the number of sections until the results remain the same[1]. Note however that using more than 3200 sections is meaningless, since the real cochlea has only that amount of inner hair cells. When using Zweig impedance, one should use at least 600 sections.

[1] Hint: to compare the output images, it is useful to make sure that the images have the same size. So multiply the `NUMBER OF SECTIONS PER PIXEL` by the same amount with which you multiply the `NUMBER OF MEMBRANE SECTIONS`. In the previous section is explained how you can keep images on screen to compare them with a new run.

The 'Computational frequency' field

The simulated time is also divided in discrete intervals. Again, the smaller the intervals, the more accurate the results, but computation time will also increase. Try to increase the computation frequency until the results remain the same. For a human cochlea with normal impedance (see cochlea tab page), use at least 80 kHz. When modeling the guinea pig, one should use a higher frequency (caused by the higher frequencies present in the cochlea of the guinea pig). Zweig impedance requires an computational frequency of at least 150 kHz.

The 'Simulation time' field

This is the place to set the total simulation time.

The cochleogram size is updated when you change one of the settings above. See the Graphs-tab information for more details. In the lower section of this menu page, one can choose to store the final values of the membrane velocity and displacement to the binary files `FINAL_V.BIN` and `FINAL_Y.BIN`. In a next run, one can continue the computation by using the stored values as the initial ones (otherwise the membrane is initiated at its equilibrium positions, i.e. displacement and velocity both are zero everywhere along the membrane).

1.4.2 Compose stimulus TAB

On this tab page, you can compose your own stimulus. A stimulus can be composed out of a pulse sequence and two sine-waves. Each of the components can be switched on or off and specified independently. The composed summed stimulus can be switched on smoothly by setting the onset duration. This reduces onset spectral splatter.

1.4.2.1 Pulse:

The pulse has an amplitude set by `Level` (dB SPL); the `Duration` of the pulse can be set in ms after which the signal is reset to zero. This pulse is repeated after `Period` ms. The `Duration` has to be at least the length of one computational time interval (i.e. 1/(internal computational frequency); see General Tab information).

1.4.2.2 Sine waves:

The sine waves have a fixed frequency, phase, and amplitude (`Level`). The frequency can be modified either by changing the `Period` or by modifying the `Frequency` (= 1/`Period`).

1.4.3 Audio file TAB

In this tab page you can select an external audio file (not compressed wave-format, .wav)[2] as stimulus (press `Open`-button). By pressing the `PLAY`-button the audio file will be played. By setting the `Level` (dB SPL), you specify the level of the largest possible value in the audio file. Furthermore, the sampling frequency of the audio file has to match the internal computational frequency. To match these two frequencies, the audio file is resampled with a rational fraction to get a match of at least 99%. To get an exact match, the computational frequency is adjusted accordingly. The user is asked if he wants the total simulation time to be set equal to the duration of the audio file. Mono, stereo and three-way (left, right, center) signals are supported, but only one channel (either left or right) will be used.

Several programs or routines are available to enable the generation of well defined .wav files. For instance, Matlab© provides the command *wavwrite* and Adobe Audition© has several generation and mixing options.

1.4.4 Cochlea TAB

Currently the program enables the user to simulate basilar membrane movement of humans and of the guinea pigs. Both settings have their own set of parameters (see table). Other settings can be added.

In the model, the CP is divided into a number of sections, each section having its own stiffness and damping. The number of sections is considered a general parameter, and has been introduced above (General TAB). For a given mass per unit area the frequency map, which gives the resonance frequency for each position, and the quality factor (which determines the shape of the resonance curve) together define the stiffness and damping. Damping and stiffness can independently be set linear or nonlinear.

[2] The .wav format is not a fixed standard, it allows for specifications in the file header. The cochlea program processes simple versions only.

1.4.4.1 Damping profile

The damping profile is defined by the quality factor, Q_{3dB}. The higher the quality factor (i.e. less damping), the sharper the profiles will be. One can choose a constant quality factor or a quality factor that is dependent on position in the cochlea. In the latter case the Q is proportional to the square root of the resonance frequency and is 0.5 at the apex to get a frequency independent matching at the end (so to prevent reflections). When Zweig impedance is used, the quality factor should be negative in order to get negative damping. Do not to set a value too close to zero, because then the model will become unstable. The more negative the quality factor in the Zweig impedance case (i.e. less negative damping), the more stable the model will be. Selecting the position dependent quality factor while using Zweig impedance is not very useful, because Zweig impedance is based on negative damping, stabilized by a feedback force which results in positive damping.

