USF Neural Simulator

Table of Contents

Chapter 1

Overview

The USF Neural Simulator[1] started out as an implementation of Ronald J. MacGregor's SYSTM11 simulator as documented in his 1987 book *Neural and Brain Modeling*, together with the NEWSNED network editor and the SNEDSKOP waveform display program. It has now been rewritten in the C language. The new program is compatible with the previous program's functionality, and has many new features.

The software is written for the UNIX environment. The UNIX environments it has been run on include Linux, Cygwin on Windows, and HP-UX.

The input to the simulator is in two forms: the first is a ".sim" format text file that lists the values of all the parameters, and the second is answers to questions that the simulator asks on the command line. The answers can be listed in advance in a second text file and then fed to the simulator. These files could be created by hand in any text editor (run Cygwin's dos2unix on them if you use a Windows program to create them), but NEWSNED provides a graphical interface for drawing the network, specifying the parameters, creating the two files, and invoking the simulator, that makes the process much easier.

The output from the simulator is in the form of a ".bdt" format text file that lists the spike times of selected cells and the integrated activity of a population, and "wave" text files that record selected membrane potentials and other values for display by SNEDSKOP. The ".bdt" files can be displayed with the "scope" application, also available from this laboratory.

[1]The program was developed in laboratory of B. G. Lindsey at the University of South Florida by U.J. Balis, Kendall Morris and others using Fortran, the K&R C language, X-windows and Motif. The latest version was modified and enhanced as described in this document by Russell O'Connor in 2005-2007. Development of the program was supported by NIH grants NS19814 and NS46062 as part of the NSF/NIH Collaborative Research in Computational Neuroscience Program.

Chapter 2

Installation

2.1 Installing on Windows

2.1.1 The Cygwin UNIX Environment

In order to run the simulator on Windows, you first need to install the Cygwin UNIX environment, which is available at no cost from http://www.cygwin.com. Installation instructions are available at that website, and a few additional details are in the next section of this manual.

2.1.2 Installing Cygwin

The Cygwin environment is installed using the Cygwin installer program, `setup.exe`, which can be downloaded from http://www.cygwin.com/setup.exe. Run the program, and you will be presented with a series of dialog boxes asking questions, as is typical for Windows software install programs. The default responses for most of the questions are correct for most installations, but you will need to make a choice at the "Choose Download Site" screen and the "Select Packages" screen.

The Cygwin software is available from nearly 100 download sites worldwide but many of the sites are slow or unavailable at any particular time. Choosing a site from the list is mostly a matter of trying one and seeing how it works. The site http://cygwin.paracoda.com/ has been fast and reliable here lately (October 11, 2005 in Tampa, FL).

By default, Cygwin setup installs a minimal set of packages. In order to run the simulator, you will need more than that. The easiest thing to do it to install all the packages, if you have the disk space. The total disk space requirement is, as of October 11, 2005, 2.4 gigabytes plus 1.25 gigabytes for the package files (which can be deleted after installation). The procedure for installing all the packages is documented at http://www.cygwin.com/faq/faq.setup.html. The key paragraph is:

At the "Select Packages" screen, in "Categories" view, at the line marked "All", click on the word "default" so that it changes to "install". (Be patient, there is some computing to do at this step. It may take a second or two to register the change.) This tells Setup to install everything, not just what it thinks you should have by default.

If you are short on disk space, you can install the default packages plus at least the gcc, lesstif, xorg-x11-base, and xterm packages, but I have not tried it so I don't know what other packages you may need, or how much disk space you save.

After setup is finished, copy the file `C:\cygwin\usr\X11R6\bin\startxwin.bat` to your desktop so that you can get at it easily. Then double-click on it and you should get a command-line window. (If you installed Cygwin somewhere other than `C:\cygwin`, you will have to modify the above instruction accordingly.)

2.1.3 Installing the Simulator

From this point on, installing the simulator on Windows is the same as installing it on UNIX. see Section 2.2: UNIX Install, page 3.

2.2 Installing on UNIX

In order to install the simulator on UNIX, your UNIX system must have a C compiler, the X Window System and the Motif (of Lesstif) GUI libraries.

The simulator is packaged in a .tar.gz file. Copy this file (for example, the file named '`simulator-0.1.28.tar.gz`') to the directory under which you would like to install the simulator. For example, if you received the file via email, you might save the attachment to `C:\cygwin\home\myname`. Then, at the cygwin command line, *`cd`* to that directory, and type something like the following:

```
gunzip -c simulator-0.1.28.tar.gz | tar xf -
cd simulator-0.1.28
./configure > configure.out
make > make.out
make install > make_install.out
```

You may need to have admin privileges for "make install" to work.

If it succeeds, you can invoke NEWSNED by typing:

```
newsned
```

Chapter 3

Running the Simulator

3.1 Model Parameters

The model parameters can be specified either by using NEWSNED or by editing the ".sim" file. There are two formats of ".sim" file. The simulator will read either format, but NEWSNED only writes the new format. The labels used for the parameters are different in NEWSNED than they are in the ".sim" files. In the following sections, each parameter is described and the labels used in NEWSNED and in the ".sim" files, and the variable names used in *Neural and Brain Modeling*, and the symbols used in Breen et al, are given in the following format:

SNED: label used in NEWSNED
SIM: label used in the old format ".sim" file
SIM2: label used in the new format ".sim" file
BREEN: label used in Breen et al (see [Breen et al], page 9)
VAR: variable name used in *Neural and Brain Modeling*
UNIT: unit of measure for the parameter
TYP: typical values for the parameter

If a particular entry does not appear for a particular parameter, that parameter is not used in that place.

Several of the parameter descriptions refer to variables that decay exponentially toward a value, with a specified time constant. That means that the variable is governed by an equation of the form

$$A + (V - A) * exp(-t/t0)$$

where t0 is the time constant and A is the value toward which the variable is decaying. (V is the value of the variable at t = 0).

3.1.1 MacGregor Cell Parameters

SNED: Accommodation Time
SIM: Time constant for accommodation
SIM2: DCTH
VAR: TTH
UNIT: milliseconds
TYP: 20-25

The firing threshold moves up and down in proportion to membrane potential changes, but it doesn't move immediately to the new value. Instead, it approaches it exponentially. This parameter is the time constant of the exponential decay toward the new value, except for SIM2, where it is exp(-step / TTH).

SNED: Potassium Conductance Time
SIM: Time constant for potassium conductance
SIM2: TGK
VAR: TGK
UNIT: milliseconds
TYP: 3-10

In the absence of spikes, the potassium conductance decays exponentially toward zero with the time constant specified by this parameter. This controls the refractory period. During a spike, the potassium conductance decays exponentially toward a specified value (see [B], page 6), with this same time constant. If this parameter is set to 0, the cell model changes to Hybrid IF (see [Hybrid IF Parameters], page 9).

SNED: Membrane Time Constant
SIM: Time constant for membrane potential
SIM2: TMEM
VAR: TMEM
UNIT: milliseconds
TYP: 5-11

In the absence of potassium or synaptic conductances, the membrane potential will decay exponentially toward its resting potential (which is taken as 0 in this model) with the time constant specified by this parameter. If this parameter is set to 0, the cell model changes to a PSR model (see [PSR Parameters], page 11).

SNED: Resting Threshold
SIM: Resting threshold
SIM2: Th0
VAR: THO
UNIT: millivolts
TYP: 10-20

When the membrane potential has been at its resting value so that the firing threshold is not affected by accommodation, the firing threshold will have the value specified by this parameter, relative to the resting membrane potential. If the value of this parameter in NEWSNED's Cell Population Definition Panel is followed by a slash and a second number, the value of this parameter will be distributed around the specified value (the first number) with a normal distribution with a standard deviation equal to the second number.

SNED: K conductance change with AP
SIM: Change in potassium conductance with action potential
SIM2: B
VAR: B
UNIT: dimensionless
TYP: 25-27

During an action potential (which has a duration of one time step), the potassium conductance will decay exponentially toward this value. This value is in units of the resting membrane conductance. In other words, a value of 2 means twice the resting membrane conductance.

SNED: Accommodation Parameter
SIM: Accommodation parameter
SIM2: MGC
VAR: C
UNIT: dimensionless
TYP: .04-.85

As the membrane potential rises above its resting value (taken to be 0 in this model), the firing threshold will eventually rise in proportion from its resting value. This parameter is the proportionality constant. For example, if this parameter is .5, then when the membrane potential rises 2 mV above its resting potential, the firing threshold will rise exponentially toward 1 mV above its resting value, with a specified time constant (see TTH, above).

SNED: Population Size
SIM: Number of cells in population
SIM2: cell_count
VAR: N
UNIT: count
TYP: 1-1000

All the cells in a population have the same values of the other parameters specified in this section, but each cell in the population can have a different membrane potential and firing pattern because it can have a different firing pattern at its synapses than the other cells in the population, so each cell is simulated separately. This parameter specifies how may cells in the population are simulated.

SNED: DC Injected Current
SIM: dc injected current for each population
SIM2: GE0
VAR: SC
UNIT: millivolts
TYP: 0-17

An injected current will raise the membrane potential by an amount that is inversely proportional to the membrane conductance. Instead of being specified directly as a current, the "DC Injected Current" is specified in terms of the effect it has on the membrane potential. This parameter is specified in millivolts, and it is interpreted as the current that is required to raise the membrane potential by the specified number of millivolts when the membrane conductance has its resting value. The effect on the membrane potential at other membrane conductances will be inversely proportional to the conductance.

SNED: Cell Population Comment
SIM: [unused]
SIM2: [unused]
VAR: [unused]
UNIT: n/a
TYP: n/a

This is not really a simulation parameter, but instead is just a label for a cell population on NEWSNED's graphical display. It does not appear in the ".sim" file. It is listed here because it appears in NEWSNED on the cell population parameters screen.

SNED: [implicit]
SIM: Number of targets of cell populations
SIM2: targetpop_count
VAR: NTGR
UNIT: count
TYP: n/a

This parameter is the number of connections between populations whose source is this cell population. This is not explicitly entered in NEWSNED, because NEWSNED determines it from the network drawing. A single "connection between populations" represents a connection from each cell of the source population to some number of cells of the target population. There can be multiple "connections between populations" from the same source population to the same target population, and each of those connections is counted separately for this parameter. However, if you set up multiple connections that all have the same source and target population, NEWSNED will only allow you to edit the parameters of the first of the connections.

SNED: Noise Amplitude
SIM2: noise_amp
UNIT: nanosiemens
TYP: 0 - .3

Each cell has an internal noise generator that acts like two synapses, one with an equilibrium potential of 70 mV above resting and the other with -70mV. Each acts like it has an incoming firing probability of .05 per time step, and a synapse time constant of 1.5ms. This parameter is the conductance that gets added to the synapse conductance on each (virtual) spike.

Unused Parameters

The old format ".sim" file specifies several cell population parameters that are not used in this version of the simulator. Their values in the ".sim" file have no effect on the simulation. The labels for these parameters as they appear in the sim file are as follows:

```
Rebound parameter:
Rebound time constant:
Threshold for removal of ika inactivation:
Threshold for ika activation:
Maximum ika conductance:
Proportionality constant for removal of ika inactivation:
Proportionality constant for ika activation:
Time constant for ika:
```

3.1.2 Hybrid IF Cell Parameters

If the Potassium Conductance Time of the MacGregor model (see [TGK], page 5) is set to 0, the cell model changes to Hybrid IF ("burster"). The Hybrid IF model is derived from the model described in

> Barbara J. Breen, William C. Gerken, Robert J. Butera, Jr., Hybrid integrate-and-fire model of a bursting neuron, *Neural Computation, v.15* n.12, p.2843-2862, December 2003.

The TYP values shown below are from Breen et al. If we have used a different value, it is shown in parentheses.

There is no description of these parameters unless the description differs from the parameter used in Breen et al.

If a parameter that appears on the NEWSNED Cell Population Definition Panel for a Hybrid IF population does not appear in this list, it is documented under the MacGregor Parameters (see [MacGregor Parameters], page 5).

SNED: Time Constant for h
SIM2: taubar_h
BREEN: $\bar{\tau}_h$
UNIT: milliseconds
TYP: 10,000 (2,000)

SNED: NaP conductance
SIM2: g_NaP_h
BREEN: $g_{\mathrm{NaP-h}}$
UNIT: nanosiemens
TYP: 2.8 (3)

SNED: Half-voltage for h
SIM2: theta_h
BREEN: Θ_h
UNIT: millivolts
TYP: -48 (-51)

SNED: Slope for h
SIM2: sigma_h
BREEN: σ_h
UNIT: millivolts
TYP: 6 (5)

SNED: Half-voltage for activation
SIM2: theta_m
BREEN: $\Theta_{m-\mathrm{NaPh}}$
UNIT: millivolts
TYP: -40 (-43)

SNED: Slope for activation
SIM2: sigma_m
BREEN: $\sigma_{m-\mathrm{NaPh}}$
UNIT: millivolts
TYP: -6

SNED: Reset Voltage @h=0
SIM2: Vreset
BREEN: $V_{\mathrm{Reset}}(0)$
UNIT: millivolts
TYP: -47.359 (-42)

SNED: Threshold Voltage
SIM2: Vthresh
BREEN: $V_{\mathrm{Thresh}}(h)$
UNIT: millivolts
TYP: (-37)

In Breen et al, this parameter is a function of h. In the simulator, it is a constant.

SNED: Delta_h @ h=0
SIM2: delta_h
BREEN: $\Delta h(0)$
UNIT: dimensionless
TYP: -0.00078 (0)

SNED: Applied Current (Iapp)
SIM2: GE0
BREEN: I_{app}
UNIT: picoamps
TYP: 13-25 (0)

3.1.3 Pulmonary Stretch Receptor Parameters

If the Membrane Time Constant of the MacGregor model (see [TMEM], page 5) is set to 0, the cell model changes to a PSR model. The PSR model is intended for modeling Pulmonary Stretch Receptors. The steady-state outgoing firing rate from a PSR cell is the same as the steady-state incoming firing rate, but there is an exponential delay between changes in the incoming firing rate and changes in the outgoing firing rate. Only the firing rate of the incoming connections matters, not the strength or the equilibrium potential or the time constant. The firing rate is converted in the PSR model to a firing probability per time step.

SNED: Rise Time
SIM2: DCTH
UNIT: milliseconds
TYP: 500

If the incoming firing probability (IFP) is higher than the outgoing firing probability (OFP), the outgoing firing probability is updated on each time step as follows: $OFP = IFP + (OFP - IFP) * exp(-step/RiseTime)$. So the OFP will get most of the way up to the IFP in "Rise Time" milliseconds.

SNED: Fall Time
SIM2: DCG
UNIT: milliseconds
TYP: 500

If the incoming firing probability (IFP) is lower than the outgoing firing probability (OFP), the outgoing firing probability is updated on each time step as follows: $OFP = IFP + (OFP - IFP) * exp(-step/FallTime)$. So the OFP will get most of the way down to the IFP in "Rise Time" milliseconds.

SNED: Output Threshold
SIM2: Thr
UNIT: dimensionless
TYP: 0

The probability that there will be a spike on a particular time step is calculated as $OFP - OutputThreshold$ (OFP = outgoing firing probability). If the result is negative, there are no output spikes.

3.1.4 Fiber Population Parameters

SNED: Probability of Firing
SIM: Probability of fiber population firing
SIM2: probability
VAR: P
UNIT: dimensionless (probability)
TYP: .01-.05

Each fiber in the population generates an action potential at each time step with the probability specified by this parameter. Therefore, if the time step in seconds is T, the average firing rate of each fiber will be P/T.

