Nektar1D Reference Manual

Marie Willemet[1], Jordi Alastruey[1]
[1]Department of Biomedical Engineering, Division of Imaging Sciences and Biomedical Engineering, King's College London, King's Health Partners, St. Thomas' Hospital, London SE1 7EH, UK

June 20, 2017

Contents

1 Introduction

Nektar1D is our in-house computer code for solving the nonlinear one-dimensional (1-D) equations of blood flow in a given network of compliant vessels subject to boundary and initial conditions. This document describes how to compile Nektar1D (Section 2), create a text file containing all the input data for a specific simulation (Section 3), run the input file (Section 4), and interpret the output files with the results of the simulation (Section 5). It also provides some examples of Nektar1D simulations used in our peer-reviewed publications (Section 6) and a brief summary of the structure of the source code (Section 7).

For a review on arterial pulse wave haemodynamics and a description of the 1-D equations and the numerical scheme used in Nektar1D to solve them we refer to [1]. We have verified the accuracy of the 1-D formulation by comparison against *in vivo* [2] *in vitro* [3] and 3-D numerical [2, 4] data. Any comment/improvements on this document and Nektar1D is very much welcome and can be sent to `jordi.alastruey-arimon@kcl.ac.uk`.

2 Compiling Nektar1D

The easiest way of compiling Nektar1D is on a Linux operating system. The code is hosted in Bitbucket (`www.bitbucket.org`). To download the code, first install `hg` by typing the following command into a terminal:

```
sudo apt-get install mercurial
```

The `clone` command downloads the code to your home directory (*i.e.* `/home/username`):

```
hg clone https://YOUR_USER_NAME@bitbucket.org/JordiAlastruey/nektar
```

You will need to edit the configuration file of mercurial. Open this file using a text editor (for example, `gedit`, `nano` or `vim`):

```
vim ~/nektar/.hg/hgrc
```

where the `~/` symbol is a shortcut to your home directory (= `/home/username/`). Within the `vim` text editor, press `i` to insert text, and add the following lines:

```
[paths]
default = https://YOUR_USER_NAME@bitbucket.org/JordiAlastruey/nektar
[ui]
username = YOUR_USER_NAME <YOUR_EMAIL_ADDRESS@kcl.ac.uk>
```

Save the changes and exit the `vim` editor by pressing `Esc` and then `:wq`.

To compile the code, first of all make sure that `g++` and `gfortran` compilers are installed (they normally come with the Linux distribution). Otherwise, use

```
sudo apt-get install g++
sudo apt-get install gfortran
```

The code requires the compilation of two libraries in `/Hlib` and `/Veclib` and of the source code in `1DBio/src`. All makefiles assume that the application `make` is available. First, make sure that the following symbolic links have been created:

```
ln -s ../Makefile Makefile       in nektar/Hlib/Linux
ln -s ../MakeHybrid MakeHybrid   in nektar/Hlib/Linux
ln -s GCC.inc Linux.inc          in nektar/Flags
```

Next, install additional tools:

- The Yacc compiler: `sudo apt-get install byacc`
- LAPACK and BLAS libraries for linear algebra operations:

```
sudo apt-get install liblapack-dev
```

You can then build Veclib using the following commands:

```
cd nektar/Veclib
make
```

The file `libvec.a` should have been generated in `/Veclib`. In order to build Hlib, go into the `Hlib/Linux` directory and compile using `make`:

```
cd ../Hlib/Linux
make dbx
make opt
```

The files `libhybridg.a` and `libhybridopt.a` should have been generated in `/Hlib/Linux`. Copy the Veclib library into the Hlib/Linux directory:

```
cd ../..
cp Veclib/libvec.a Hlib/Linux/.
```

Finally, compile Nektar1D using

```
make dbx    in nektar/1DBio/Linux
```

Once the compilation is complete, the Nektar1D executable file, called `oneDbio`, will be located in the folder `nektar/1DBio/Linux`. To run it from this directory, use the command `./oneDbio`.

It might be convenient to access the executable from any directory. To do so, you need to export this path into your `.bashrc` file. The `.bashrc` file contains some initialisation commands for the shell, such as the definition of aliases, of new environment variables, or the addition of directories to the `PATH` variable, which will interest us in this case. Open the `.bashrc` file in a text editor:

```
vim ~/.bashrc
```

At the end of the file, insert the following line:

```
export PATH=$PATH:/home/username/nektar/1DBio/Linux
```

To refresh the environment variables previously defined, type the following command in your terminal:

```
source ~/.bashrc
```

Your executable `oneDbio` should now be accessible from any directory.