1.4.4.2 Frequency map

A choice between two different frequency maps has to be made. The frequency maps are given by: $f(x) = A.10^{-\alpha x} - B$, where x is the position on the membrane measured from the stapes.

Frequency map	A	B	α	f_{stapes} (Hz)	f_{apex} (Hz)
Exponential (human)	22507	0	65.1	22507	138
Greenwood (human)	18136	145.4	60	17991	20
Exponential (guinea pig)	38346	0	113.5	38346	347
Greenwood (guinea pig)	43694	297.5	113.5	43397	98

1.4.4.3 Impedance

Stiffness and damping can be set to linear and several nonlinear parameters. Moreover, the Zweig-impedance can be selected; its characteristics are somewhat different from the regular impedance settings. Some combinations are possible with others, and some combinations are excluded (see also Chap. 5, Table 5.4 in the CM-book). The most common combinations are indicated in the next table:

		damping								
		Lin	NL	vdP	G&F	06	07	Hopf	Zw.Lin	Zw.NL
stiffness	Lin	×	×	×	×	×	×	×		
	NL	×	×		×					
	Zw.Lin								×	
	Zw.NL									×

`Linear stiffness:` the stiffness is dependent on position on the membrane only, not on the displacement of it.

`Nonlinear stiffness:` the linear stiffness is multiplied by a nonlinearity factor that is dependent on membrane displacement.

`Linear damping:` the damping is dependent on position on the membrane only, not on its displacement or velocity.

`Nonlinear damping:` parabolic damping shape, with positive damping.

`Van der Pol damping:` default Van der Pol damping profile (parabolic, negative around 0) with $\varepsilon = 0.05$.

`Hopf damping:` parabolic damping profile, similar to Van der Pol, except that now the damping $d \equiv 0$ for small deflection or velocity response.

`Goldstein & Furst damping:` in this case the nonlinearity factor is given by $1+4|y|^{0.1}$, where y represents the displacement of the section involved.

`06 or 07 damping:` similar to NL damping, with different shapes for small amplitudes (different active behavior).

`Zweig-linear or Zweig-nonlinear:` are determined by feedback. The linear version has an overall negative damping except at the borders of the membrane. For the nonlinear system the feedback becomes negligible at high levels.

1.4.4.4 Active damping

The nonlinear damping case can be made active in certain sections of the membrane, which means that parts of the membrane will start to move even without an external stimulus. Hereto a negative term is added to the nonlinearity term for small velocities (for large velocities, damping is still positive (and nonlinear). The nonlinearity factor in the case of active damping is given by:

$$d(x,v) = \frac{\sinh(\alpha v)}{\alpha v} - \frac{\gamma}{\cosh(\beta v)},$$

with $\alpha = 10^4$ and $\beta = 10^6$.

The values for gamma are read from the ASCII-file `GAMMA.DAT`. This file must be in the same directory as the main program. For every section of the membrane, a value for gamma is read from the file, so the file should contain (at least) as many gamma-values as there are sections. For each section where $\gamma > 1$ and for small velocities, the active damping predominates and energy is produced in the ear. For larger velocities, the damping is positive and will dissipate energy.

1.4.4.5 Cochlea options

The apex is the last part of the cochlea where the fluid can flow from upper to lower duct. In earlier versions no mass and damping were incorporated for the apex. This is corrected, but one can still check `SET SHORTCUT APEX` to compare results with those of previous versions. Note however that shortcutting the apex is physically incorrect. It is possible to simulate the presence of a microphone positioned in the outer ear (for instance to measure OAEs) by checking `USE EAR CANAL COUPLER`.

1.4.5 Graphs TAB

This tab allows selection of displayed output.

1.4.5.1 Select Graphs

`Legend`	displays the settings used for the current run
`Stimulus`	displays the stimulus. Be aware of aliasing when NUMBER OF TIME STEPS PER PIXEL is set greater than one. Simulation time is on the horizontal axis; the level of the stimulus is on the vertical axis
`Membrane`	displays the displacement or velocity (depending on plot options) of the basilar membrane (for each section with an interval of NUMBER OF SECTIONS PER PIXEL and for each time step with an interval of NUMBER OF TIME STEPS PER PIXEL). Horizontal axis represents the distance along the cochlea measured from the stapes (high frequencies on the left; low frequencies on the right); the vertical axis is the displacement or the velocity of the membrane. This graph is actually a vertical cross section of the cochleogram.
`Cochleogram`	displays displacement or velocity of the basilar membrane as a color image. Along the horizontal axis is the simulation time; the vertical axis is the distance along the cochlea (stapes, so

high frequency-side, is above; apex or low frequency-side is beneath). The color indicates the displacement or the velocity of the basilar membrane section (ranging from dark blue through green and yellow to dark red; see the color bar further on).