SNED: Time to begin firing
SIM: Time fiber population begins firing
SIM2: start
VAR: INSTR
UNIT: time steps
TYP:

The probability of firing for all the fibers in the population is 0 until the time specified by this parameter, at which point the "Probability of Firing" parameter takes over. This parameter is in time steps from the start of the simulation.

SNED: Time to end firing
SIM: Time fiber population stops firing
SIM2: stop
VAR: INSTP
UNIT: time steps
TYP:

The probability of firing for all the fibers in the population is 0 after the time specified by this parameter. The "Probability of Firing" parameter has no effect after this time. This parameter is in time steps from the start of the simulation.

SNED: Random Number Seed
SIM: Random number seed for fiber population firing
SIM2: infsed
VAR: INFSED
UNIT:
TYP:

The firing patterns of the fibers in the population is determined by a pseudo-random number generator, and this parameter determines the sequence of numbers generated by the pseudo-random number generator. The firing pattern will always be the same for the same value of this parameter (but different for different fibers in the population). Different fiber populations should have different values of this parameter in order to generate different firing patterns.

SNED: Fibers in population
SIM: Number of fibers in population
SIM2: fiber_count
VAR: N
UNIT: count
TYP:

All the fibers in a population have the same values of the other parameters specified in this section, but each fiber in the population will have a different firing pattern because it uses different numbers from the pseudo-random number generator, so each fiber is simulated separately. This parameter specifies how many fibers in the population are simulated.

SNED: Fiber Comment
SIM: [not used]
SIM2: [not used]
VAR: [not used]
UNIT:
TYP:

This is not really a simulation parameter, but instead is just a label for a fiber population on NEWSNED's graphical display. It does not appear in the ".sim" file. It is listed here because it appears in NEWSNED on the fiber population parameters screen.

SNED: [not used]
SIM: Number of targets of fiber populations
SIM2: targetpop_count
VAR: NTGR
UNIT:
TYP:

This parameter is the number of connections between populations whose source is this fiber population. This is not explicitly entered in NEWSNED, because NEWSNED determines it from the network drawing. A single "connection between populations" represents a connection from each fiber of the source population to some number of cells of the target population. There can be multiple "connections between populations" from the same source population to the same target population, and each of those connections is counted separately for this parameter. However, if you set up multiple connections that all have the same source and target population, NEWSNED will only allow you to edit the parameters of the first of the connections.

3.1.5 Axon/Synapse Parameters

SNED: Conduction Time
SIM: Conduction time
SIM2: NCT
VAR: NCT
UNIT: time steps
TYP: 0-4

When an action potential fires in a cell of a source population, the effect will be felt at the target cell after some number of time steps. This parameter specifies the maximum number of time steps for this connection between populations. Each individual cell-to-cell or fiber-to-cell connection within this connection between populations will have its own conduction time, randomly chosen between 1 and the value of this parameter, inclusive, but always the same for a particular cell-to-cell or fiber-to-cell connection.

SNED: Number of terminals
SIM: Number of terminals per fiber
SIM2: NT
VAR: NT
UNIT: count
TYP: 10-100

Each cell or fiber in the source population makes the same number of connections in the target population, and this parameter specifies that number. The particular target cells that a particular source fiber or cell is connected to are chosen at random from among the target population, and they remain the same for the duration of the simulation. Multiple source cells or fibers can be connected to the same target cell, and more than one of the terminals from single source fiber or cell can be on the same target cell.

SNED: Synapse Strength
SIM: Synaptic strength
SIM2: STR
VAR: STR
UNIT: dimensionless - ratio to resting conductance
TYP: .0025-1.4

When a synapse fires on a cell in the target population (after the conduction time), the membrane conductance of the target cell increases instantaneously by the amount specified by this parameter. The value is specified in units of the resting membrane conductance, so 2 means twice the resting membrane conductance.

SNED: Random Number Seed
SIM: Random number seed
SIM2: INSED
VAR: INSED
UNIT:
TYP:

The particular target cells that are chosen for each source cell or fiber, and the associated conduction time, are determined by the value of this parameter. The same choices are always made for the same value of this parameter. This parameter should be set to a different value for each connection in order to get different connection patterns.

SNED: Synapse type
SIM: Synapse type
SIM2: TYPE
VAR: TYPE
UNIT: index or name
TYP: 1-6 or name

All the synapses associated with a particular connection between populations are of the type specified by this parameter. In the ".sim" file, this parameter is an index into the list of synapse types. The properties of a synapse type are specified by other parameters (see Synapse Parameters). In NEWSNED, this parameter is specified by using a name associated with the synapse index number (see the synapse type parameters "Synapse Name" and "Synapse Number").

SNED: [implicit]
SIM: Identity
SIM2: IRCP
VAR: IRCP
UNIT: index
TYP: 1-10

In the ".sim" file, for each source population, this parameter specifies the identity of the target cell population for each connection. It is not entered explicitly in NEWSNED, because NEWSNED determines it from the network drawing.

3.1.6 Synapse Type Parameters

SNED: Synapse Eq. Potential
SIM: Equilibrium potential for synaptic type
SIM2: EQ
VAR: EQ
UNIT: millivolts
TYP: -25 - 115

Each of the per-synapse conductances and the potassium conductance and the resting membrane conductance has an equilibrium potential associated with it. The membrane potential decays exponentially toward the weighted average of these equilibrium potentials, each weighted by its conductance. This parameter specifies the equilibrium potential for synapses of this synaptic type, relative to the resting membrane potential.

SNED: Synapse Time Constant
SIM: Time constant for synaptic action
SIM2: DCS
VAR: T
UNIT: milliseconds
TYP: 0.1 - 2.0

When a synapse fires on a target cell, the membrane conductance of the target cell increases instantaneously by the synaptic strength (see the STR Axon/Synapse parameter). This additional conductance decays exponentially toward 0 with the time constant specified by this parameter (except DCS = exp(-step/T).

SNED: Synapse Number
SIM: [implicit]
SIM2: syntype
VAR: [varies]
UNIT: index
TYP: 1-6

In NEWSNED, the synapse type parameters for each synapse type are associated with a synapse type index number, which is referenced by the Axon/Synapse parameter TYPE. This parameter specifies that index number. In the old format ".sim" file, this parameter is implicit in the order in which the synapse parameters are listed. In the new format ".sim" file, "syntype" starts with 0, so "syntype" is the NEWSNED synapse type minus 1.

SNED: Synapse Name
SIM: [not used]
SIM2: [not used]
VAR: [not used]
UNIT:
TYP:

This is not a simulation parameter, but just a label specified on NEWSNED'S "Synaptic Definitions Panel" for a synapse type, for use on NEWSNED's "Synapse Selection Panel", instead of using the synapse type index number. It does not appear in the ".sim" file.

3.1.7 Global Parameters

SNED: Length of simulation
SIM: Length of simulation in basic time steps
SIM2: step_count
VAR: LTSTOP
UNIT: time steps
TYP: 60000

The duration of the simulation in simulated time, in time steps. The size of a time step is specified by the Simulation step size in milliseconds, page 18.

SNED: Potassium Equilibrium Potential
SIM: Potassium equilibrium potential
SIM2: EK
VAR: EK
UNIT: millivolts
TYP: -10

Each of the per-synapse conductances and the potassium conductance and the resting membrane conductance has an equilibrium potential associated with it. The membrane potential decays exponentially toward the weighted average of these equilibrium potentials, each weighted by its conductance. This parameter specifies the equilibrium potential for the potassium conductance, relative to the resting membrane potential. This potassium equilibrium potential is the same for all cells in all populations in the model.

SIM: Simulation step size in milliseconds
SNED: Step size
SIM2: step
VAR: STEP
UNIT: milliseconds
TYP: .5 - 1

The value of the membrane potentials and the firing state are calculated at the interval specified by this parameter. This is also taken to be the duration of an action potential (see the cell population parameter “Change in potassium conductance with action potential”.) The value of this parameter also affects the time indicated by parameters specified in time steps.

SNED: Global Comment
SIM: [not used]
SIM2: [not used]
VAR: [not used]
UNIT:
TYP:

This is not really a simulation parameter, but just a label for the model on NEWSNED's graphical display. It does not appear in the ".sim" file. It is listed here because it appears in NEWSNED on the "Global Variable Definition Panel".

SNED: [implicit]
SIM: Total number of populations
SIM2: [not used]
VAR: NTPOPS
UNIT:
TYP:

The number of cell populations plus the number of fiber populations. Specified in ".sim" file. Determined by NEWSNED from the network drawing.

SNED: [implicit]
SIM: Number of fiber populations
SIM2: fiberpop_count
VAR: NFPOPS
UNIT:
TYP:

Specified in ".sim" file. Determined by NEWSNED from the network drawing.

SNED: [implicit]
SIM: Number of cell populations
SIM2: cellpop_count
VAR: NCPOPS
UNIT:
TYP:

Specified in ".sim" file. Determined by NEWSNED from the network drawing.

SNED: [implicit]
SIM: Maximum conduction time plus 1
SIM2: [not used]
VAR: MCTP1
UNIT:
TYP:

Maximum of the Axon/Synapse "Conduction Time" parameters, plus 1. Specified in ".sim" file. Determined by NEWSNED from the Axon/Synapse "Conduction Time" parameters.

SNED: [implicit]
SIM: Number of synaptic types
SIM2: syntype_count
VAR: SNTP
UNIT:
TYP:

Specified in ".sim" file. Determined by NEWSNED from the synapse type definitions.

SNED: [implicit]
SIM: Maximum number of targets
SIM2: [not used]
VAR: NTGMX
UNIT:
TYP:

Maximum of the Cell Population " Number of targets of cell populations" parameters. Specified in ".sim" file. Determined by NEWSNED from the network drawing..

SNED: [implicit]
SIM: Maximum number of cells per population
SIM2: [not used]
VAR: NCLS
UNIT:
TYP:

Maximum of the Cell Population "Population Size" parameters. Specified in ".sim" file. Determined by NEWSNED from the Cell Population "Population Size" parameters.

3.2 The NEWSNED Network Editor

3.2.1 Overview of NEWSNED

NEWSNED makes it possible to create a model by placing cell and fiber populations on a graphical display, drawing connections between them, and entering parameters in dialog boxes. NEWSNED can save the model and its graphical representation in a file with a ".snd" filetype, and can create the ".sim" file version of the model for use by the simulator. It is also possible to specify the output of the simulator from NEWSNED, and to invoke the simulator with those choices.

The tools for drawing the network are found on a menu that pops up when you right-click in the drawing area. Items on the '`File`' menu are for reading and writing ".snd" and ".sim" files, and for invoking the simulator. Global and synapse type parameters are accessed from the '`Build`' menu, and options associated with the network drawing are on the '`Option`' menu.

3.2.2 The Right Click Menu

Cell Population

When you select this item, the cursor will change to a circle. Left click where you want to place the cell population, and the "Cell Population Definition Panel" will pop up. The meaning of the parameters is documented in MacGregor Parameters, page 5. When you are happy with the parameter settings, left-click on '`ACCEPT VALUES`'.

Fiber Population

When you select this item, the cursor will change to a square. Left click where you want to place the fiber population, and the "Fiber Population Definition Panel" will pop up. The meaning of the parameters is documented in Fiber Parameters, page 12. When you are happy with the parameter settings, left-click on '`ACCEPT VALUES`'.

Create Axon/Synapse

When you select this item, the cursor will change to a cross. Left click near the source fiber or cell population, then at each corner of the path you want the connection to follow, and then near the target cell population. The path will be drawn as you click at the corners. When you click near the target population, the path will disappear and the "Synapse Selection Panel" will pop up. The meaning of the parameters is documented in Axon Parameters, page 14. When you are happy with the parameter settings, left-click on '`ACCEPT VALUES`', and the path will reappear.

Edit Fiber/Cell Population

This item will allow you to change the parameters of a previously defined population. When you select this item, the cursor will change to a question mark. Left click on the population you want to edit, and the appropriate parameter panel will pop up. When you are happy with the parameter settings, left-click on '`ACCEPT VALUES`'.

Edit Axon

This item will allow you to change the parameters of a previously defined connection. When you select this item, the cursor will change to a question mark. If there is only one connection between the source population and the target populations, left click on the source population and then the target population and the "Synapse Selection Panel" will pop up. When you are happy with the parameter settings, left-click on '`ACCEPT VALUES`'.

There are three different types of connections: normal, presynaptic, and postsynaptic (see [presynaptic], page 27). It is possible to have as many as three connections between the same source and destination population if the three connections are of different types. In that case, you can edit the the presynaptic connection by holding down the control key while you select the populations, and you can edit the the postsynaptic connection by holding down the shift key while you select the populations.

If you have more than one connection of the same type between the same source and destination population, you will only be able to edit one of them.

Delete Fiber/Cell

When you select this item, the cursor will change to a skull and crossbones. Left click on a population, and it will be deleted, along with all the connections for which it is a source or a target.

Delete Axon

When you select this item, the cursor will change to a skull and crossbones. Left click on the source population and then the target population and if there is a connection from that source to that target, it will be deleted. If there is more than one connection, it is handled the same way as selecting a connection to edit (see [Edit Axon], page 22).

Anterograde Rendering

When you select this item, the cursor will change to a cross. Left-click on a population, and all the connections for which that population is a source will be highlighted. All the other connections will be colored blue. The cursor will remain a cross. To return to normal mode,

left-click again in the display area. The color changes will only be visible if color mode is active (see [Option Menu], page 29).

Retrograde Rendering

When you select this item, the cursor will change to a cross. Left-click on a population, and all the connections for which that population is a target will be highlighted. All the other connections will be colored blue. The cursor will remain a cross. To return to normal mode, left-click again in the display area. The color changes will only be visible if color mode is active (see [Option Menu], page 29).

3.2.3 The File Menu

3.2.3.1 Miscellaneous

New

Clears the screen and sets the global parameters to their default values.

Load .snd

Load a file in ".snd" format, which was previously written by NEWSNED. Contains all the information necessary to run a simulation (and to display the model graphically), except that is does not specify what data is to be output by the simulation. If there is a file with the same name as the ".snd" file but with a filetype of ".ols" in the same directory as the ".snd" file, the data to output will be read from that ".ols" file. Otherwise, it can be specified by selecting "Spawn" from this menu (see [Spawn], page 24). Any data read from a ".ols" file can be changed there, as well.

Save .snd

Saves the current model in ".snd" format. The ".snd" file contains all the information necessary for NEWSNED to display the model, create a ".sim" file, and run a simulation. This file is in binary format and cannot be edited. The list of data to be output by the simulation (as specified on the Simulator Invocation Panel (see [Spawn], page 24)) is saved in a file with the same name as the ".snd" file but with a filetype of ".ols" in the same directory as the ".snd" file. The ".ols" file is a text file, and can be edited by hand.

Save .sim

Saves the current model in ".sim" format. Contains all the information necessary for the simulator to run a simulation, except for what data is to be output by the simulation. It does not contain the information necessary for NEWSNED to display the model, and it cannot be read by NEWSNED. This file is useful if you want to invoke the simulator "by hand", perhaps because you want to edit the ".sim" file by hand before you run it.

Postscript View/Print

Saves the current model to a PostScript® file named "newsned_network.ps" in the current directory, overwriting any existing file by that name, and opens it with an application named "gv" if one exists on the system. (The program "gv" is not part of this package, but it is available free of charge on most Linux systems.) The PostScript® file is a vector-graphics representation of the network as displayed on the screen, except that it is the entire network, not just the displayed portion.

Shell

If NEWSNED was invoked from the command line without placing it in the background, you will not get your command prompt back until you exit NEWSNED. This menu selection will give you a prompt on the command line, but you will need to exit from it before you go back to NEWSNED. A better way to do it is to start NEWSNED in the background, by typing `newsned&`, and you will have your command prompt while NEWSNED is running. Or just open a new command line window.