Finally, note that an input file transferred from a DOS-platform into a Linux -based system might create errors at the execution. This is because both systems have different character encodings and end-of-line commands. The `fromdos` command converts text files between DOS and UNIX formats. You should be able to install it using the command:

```
sudo apt-get install tofrodos
```

`fromdos` should be used once to convert your input file: `fromdos myInputFile.in`

3 Creating an Input File

This section describes how to write a text input file containing all the parameters of a specific simulation. You will find examples of input files in the folder `nektar/examples`. At the bottom of each file you will find the command line required to execute the file. A full description on how to run an input file is given in Section 4.

The input file must have extension `.in` (*e.g.* `input_file_name.in`) and contain the following 5 sections in the given order:

1. Parameter list
2. Mesh definition
3. Boundary conditions
4. Initial conditions
5. History points

Below we describe each section and provide examples.

3.1 Parameter List

The parameter list contains several general parameters of the simulation. The first line of the list must contain a number indicating the total amount of parameters in the list, followed by the list of parameters, each one written in a different line; the value of the parameter must be followed by the parameter identifier. The order of the parameters is not important, but their value must precede their name. For any simulation, the following parameters must always be included:

`EQTYPE`	Integer number with the value 0 for the nonlinear formulation and 1 for the linear formulation. Note that the linear formulation is not as developed as the nonlinear formulation
`DT`	Time step (in s)
`NSTEPS`	Number of time steps
`HISSTEP`	Number of time steps until the next step solution is dumped in the output file `input_file_name.his`
`Rho`	Blood density (in Kg m^{-3})
`Viscosity`	Blood viscosity (in Pa s)
`Alpha`	Velocity profile parameter α in the frictional force per unit length of the momentum equation, as defined in [1] (after Eq. (1); the velocity profile is given by Eq. (3)). `Alpha` = 4/3 for Poiseuille flow; `Alpha` = 1 for inviscid flow

The following parameters are optional:

`INTTYPE`	Time integration order (1, 2 or 3) used by the numerical scheme. By default, `INTTYPE` = 2
`IOSTEP`	Number of time steps until the next step solution is dumped in the output file `input_file_name.out`. By default `IOSTEP` = 0 and .out files are not generated.
`Beta`	Constant stiffness parameter $\beta = 4/3\sqrt{\pi}Eh/A_{\rm d}$ (in Pa m^{-1}) applied to all the arterial segments of the model, where E is the Young's modulus of the arterial wall, h is the wall thickness, and $A_{\rm d}$ is the luminal area at the reference pressure (usually the diastolic pressure) (see Eq. (3.3))
`Gamma`	Wall visco-elasticity (in Pa s). Note that wall visco-elasticity is only available for the nonlinear solution
`Ao`	Constant luminal cross-sectional area $A_{\rm d}$ (in m^2) applied to all the arterial segments of the model
`pinf`	Pressure (in Pa) at the outflow of each lumped parameter model. By default `pinf` = 0
`Pext`	External Pressure $P_{\rm ext}$ (in Pa) to the vessel wall (see Eq. (3.2)). By default `Pext` = 0
`Junc_losses`	1: Switch on energy losses at junctions; 0: no energy losses at junctions. By default `Junc_losses` = 0
`Bifurc_losses`	1: Switch on energy losses at bifurcations; 0: no energy losses at bifurcations. By default `Bifurc_losses` = 0
`Periodic`	The inflow boundary condition is periodic. The value of the period is the number given in front of `Periodic` in seconds
`T_initial`	Initial time (in s) for the time-average calculations in `input_file_name_periodic.tex`. By default `T_initial` = 0
`T_final`	Final time (in s) for the time-average calculations in `input_file_name_periodic.tex`. By default `T_final` = `NSTEPS` x `DT`
`SCAL_F`	Scaling factor for the inflow waveform

Additional parameters can be defined by the user; *e.g.* `ELASTIC` is used in `nektar/examples/Experimental/exp37.in` to define a constant Young's modulus for all arterial segments. (This model produces the purely elastic results described in [3].) Note that the number π is defined as `PI` by default throughout the input file.

3.2 Mesh Definition

This section contains the geometrical and mechanical properties of each arterial segment, which is called a 'domain' in Nektar1D. The first line of the section must contain the string 'Mesh' followed by 'Ndomains = *integer*', where the integer indicates the number of domains present. If the simulation has only one domain, we just need to write 'Mesh'.

Each domain definition is started with an opening line containing the number of finite elements (Nel) that make up the domain. For each element, a line is required with four numbers which indicate the (i) lower spatial coordinate x; (ii) upper spatial coordinate x; (iii) polynomial order of the element (p); and (iv) quadrature order of the element (q). For example, a domain with first point $x = -0.075$ m and last point $x = 0.075$ m which is divided into three equispaced elements with a quadrature and polynomial order of 6 is defined as

```
Mesh
3       # Nel
-0.075  -0.025  6  6  # x_lower x_upper p q
-0.025  0.025   6  6  # x_lower x_upper p q
0.025   0.075   6  6  # x_lower x_upper p q
```

Material and geometrical properties can be defined for each domain in *Mesh Definition*, as detailed below for a purely elastic (Section 3.2.2) and visco-elastic (Section 3.2.3) arterial wall. If all the domains have the same properties, their values can be defined in *Parameter List* (see Section 3.1 for more details). First, we present the most generic tube law – relating changes in blood pressure to changes in cross-sectional area – currently available in Nektar1D (Section 3.2.1).