`Profile` displays the maximum displacement or velocity of each section. Horizontal axis is represents the distance along the cochlea measured from the stapes; vertical axis is the maximum value in dB relative to 1 nm or 1 nm/s respectively. The vertical axis has a range of 60 dB. When only one sine wave is used as stimulus, the phase of each section relative to the stimulus is plotted as well. The phase difference is calculated in the last period and 2π-jumps are removed. The Profile will be shown at the end of a run.

1.4.5.2 Graph size

For accuracy and stability reasons the time intervals and the sections (see General tab page) has to be small enough. You may however not want to plot all the calculated points. For both you can define the number of calculated points per pixel.

The total number of calculated time points is equal to the total simulation time times the computational frequency. The width of the cochleogram is this value divided by the number of time steps per pixel. The height of the cochleogram is the number of sections divided by the number of sections per pixel.
Note that the cochleogram size may not be too large, else the cochleogram window cannot be initialized correctly.

When too few points are plotted, aliasing-effects can become visible. It is possible to remove this aliasing by low pass filtering the output. Be aware that this also does not give you a good representation of what is going on. Parts of the membrane now seem not to move at all or just at low frequencies, while they may move at a high frequency. The only way to get a good representation of reality is by plotting more points per time unit (i.e., increasing the computational frequency or decreasing the number of time steps per pixel). The dialog box shows the frequency above which the frequencies will be removed. This frequency has to be larger than the highest frequency present in the cochlea (see frequency map) to get a correct representation.

Plot options

The program calculates both displacement and velocity of the membrane. Here you can specify which one has to be plotted in the graphs selected above.

Scale options

The membrane and cochleogram both plot displacements or velocities. In Scale options one can set the maximum and minimum value to plot. All the values in between are scaled linearly. In the membrane graph, values outside this range fall outside the graph window; in the cochleogram values lower than the minimum value are plotted black, values exceeding the maximum value are plotted white. So, when the minimum value is set to zero, negative values are all black and the positive values till the maximum value are scaled linearly on the color map (ranging from dark blue to dark red). When the minimum and maximum value are centered around zero, negative values range in the blue, zero is represented by light green and positive values are plotted in the colors from yellow to red.

Fig. 1.3 In the cochleogram values are plotted using this color range. High values (relative to the maximum value to plot) are plotted in the colors on the right, low values (relative to the minimum value to plot) are plotted in the colors on the left). Values lower than the minimum value to plot are plotted in black; values exceeding the maximum value to plot are plotted in white.

1.4.6 Output TAB

On this tab page you can select the files to store several quantities. All the files are placed in the general output directory. This directory may be given relative to the main programs location, but you can also type a absolute location (beginning with the drive letter). Leaving this field blank causes the files to be written to the directory of the main program. The file names may not contain any subdirectories.

1.4.6.1 Pressure ear canal

Two columns of data are written:

1. simulation time (s)
2. pressure in the ear canal (Pa)

For every calculated point (equal to the computational frequency times the simulation time) these values are stored in this file.

1.4.6.2 Profile

Four columns of data are written:

1. Section number
2. Resonance frequency (Hz)
3. Maximum displacement (m)
4. Maximum velocity (m/s)

1.4.6.3 Probing

The simulation time is written to the first column. In the next n columns the displacement of the probe points is written; in the last n columns the velocity of the probe points is written (the number n being the number of selected probe points). To select the sections to store in the file, type the number of the section in the field at the bottom of the tab page. When you do not want more sections, put a zero or a number higher than the number of membrane sections that was set in the general tab field in the next field.

1.4.6.4 Store extra information into files

When this option is selected, header rows will be written at the beginning of the file and the parameters used in the run are written at the end of the file. The header rows inform what data is stored in which column.