Quit

Exits NEWSNED immediately, without saving anything.

3.2.3.2 Spawn

This selection will pop up the '`Simulator Invocation Panel`', which allows you to select what output you want the simulator to generate, and to start ("spawn") the simulator.

The simulator can create two kinds of output file: ".bdt" files and wave files. The ".bdt" file is a text file that lists the spike times of selected cells or fibers in two columns, the cell/fiber ID in first column and the time (in time steps) in the second column. The ".bdt" files can also have analog data encoded in the ID field of additional entries. The wave files are text files that record the membrane potential at each time step for selected cells, for display by SNEDSKOP.

The '`Simulator Invocation Panel`' has buttons for selecting whether to '`Create .bdt File`', '`Create sNeD Scope File`', and/or '`Create analog entries`'. By default, all three of these buttons are selected.

Below the '`Create .bdt File`' button are widgets for selecting which cells' or fibers' spikes to include in the .bdt file. For each entry, you need to specify whether the spikes are coming from a cell or a fiber, the number of the population (cell and fiber populations are numbered separately), and the number of the cell or fiber within the population. (The population numbers and the cell/fiber counts appear on the network drawing.) As you choose the parameters for each entry, they appear in the text box below the entry widgets, but although it can be edited in the text box, the changes are ignored. The cell, fiber, and population numbers start with 1. If you set any of the numbers to 0, that entry and all subsequent ones will be removed from the list, but they will come back when you correct the entry. If the '`Create .bdt File`' button is not selected, these entries will be ignored. Up to 999 entries can be made in this list.

Below the '`Create sNeD Scope File`' button are widgets for selecting which cells' state variables to include in the wave files. They work like the ".bdt" file widgets, except that there is no button to select cell vs. fiber (because only cells have state in the model), and there is an extra entry labeled "Var #". The "Var #" determines the state variable to be plotted, as follows;

Var#	state variable
1:	membrane potential
2:	potassium conductance or hybrid IF "h" value
3:	threshold
4:	instantaneous population activity
>4:	binned population activity, where the "Var #" is the width of the bin in milliseconds

If the '`Create sNeD Scope File`' button is not selected, these entries will be ignored. Up to 1000 entries can be made in this list, but SNEDSKOP can only display 181 of them. The PostScript® output (see [Postscript View/Print], page 37) can display more, however.

If the variable being plotted is population activity ('`Var #`' >= 4), the value being plotted is not specific to any one cell in the population, so the '`Memb #`' entry is used to provide a scaling factor for the plot of population activity instead of being used to specify a member number. The vertical size of the plot in SNEDSKOP will be reduced by the indicated factor. The displayed spikes/second will still be correct.

Below the '`Create analog entries`' button are widgets that determine the analog data that will be included in the ".bdt" file. The ID field of the analog entries is

$$AnalogID * 4096 + 2048 + analogvalue.$$

The value of the '`Analog ID`' does not matter unless you will be merging the ".bdt" file with others, except that the value has to be greater than 0 and less than 25. If you are merging you probably want to use different '`Analog ID`' values.

The analog value is the overall firing rate for a specified '`Cell Population`'.

If '`Create analog entries`' is selected, a power spectrum of the analog signal will be displayed at the end of the simulation. see Section 3.3.3: Power Spectrum, page 32.

The '`Rate`' entry determines how often analog data will be written to the ".bdt" file. An analog entry will be made every 1000/rate milliseconds of sample time. Only the integer part of '`Rate`' is read, and the result of the division rounded down to an integer. If you enter a number greater than 1000, no analog data will be written.

The analog value is updated each time it is written, as follows:

$$analog = analog * exp(-(1000/Rate)/(IntegrationTime)) + (spikes * ScalingFactor)$$

The analog value is an integer. The quantity '`spikes`' is the number of action potentials that have occurred in the specified population since the last time the analog data was written. The '`Integration Time`' and '`Scaling Factor`' should be chosen so that the analog value does not exceed 2047. Note that the '`Integration Time`' is in milliseconds.

To generate a power spectrum, the '`Analog ID`' must be set to 1, the '`Cell Population #`' should be set to the number of the phrenic population, '`Rate`' should be set to 1000, '`Integration Time K`' should be set to 1, and '`Scaling Factor`' should be set to 1. see Section 3.3.3: Power Spectrum, page 32.

It is possible to run multiple invocations of the simulator in parallel from the '`Simulator Invocation Panel`'. Each invocation is given its own "spawn number". The sim file for each invocation is named "spawnN.sim", the ".bdt" file is named "spawnN.bdt', and the wave files are named "waveNN.nnnn", where "N" represents the spawn number (two digits in the wave file name). Your choices about what data to output are provided to the simulator in a file named "scriptN.txt". The "nnnn" in the wave file name represents a sequence number. No more than 100 time steps are written to each wave file, and then the sequence number is incremented and a new file is started. The sequence number starts with 0000, and the simulation will end after it gets to 9999.

The spawn number that will be used is indicated in the upper left of the panel, where there are up and down arrow buttons to change it. The spawn number will be highlighted if a simulator is running with that spawn number. Just below that is a button labeled "KILL", which will terminate the simulation running with that spawn number if there is one.

When a simulation run ends, either because it completed or because it was killed, the final state is written to a file named "end_state.NN", where NN is the spawn number. You can rename this file "initial_state.NN" by clicking the button labeled '`END->INITIAL`' after the simulation ends. When a simulation starts, it will load its initial state from the file initial_state.NN if it exists, so that you can pick up a simulation where you left off a previous one. To the right of the '`END->INITIAL`' button is a button labeled '`CLEAR INITIAL`', which will delete the initial_state.NN file if it exists, in case you want to start the simulation from scratch.

The simulator will start when you click the '`LAUNCH SIMULATOR`' button, and a SNEDSKOP window will pop up. If you did not select '`Create sNeD Scope File`', there will be nothing to look at with SNEDSKOP.

The files that are associated with a spawn number are written to the directory you were in when you started NEWSNED.

It is possible to update the ".sim" file during a simulation run. After making changes to the model, click on the '`MID-RUN UPDATE`' button, and the new model will be written to the ".sim" file for the currently displayed spawn number. The simulator re-reads the ".sim" file at intervals specified by the '`Sim Read Interval`' entry (0 for never), and the simulation continues with the new parameter values. The value in this field is the number of time steps between re-reads of the ".sim" file, and it only has an effect when a simulation is started by clicking on the '`LAUNCH SIMULATOR`' button.

The '`COPY TO NEXT PANEL`' button will copy the values in the currently displayed fields of the panel to the next spawn number, so that when you increment the spawn number, you will see all the same values instead of the default values, in case you want to run another simulation with the same or similar values.

The '`CANCEL`' button just makes the '`Simulator Invocation Panel`' go away, it does not stop any simulations in progress. In fact, the simulations will continue even after you quit NEWSNED.

The '`presynaptic`' button is for modeling presynaptic and postsynaptic connections. It does it by changing the behavior of the synapse types. When the button is pressed, the synapse types are grouped in three's. The first of the three is a normal synapse. The second of the three is "presynaptic": it modifies conductance changes before they reach the normal synapse. The last of the three is "postsynaptic": it modifies the membrane conductance due to the normal synapse.

A strength of 1 in a pre- or post-synaptic synapse has no effect on the normal synapse. A strength of less than 1 multiplies the normal synapse conductance, and if the strength is greater than 1, the amount by which it is greater than 1 is added to the normal synapse conductance.

A pre- or post-synaptic connection to a target cell simultaneously affects all the normal connections of the appropriate type to that target cell .

The '`turbo`' button causes "LAUNCH SIMULATOR" to run a (very) approximate version of the simulation that runs much faster.

The '`condi`' button causes "LAUNCH SIMULATOR" to run a simulation that writes a file named "condi.csv" before starting the simulation. The file contains a cell-by-cell list of all the connections in the network being simulated. The file is in comma-separated-value format for import to a spreadsheet. Each line of the file has the source population, source cell, target population, target cell, number of terminals, and synapse type for a connection.

3.2.3.3 Parallel Simulations

Newsned is capable of running multiple simulations in parallel, simply by running them with different spawn numbers. After starting the first simulation in the usual way, click the '`COPY`

`TO NEXT PANEL`' button (on the Simulator Invocation Panel) in order to copy the fiber and cell lists on that panel to the next spawn number. Then click the up arrow next to the Spawn # in the upper left to increment the spawn number. Then make any changes you wish to the model for the next simulation, and click '`LAUNCH SIMULATOR`'. Then go back to '`COPY TO NEXT PANEL`' and repeat, up to 20 times.

If you find yourself making the same set of changes repeatedly, this could get tedious. You might change a parameter in the model, and then create different versions of the the new model where each version runs with a different spawn number. If, each time you change the parameter, you have to generate the same versions for the different spawn numbers, you will be doing a lot of repetitive work.

For example, in a respiratory simulation, you might have variations of the model to simulate vagotomy, hypoxia, apneusis, and eupnea, where vagotomy, hypoxia, and apneusis are slight variations of the eupnea model. You might then change a parameter, and then run the four models with this new parameter value, and do this repeatedly.

Newsned has a facility that can automate this.

If there is a program in the working directory named "make_spawn1", `newsned` will run it when you start a simulation with spawn number 0 to create a version of the ".sim" file (see [Parameters], page 4) for spawn number 1 from the version for spawn number 0, and `newsned` will then start the spawn #1 simulation. It will then look for "make_spawn2", and if it exists, start spawn #2, and so on, for as many consecutive make_spawnN programs as it finds, up to 19.

So once you have created the make_spawnN programs, each time you change a parameter, all you have to do is to click '`LAUNCH SIMULATOR`' once to run all the versions of the model.

Included with the simulator are three sample make_spawn source files: make_spawn1.c, make_spawn2.c, and make_spawn3.c. Each one is a program that takes the original .sim file (spawn0.sim), modifies it, and creates a new .sim file. The program "make_spawn1" sets fiber population 1 to turn on at time 0 (the original model would have had it turning on at a different time). There is just one line of code in the program to make the change. The rest is boilerplate. You should be able to figure out how to make similar changes by looking at the code. Note that the population numbers are offset by 1 in the program vs. the model (0 in the program is 1 in the model).

After modifying the three source files to your requirements (and maybe creating more), put the modified files in the directory in which you compiled the simulator, and compile them by typing the following lines in that directory:

```
make make_spawn1
make make_spawn2
make make_spawn3
```

Then you would move the resulting make_spawn1, make_spawn2, and make_spawn3 programs into the directory from which you will run the simulator. Then when you click '`LAUNCH SIMULATOR`', you will get four simulations running in parallel with different parameters.

If you have a four processor system, the four simulations will run in the same time that a single simulation would take running by itself.

3.2.4 The Build Menu

Synapses

This menu item pops up the '`Synaptic Definitions Panel`', which allows you to define synapse types. The parameters are documented in Synapse Type Parameters, page 16.

Globals

This menu item pops up the '`Global Variable Definition Panel`', which allows you to set the parameters documented in Global Parameters, page 18.

In addition, it allows you to set the '`Phrenic recruitment equation`' and the '`Lumbar recruitment equation`'. See Recruitment Equations, page 41.

3.2.5 The Option Menu

Snap Engaged

If this option is selected, mouse click positions for placing or selecting items on the screen are rounded to the nearest multiple of 20 pixels.

Visible Identification

If this option is selected, the population number, population count, and (for fiber populations) firing probability are displayed inside each population circle or square.

Visible Comments

If this option is selected, the '`Fiber Comment`"s, '`Cell Population Comment`"s, and the global variables are displayed on the network drawing.

Tablature visible

If this option is selected, a window is displayed with the contents of the ".sim" file that would be written for the current model. It is updated as parameters are changed.

Documentation visible

If this option is selected, a window is displayed with documentation for the model. You can enter text in this window, and the first 1024 characters will be saved and restored with the ".snd" file.

Color Mode Active

This option determines whether the network drawing will be in color or black and white. Color mode needs to be active in order for Anterograde and Retrograde Rendering to work see Section 3.2.2: Right Click, page 21.

3.3 The Simulator

3.3.1 Cell Coordinates

When the simulator starts, it looks for symbolic links in the current directory with the same names as the cell populations in NEWSNED. The links should point to the .dx files created for the atlas to display the populations. Each link can be created at the command line with a command like the following:

```
ln -sf ~/atlas/coord_files/expanded_E-Aug_naa_nt_atlas_sim_v1.dx E-AUG
```

The simulator will create a file named "pop_names" with one line for each cell population. Each line will start with "ln -sf", followed by the .dx file name in quotes, followed by the cell population name (which is also the link name) in quotes. If the link is not found, the .dx file name will be empty, leaving just a pair of quotes where the .dx file name should be. This file is just for the user's information, though it can be used as a script to create the links after filling in the empty quotes.

The simulator will read the cell coordinates from the .dx files. If it can't, the simulation will still run but the cell coordinates will be all 0.

The simulator will also look for a file named "slice.sim" in the current directory. This file specifies the time and location of slices to be made through the brainstem during the simulation. Each slice cuts all the connections in the path of the slice at the specified time. If it can't read the file, the simulation will run but there will be no slices.

An example of the format of the "slice.sim" file is as follows:

```
file_format_version 5
slice_count 2
slice 0
stepnum 1372
ap 1
rl 1
dp 1
origin 1

slice 1
stepnum 46973
ap 2
```

The `ap`, `rl`, and `dp` lines are the A/P, R/L, and depth coordinates of a point through which the slice will pass. The plane of the slice will be perpendicular to the line between the point and obex. Any one or two coordinates can be omitted and they will be assumed to be 0. At least one coordinate must be non-zero so the orientation of the plane can be determined. If you want the plane to go through obex, include the "`origin 1`" line. Otherwise, it must be omitted. If it is included, the plane will go through obex instead of the specified point. The specified point will only be used to determine the orientation of the slice plane. As many slices as desired can be specified, and the "`slice_count`" line must match the number of slices that follow, and the "`slice`" lines must be consecutive and must start with 0. Each slice must be followed by an empty line, even the last slice, and the slices must be in `stepnum` order. The first line must be "`file_format_version 5`".

After reading the cell coordinates, the simulator will write a file in the working directory named "slice_groups.txt". This file will have one line for each cell. Each line will have a "`+`", "`-`", or a "`0`" for each slice, indicating which side of the slice the cell is on. A "`+`" means the side away from obex, a "`-`" means the same side as obex. If the slice goes through obex, a "`+`" means the same side as the point used to specify the orientation of the slice, and a "`-`" means the other side. A "`0`" means the slice goes through the cell. The "`+/-/0`" indications are followed on the line by the population number and cell number. The "`+/-/0`" indications are left-to-right in slice number order.

A connection between two cells will be cut by a slice if they are on opposite sides of the slice. If a slice goes through a cell, all its connections will be cut.

3.3.2 OLS Generation

At the beginning of every run, the simulator will write a file named "sample_cells.ols" to the current directory, overwriting any existing file by that name. It contains a subset of the cells in the model such that one of each endpoint of every connection between populations

in the model is represented. The file can be renamed and used as a ".ols" input file to NEWSNED. (see Section 3.2.3.1: Load .snd, page 23, for more information on .ols files.) It only contains information on cells to be written to the ".bdt" file, however, which corresponds to the '`Create .bdt File`' section of the '`Simulator Invocation Panel`'. The rest of the information can be entered by hand on the '`Simulator Invocation Panel`'. see Section 3.2.3.2: Spawn, page 24.

In addition, the simulator will write a file named "sample_cells_rosetta.txt", which documents the mapping between the cell numbers as seen in the ".bdt" file and the population and cell numbers as seen in NEWSNED. It also documents the same information listed by connection, together with the synapse type.