3.2.1 Tube Law

We have implemented a Voigt-type visco-elastic tube law given by

$$P = P_{\mathrm{e}}(A;x) + \frac{\Gamma(x)}{\sqrt{A}}\frac{\partial A}{\partial t}, \tag{3.1}$$

with

$$P_{\mathrm{e}}(A,x) = P_{\mathrm{ext}} + \beta(x)\left(\sqrt{A} - \sqrt{A_{\mathrm{d}}(x)}\right), \tag{3.2}$$

$$\beta(x) = \frac{4}{3}\frac{\sqrt{\pi}\,E(x)h(x)}{A_{\mathrm{d}}(x)}, \tag{3.3}$$

$$\Gamma(x) = \frac{2}{3}\frac{\sqrt{\pi}\,\varphi(x)h(x)}{A_{\mathrm{d}}(x)} = \frac{4}{3}\varphi(x)\frac{h(x)}{D_{\mathrm{d}}(x)}\frac{1}{\sqrt{A_{\mathrm{d}}(x)}}, \tag{3.4}$$

where $P(x,t)$ is blood pressure, $P_{\mathrm{e}}(x,t)$ is the elastic component of pressure, P_{ext} is the external pressure, $A(x,t)$ is the luminal cross-sectional area, $D_{\mathrm{d}}(x)$ is the luminal diameter at the reference pressure (usually the diastolic pressure), $h(x)$ is the wall thickness,

$E(x)$ is the Young's modulus, and $\varphi(x)$ is the wall viscosity. Therefore, $\beta(x)$ is related to the wall elasticity and $\Gamma(x)$ to the wall viscosity. $\beta(x)$ and $\Gamma(x)$ are both independent of the transmural pressure $P - P_{\text{ext}}$. The reference area $A_{\text{d}}(x)$ is the area at $P = P_{\text{ext}}$ and $\frac{\partial A}{\partial t} = 0$, which are typical initial conditions for numerical simulations.

3.2.2 Purely-Elastic Simulations

By default $\Gamma = 0$; *i.e.* all domains have a purely-elastic arterial wall (see Eq. (3.1)). The mechanical properties for a purely-elastic arterial wall can be specified in three different ways; *i.e.* using:

- The stiffness parameter β;
- The product Eh;
- An empirical law relating the pulse wave velocity c and the diameter D_{d} at the reference area A_{d};

The stiffness parameter β
For each domain, the opening line must contain the strings `Beta` and `Area`, as shown in the example below. For each element, β and A_{d} are defined in two different lines, starting with '`Beta =`' and '`Area =`' (or `Ao =`), respectively. (The order matters; first `Beta` and then `Area`.) Elements can have either constant β and A_{d} or varying β and A_{d} with the axial coordinate x along the domain. For example,

```
2  # Nel domain 5  Beta Area
0.0  0.0175  6  6  # x_lower x_upper p q
Beta = 404.063553/(3.2000e-03 + -2.8571e-02*x)/(2.7628e-03 + -2.1984e-02*x)
Area = 4.7022e-06 + -6.7010e-05*x + 2.3874e-04*x*x
0.0175  0.035  6  6  # x_lower x_upper p q
Beta = 404.063553/(3.2000e-03 + -2.8571e-02*x)/(2.7628e-03 + -2.1984e-02*x)
Area = 4.7022e-06 + -6.7010e-05*x + 2.3874e-04*x*x
```

This example is taken from `nektar/examples/Rabbit/Rabbit.in`, which is the model of the rabbit systemic circulation used in [5].

Note that it is possible to define a constant β in *Parameter List* and a variable A_{d} in *Mesh Definition*, and viceversa.

The product Eh
We can prescribe the quantity Eh, with $E(x)$ the Young's modulus of the arterial wall and $h(x)$ the wall thickness. The value of $\beta(x)$ is then computed by Nektar1D as

$$\beta = \frac{4}{3}\frac{\sqrt{\pi}\,Eh}{A_{\text{d}}}. \tag{3.5}$$

For each domain, the opening line must contain the strings `Eh` and `Area`. For each element, Eh is then defined in a new line starting with '`Eh =`'. Note that A_{d} must be defined before Eh. For example:

```
1  # Nel  Eh Area
0.0  0.4  30  30  # x_lower x_upper p q
Area = PI*1E-4
Eh = 480
```

This example is taken from `nektar/examples/Aorta/Ao_XX_It7_Eh.in`. The material properties in this model can also be prescribed using the stiffness parameter β as shown in `nektar/examples/Aorta/Ao_XX_It7.in`. Another example in which Eh is specified is `nektar/examples/Experimental/exp37.in`. In this case a constant Young's modulus (defined in *Parameter List* as `ELASTIC`) is used for all arterial segments [3].