1.4.6.5 Store data in binary format

Data can either be stored in ASCII or in binary format. ASCII has the advantage that data can be read with every normal text editor; binary written files, however, are smaller in size and are processed faster. In binary format all values are written consecutively as 4 bytes reals (no extra info is stored at the beginning or at the end). The following code fragment shows how to read for example the ear canal pressure file with Matlab^{TM}:

```
columns = 2;
fid  = fopen (EarCanalPressureFilename)
data = fread (fid, [columns, inf], 'real*4');
t    = data (1, :);
p    = data (2, :);
```

For the profile file, the value of columns must be set to 4; for the probing file, columns must be set to $2n+1$ (n indicating the number of probe points).

1.5 Hardware requirements

Cochlea is a small program, but can require large amounts of memory and computational time, depending on the settings made in the dialog box or parameter file.

1.5.1 Notes about computation time

The simulation of the movement of the basilar membrane requires a considerably amount of computational power. The computation time depends mainly on the number of sections and the total number of time steps (= computation frequency $\times$ the simulation time). The selected impedance (linear or nonlinear) is an important determinant. Plotting graphs and writing data to output files also adds to the run time.

Apart from the initialization (which takes negligible time), the computational time is linear in the number of sections, the computational frequency and the simulation time. On a 3200 MHz computer the computation time (CT) can be approximately related to the three factors above (n, f and t) by $CT = anft$ where a is a constant (plotting and storing the output both have been switched off, cochleogram size has been set to 1×1 and a linear cochlea is used). The value of a depends on the computer processor and software. Our current score with an AMD 3200 MHz processor is a run time that equals $10 \times$real time, for $n = 400$ and $f = 100$ kHz.

Introducing nonlinearity into the system increases the computational time considerably. The normal nonlinear damping causes the model to run a factor 1.5 slower. Active nonlinear damping is at least a factor 3 to 4 slower than the linear case.

Of course the program is not very useful without output. First of all a larger cochleogram size will require more computational time (because values have to be stored in memory, so they can be used while rescaling the image). This is necessary only if graphs are plotted, so set the cochleogram size to 1×1 if the output consists of writing to files only (this is equivalent to switching of the cochleogram). The legend and profile do not take noticeable time, but the membrane response and cochleogram do.

1.5.2 Notes about memory usage

The program uses a few structures that need a 'significant' amount of memory.

When a .wav-file is used as stimulus, the file is resampled to match the computational frequency and those values are stored in a real(4)-array. The array-size is equal to twice (see footnote 4) the number of time samples (= twice the computational frequency times the simulation time). For a computational frequency of 400 kHz and a simulation time of 2 seconds, this requires 6.4 MB.

The displacements and velocities of the membrane are stored in two real(4)-matrices (to be able to rescale the image afterwards). Both have the size of cochleogram size. A cochleogram size of 800 $\times$5000 requires 16 MB for each matrix.

Finally, Zweig impedance uses an array to store previous displacement-values, necessary for the feedback force. This array is linear in size with n and the computational frequency. Eight hundred sections and a computational frequency of 400 kHz, require about 8 MB, depending on the frequency map used.

Chapter 2
Program

Abstract
The programs **CochleaDialog (cochlea2)** and **Cochlea** are presented, with example scripts for specific application.

Introduction

Basically the sections present the source codes of the programs and underlying structure, without additional comments. Some comments are contained in the program code. The structure of the Sects. and Subsects. follows the structure in the program. The routines are ordered alphabetically within the (sub-)sections.

Local parameters are specified within routines, global parameters are defined within 'modules'. Implicit parameters (optional in Fortran) are explicitly excluded with the `IMPLICIT NONE` command in each header. Short comments appear on several places.

The numbers in the right-hand column represent the number of lines in each routine. It includes comment lines as well as blank lines, included for readability.

2.1 Cochlea Dialog

We present the elements of the program, i.e., the constituting subroutines and functions, in the following list. The files are presented on the CD in original format, using the indicated directory or folder structure.

Dialog
Buttons

`ErrorHandling.f90`	117
`FileNameBox.f90`	95
`LoadButton.f90`	26

Most .f90 files are compact (20 - 40 lines) subroutines or functions. Some of these are used in both the dialog program and in the execution program. The `WinMain.f90` file is the file that defines the structure of the graphical interface under Microsoft Windows.

2.2 Cochlea

This section presents the similar structure for the program. The basic routine is the RK4-subroutine with the associated g-function definition (damping + stiffness

parts of the local 2nd order ODEs) as CONTAINed parts. This includes the 'Zweig'-damping and stiffness specification.

```
   ReadData.f90                          43
   WaveRead.f90                          26
   WaveReadModule.f90                    40
Cochlea.f90                             127
Cochlea.rc                              structure file
```