The cells included in "sample_cells.ols" are chosen using a pseudo-random number generator. If the file "sample_cell_seed" is found in the working directory, the seed for the random number generator it taken from it. Otherwise, the current time in seconds is used as a seed, and is written to "sample_cell_seed".

3.3.3 Power Spectrum

The simulator can create a power spectrum from a simulated phrenic signal.

If the option '`Create analog entries`' is checked on the '`Simulator Invocation Panel`' (see [Spawn], page 24), then NEWSNED will answer 'Y' to the simulator when it asks "SAVE POPULATION TOTAL TO DISK?" (see [Simulator Options], page 33), in which case at the end of a run, the simulator will generate a power spectrum from the analog signal written to the "spawnN.bdt" file during the run. The parameters for the analog signal should be set properly on the '`Simulator Invocation Panel`' as explained in Spawn, page 24.

The plot data for the spectrum is written to a file named "power_spectrum_plot_data_N", in the current directory, overwriting any existing file by that name. The "N" in the file name "power_spectrum_plot_data_N" is the spawn number (see [SNEDSKOP Waveform Display], page 36). If there is a program named "plotmtv" on the system, the simulator will use that to open the plot data. Otherwise, it will use a program named "gnuplot" if it exists. The programs "plotmtv" and "gnuplot" are not part of this package, but they are available free of charge on most Linux systems.

The simulator assumes that the analog data in the ".bdt" file is an integrated phrenic signal, and it processes it to determine the beginning and end of each burst. The spectrum is the average spectrum of the first 512 ms of each burst. The timing marks for the beginning and end of each burst are written as channels 97 and 98 to a file named "ieN.edt", and then that file and spawnN.bdt are combined to create mergeN.edt.

3.3.4 Simulator Options

When the simulator is invoked from the command line rather than from NEWSNED, it will ask the user a series of questions in order to determine the name of the input file and what output it should generate. When the simulator is invoked from NEWSNED, the answers to these questions are written to a file named scriptN.txt (N is the spawn number), and fed to "sim". It is also possible to invoke the simulator from the command line with answers previously generated by NEWSNED by typing:

```
./sim < scriptN.txt
```

at the command line, where N is the "spawn number" used by NEWSNED.

You can also create the script file in a text editor, but if you use a Windows program to do it, you should use the following command in Cygwin instead of the one above:

```
dos2unix < scriptN.txt | ./sim
```

The prompt for the first question asked by the simulator is

```
ENTER INPUT FILENAME ...OR...  <CR> TO EXIT >>
```

Don't put leading or trailing spaces in your answer, unless you want them in the file name. NEWSNED creates an input file named spawnN.sim.

The second question is

```
update parameters every how many steps?  (0 for never)
```

The simulator will consider re-reading the input ".sim" file at the intervals specified in answer to this question. The number represents simulator time steps. A ¡CR¿ is the same as a 0. This corresponds to '`Sim Read Interval`' on the NEWSNED '`Simulator Invocation Panel`'.

When it comes time to consider re-reading the input file, the simulator will first read from a file in the current directory named "pipe.nn", where nn is the "spawn number", with a leading zero if necessary to make it two digits. The spawn number is zero unless a different number is specified in answer to a later question. If it cannot open a file of that name, it will not re-read the ".sim" file this time. If it succeeds in opening the file, it will try to read a 1 or 2 digit (ASCII) number from the file. If it fails, it will keep trying to read from the open file. If the number it reads is 0, it will not re-read the ".sim" file this time. If the number it reads is a 1, it will keep trying, closing and opening pipe.nn between tries. If the number it reads is a 2, the simulator will immediately exit. If the number it reads is a 3, it will re-read the ".sim" file and write a 0 to pipe.nn. The purpose of all this is to communicate with another program (like NEWSNED) which can update the ".sim" file.

The third prompt is

```
E -- DRAW MEMBRANE POTENTIALS ON O-SCOPE
<CR> -- SKIP
```

If your response begins with an 'E' (upper or lower case), the simulator will write the current value of the membrane potential and spiking state of specified cells to a file on each time step. The desired cells are specified in answer to later questions. This question corresponds to '`Create sNeD Scope File`' on the NEWSNED '`Simulator Invocation Panel`'.

The values are written to files named "wave.nn.xxxx", where nn is the "spawn number", specified below, and xxxx is a four-digit sequence number. No more than 100 time steps are written to a wave file. The sequence number starts at 0. When 100 time steps have been written to that file, the sequence number is incremented and a new file is started. When the sequence number gets to 9999, it will roll over to 0, and the simulator will exit if wave.nn.0000 still exists.

The first line of each wave file is the number of time steps that are written to the file, and the size of the step in milliseconds. The second line is the number of cells whose value pairs are written to the file. Each of the next lines is the population number, the cell number, the variable number (from the Simulator Invocation Panel (see [Spawn], page 24)), the population type (MacGregor = 0, Hybrid IF = 1), and the population comment for one of the cells. And following that are the value pairs, one pair per line, a pair for each cell at the first time step, followed by a pair for each cell for the second time step, etc.

If you answered 'E' to the "MEMBRANE POTENTIALS" question, the next prompt will be:

```
What spawn number?
```

Your answer will affect the file names of the pipe.nn and wave.nn files as described above. The purpose of this is to communicate with other programs (like NEWSNED and SNEDSKOP) that are changing the ".sim" file or displaying the wave files, and that might have started up (spawned) the simulator more than once. Your answer must be a maximum of 5 characters. This question corresponds to the spawn number in the upper left corner of the NEWSNED '`Simulator Invocation Panel`'.

If you answered 'E' to the "MEMBRANE POTENTIALS" question, you will next get a prompt that looks like:

```
CELL #1 >>
```

In response to this, you should enter a cell population number and a cell number within the population, and (optionally) a variable number to plot (see [Variable Numbers], page 25), separated by commas. You will get the prompt again, until you enter a blank response, so that you can plot as many variables as you wish. This corresponds to '`Entry`' and '`Cell Pop #`', '`Memb #`', and '`Var #`' under '`Create sNeD Scope File`' on the NEWSNED '`Simulator Invocation Panel`'.

The next question will be:

```
SAVE SPIKE TIMES TO DISK? (Y/N)
```

You can answer with an upper- or lower-case Y or N. If you answer Y, the spike times of cells or fibers that you specify will be written to disk each time one of them occurs. The

file format will be ".adt" or ".bdt", depending on the answer to the next question. This corresponds to '`Create .bdt file`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "SAVE SPIKE TIMES" question, the next question will be:

```
SAVE POPULATION TOTAL TO DISK? (Y/N)
```

If you answer Y, the file format for the spike times will be .bdt, and it will include analog information that represents the overall firing rate for the cell population that you specify. This corresponds to '`Create analog entries`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "POPULATION TOTAL" question, the next prompt will be:

```
ENTER ANALOG ID (1-3)
```

Each line in a ".bdt" file has a 5 digit ID field and an 8 digit timestamp field. For analog data, the value in the ID field is $analogid*4096+value+2048$, where "analogid" is the analog ID and "value" is the analog value. Your answer to this question sets the value of analogid. The value doesn't really matter, unless you have other analog data from another source that you want to merge later with the data from this file. The value must be between 1 and 3 inclusive. This corresponds to '`Analog ID`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "POPULATION TOTAL" question, the next prompt will be:

```
ENTER POP. NO.
```

Enter the number of the cell population for which you want analog data recorded in the ".bdt" file. This corresponds to '`Cell Population #`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "POPULATION TOTAL" question, the next prompt will be:

```
ENTER RATE (/sec.)
```

A line will be added to the ".bdt" file with analog data this many times per second. It is read in as an integer, and data will be written to the file every 1000/RATE milliseconds (the result of the division is rounded down to an integer). If you don't want the analog data, just enter a number larger than 1000. This corresponds to '`Rate`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "POPULATION TOTAL" question, the next prompt will be:

```
ENTER TIME CONSTANT
```

The analog value written to the file is updated each time it is written, as follows:

$analog = analog * exp(-(1000/RATE)/TC) + (spikes * SF)$

where '`spikes`' is the total number of action potentials in the population since the last time analog data was written, SF ('`SCALING FACTOR`') is specified in the next question, '`RATE`' is the answer to the previous question and TC ('`TIME CONSTANT`') is the answer to this question.

The '`TIME CONSTANT`' and '`SCALING FACTOR`' should be chosen so that the analog value does not exceed 2047 (larger values will be truncated). Note that the '`TIME CONSTANT`' is in milliseconds. This corresponds to '`Integration Time K`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "POPULATION TOTAL" question, the next prompt will be:

```
ENTER SCALING FACTOR
```

This is the '`SCALING FACTOR`' parameter discussed in the previous question. This corresponds to '`Scaling Factor`' on the NEWSNED '`Simulator Invocation Panel`'.

If you answered Y to the "SAVE SPIKE TIMES" question, the next question will be:

```
ENTER OUTPUT FILENAME
```

This is the name of the .adt or .bdt file that the spike times will be written to. You should specify the whole file name, including the .adt or .bdt extension if you want it. Don't put leading or trailing spaces in your answer, unless you want them in the file name. NEWSNED answers "spawnN.bdt" to this question, where "N" is the spawn number.

If you answered Y to the "SAVE SPIKE TIMES" question, the next question will be:

```
ENTER I.D. CODES OF CELLS AND FIBERS WHOSE
SPIKE TIMES WILL BE INCLUDED IN THE OUTPUT FILE.
FORMAT: F(iber) or C(ell), POPULATION ID,CELL/FIBER ID <CR> (A1I5,I5)
NOTE: NO COMMA ALLOWED BETWEEN F/C AND POPULATION ID
CHANNEL # 1 >>
```

The first character of your response should be an upper- or lower-case C or F to indicate whether the next number is a cell or fiber population number, followed by the population number and the number of the cell or fiber within its population, with a comma between population= and cell. You will get the prompt again, until you enter a blank response or a 0, so that you can enter all the cells or fibers for which you want spike times recorded. This corresponds to '`Entry`' and '`C/F`' and '`Pop #`' and '`Memb #`' under '`Create .bdt File`' on the NEWSNED '`Simulator Invocation Panel`'.

3.4 SNEDSKOP Waveform Display

SNEDSKOP is invoked, either by NEWSNED or "by hand" from the command line, with a numeric argument indicating the spawn number of the wave files that it should look for, and it will display the contents of those files as waveforms on the screen. It looks for files in the current directory with names of the form "wave.00.0000". The first two digits in the name are the spawn number, and the last four are the sequence number. Each wave file contains 100 time steps of the waveforms in sequence number order. Once per second, SNEDSKOP checks to see whether there are new wave files, and if so, it adds them to the displayed waveforms, so it can display the waveforms as NEWSNED generates them.

3.4.1 SNEDSKOP File Menu

Previous Block

Next Block

The horizontal scrollbar on the waveform display allows you to scroll around in an underlying 20,000 pixel wide window. But for a waveform longer than 1,250 time steps with .25 time compression or 320,000 time steps with 64 time compression see Section 3.4.3: Timebase Menu, page 38, that won't be enough. '`Next Block`' and '`Previous Block`' allow you to adjust the portion of the waveform that is displayed in the underlying 20,000 pixel wide window. The amount by which they move the displayed portion is 200 * Time Compression time steps, except that it moves at least 100 time steps.

Postscript View/Print

Saves the waveforms to a PostScript® file named "snedskop_N.ps" in the current directory, overwriting any existing file by that name, and opens it with an application named "gv" if one exists on the system. (The program "gv" is not part of this package, but it is available free of charge on most Linux systems.) The "N" in "snedskop_N.ps" is the spawn number (see [SNEDSKOP Waveform Display], page 36). The PostScript® file is a vector-graphics representation of the waveforms as displayed on the screen, except that it includes the entire duration of all the waveforms, not just the displayed portion.

Quit

Exits SNEDSKOP immediately.

3.4.2 SNEDSKOP Option Menu

Action Potentials Visible

For each cell whose membrane potentials are recorded, the wave files record both the membrane potential and whether there was an action potential, at each time step. For each cell at each time step, SNEDSKOP will draw a single vertical line of fixed height in a different color from the membrane potential, on top of the membrane potential, to indicate the presence of an action potential at that time step. This menu item allows you to turn that display of action potentials on and off.

Color Mode Enabled

By default, the membrane potentials, action potentials, time markers, and voltage markers are drawn in different colors on a black background. This menu item will toggle between that color display in one mode, and white lines on a black background in the other mode.

3.4.3 Timebase Menu

The Timebase Menu allows you to select timebase settings that range from '`.25 Time Compression`' to '`64 Time Compression`' in powers of 2. The waveforms are displayed st $4/(TimeCompression)$ pixels per time step. If '`64 Time Compression`' is not enough, repeatedly selecting '`64 Time Compression`' will continue to give more time compression, with no upper limit, but the menu will continue to indicate '`64 Time Compression`'. The time markers (see [Time Markers Menu], page 38) will be the only indication of the degree of time compression beyond 64.

3.4.4 Spacing Menu

This menu allows you to select a vertical spacing between waveform baselines of 11, 15, 25, 50, 75, 100, 150, or 300 pixels. The menu items are labeled '`Point Spacing`', but the numbers are actually pixel spacings.

3.4.5 Vertical Gain Menu

This menu allows you to select a vertical scale of .1, .25, .5, 1, 1.5, 3, or 6 pixels per millivolt. The menu item labeled '`1.5 X Gain`', for example, means a vertical scale of 1.5 pixels per millivolt.

3.4.6 Time Markers Menu

SNEDSKOP normally draws vertical red lines the height of the window at regular intervals across the window. This menu allows you to select whether those intervals will be 5, 10, 50, 100, 500, or 1000 time steps starting from the left edge of the underlying 20,000 pixel window see Section 3.4.1: Block, page 37. For example, the menu item that says '`5 Unit Markers`' means time markers at 5 time-step intervals. The default is 100 time steps, which corresponds to the number of time steps normally stored in a wave file, and therefore to the increments in which the window is updated. This menu also gives you the option to turn the time markers off.

3.4.7 Voltage Markers Menu

SNEDSKOP will draw a horizontal line for each trace as a reference level for that trace. The Voltage Markers Menu makes it possible to change the reference level. The menu items are labeled in millivolts, but each trace can have different units, and the reference level setting is displayed with each trace in the correct units for that trace. The reference levels can also be changed with the up and down arrow keys, and this way you are not limited to the selections on the Voltage Markers Menu. The last item on the Voltage Markers Menu, "No Voltage Markers", turns of all the voltage markers.

The label displayed with each trace starts with the population number and the Cell Population Comment.

If the trace is displaying population activity, the next item in the label is the time interval over which the firing rate is calculated (similar to the bin size of a CTH). And the final item is the firing rate in spikes per second at the voltage marker line, which changes as the voltage marker lines are moved.

If the trace is displaying some value other than population activity, the Cell Population Comment is followed by the name of the variable being displayed, which will be one of the following:

Vm: membrane potential
gK: potassium conductance
Thr: firing threshold
h: hybrid IF inactivation variable

followed by the value of the variable at the voltage marker line and the units in which it is measured. The value of the variable at the voltage marker line changes as the voltage marker lines are moved.

If "h" is being displayed, there are no units because "h" is dimensionless. If "gK" is being displayed, there are no units because it is a ratio to resting membrane conductance. If "Vm" or "Thr" are being displayed, the units are "mV abs" for a hybrid IF cell because the voltage is absolute (the actual voltage across the membrane), and "mV rel" for the MacGregor model because it is relative to the resting potential.

Which value is being displayed is set on the Simulator Invocation Panel of NEWSNED (see [Variable Numbers], page 25).