Empirical law
Material properties can be prescribed through the local pulse wave velocity, $c(x,t)$, since this is directly related to $\beta(x)$ as follows,

$$\beta(x) = \frac{2\,\rho\,c^2}{\sqrt{A(x,t)}}. \tag{3.6}$$

Pulse wave velocities can be calculated using the following empirical relationship [6],

$$c = \frac{a}{(D_{\mathrm{d}})^b}, \tag{3.7}$$

where D_{d} is the luminal diameter (expressed in mm) at the reference pressure, and a and $b = 0.3$ are empirical coefficients. The value of β is computed by Nektar1D as

$$\beta = \frac{2\,\rho}{\sqrt{A_{\mathrm{d}}}}\,\frac{a^2}{(D_{\mathrm{d}})^{2b}} \tag{3.8}$$

with $D_{\mathrm{d}} = \sqrt{\frac{4A_{\mathrm{d}}}{\pi}}$ expressed in mm.
For each domain, the opening line must contain the strings `Empirical_I` and `Area`. For each element, a is defined in a line starting with '`a =`'. Note that the reference area A_{d} must be defined before a. For example:

```
1  # Nel domain 3  Empirical_I Area
0.0  0.023  3  3  # x_lower x_upper p q
Area = (4.94808594E-04 - 1.88563815E-03*x + 1.79646802E-03*x*x)
a = CSCAL_a*11.0
```

This example is taken from `nektar/examples/55art/55art_Pe.in`, which corresponds to a model of the 55 larger systemic arteries [1]. In this example, `CSCAL_a` is defined by the user in *Parameter List*.

3.2.3 Visco-Elastic Simulations

To define a domain with a visco-elastic arterial wall we must provide the visco-elastic parameter Γ using the string `Gamma` in the opening line of the domain definition. Note that the reference area A_{d} must be defined before Γ. For example:

```
1  # Nel domain 3  Empirical_I Area Gamma
0.0  0.023  3  3  # x_lower x_upper p q
Area = (4.94808594E-04 - 1.88563815E-03*x + 1.79646802E-03*x*x)
a = CSCAL_a*11.0
Gamma = 4/3*Varphi4*hD/(CSCAL_Ao*(4.94808594E-04 - 1.88563815E-03*x
    + 1.79646802E-03*x*x))^0.5
```

This example is taken from `nektar/examples/55art/55art.in` which corresponds to a model of the 55 larger systemic arteries [1]. In this example, `CSCAL_a` (a scaling factor for a in Eq. (3.8)), `Varphi4` (the value of φ in Eq. (3.4)) and `hD` (the value of the ratio h/D_{d} in Eq. (3.4)) are defined by the user in *Parameter List*.

3.3 Boundary Conditions

This section defines the boundary conditions (BCs) for all the domains (arterial segments) that can be prescribed in Nektar1D. The first line of this section must contain the string `Boundary`. Then, for each domain, at least four lines with BC information must be included: two with information on the BC at the inlet of the domain, two with information at the outlet. Each line must start with a letter defining the type of BC.

In Nektar1D, there are three types of BCs that can be prescribed at the boundaries of a domain:

1. *Prescribed waveform* (usually at the inlet of the network);
2. *Lumped parameter (0-D) models* (usually at the outlets of terminal branches);
3. *Connection with other domains.*

3.3.1 Prescribed Waveform

A blood flow rate, flow velocity, or pressure waveform can be prescribed as BC. This type of BC treats incoming waves in two different ways: they can be reflected back into the domain (*reflective BC*), or they can be fully absorbed by the BC (*absorbing BC*).

A pulse waveform can be described in an external file as a sum of Fourier harmonics (for flow rate signals only), or as the evolution of the variable with time. The file must be called `FileName_IN.bcs` if there is only one domain or `FileName_IN_1.bcs` if there are multiple domains. Alternatively, a waveform can also be defined in the input file as an algebraic function of time (with time denoted by the variable 't').

The following table describes the Nektar1D commands to impose a velocity, pressure and flow rate waveform as BC, in a reflective or absorbing way.