The label display can be turned on and off by pressing the "`l`" (ell) key. When scrolling left and right with the scroll bar, the labels can get duplicated across the screen. This can be fixed by hitting the "`l`" key twice.

3.5 The Lung Model

3.5.1 Interface to the Network Model

3.5.1.1 Motor Populations

The simulator includes a model of the lungs. It takes as input the mean firing rate of the phrenic motor, the lumbar (abdominal muscle) motor, and the inspiratory and expiratory laryngeal motor neuron populations, and produces as output the lung volume, alveolar pressure, and tracheal flow.

To use the lung model, you must label the above-mentioned populations appropriately (in Newsned), as follows:

Population	Name must contain the word
Phrenic	phrenic
Lumbar	lumbar
Inspiratory Laryngeal	ILM or PCA
Expiratory Laryngeal	ELM or TA

Any of the letters in the word may be upper or lower case.

Also, the word '`pre`' must not appear in the name. This is also case insensitive.

You may have more than one phrenic and more than one lumbar population. The word "phrenic" or "lumbar" in the additional populations names must be followed immediately by a small positive integer (e.g. phrenic1).

3.5.1.2 Recruitment Equations

The activation of the diaphragm by the phrenic and the abdominal muscles by the lumbar are determined by equations provided by the user. The equations are entered in the appropriate fields of the '`Global Variable Definition Panel`' on the Build menu of `newsned` (see The Build Menu, page 29). The phrenic recruitment equation (actually an expression) should be in terms of variables named '`P0`', '`P1`', etc., – one for each of the phrenic populations. Each of these variables represents the mean instantaneous firing rate of the corresponding population. '`P0`' is the variable for the un-numbered phrenic population. If no equation is entered, it defaults to '`P0/100`'. The value of the expression is the muscle activation, where 0 is no activation and 1 is maximum activation. If the value is greater than 1, it acts like 1, and if it is less than 0, it acts like 0. The lumbar recruitment equation is similar, except that the variables are '`L0`' etc., and the default equation is '`L0/20`'.

If you have two phrenic populations, and the firing rate of individual phrenic neurons maxes out at 100 spikes/second, an appropriate phrenic recruitment equation might be '`(.3*P0+.7*P1)/100`'. This approximates the plot in Fig. 1 of Mantilla and Sieck (2011). A max rate of 100 spikes/second for phrenic is suggested by Nail et al. (1972).

The laryngeal motor neurons don't have recruitment equations, but if the simulator finds a file named "lmmax" in the current directory, it will use the first number in that file as the firing rate for maximum activation for both ILM and ELM, instead of the default of 40 spikes/second.

3.5.1.3 Pulmonary Stretch Receptors

The volume output of the lung model can be use to control the injected current of any cell population, thereby creating a pulmonary stretch receptor. To do this, simply enter an expression involving the variable '`V`' (representing lung volume) in the DC Injected Current entry of the Cell Population Definition Panel instead of just a number. The simulator will evaluate the expression at every step using the lung volume at that step to determine the injected current at that step. If no cell population has such an expression, the lung model will not be used, which will speed up the simulation, and the lung model outputs will not be available for plotting with Snedskop.

For example, if you want an injected current that is proportional to lung volume, with 0 at RV, an appropriate expression might be '`.5*RV`'. Or if you want an injected current that is 0 at TLC and rises with decreasing lung volume, you might use '`.5*(RV-TLC)`'.

3.5.1.4 Variable Plots

The volume, pressure, and flow and other variables can be plotted with Snedskop, by specifying the appropriate Var# in the Snedskop section of the Simulator Invocation Panel of Newsned.

Var#	State Variable	Units
-1:	lung volume	%VC, relative to RV
-2:	tracheal flow	%VC per second expiration positive (up)
-3:	alveolar pressure	cmH_2O
-4:	Phr_d: diaphragm activation	dimensionless ratio to max
-5:	u: abdominal muscle activation	dimensionless ratio to max
-6:	lma: net laryngeal muscle activation	dimensionless ratio to max
-7:	Vdi: diaphragm volume	liters
-8:	Vab: abdominal volume	liters
-9:	Vdi_t: derivative of Vdi	liters/sec
-10:	Vab_t: derivative of Vab	liters/sec
-11:	Pdi: transdiaphragmatic pressure	cmH_2O
-12:	Pab: abdominal pressure	cmH_2O
-13:	PL: transpulmonary pressure	cmH_2O
-14:	Phr_d: diaphragm activation limited, $0 \rightarrow 1$	dimensionless ratio to max
-15:	u: abdominal muscle activation limited, $0 \rightarrow 1$	dimensionless ratio to max
-16:	lma: net laryngeal muscle activation limited, $-1 \rightarrow 1$	dimensionless ratio to max

The population and member number entries are not needed for their original purposes for these variables, so they are used to specify the offset and scaling of the plot. The Cell Pop # specifies the value at the baseline in units of the variable and the Memb # specifies the number of units of the variable per pixel (at 1X gain). Since the user may wish to specify a fractional value for the offset or scale, and since these fields can only take integers, the user must multiply the desired value by 10,000 and round to the nearest integer before entering the value.

3.6 Utilities

3.6.1 renumber

The "renumber" program renumbers ".ols" files.

The ".ols" file generated by NEWSNED contains the list of cells, fibers and populations that are to be written by the simulator to the "spawnN.bdt" file and to the "wave.NN.*" files. It

is a text file and can be edited by hand, but the lines are numbered and the numbers must be in order. This makes it awkward to edit by hand, because if the lines are rearranged or deleted, every line number might have to be updated.

The "renumber" utility will do this renumbering for you. After editing the ".ols" file without regard to the line numbers, type

```
renumber whatever.ols
```

and the file '`whatever.ols`' will be renumbered. A backup file is created with the name '`whatever.ols.orig`', before the renumbering is done. Any previous backup file will be overwritten.

An interpreter for the "perl" programming language must be installed on the system at /usr/bin/perl for this utility to work.

3.6.2 pickwave

The "pickwave" program picks a waveform out of a specified set of wave.* files, and prints the values as text to standard output (the screen). If you type

```
pickwave --help
```

at the command line, it will type

```
Usage: pickwave [OPTION...] [FILE...]
picks a waveform out of a specified set of wave.* files

  -n, --spawn=N           take input from wave.* files with spawn number N
  -s, --spikes            pick spike data instead of waveform data
  -w, --wave=N            output data for wave number N
  -?, --help              Give this help list
      --usage             Give a short usage message
  -V, --version           Print program version

Mandatory or optional arguments to long options are also mandatory or
optional for any corresponding short options.

The wave number for the -n option is the position in the wave.* file.
The first position is 1, not 0.

Report bugs to <roconnor@hsc.usf.edu>.
```

The program is run from the command line, and may be useful for importing the data into other programs such as Matlab.

3.6.3 pickedt

The "pickedt" program picks a waveform out of a specified .bdt file, and prints the values as text to standard output (the screen). If you type

```
pickedt --help
```

at the command line, it will type

```
Usage: pickedt [OPTION...] [FILE...]
picks a waveform out of specified .bdt or .edt file(s)

  -a, --analog_id=N          output data for analog id N
  -c, --ids                  output all ids on stderr
  -o, --offset=N             add N to each analog value
  -s, --spike_id=N           output data for spike id N
  -t, --starttime            first line of output is start time in units of
                             the sample period
  -?, --help                 Give this help list
      --usage                Give a short usage message
  -V, --version              Print program version

Mandatory or optional arguments to long options are also mandatory or
optional for any corresponding short options.

The output is text, one value per line, on standard output

Report bugs to <roconnor@hsc.usf.edu>.
```

This program is used by the power_spectrum.sh utility. It can also be run from the command line, and may be useful for importing the data into other programs such as Matlab.

3.6.4 spectrum

The "spectrum" program computes the average power spectrum of phrenic bursts. Type

```
spectrum --help
```

at the command line, and it will type

```
Usage: spectrum [OPTION...] [FILE...]
computes the average power spectrum of phrenic bursts

  -d, --detrend              remove the best-fit line from the phrenic burst
                             data
  -f, --filter               bandpass filter the input .3 to 200 Hz before
                             processing (default is no filter)
  -m, --spawnnum=N           the I and E pulses are written to ieN.edt
  -n, --fftsz=N              power spectrum is of first N samples of each
                             phrenic burst (overrides -p)
  -p, --period=N             power spectrum is of first N seconds of each
                             phrenic burst (default .5)
  -r, --rectify              subtract the mean and then take the
                             absolute value of the input data
  -s, --stepsize=N           set stepsize of input data to N ms (default .5)
  -t, --threshold=N          controls the placement of the I pulse - larger
                             = later (default .025)
  -w, --window               apply a Hann window to the phrenic burst data
                             after any detrend
  -?, --help                 Give this help list
      --usage                Give a short usage message
  -V, --version              Print program version

Mandatory or optional arguments to long options are also mandatory or
optional for any corresponding short options.

The input is binary C floats representing phrenic population activity.
The output is lines of text on standard output, two values per line
The first value is frequency in Hz, the second is power

If the -f option is used, the filtered input is written to a file named
"filtered" in the current directory

FFT optimization information is read from and written to a file named
"wisdom" in the current directory.

Report bugs to <roconnor@hsc.usf.edu>.
```

This program is used by power_spectrum.sh. It can also be run from the command line.

3.6.5 txt2flt

The "txt2flt" program converts text numbers to binary numbers.

If you type

```
txt2flt --help
```

at the command line, it will type

```
Usage: txt2flt [OPTION...] [FILE...]
converts text numbers to binary numbers

  -d, --double             output binary double precision C floating point
                           values
  -f, --float              output binary single precision C floating point
                           values (default)
  -i, --int                output binary C int (integer) values
  -s, --short              output binary C short integer values
  -?, --help               Give this help list
      --usage              Give a short usage message
  -V, --version            Print program version

The output is binary numbers on standard output.

Report bugs to <roconnor@hsc.usf.edu>.
```

This program is used by the power_spectrum.sh utility. It can also be run from the command line.

3.6.6 wave2ps

The "wave2ps" program generates a PostScript® file from a specified set of wave.* files. It is invoked as follows:

```
wave2ps -s N
```

where N is the spawn number of the wave files. The PostScript® text will be sent to standard output (the screen), so you would normally want to redirect it to a file, as follows:

```
wave2ps -s N > whatever.ps
```

This program is used by SNEDSKOP to generate PostScript® output, but can also be run from the command line.

3.6.7 merge

The "merge" program combines two ".bdt" or ".edt" files into a single file. The input files can be of different types. The types are determined by the contents of the files, not by the names. The type of the output is ".edt" if either input is, otherwise it is ".bdt".

It is invoked as follows:

```
merge input1.[eb]dt input2.[eb]dt > output.[eb]dt
```

The output file will be sent to standard output (the screen), so you would normally want to redirect it to a file, as shown above.

The "merge" program can also be used to merge more than two input files, as in this example:

```
merge input1.bdt input2.edt | merge input2.bdt > output.edt
```

This program is used by power_spectrum.sh, but can also be run from the command line.

3.6.8 power_spectrum.sh

The "power_spectrum.sh" program generates a power spectrum from analog channel 1 of a ".bdt" file named spawnN.bdt. It takes two arguments. The first is the the value of N in spawnN.bdt. The second is the step size of the analog data in milliseconds.

The plot data for the spectrum is written to a file named "power_spectrum_plot_data_N", in the current directory, overwriting any existing file by that name. If there is a program named "plotmtv" on the system, power_spectrum.sh will use that to open the plot data. Otherwise, it will use a program named "gnuplot" if it exists. The programs "plotmtv" and "gnuplot" are not part of this package, but they are available free of charge on most Linux systems.

The "power_spectrum.sh" program assumes that the analog data in the ".bdt" file is an integrated phrenic signal, and it processes it to determine the beginning and end of each

burst. The spectrum is the average spectrum of the first 512 ms of each burst. The timing marks for the beginning and end of each burst are written as channels 97 and 98 to a file named "ieN.edt", and then that file and spawnN.bdt are combined to create mergeN.edt.

The "power_spectrum.sh" program is used by the simulator to automatically generate a power spectrum at the end of a run, but it can also be run from the command line.

Chapter 4

Lung Model Internals

4.1 The Top Level

At the top level, there are just two equations in the model (the variable names are the same as those used in the source code found in the file named lung.c):

```
-VL_t * Rrs - sigma_L + (fa + Fdi) * sigma_di + Pica - sigma_rc = 0

sigma_ab + VL_t * Rrs + sigma_L - sigma_di = 0
```

VL_t is $d(VL)/dt$, the rate of change of lung volume in liter/sec, and is therefore the tracheal flow, with positive values during inspiration.

Rrs is airway resistance in $cmH_2O/(l/s)$.

sigma_L is the recoil pressure of the lung in cmH_2O due to its elastance, positive values acting to contract the lung.

fa is the fraction of the rib cage exposed to abdominal pressure. The remainder is exposed to pleural pressure. This is a dimensionless value between 0 and 1.

Fdi is the fraction of the diaphragm pressure that acts to expand the rib cage by way of insertional forces from the diaphragm acting on the lower ribs. This is a dimensionless value equal to .15, from Loring and Mead (1982) where it is $c \cdot \text{Adi}/\text{Arc} = 0.25 \cdot 0.6$. (Their Fdi is defined differently.)

sigma_di is the recoil pressure of the diaphragm in cmH_2O due to muscle tension, elastance, and resistance, positive values acting to contract the diaphragm.

Pica is the pressure in cmH_2O generated by the intercostal and accessory muscles, positive values acting to expand the rib cage.

sigma_rc is the recoil pressure of the rib cage due to its elastance and resistance, positive values acting to contract the rib cage.

sigma_ab is the recoil pressure of the abdominal wall as seen at the level of the diaphragm in cmH_2O due to muscle tension, elastance, and resistance in the abdominal wall,

positive values acting to contract the abdominal wall.

Each of these values is a function of the four motor outputs of the network model (see next section), and the diaphragm and abdominal wall volumes (`Vdi` and `Vab`) and their time derivatives (`Vdi_t` and `Vab_t`). At each time step, the motor outputs are known from the network model, the volumes are known from the end of the last time step, and the two equations are solved for the two derivatives to calculate the volumes at the next time step.

The volumes are in liters and their derivatives are in liters/sec.

4.1.1 The Motor Outputs

The phrenic motor output from the network model at each time step is calculated by first counting the total spikes from all the cells in each phrenic population, and dividing by the duration of a time step in seconds and the number of cells in each population, to get an instantaneous firing rate in spikes/sec/cell for each phrenic population. These numbers are combined by the phrenic recruitment equation (see see Section 3.5.1.2: Recruitment Equations, page 41) to generate the raw diaphragm muscle activation, with 0 indicating no activation and 1 indicating maximum activation. This raw activation is integrated by means of a simulated RC filter with a time constant of 60 ms, and then limited to a range of 0 to 1 to generate the diaphragm activation '`Phr_d`'.

The lumbar motor output is handled the same way, except that the lumbar populations and recruitment equation are used to generate the abdominal muscle activation '`u`'.

The Inspiratory Laryngeal Motor output from the network model at each time step is calculated by first counting the total spikes from all the cells in the ILM population, and dividing by the duration of a time step in seconds and the number of cells in the population, to get an instantaneous firing rate in spikes/sec/cell.

The Expiratory Laryngeal Motor output is handled the same way, except with the ELM population.

The ELM firing rate is then subtracted from the ILM firing rate, and the difference is divided by a maximum firing rate (see Section 3.5.1.2, last paragraph), and integrated by means of a simulated RC filter with a time constant of 35 ms, and then limited to a range of -1 to 1, to generate the net laryngeal muscle activation, '`lma`', where 1 indicates a maximally open airway, -1 indicated fully closed, and 0 indicates resting diameter.

4.2 The Pressure Equations

To derive the top level model equations, we start with pressure balance equations for the rib cage, lungs, diaphragm, and abdominal wall, and the pressure drop across the larynx. In this model, we ignore inertial forces, which are forces necessary to accelerate masses, in effect treating the components of the respiratory system as if they were massless. For evidence that

(at least some) inertial forces are negligible, see Mead (1956). For a precedent for ignoring inertial forces in a lung model, see Younes and Riddle (1981). In the absence of inertial forces, the forces on an object must sum to zero (Newton, 1687).

All the pressure variables below are in units of cmH_2O.

4.2.1 Pressure balance on the rib cage

```
(1 - fa) * Ppl + fa * Pab + Pica + Fdi * sigma_di = sigma_rc
```

The inner surface of the rib cage above the diaphragm sees the pleural pressure, `Ppl`. The rest of the inner surface of the rib cage (the zone of apposition) sees the abdominal pressure, Pab. The variable '`fa`' represents the fraction of the rib cage that sees `Pab`, so the total pressure from those sources is `(1 - fa) * Ppl + fa * Pab`. Positive values of these pressures tend to expand the rib cage.

The action of the intercostal and accessory muscles also tends to expand or contract the rib cage. This effect is represented by the variable `Pica`, which is an equivalent pressure defined so that positive values tend to expand.

When the diaphragm contracts, the force at the diaphragm insertions tends to lift the rib cage, and through the pivoting action of the ribs at the spine, tends to expand it as well. The equivalent pressure generated by this effect is taken to be proportional to the pressure generated by the diaphragm, `sigma_di`, with `Fdi` being the proportionality constant.

These pressures that tend to expand the rib cage have to be balanced by the recoil pressure of the rib cage generated by its elastance. This pressure, `sigma_rc`, is positive when it tends to contract the rib cage.

4.2.2 Pressure balance on the lung

```
Palv - Ppl = sigma_L
```

The alveolar pressure, `Palv`, is the air pressure inside the lung, with positive values tending to expand. The pressure outside the lung is the pleural pressure, `Ppl`, and positive values of this tend to contract the lung. The difference in these pressures, `Palv-Ppl`, positive values of which tend to expand the lung, must be balanced by the recoil pressure of the lung, `sigma_L`, generated by its elastance, positive values of which tend to contract the lung.

4.2.3 Pressure balance on the diaphragm

```
Pab - Ppl = sigma_di
```

The abdominal pressure, Pab, is the pressure inside the abdomen, with positive values tending to expand the diaphragm. The pressure on the other side of the diaphragm is

the pleural pressure, Ppl, and positive values of this tend to contract the diaphragm. The difference in these pressures, Pab-Ppl, positive values of which tend to expand the diaphragm, must be balanced by the recoil pressure of the diaphragm, sigma_di, generated by its elastance and muscle contraction, positive values of which tend to contract the diaphragm.

4.2.4 Pressure balance on the abdominal wall

```
Pab = sigma_ab
```

The abdominal pressure, `Pab`, is the pressure inside the abdomen, with positive values tending to expand the abdominal wall. This must be balanced by the recoil pressure of the abdominal wall, `sigma_ab`, generated by its elastance and muscle contraction, positive values of which tend to contract the abdominal wall.

4.2.5 Pressure drop across the larynx

```
Palv = -VL_t * Rrs
```

The alveolar pressure, `Palv`, is the air pressure inside the lung. This must be balanced by the pressure drop across the larynx, which, in the hydraulic analog of Ohm's law, is proportional to the flow out of the lungs, with `Rrs` being the proportionality constant. `VL_t` is the time derivative of lung volume, so positive values represent flow into the lungs, so `-VL_t` is the flow out of the lings.

4.3 Deriving the Model Equations from the Pressure Equations

By combining the pressure equations above, we can eliminate the variables `Pab`, `Ppl`, and `Palv`, leaving two equations.

To derive the first model equation from the pressure equations, start with the pressure balance on the rib cage:

```
(1 - fa) * Ppl + fa * Pab + Pica + Fdi * sigma_di = sigma_rc
```

Rearrange by collecting `Pab` and `Ppl` together, which both multiply `fa`:

```
Ppl + (Pab - Ppl) * fa + Pica + Fdi * sigma_di - sigma_rc = 0
```

Using the diaphragm pressure balance equation, replace `(Pab - Ppl)` with `sigma_di`:

```
Ppl + sigma_di * fa + Pica + Fdi * sigma_di - sigma_rc = 0
```

Using the lung pressure balance equation, replace `Ppl` with `Palv-sigma_L`:

```
Palv - sigma_L + sigma_di * fa + Pica + Fdi * sigma_di - sigma_rc = 0
```

Using the equation for the pressure drop across the larynx, replace `Palv` with `-VL_t*Rrs`:

```
-VL_t * Rrs - sigma_L + sigma_di * fa + Pica + Fdi * sigma_di - sigma_rc = 0
```

Rearrange by collecting `fa` and `Fdi`, which both multiply `sigma_di`:

```
-VL_t * Rrs - sigma_L + (fa + Fdi) * sigma_di + Pica - sigma_rc = 0
```

To derive the second model equation from the pressure equations, start with the pressure balance on the diaphragm:

```
Pab - Ppl = sigma_di
```

Using the abdominal wall pressure balance equation, replace `Pab` by `sigma_ab`:

```
sigma_ab - Ppl = sigma_di
```

Using the lung pressure balance equation. replace `Ppl` with `(Palv-sigma_L)`:

```
sigma_ab - (Palv - sigma_L) = sigma_di
```

Using the equation for the pressure drop across the larynx, replace `Palv` with `-VL_t*Rrs`:

```
sigma_ab - (-VL_t * Rrs - sigma_L) = sigma_di
```

Rearrange:

```
sigma_ab + VL_t * Rrs + sigma_L - sigma_di = 0
```

So the two model equations are:

```
-VL_t * Rrs - sigma_L + (fa + Fdi) * sigma_di + Pica - sigma_rc = 0
```

```
sigma_ab + VL_t * Rrs + sigma_L - sigma_di = 0
```

Each variable in these equations is a function of the diaphragm and abdominal wall volumes and their derivatives, and the muscle activations, represented by these variables:

`Vdi`:	diaphragm volume
`Vab`:	abdominal wall volume
`Vdi_t`:	time derivative of the diaphragm volume
`Vab_t`:	time derivative of the abdominal wall volume
`Phr_d`:	phrenic activation of the diaphragm
`u`:	lumbar activation of the abdominal muscles
`lma`:	net laryngeal muscle activation

In the following sections, we will show the function of these variables for each variable in the model equations. But first, we will show the volume equations, wich are used in more than one of the variable equations and are fundamental to the model.

4.4 The Volume Equations

4.4.1 Abdominal Volume

`Vdi + C1 * Vrc + Vab = Vsum`

The diaphragm volume, `Vdi`, is the volume above the level of the diaphragm insertions on the rib cage and below the dome of the diaphragm. The rib cage volume, `Vrc`, is defined in the next section. The abdominal wall volume, `Vab`, is the volume between the abdominal wall and a frontal plane that coincides with the contracted position of the abdominal wall. The equation says that a properly weighted sum of these three volumes is a constant, `Vsum`, which is a computed parameter.

It has frequently been assumed that the abdominal contents are incompressible, and that the abdominal cavity has only two movable walls, the diaphragm and the abdominal wall, and that therefore a displacement in one must be met by an equal displacement of the other (Mead and Loring (1982), Grassino et al. (1978), Grimby et al. (1976), Macklem et al. (1978), Lichtenstein et al. (1992)); in other words, that `Vdi + Vab = Vsum` (`Vsum` constant). In particular, Lichtenstein et al. (1992), from which this model was adapted, includes the equation $[\texttt{Vab} - \texttt{Vab}_0] + [\texttt{Vdi} - \texttt{Vdi}_0] = 0$.

If you assume, as we do and as Lichtenstein et al. (1992) does, that the static contractile pressure generated by the diaphragm at a given muscle activation depends only on the diaphragm volume `Vdi`, then `Vdi + Vab = Vsum` implies that the abdominal volume and the diaphragm activation determine the static pressure generated by the diaphragm. But this is inconsistent with the data in Grassino et al. (1978) Figure 4, which shows that the pressure depends also on rib cage volume. This can be accounted for by the fact that a change in volume at the abdominal wall can be reflected in a change in rib cage volume as well as diaphragm volume. Mead and Loring (1982) point out the importance of including the rib cage as one of the movable walls of the abdominal container.

We account for the rib cage as a movable wall of the abdominal container by adding a term, `C1 * Vrc`, to the equation. This allows changes in abdominal volume to be reflected in rib cage volume even if diaphragm volume is constant, though not necessarily 1-for-1, depending on the value of `C1`.

We can determine the value of `C1` by fitting our model to the data. From Grassino et al. (1978) Figure 4, we can read 33 data points over 4 subjects giving diaphragm pressure as a function of rib cage and and abdominal volume at 30% diaphragm activation. Using our model equations, we can calculate the diaphragm pressure from the volumes, given a value of `C1`. We can then find the value of `C1` that gives us the best fit of the calculated pressure values to the measured values, and we get a value of 0.369 ± 0.035.

4.4.2 Rib Cage Volume

`Vrc = VL + Vdi + Vc`

The rib cage volume, `Vrc`, is the sum of the lung volume, `VL`, the diaphragm volume `Vdi`, and a constant volume, `Vc`. The constant volume represents the mediastinal volume (Vms) plus the pulmonary blood volume and lung tissue volume (Vbt). Cluzel et al. (2000) measured Vms using MRI in 5 human subjects and got an average of 586 ml at TLC and 725 ml at RV. We use the harmonic mean of these, 648, to minimize the percentage error of assuming a constant value. Cluzel et al. (2000) estimates Vbt from the literature at 1100 ml. We use a value of 1108 ml to give the best fit of this rib cage volume equation to measured values from Cluzel et al. (2000), giving a `Vc` value of 1756 ml.

4.5 Tracheal Flow

```
VL_t = (-(1 + C1) * Vdi_t - Vab_t) / C1
```

The tracheal flow is defined internally as the time derivative of the lung volume, `VL_t`, so positive flow is inspiration (but this is inverted for display). The equation is derived by using the rib cage volume equation to substitute `VL + Vdi + Vc` for `Vrc` in the abdominal volume equation, solving for `VL`, and taking the time derivative.

4.6 Airway Resistance

```
Rrs = k1 + k2 * |VL_t| + .72 + .44 * |VL_t|
```

For the airway resistance we use the widely adopted Rohrer's equation (Hey and Price (1982), Rohrer (1915)):

$$Pressure = K_1 \cdot Flow + K_2 \cdot Flow^2$$

Dividing through by flow gives us:

$$Pressure/Flow = Resistance = K_1 + K_2 \cdot Flow$$

The equation is applied twice, once for laryngeal resistance (`k1 + k2 * |VL_t|`) and once for the resistance of the oropharynx and lower airway (`.72 + .44 * |VL_t|`).

The values .72 and .44 come from Renotte et al. (1998). Their Equation 8 implies that K_2 for the oropharynx is 0 for a constant oropharyngeal volume, and their Table 2 gives .44 for the lower airway, for a total of .44. Assuming K_1 for the larynx is negligible during quiet breathing (see below), Table 2 gives .34 for the lower airway and .38 for the oropharynx, for a total of .72.

The values of k_1 and k_2 for the larynx depend on the diameter of the glottis. We use the following equations:

```
k1 = .153 / (d^2 * B^2)

k2 = .167 * ((1 - B^2) / B^4 - (1 - B^2))

d = min (max (10.9 * (1 + lma), 0), D)

D = 18

B = d / D
```

`D` is the diameter of a human trachea, and the value is based on numbers in Breatnach et al. (1984) and Kamel et al. (2009).

The variable `d` is the diameter of the glottis, or more precisely, the diameter of a circle with the same area as the opening of the glottis. The resting diameter (when `lma` = 0) is taken to be 10.9 mm, based on numbers in D'Urzo et al. (1988), Brancatisano et al. (1983), and Baier et al. (1977), and is assumed to change in proportion to `lma`, based on the results in Tully et al. (1990) and Tully et al. (1991).

The coefficient `k2` is assumed to be proportional to `((1 - B^2) / B^4 - (1 - B^2))`, as in the equation for flow through an orifice in Table II of Simpson (1968). Renotte et al. (1998) Table 2 gives two different values for `k2` for the upper airway, one for inspiration and one for expiration, which we assume is due to changes in `d`. Assuming equal excursions from the resting diameter, we solved for the coefficient that would give us the reported values of `k2`, and got .167. This results in a resting value for `k2` of 0.681.

Using Tully et al. (1990) Table 2 to get the ratio of the mean resting value of `k1` to the mean resting value of `k2`, we applied that to 0.681 to get a resting value of `k1` of .0035 for the larynx, which is indeed small relative to K_1 for the rest of the airway.

For the `k1` equation, we use the Darcy-Weisbach equation for laminar flow, which says that the pressure drop is proportional to $flow/d^4$ (Kreith et al., 2004). This makes it proportional to the `(d^2 * B^2)` used in the equation, because `D` is constant, making `B^2` proportional to `d^2`. Plugging in the resting value of `k1` determined above, and the resting value of `d`, we can solve for the coefficient, which gives us .153.

4.7 Recoil Pressure of the Lung

```
sigma_L = (VL - VL0) / CL
```

This equation assumes a linear relationship between lung volume and recoil pressure.

`CL = .201` is lung compliance, and the value is the average of the inspiratory compliance of 28 men in Table 2 of Permutt and Martin (1960).

`VL0` is the lung volume at zero recoil pressure, which we take to be equal to the Residual Volume (RV) after a maximal exhalation based on the results in Permutt and Martin (1960) and Harris (2005), and which we calculate as follows:

```
VL0 = VLFRC - (VrcKM_FRC + VabKM_FRC) * VC
```

`VLFRC = 2.290` is the lung volume in liters at FRC (the end of a passive expiration) from Cluzel et al. (2000), Table 1. The value is the mean spirometric supine lung volume of 5 men.

`VrcKM_FRC = .1282` is the rib cage contribution to lung volume at FRC as a fraction of VC relative to RV from Konno and Mead (1967) Figure 14B. The value is an average over 6 supine human subjects.

`VabKM_FRC = .1282` is the abdominal contribution to lung volume at FRC as a fraction of VC relative to RV from Konno and Mead (1967) Figure 14B. The value is an average over 6 supine human subjects.

`VC = 5.37` is the mean Vital Capacity in liters from Roca et al. (1998) for 6479 men.

4.8 Abdominal Fraction of Rib Cage

```
fa = (Vdi - VdiTLC) / (Vdi - VdiTLC + VL) / (1 + C1) + .15
```

`fa` is the fraction of the rib cage exposed to abdominal pressure.

At TLC (Total Lung Capacity), none of the diaphragm is apposed to the rib cage (Mead and Loring, 1982), so we assume that at all lung volumes, a portion of the diaphgram volume equal to the diaphragm volume at TLC (`VdiTLC`) does not contribute to the zone of apposition. The remainder (`Vdi - VdiTLC`) is divided by the remainder plus the lung volume (`Vdi - VdiTLC + VL`) to give an estimate of the fraction of the rib cage surface above the diaphragm insertions that is exposed to abdominal pressure.