		Reflective BC	Absorbing BC
Velocity u	File .bcs (time - velocity)	`u 2`	`u 3`
	Algebraic function	`u 5`	`u 1`
Pressure p	File .bcs (time - pressure)	`p 2`	`p 3`
	Algebraic function	`p 4`	`p 1`
Flow rate q	File .bcs (time - flow rate)	`q 2`	`q 3`
	File .bcs ($\sum$ harmonics)	`F 0`	`F 8`
	Algebraic function	`q 0`	`q 1`

For example, to prescribe a half sine pressure wave in a reflective the following commands must be used in the *Boundary condition* section of the input file:

```
p 4
p = 1E6*sin(3.1415*t/0.165)*step(0.165-t,0)
p 4
p = 1E6*sin(3.1415*t/0.165)*step(0.165-t,0)
```

To prescribe an *in vivo* flow wave expressed as a sum of harmonics in the file `FileName_IN.bcs`, in a reflective way, the commands at the inlet (or outlet) of the domain are

```
F 0
F 0
```

3.3.2 Lumped Parameter Model

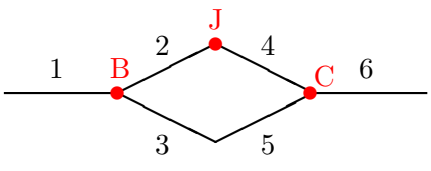
The main types of outlet BC in Nektar1D are summarised hereunder, together with the two lines of commands that are required in the *Boundary condition* section of the input file.

Reflection coefficient	`T` #	if # = 0, complete absorption
	`T` #	if # = 1, complete reflection
Single resistance	`R` #	# is the resistance value in [Pa s m^{-3}]
	`R` #	
2-element windkessel	`w` $\#_C$	$\#_C$ is the compliance value in [m^3 Pa^{-1}]
	`w` $\#_R$	$\#_R$ is the resistance value in [Pa s m^{-3}]
3-element windkessel	`W` $\#_C$	$\#_C$ is the compliance value
	`W` $\#_R$	$\#_R$ equals the sum of resistances $R_1 + R_2$
4-element windkessel	`Z` $\#_C$	$\#_C$ is the compliance value
	`Z` $\#_R$	$\#_R$ equals the sum of resistances $R_1 + R_2$
		The L value is specified in *Parameter List* as `inductance`.

For the 3-element windkessel BC, the value of the first resistance (R_1) is by default computed as the characteristic impedance of the end point of the domain (Z_0). If a numeric value is specified after the total resistance value (*i.e.* `W` $\#_R$ $\#_r$), the value of the first resistance R_1 will be multiplied by the absolute value of this factor: $R_1 = |r| * Z_0$. By default, if not specified, $r = 1$.

3.3.3 Connection with Other Domains

Three types of domain connections can be defined in Nektar1D. The Nektar1D commands for each type are shown below and illustrated using the following example of a network.



- **J: Connection between two domains:** For example, at the outlet of Domain 2 the following two lines of commands must be written to indicate that Domain 2 is connected to Domain 4:
  `J 4 4`
  `J 4 4`

- **B: Splitting Flow Bifurcation:** For example, at the outlet of Domain 1 the following two lines of commands must be written to indicate that Domain 1 is connected to the daughter Domains 2 and 3:
  `B 2 3`
  `B 2 3`
  Similarly, at the inlet of Domain 3 the following two lines of commands must be written to specify the number of the parent domain (Domain 1) and the other daughter domain (Domain 2):
  `B 1 2`
  `B 1 2`

- **C: Merging Flow Bifurcation:** For example, at the outlet of Domain 4, the following two lines of commands must be written to indicate that Domain 4 is connected to Domains 5 and 6:
  `C 5 6`
  `C 5 6`
  Similarly, at the inlet of Domain 6, the commands are:
  `C 4 5`
  `C 4 5`

3.4 Initial Conditions

This section contains the initial values for luminal cross-sectional area (A_0) and blood velocity (U_0) in all the domains of the simulation. The first line of the section must contain the string `Initial condition`. Next, two lines must be included for each domain: the first provides the value of A_0 and the second the value of U_0. Both A_0 and U_0 can either be constant values or functions of x. For example,

```
Initial condition
a = PI*1E-4          #Initial value of area (in m^2)
u = 0                #Initial value of flow velocity (in m/s)
```

The initial area A_0 can be prescribed to be equal to the area A_d defined in *Mesh Definition* by using `a = Ao`. For example,

```
Initial condition
a = Ao               #Initial value of area equal to the area in Mesh Definition (in m^2)
u = 0                #Initial value of flow velocity (in m/s)
```

The initial area is usually computed as the area that yields the prescribed area A_d in *Mesh Definition* at a given pressure P_d. This is achieved by replacing $P_\mathrm{e} = 0$, $P_\mathrm{ext} = P_\mathrm{d}$, and $A = A_0$ in Eq. (3.2), which yields

$$A_0 = \left(\sqrt{A_\mathrm{d}} - \frac{P_\mathrm{d}}{\beta} \right)^2 . \tag{3.9}$$

Nektar1D can calculate A_0 using Eq. (3.9) if the simulation is launched using the flag `-i`, followed by a positive real number with the value of the pressure P_d (in Pa). For example,

```
./oneDbio -i 12666.66 55art_Pe.in
```

Using the flag `-i` overwrites all values of A_0 in *Initial Condition*. Note, however, that the section *Initial Condition* must still be included in the input file. This example is taken from `nektar/examples/55art/55art.in` [1].