The "obligatory ring" below the diaphragm insertions is always exposed to abdominal pressure. Mead and Loring (1982) estimate this as 15% of the rib cage surface. But their rib cage volume is larger than ours, because they include parts below the diaphragm insertions. They measure rib cage volume by the method of Konno and Mead (1967), which means that a rib cage volume change is equal to a lung volume change with the abdominal volume held constant. From our volume equations, this means that their rib cage volume is larger than ours by a factor of `(1 + C1)`. So we divide the previously calculated fraction of the smaller rib cage by `(1 + C1)` to turn it into a fraction of the larger rib cage before adding it to the `.15`.

4.9 Recoil Pressure of the Diaphragm

```
sigma_di = Phr_d * Pdimax * Ffl * Ffv
           + max (Vdi - VdiFRC, 0) * Kdi_psv * (Vdi - VdiFRC)^2 + Rdi * Vdi_t
```

`Phr_d * Pdimax * Ffl * Ffv` corresponds to the Hill muscle model equation A.6 from Ratnovsky et al. (2003), except that `Pdimax` is a pressure rather than a force, and `Ffl` and `Ffv` are functions of volume and its derivative (flow) respectively rather than length and velocity. We can substitute pressure for force, because by Laplace's Law (Laplace, 1808), the pressure is proportional to the force when the curvature is constant, and Braun et al. (1982) found that the curvature of the human diaphragm dome does not change significantly with volume. Also, Kim et al. (1976) found a constant ratio between diaphragm force and pressure in the dog. Substituting volume for length (with an offset) is supported by the numbers in Cluzel et al. (2000), where the relationship between diapghram pressure and length is not clearly different from linear when measured at RV, FRV, and TLC. To the extent that the action of the diaphragm resembles that of a piston (Kim et al., 1976), this is to be expected. Younes and Riddle (1981) provide a precedent for a Hill style model in terms of pressure and flow for the respiratory system.

`Phr_d` is phrenic activation of the diaphragm (see discussion of `Phr_d` in Section 4.1.1).

`Pdimax` is static diaphragm recoil pressure at optimum length and maximum activation in cmH_2O. This parameter is calculated to make `sigma_di` at active TLC equal to the sum of abdominal and lung recoil pressure at active TLC. These pressures depend on the corresponding volumes, which are calculated as described in Volume Parameters.

`max (Vdi - VdiFRC, 0) * Kdi_psv * (Vdi - VdiFRC)^2` is the passive transdiaphragmatic pressure as a function of diaphragm volume. This is taken to be zero at resting diaphragm volume `VdiFRC` (Agostoni and Rahn, 1960), and zero below and quadratic above (Reid et al., 1987). [NOTE — the Grassino data may give a better estimate of where the quadratic starts.] The parameter `Kdi_psv` is calculated to make the passive transdiaphragmatic pressure at `VdiRV` (`PdiRV`) equal to 20 cmH_2O. The value 20 was chosen to be roughly consistent with the results in Siafakas et al. (1979), Grassino et al. (1978), and Agostoni et al. (1966). [NOTE - PdiRV needs a better justification.]

`Rdi * Vdi_t` represents the passive resistance of the diaphragm. The `Ffv` factor in the first term of the `sigma_di` equation is a resistance effect too, but the resistance it represents goes to zero when the diaphragm is not activated, which is not realistic — it would result in any pressure difference causing an infinite flow. So we add a small passive resistance, `Rdi` = 1 $cmH_2O/(L/s)$. The Ffv resistance is, by contrast, about 300 $cmH_2O/(L/s)$ at full activation. [NOTE — need a better justification for the value of `Rdi`.] `Vdi_t` is the time derivative of the diaphragm volume.

4.9.1 Ffl

```
Ffl = exp (-0.5 * (((1 - Ldi_min) / Vdi0 * Vdi + Ldi_min - 1.05) / 0.19)^2)
```

`Ffl` is the static pressure-volume relationship of the diaphragm, corresponding to equation A.7 from Ratnovsky et al. (2003), with the relative length replaced by a linear function of volume, as described above. Equation A.7 builds in the assumption that the peak tetanic length is 5% greater than the resting length.

`Vdi0` is the volume of the diaphragm at the resting length, in liters. This is taken to be `VdiRV`, the diaphragm volume at RV (see Volume Parameters), based on the observation in Smith and Bellemare (1987) that the "highest Pdi twitch amplitude was recorded at RV". [NOTE — the optimum twitch length is not necessarily the same as the resting length.]

4.9.1.1 `Ldi_min`

```
Ldi_min = (VdiTLC - .65 * VdiRV) / (VdiTLC - VdiRV / 1.05);
```

`Ldi_min` is the length of the diaphragm at zero diaphragm volume, in units of the resting length. This parameter is calculated by assuming that the length of the diaphragm at TLC is 65% of that at RV, and that the peak tetanic tension is at RV (both from Smith and Bellemare (1987)). [NOTE — Smith said peak twitch was at RV, which may not be the same as peak tetanic, and for `Vdi0` we assumed that resting length was at RV, which is not the same as peak tetanic.]

`VdiRV` is described in Volume Parameters.

`VdiTLC` is described in Volume Parameters.

`Vdi` is the current volume of the diaphragm, in liters.

4.9.2 `Ffv`

```
Ffv = 0.1433 / (0.1074 + exp (-1.409 * sinh (3.2 * Vdi_t / Vdi_t_max
                                              + 1.59443531272566456619)));
```

`Ffv` is the pressure-flow relationship of the diaphragm, corresponding to equation A.8 from Ratnovsky et al. (2003) with the velocity replaced by flow, as discussed above. Also, a typo has been corrected – the corrected version is as found in Artemiadis and Kyriakopoulos (2005), Rosen et al. (1999), and Hatze (1981). Also, the value 1.6 has been replaced with the exact value that will make `Ffv` equal to 1 when the flow is zero. [NOTE — adjusting one of the other constants may make more sense - 1.6 is 3.2/2 in Hatze.]

`Vdi_t` is the current rate of change of diaphragm volume.

`Vdi_t_max` = 2.449 is the maximum rate of change of diaphragm volume. This parameter is derived from Fig. 6 in Goldman et al. (1978), which gives transdiaphragmatic pressure as a function of flow at several levels of diaphragm activation up to 45%. Because they held the rib cage still, the flow represents the rate of change of diaphragm volume. Fitting the curves to a Hill-style relation between pressure and flow (Younes and Riddle, 1981), we can find a maximum flow (where the pressure goes to zero) for each level of diaphragm activation. Chow

and Darling (1999) Fig 4 implies that the maximum flow increases somewhat linearly to 80% activation and then levels off. That assumption together with the results for the maximum flow at lower activations allows us to compute a maximum flow at 100% activation, which we use for `Vdi_t_max`. [NOTE — we are using the fact that `Vdi_t_max` changes with activation, but in our model, it doesn't.]

4.10 Pressure of the Intercostal and Accessory Muscles

```
Pica = (if (Vdi < VdiFRC) (Vdi - VdiFRC) / (VdiTLC - VdiFRC) * Pdirc
        else 0) * sigma_di
       + u * (Pica_ab_RV
              + (Vrc - VrcRV) / (VrcTLC - VrcRV) * (Pica_ab_TLC - Pica_ab_RV))
```

`Pica` is the effective pressure in cmH_2O generated by the intercostal and accessory muscles, positive values acting to expand the rib cage.

The first half of this expression is the pressure due to the action of the inspiratory intercostals, which are assumed to be inactive when the diaphragm volume is above `VdiFRC` (low lung volumes). Below `VdiFRC`, the pressure exerted by the inspiratory intercostals is assumed to be proportional to the pressure exerted by the diaphragm (`sigma_di`), and the proportionality constant itself is assumed to scale linearly from 0 at `VdiFRC` to its maximum value of `Pdirc` at `VdiTLC`.

The parameter `Pdirc` is calculated as the ratio of `Pica` at TLC to `sigma_di` at TLC.

We calculate `Pica` and `sigma_di` at TLC by solving the model equations for them while assuming TLC conditions: maximal phrenic, zero abdominal, previously calculated values of diaphragm and abdominal volumes (VdiTLC and VabTLC), and zero derivatives. This gives us the intercostal pressure necessary to reach TLC.

The second half of this expression is the pressure due to the action of the expiratory intercostals, which is assumed to be proportional to the abdominal muscle activation (`u`), and the proportionality constant itself is assumed to scale linearly with the rib cage volume, changing from `Pica_ab_RV` to `Pica_ab_TLC`.

`Pica_ab_TLC` = -135 is the pressure due the expiratory intercostals at TLC and maximal abdominal activation. This parameter's value is calculated by taking the mean male maximal mouth pressure at TLC from Ratnovsky et al. (2008) Table 1 and subtracting it from the rib cage recoil pressure at TLC. [NOTE — this neglects the lung recoil pressure, which is less than 1 SD of this value.]

`Pica_ab_RV` is the pressure due the expiratory intercostals at RV and maximal abdominal activation. We calculate this parameter by solving the model equations for Pica while assuming RV conditions: zero phrenic, maximal abdominal activation, previously calculated values of diaphragm and abdominal volumes (VdiTLC and VabTLC), and zero derivatives. This gives us the intercostal pressure necessary to reach RV.

Vdi is the current volume of the diaphragm, in liters.

Vrc is the current volume of the rib cage, in liters.

VdiFRC, VdiTLC, VrcRV, and VrcTLC are parameters calculated as described in Volume Parameters.

4.11 Recoil Pressure of the Rib Cage

```
Prc = log ((Vrc_max - Vrc) / (Vrc - Vrc_min)) / Prc_div + Prc_add
```

The volume of the rib cage is assumed to be a sigmoid function of the recoil pressure of the rib cage, Prc. With increasing pressure the volume asymptotically approaches a maximum, and with decreasing pressure it asymptotically approaches a minimum. A generalized logistic function is used for the sigmoid, Vrc as a function of Prc:

```
Vrc = Vrc_min + (Vrc_max - Vrc_min) / (1 + exp (Prc_div * (Prc - Prc_add)))
```

This equation is then solved for Prc to give Prc as a function of Vrc, which is the equation above for Prc.

4.11.1 Vrc_max

```
Vrc_max = VrcTLC + .05 * (VrcTLC - VrcRV)
```

Vrc_max is the upper asymptote. This parameter is calculated to be 5% of the range of Vrc above VrcTLC.

VrcRV, and VrcTLC are parameters calculated as described in Volume Parameters.

4.11.2 Vrc_min

```
Vrc_min = VrcRV  - .99 * (VrcTLC - VrcRV)
```

Vrc_max is the lower asymptote. This parameter is calculated to be 99% of the range of Vrc below RV, so that it is not approached during the normal range of breathing. [NOTE — putting it 5% below resulted in simulation failure. This bears further investigation]

VrcRV, and VrcTLC are parameters calculated as described in Volume Parameters.

4.11.3 Vrc_div

```
Prc_div = -4 * Crc / (Vrc_max - Vrc_min) / (1 + C1)
```

`Prc_div` is a parameter that controls the maximum slope of the sigmoid, which is the compliance of the rib cage. It is calculated to make the compliance equal to `Crc / (1 + C1)`. The factor of `(1 + C1)` appears because `Crc` is for a rib cage volume defined differently than `Vrc` (see Section 4.8).

`Crc = .110` is the compliance of the rib cage in liters per cmH_2O, an average of values from Table 1 of Gilroy et al. (1985) for three sitting subject.

4.11.4 Vrc_add

```
Prc_add = log ((Vrc0 - Vrc_min) / (Vrc_max - Vrc0)) / Prc_div
```

`Prc_div` is a parameter that controls the location of the sigmoid along the `Prc` axis. It is calculated to make the volume equal to Vrc0 when the pressure is 0.

`Vrc0` is the volume of the rib cage at zero recoil pressure. It is set equal to `VrcFRV` (see Volume Parameters).

4.12 Recoil Pressure of the Abdominal Wall

```
sigma_ab = u * FCEmax * Ffl * Ffv * k * (1/rt + 1/rs)
           + (Vab - Vab0) / Cab + Rab * Vab_t
```

The abdominal wall is modeled as a surface with a circular segment cross-section in each transverse plane, all with the same radius, and a circular segment cross-section in each sagittal plane, all with another radius. The volume of the abdominal wall, `Vab`, is bounded by this surface and by a frontal plane. The values for the sagittal and transverse radius were derived from the results in Song et al. (2006), who measured the radii during insufflation for laparoscopic surgery in humans. Exponential curves were fit to the data points in Fig 3, and the resulting relationship between the fitted sagittal and transverse radii were found to be well approximated by a linear function:

```
rs = 8.00479 * rt - 1.10158
```

The length of the longest transverse chord in the bounding frontal plane was found which gave the volume change stated in the paper, $1.27 \times 10^{-}3$ m^3. The resulting chord length in meters was

```
ct = 0.320496
```

The volume of the abdominal wall is calculated by numerical integration at different radii, to generate a relationship between the transverse radius and the volume:

```
rt = radius (Vab)
```

`u * FCEmax * Ffl * Ffv` is the Hill muscle model equation A.6 from Ratnovsky et al. (2003).

`u` is activation of the diaphragm by the lumbar motor neurons (see Section 4.1.1).

`FCEmax = 33` is the maximal force capacity of the contractile element, in Newtons, for 1.5 cm^2 cross-section of canine external oblique muscle, from Ratnovsky et al. (2003) Table 1.

Using the Law of Laplace (Laplace, 1808), we can convert the tension in the abdominal wall to a pressure. [NOTE — an unduloid would meet the assumptions of the Law of Laplace more exactly.]

`k = .67981067` is a constant to convert from a force in Newtons for 1.5 cm^2 of muscle to a surface tension in meter-cmH2O.

`(1/rt + 1/rs)` converts the surface tension to a pressure, using the Law of Laplace.

`(Vab - Vab0) / Cab` is the passive recoil pressure of the abdominal wall.

`Vab0` is the volume of the abdominal wall at which the recoil pressure is 0. This is taken to be equal to VabFRC (see Volume Parameters).), since we are assuming a supine position.

`Cab = .108` is the compliance of the abdominal wall in liters/cmH2O. Estenne et al. (1985) reported a mean value of 0.0880.027 in approximately 10 supine men aged 24-39 for diaphragm-abdomen compliance assuming that supine lune compliance is the same as sitting. Cala et al. (1993) gives a value of .196. [NOTE — this needs more work.]

`Rab * Vab_t` represents the passive resistance of the abdominal wall. The `Ffv` factor in the first term of the `sigma_ab` equation is a resistance effect too, but the resistance it represents goes to zero when the abdominal wall muscles are not activated, which is not realistic — it would result in any pressure difference causing an infinite flow. So we add a small passive resistance, `Rab` = 1 $cmH_2O/(L/s)$. [NOTE — need a better justification for the value of `Rab`.] `Vab_t` is the time derivative of the abdominal wall volume.

4.12.1 Ffl

```
Ffl = exp (-0.5 * ((LCE / LCE0 - 1.05) / 0.19)^2)
```

`Ffl` is the static pressure-volume relationship of the abdominal wall, equation A.7 from Ratnovsky et al. (2003),

`LCE0 = 19.1` is the resting length of the contractile element, in cm. The value is from Gaumann et al. (1999), an average value for the transversus abdominis of 15 male and 11 female cadavers aged 62-87.

4.12.1.1 LCE

```
LCE = 2 * rt * asin (ct / (2 * rt)) * 100 / 2
```

`LCE` is the length of the contracting element. It is calculated by this equation from `ct`, and from `rs`, `rt`, which are calculated from `Vab` as described above.

4.12.2 Ffv

```
Ffv = 0.1433 / (0.1074 +  exp (-1.409 * sinh (3.2 * LCE_t / VCEmax
                                              + 1.594435312725664566 19)))
```

Ffv is the force-velocity relationship of the abdominal wall muscles, corresponding to equation A.8 from Ratnovsky et al. (2003) A typo has been corrected – the corrected version is as found in Artemiadis and Kyriakopoulos (2005), Rosen et al. (1999), and Hatze (1981). Also, the value 1.6 has been replaced with the exact value that will make Ffv equal to 1 when the flow is zero. [NOTE — adjusting one of the other constants may make more sense - 1.6 is 3.2/2 in Hatze.]

LCE_t is the velocity of the contractile element, calculated from the time derivative of Vab, from the relationships described above.

VCEmax = 34.7 is the maximal contractile velocity of the contractile element, in cm/s, for canine external oblique muscle, from Ratnovsky et al. (2003) Table 1.

4.13 Volume Parameters

The volumes of the rib cage, lung, diaphragm, and abdomen at RV, FRC, and TLC are calculated as parameters to the model. The names of the resulting parameters are

```
VrcRV  VdiRV  VLRV  VabRV
VrcFRC VdiFRC VLFRC VabFRC
VrcTLC VdiTLC VLTLC VabTLC
```

VrcFRC, VdiFRC, and VLFRC are taken from Cluzel et al. (2000), the average of 5 supine healthy males age 27-33.

VLFRC = 2.290 is a spirometric lung volume from Cluzel et al. (2000) Table 1.

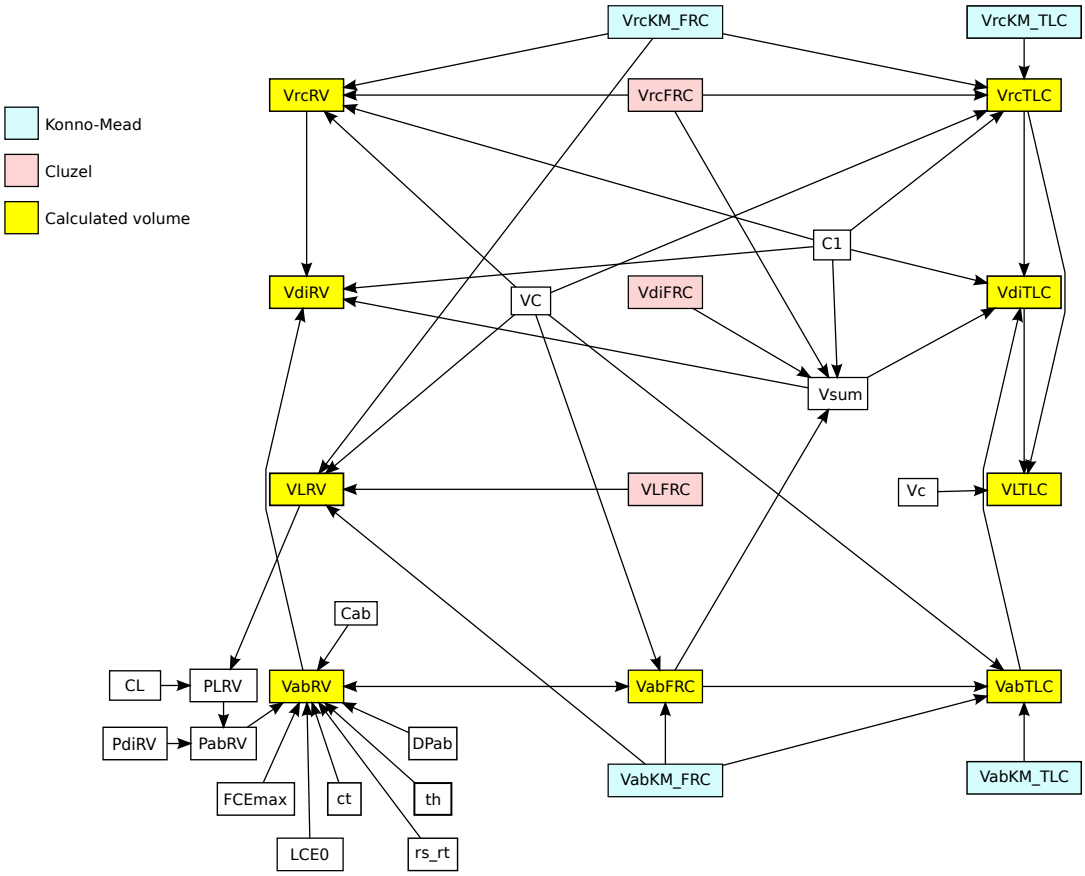
VrcFRC = 7.013 and VdiFRC = 2.967 are supine MRI volumes from Cluzel et al. (2000), adjusted to fit with Cluzel's spirometric lung volumes with minimimum implied percentage error and assuming a constant Vms (mediastinal volume) of .648, which minimize the maximum implied percentage error of Cluzel's Vms volumes.

Konno and Mead (1967) Fig 14B gives us the rib cage and abdominal volumes in %VC relative to FRC, average of 6 supine human subjects.

For VC, we use 5.37, the mean Vital Capacity in liters from Roca et al. (1998) for 6479 men.

Starting from the Cluzel and Konno-Mead volume numbers, and using the volume equations (Section 4.4), we calculate VrcRV, VLRV, and VrcTLC. Then bringing in the recoil pressure equations, we can calculate the rest of the volume parameters. The following diagram shows the relationships:

►


Konno-Mead
Cluzel
Calculated volume
VrcKM_FRC
VrcKM_TLC
VrcRV
VrcFRC
VrcTLC
C1
VdiRV
VC
VdiFRC
VdiTLC
Vsum
VLRV
VLFRC
Vc
VLTLC
Cab
CL
PLRV
VabRV
VabFRC
VabTLC
PdiRV
PabRV
DPab
FCEmax
ct
th
VabKM_FRC
VabKM_TLC
LCE0
rs_rt

Bibliography

Agostoni, E., Gurtner, G., Torri, G., and Rahn, H. (1966). Respiratory mechanics during submersion and negative-pressure breathing. *Journal of Applied Physiology*, 21(1):251–258.

Agostoni, E. and Rahn, H. (1960). Abdominal and thoracic pressures at different lung volumes. *Journal of Applied Physiology*, 15(6):1087–1092.

Artemiadis, P. and Kyriakopoulos, K. (2005). Teleoperation of a robot manipulator using emg signals and a position tracker. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1003 – 1008.

Baier, H., Wanner, A., Zarzecki, S., and Sackner, M. A. (1977). Relationships among glottis opening, respiratory flow, and upper airway resistance in humans. *Journal of applied physiology: respiratory, environmental and exercise physiology*, 43(4):603–11. Research Support, U.S. Gov't, P.H.S.,.

Brancatisano, T., Collett, P. W., and Engel, L. A. (1983). Respiratory movements of the vocal cords. *Journal of Applied Physiology*, 54(5):1269–1276.

Braun, N. M., Arora, N. S., and Rochester, D. F. (1982). Force-length relationship of the normal human diaphragm. *Journal of Applied Physiology*, 53(2):405–412.

Breatnach, E., Abbott, G., and Fraser, R. (1984). Dimensions of the normal human trachea. *American Journal of Roentgenology*, 142(5):903–906.

Cala, S. J., Edyvean, J., and Engel, L. A. (1993). Abdominal compliance, parasternal activation, and chest wall motion. *Journal of Applied Physiology*, 74(3):1398–1405.

Chow, J. W. and Darling, W. G. (1999). The maximum shortening velocity of muscle should be scaled with activation. *Journal of Applied Physiology*, 86(3):1025–1031.

Cluzel, P., Similowski, T., Chartrand-Lefebvre, C., Zelter, M., Derenne, J.-P., and Grenier, P. A. (2000). Diaphragm and chest wall: Assessment of the inspiratory pump with mr imaging—preliminary observations. *Radiology*, 215(2):574–583.

D'Urzo, A. D., Rubinstein, I., Lawson, V. G., Vassal, K. P., Rebuck, A. S., Slutsky, A. S., and Hoffstein, V. (1988). Comparison of glottic areas measured by acoustic reflections vs. computerized tomography. *Journal of Applied Physiology*, 64(1):367–370.

Estenne, M., Yernault, J. C., and De Troyer, A. (1985). Rib cage and diaphragm-abdomen compliance in humans: effects of age and posture. *Journal of applied physiology (Bethesda, Md. : 1985)*, 59(6):1842–1848.

Gaumann, A., Hoeckel, M., and Konerding, M. (1999). The anatomic basis of the transversus and rectus abdominis musculoperitoneal (tramp) composite flap. *Hernia*, 3:39–41. 10.1007/BF01576742.

Gilroy, R. J., Lavietes, M. H., Loring, S. H., Mangura, B. T., and Mead, J. (1985). Respiratory mechanical effects of abdominal distension. *Journal of Applied Physiology*, 58(6):1997–2003.

Goldman, M. D., Grassino, A., Mead, J., and Sears, T. A. (1978). Mechanics of the human diaphragm during voluntary contraction: dynamics. *Journal of Applied Physiology*, 44(6):840–848.

Grassino, A., Goldman, M. D., Mead, J., and Sears, T. A. (1978). Mechanics of the human diaphragm during voluntary contraction: statics. *Journal of Applied Physiology*, 44(6):829–839.

Grimby, G., Goldman, M., and Mead, J. (1976). Respiratory muscle action inferred from rib cage and abdominal v-p partitioning. *Journal of Applied Physiology*, 41(5):739–751.

Harris, R. S. (2005). Pressure-volume curves of the respiratory system. *Respiratory care*, 50(1):78–98; discussion 98–9.

Hatze, H. (1981). *Myocybernetic Control Models of Skeletal Muscle*. University of South Africa.

Hey, E. N. and Price, J. F. (1982). Nasal conductance and effective airway diameter. *The Journal of Physiology*, 330(1):429–437.

Kamel, K. S., Lau, G., and Stringer, M. D. (2009). In vivo and in vitro morphometry of the human trachea. *Clinical Anatomy*, 22(5):571–579.

Kim, M. J., Druz, W. S., Danon, J., Machnach, W., and Sharp, J. T. (1976). Mechanics of the canine diaphragm. *Journal of Applied Physiology*, 41(3):369–382.

Konno, K. and Mead, J. (1967). Measurement of the separate volume changes of rib cage and abdomen during breathing. *Journal of applied physiology*, 22(3):407–422.

Kreith, F., Berger, S., Churchill, S., Tullis, J. P., Tullis, B., White, F., Kumar, A., Todd, J., Chen, J., Irvine, T., Capobianchi, M., Kennedy, F., Booser, E. R., Wilcock, D., Boehm, R., Reitz, R., Kim, J., McDonald, A., Sherif, S., and Bhushan, B. (2004). Fluid mechanics. In Kreith, F., editor, *The CRC Handbook of Mechanical Engineering, Second Edition*, Handbook Series for Mechanical Engineering. CRC Press.

Laplace, P. (1808). *Traité De Mécanique Céleste: Supplément au dixième livre du traité de méchanique céleste sur l'action capillaire*. Number v. 6 in Traité De Mécanique Céleste: Supplément au dixième livre du traité de méchanique céleste sur l'action capillaire. Duprat.

Lichtenstein, O., Ben-Haim, S. A., Saidel, G. M., and Dinnar, U. (1992). Role of the diaphragm in chest wall mechanics. *Journal of Applied Physiology*, 72(2):568–574.

Loring, S. H. and Mead, J. (1982). Action of the diaphragm on the rib cage inferred from a force-balance analysis. *Journal of Applied Physiology*, 53(3):756–760.

Macklem, P. T., Gross, D., Grassino, G. A., and Roussos, C. (1978). Partitioning of inspiratory pressure swings between diaphragm and intercostal/accessory muscles. *Journal of Applied Physiology*, 44(2):200–208.

Mantilla, C. B. and Sieck, G. C. (2011). Phrenic motor unit recruitment during ventilatory and non-ventilatory behaviors. *Respiratory Physiology & Neurobiology*, 179(1):57 – 63. ¡ce:title¿Recruitment of Respiratory Motoneurons¡/ce:title¿.

Mead, J. (1956). Measurement of inertia of the lungs at increased ambient pressure. *Journal of Applied Physiology*, 9(2):208–212.

Mead, J. and Loring, S. H. (1982). Analysis of volume displacement and length changes of the diaphragm during breathing. *Journal of Applied Physiology*, 53(3):750–755.

Nail, B. S., Sterling, G. M., and Widdicombe, J. G. (1972). Patterns of spontaneous and reflexly-induced activity in phrenic and intercostal motoneurons. *Experimental Brain Research*, 15:318–332. 10.1007/BF00235915.

Newton, I. (1687). *PhilosophiæNaturalis Principia Mathematica*.

Permutt, S. and Martin, H. B. (1960). Static pressure-volume characteristics of lungs in normal males. *Journal of Applied Physiology*, 15(5):819–825.

Ratnovsky, A., Elad, D., and Halpern, P. (2008). Mechanics of respiratory muscles. *Respiratory Physiology & Neurobiology*, 163(13):82 – 89. ¡ce:title¿Respiratory Biomechanics¡/ce:title¿.

Ratnovsky, A., Zaretsky, U., Shiner, R. J., and Elad, D. (2003). Integrated approach for in vivo evaluation of respiratory muscles mechanics. *Journal of Biomechanics*, 36(12):1771 – 1784.

Reid, M. B., Feldman, H. A., and Miller, M. J. (1987). Isometric contractile properties of diaphragm strips from alcoholic rats. *Journal of Applied Physiology*, 63(3):1156–1164.

Renotte, C., Remy, M., and Saucez, P. (1998). Dynamic model of airway pressure drop. *Medical and Biological Engineering and Computing*, 36:101–106. 10.1007/BF02522865.

Roca, J., Burgos, F., Sunyer, J., Saez, M., Chinn, S., Anto, J., Rodriguez-Roisin, R., Quanjer, P., Nowak, D., and Burney, P. (1998). References values for forced spirometry. group of the european community respiratory health survey. *European Respiratory Journal*, 11(6):1354–1362.

Rohrer, F. (1915). Der stromungswiderstand in den menschlichen atemwegen und der einfluss der unregelmassigen verzweigung des bronchialsystems auf den atmungsverlauf in verschiedenen lungenbezirken. *Archiv fur die Gesamte Physiologie des Menschen und der Tiere*, 162:225–299.

Rosen, J., Fuchs, M. B., and Arcan, M. (1999). Performances of hill-type and neural network muscle models–toward a myosignal-based exoskeleton. *Computers and Biomedical Research*, 32(5):415 – 439.

Siafakas, N. M., Morris, A. J., and Green, M. (1979). Thoracoabdominal mechanics during relaxed and forced vital capacity. *Journal of Applied Physiology*, 47(1):38–42.

Simpson, L. L. (1968). Sizing piping for process plants. *Chemical Engineering"*, 75(13):192–214.

Smith, J. and Bellemare, F. (1987). Effect of lung volume on in vivo contraction characteristics of human diaphragm. *Journal of Applied Physiology*, 62(5):1893–1900.

Song, C., Alijani, A., Frank, T., Hanna, G. B., and Cuschieri, A. (2006). Mechanical properties of the human abdominal wall measured in vivo during insufflation for laparoscopic surgery. *Surgical endoscopy*, 20(6):987–990.

Tully, A., Brancatisano, A., Loring, S. H., and Engel, L. A. (1990). Relationship between thyroarytenoid activity and laryngeal resistance. *Journal of Applied Physiology*, 68(5):1988–1996.

Tully, A., Brancatisano, A., Loring, S. H., and Engel, L. A. (1991). Influence of posterior cricoarytenoid muscle activity on pressure-flow relationship of the larynx. *Journal of Applied Physiology*, 70(5):2252–2258.

Younes, M. and Riddle, W. (1981). A model for the relation between respiratory neural and mechanical outputs. i. theory. *Journal of Applied Physiology*, 51(4):963–978.