3.5 History Points

This section specifies the points in the arterial network where haemodynamic data is dumped. We call them *history points*. The first line must contain the string `History` and the second must contain the number of domains with history points. Next, tor each domain with history points, two lines must be included. The first has an integer with the number of history points within the domain, followed by another integer with the number of the domain. The second contains the axial location of each history point in the domain. The text below is an example on how to specify history points at the inlet of Domain 1 and at three points within Domain 17. It is taken from `nektar/examples/Rabbit/Rabbit.in`.

```
History Pts
2                    #Number of domains with history points
1 1                  #Number of points and domain identifier
0.0
3 17                 #Number of points and domain identifier
0.0433 0.065 0.0866
```

4 Running an Input File

To execute Nektar1D simply open a terminal window, change to the directory `nektar/1DBio/Linux` and type `./oneDbio` followed by the name and extension of the input file; *e.g.*

```
./oneDbio input_file_name.in
```

To execute Nektar1D from the directory that contains the input file, just export the path of `./oneDbio` to the shell, as described in Section 2. All output files (see Section 5) will be dumped in the directory from which `./oneDbio` is executed.

The following flags are available after the command `./oneDbio`:

1. `-p #`: Assigns the value given in `#` as the polynomial order for all the elements in the simulation. This flag replaces any values given in the input file.

2. `-q #`: Assigns the value given in `#` as the quadrature order of all the elements in the simulation. This flag replaces any values given in the input file.

3. `-N #`: Divides all domains into the number of equispaced elements given in `#`. This flag replaces the number of elements given in the input file.

4. `-O`: Generates output files (with extension `.out`) containing the variables of the simulation evaluated at the quadrature points of all the domains for different times. See Section 5 for more details.

5. `-L`: Generates output files (with extension `.lum`) containing the variables of the lumped parameter models which change with time. See Section 5 for more details.

6. `-A`: Dumps the luminal cross-sectional area, A – instead of the blood pressure, P – in the output file with extension `.out`.

7. `-B`: Generates output files (with extension `.bcs`) containing time-varying characteristic information (including the Riemann variables) at all the boundaries of the arterial network. See Section 5 for more details.

8. `-a`: Dumps the following additional variables in the history files: (i) forward- and backward-travelling Riemann (or characteristic) variables (in m s^{-1}); (ii) spatial-averaged blood pressure (Pa), blood flow velocity (m s^{-1}), blood flow rate (m s^{-3}), and luminal cross sectional area (m^2) across each domain; (iii) forward and backward-travelling components of pressure (Pa) and velocity (m s^{-1}); and (iv) space derivatives: pressure gradient term (Pa m^{-1}), convective acceleration term (m s^{-2}), flow gradient term ($m^2 s^{-1}$).

   The default variables are time (s), blood pressure (Pa), blood flow velocity (m s^{-1}), blood flow rate (m s^{-3}), luminal cross sectional area (m^2), and an integer that refers to the point label indicated in the heading (see Section 5).

9. `-d`: Dumps a report file called `input_file_name.txt` containing several parameters of the simulation, including haemodynamic variables at the initial time. The file is generated without running the simulation.

10. `-t`: Dumps a LaTeX report file called `input_file_name.tex` containing several parameters of the simulation, including haemodynamic variables at the initial time. The file is generated without running the simulation.

11. `-R`: Dumps space-averaged variables for the whole arterial network in a file called `input_file_name.avg`. See Section 5 for more details.

12. `-r #`: Scales all peripheral resistances by multiplying them by the value specified in `#`.

13. `-c #`: Scales all peripheral compliances by multiplying them by the value specified in `#`.

14. `-i #`: Sets the initial areas (A_0) in all elements to the values that will produce the areas A_d (specified in *Mesh Definition*) at the pressure given by the value specified in `#`. Equation (3.9) is used to calculate A_0.

5 Output Files

By default, Nektar1D generates the following output text files:

1. *LaTeX report file*: It is called `input_file_name.tex` and, when compiled in LaTeX, displays several parameters of the simulation and haemodynamic variables at the initial time.

2. *Report file*: It is called `input_file_name.txt` and contains similar information to the previous file in a format that can be read by a normal text editor.

3. *Period LaTeX report file*: It is called `input_file_name_period.tex` and, when compiled in LaTeX, displays several parameters of the simulation and haemodynamic variables for the time period starting at the value given by `T_initial` in *Parameter List* (Section 3.1) and ending at the value given by `T_final`.

4. *Property file*: It is called `input_file_name.prp` and contains the following information for each domain: length, inlet radius, outlet radius, inlet wave speed, outlet wave speed, inlet area, outlet area, inlet Γ, and outlet Γ. Radii, wave speeds and areas are given for the initial time.

5. *Period property file*: It is called `input_file_name_period.prp` and contains the following information for each domain: length, inlet radius, outlet radius, inlet wave speed, outlet wave speed, inlet area, outlet area, inlet Γ, and outlet Γ. Radii, wave speeds and areas are given for the time period starting at the value given by `T_initial` in *Parameter List* (Section 3.1) and ending at the value given by `T_final`. Note that this flag won't work for a reflection coefficient boundary condition (Type 'T' in Section 3.3.2).

6. *Stiffness parameter file*: A single file called `input_file_name.bet` that contains, for each domain with history points, the number of history points (initial line) and the value of the stiffness parameter β at the x position of each history point (each β in a different line).

7. *History file/s*: They are called `input_file_name.his` if the model consists of a single domain or `input_file_name_#.his` if the model consists of multiple domains, with `#` the number of each domain with history points defined in the *History Points* section of the input file (see Section 3.5). Each history file consists of a header with information on the number of history points in the domain and their x location. After the header there is a matrix of numbers with the following information: time (in s) in the first column; blood pressure (Pa) in the second; blood flow velocity (m s^{-1}) in the third; blood flow rate (m s^{-3}) in the fourth; luminal cross sectional area (m^2) in the fifth; and an integer in the sixth referring to the point label indicated in the heading. For a visco-elastic tube law, the elastic component of pressure, P_e, is dumped in the third column (in Pa), followed by flow velocity, flow rate, etc. The temporal spacing of these quantities is defined by `HISSTEP` in *Parameter List* (Section 3.1).

8. *History points location*: A single file called `input_file_name.loc` with the number of domains with history points (first line) and, for each domain with history points, the number of history points (initial line) and the x position of each history point (each in a different line).

In addition, the following output text files can be generated using the flags described in Section 4:

1. *Output file/s*: Files containing the variables of the simulation evaluated at the quadrature points of all the domains for different times. They are created using the flag `-O`. They are called `input_file_name.out` if the model consists of a single domain or `input_file_name_#.out` if the model consists of multiple domains, with `#` the number of each domain. Each output file consists of a header with information on the number of elements, points dumped (total and for each element) and time steps dumped. The number of time steps dumped is defined by `IOSTEP` in *Parameter List* (Section 3.1). For each time step, there is a matrix of numbers with the following information: x location (m) in the first column; blood pressure (Pa) in the second; blood flow velocity (m s^{-1}) in the third; forward characteristic (m s^{-1}) in the fourth; and backward characteristic (m s^{-1}) in the fifth. The flag `-A` (in addition to `-O`) dumps the luminal cross-sectional area (A), instead of blood pressure (P), in the second column.

2. *Lumped parameters file/s*: They are obtained using the flag `-L`.
It is called `input_file_name_out.lum` if there is only one domain with an outflow boundary condition or `input_file_name_out_#.lum` if there are multiple domains with outflow boundary conditions, with `#` the number of each terminal domain. Each file contains a matrix of numbers with the following information: time (s) in the first column; blood pressure (Pa) at the inflow of the lumped parameter model in the second; blood flow rate (m s^{-3}) at the inflow of the lumped parameter model in the third; blood pressure (Pa) at the compliance in the fourth; and blood flow rate (m s^{-3}) at the outflow of the lumped parameter model in the fifth.

3. *Inflow characteristic information file/s*: They are obtained using the flag `-B`. It is called `input_file_name_IN.bcs` in a model with only one domain or `input_file_name_IN_#.bcs` in a model with multiple domains, with `#` the number of each domain with an inflow boundary condition. Each file contains a matrix of numbers with the following information calculated at the first point of the domain: time (s) in the first column; $\rho c W_\mathrm{f}$ (Pa) in the second, with ρ blood density, c pulse wave velocity, and W_f the forward characteristic variable; $-\rho c W_\mathrm{b}$ (Pa) in the third, with W_b the backward characteristic variable; $W_\mathrm{f}/2$ (m s^{-1}) in the fourth; $W_\mathrm{b}/2$ (m s^{-1}) in the fifth; and $-W_\mathrm{b}/W_\mathrm{f}$ in the sixth. These information was used in [7] to study the effect of inflow boundary conditions on the shape of the pressure and flow waveforms.

4. *Outflow characteristic information file/s*: They are obtained using the flag `-B`. It is called `input_file_name_OUT.bcs` in a model with only one domain or `input_file_name_OUT_#.bcs` in a model with multiple domains, with `#` the number of each domain with an outflow boundary condition. Each file contains a matrix of numbers with the following information calculated at the last point of the domain: time (s) in the first column; $\rho c W_\mathrm{f}$ (Pa) in the second, with ρ blood density, c pulse wave velocity, and W_f the forward characteristic variable; $-\rho c W_\mathrm{b}$ (Pa) in the third, with W_b the backward characteristic variable; $W_\mathrm{f}/2$ (m s^{-1}) in the fourth; $W_\mathrm{b}/2$ (m s^{-1}) in the fifth; and $-W_\mathrm{b}/W_\mathrm{f}$ in the sixth. These information was used in [7] to study the effect of outflow boundary conditions on the shape of the pressure and flow waveforms.

5. *Average parameters file*: It is obtained using the flag `-R`. It is called

`input_file_name.avg` and contains the space-average information described in the header and in [8].

6 Examples

We provide input files for the following examples of 1-D simulations from our peer-reviewed publications. These can be found in the folder `nektar/examples/`.

`55art/` – Model of the 55 larger systemic arteries in the human under normal physiological conditions [1]. Two input files are provided: `55art/55art_elas.in` simulates the arterial wall as a purely-elastic material and `55art/55art.in` as a visco-elastic material.

`Adan56/` – Model of the 56 larger systemic arteries in the human described in [9].

`AoBif/` – Model of the human aortic bifurcation described in [4, 9].

`Aorta/` – Model of the human upper thoracic aorta described in [4, 9].

`CCA/` – Model of the human common carotid artery described in [4, 9].

`Experimental/` – 37-artery network simulating blood flow in the cardiovascular simulator rig described in [3, 9].

`Pulse/` – Single pulse propagation in a straight reflection-free vessel as described in [9, 10]. Two input files are provided: `Pulse/Pulse.in` considers an inviscid fluid and `Pulse/Pulse_vf.in` considers a viscous fluid with a viscosity of 4 mPa s^{-1}.

`Rabbit/` – Model of the 59-artery larger systemic arteries in the rabbit described in [5].

7 Source Code Structure

The main functions of the code are located in `nektar/1DBio/src/main.C` and the input file is read by functions in `setup.C`. The header files are located in `nektar/1DBio/include` and the external libraries in `nektar/Hlib` and `nektar/Veclib`.

References

[1] Alastruey J, Parker K, Sherwin S. *Arterial pulse wave haemodynamics. In Anderson (Ed.) 11th International Conference on Pressure Surges*, chap. 7. Virtual PiE Led t/a BHR Group (ISBN: 978 1 85598 133 1), 2012; 401–442.

[2] Alastruey J, Xiao N, Fok H, Schaeffter T, Figueroa C. On the impact of modelling assumptions in multi-scale, subject-specific models of aortic haemodynamics. *J. R. Soc. Interface* 2016; **13**:1–17.

[3] Alastruey J, Khir A, Matthys K, Segers P, Sherwin S, Verdonck P, Parker K, Peiró J. Pulse wave propagation in a model human arterial network: Assessment of 1-D visco-elastic simulations against *in vitro* measurements. *J. Biomech.* 2011; **44**:2250–2258.

[4] Xiao N, Alastruey J, Figueroa C. A systematic comparison between 1-D and 3-D hemodynamics in compliant arterial models. *Int. J. Numer. Meth. Biomed. Eng.* 2014; **30**:204–231.

[5] Alastruey J, Nagel S, Nier B, Hunt A, Weinberg P, Peiró J. Modelling pulse wave propagation in the rabbit systemic circulation to assess the effects of altered nitric oxide synthesis. *J. Biomech.* 2009; **42**:2116–2123.

[6] Reymond P, Merenda F, Perren F, Rüfenacht D, Stergiopulos N. Validation of a one-dimensional model of the systemic arterial tree. *Am. J. Physiol. Heart Circ. Physiol.* 2009; **297**:H208–H222.

[7] Alastruey J, Parker K, Peiró J, Sherwin S. Analysing the pattern of pulse waves in arterial networks: a time-domain study. *J. Eng. Math.* 2009; **64**:331–351.

[8] Alastruey J. On the mechanics underlying the reservoir–excess separation in systemic arteries and their implications for pulse wave analysis. *Cardiov. Eng.* 2010; **10**:176–189.

[9] Boileau E, Nithiarasu P, Blanco P, Müller L, Fossan F, Hellevik L, Donders W, Huberts W, Willemet M, Alastruey J. A benchmark study of numerical schemes for one-dimensional arterial blood flow modelling. *Int. J. Numer. Meth. Biomed. Eng.* 2015; **31**:1–31.

[10] Alastruey J, Passerini T, Formaggia L, Peiró J. Physical determining factors of the arterial pulse waveform: theoretical analysis and estimation using the 1-D formulation. *J. Eng. Math.* 2012; **77**:19–37.