CardioELSE

Version 1.0
May 31st 2011

G. Cuccuru, G. Fotia, F. Maggio
Bioinformatics Laboratory, CRS4
http://www.crs4.it/bioinformatics

Abstract

CardioELSE is a computational tool for cardiac simulation. In detail, it is a parallel solver for the bidomain equations based on spectral elements developed on unstructured all-hexahedral grids. The current cell level electrophysiology functionality in CardioELSE includes the Luo–Rudy I membrane model. It has been developed within the EU FP7 cardiac modeling project preDiCT (www.vph-predict.eu/) as a research and proof-of-principle software. CardioELSE was tested on benchmark cases and on patient-specific realistic heart models. Parallelization and portability are realized through the PETSc parallel library and the code will run on any cluster with from 1 to 128 or more CPUs. The package is being developed by a team mainly based at the Bioinformatics Laboratory of CRS4 (www.crs4.it/bioinformatics), drawing inspiration from many years expertise of research and application of spectral elements to a diverse set of compute-intensive physical problems on complex geometries from fields including geophysics, continuum mechanics, and electromagnetics.

1 What is CardioELSE?

CardioELSE is a computational tool for the simulation of the cardiac electrophysiology on large, realistic models of the heart. CardioELSE aims to simulate the propagation of electrical potential in cardiac tissue, modeled by the bidomain equations; the cell response is described by the Luo–Rudy I model. It is based on spectral elements and is designed to run in parallel on distributed memory machines.

2 Introduction

The discretization of a realistic heart geometry with a satisfactory node spacing can result in a mesh containing millions of nodes. Hence, the algebraic systems resulting from low-order methods like Finite Elements (FE) are very large and whole heart simulation using the bidomain model is a demanding scientific computing problem [2]. An alternative to mesh refinement (used by FE) is to improve the quality of numerical simulations by expressing the solution in terms of polynomial functions of high degree on a relatively coarse mesh. This is the idea at the base of spectral elements [4]. As far as we know, the only attempts to use spectral elements for cardiac simulation include the use of Fourier polynomials, when periodic boundary conditions can be assumed, algebraic polynomials in triangles (for 2D applications, see [5] and references therein), and the spectral smoothed boundary method [3].

3 CardioELSE in practice

Here we provide a quick review of how CardioELSE works, including preprocessing, analysis and visualization of the results.

3.1 Preprocessing

CardioELSE accepts hexahedral grids provided either in Abaqus format or in Chaste format (feature not included in this release). In all cases, CardioELSE will take care to add extra grid points corresponding to spectral degree > 1: this operation is completely transparent to the user. Figure 1 shows an example of an Abaqus grid (partial view). Grid point coordinates are listed in Fortran style, with indices starting from 1: gaps in the indices are not allowed. Element connectivities are listed as usual: element indices (1st column) are discarded and automatically set in consecutive order starting from 1. The element headline includes the definition of the element type (C3D8R are 8-points hexahedra for Abaqus), while ELSET=EB1 denote that following elements below to block 1.

```
*HEADING
UPF model with top cut - Author: G. Fotia, CRS4 - Jan 2011
Units: [cm] - grid size: 0.1 cm (average)
*NODE
       1,     0.0513431,    -1.7192150,    -4.2320250
       2,     0.3564533,    -1.8158830,    -4.2055330
       3,     0.4163079,    -1.7833180,    -3.9767760
       4,     0.1278995,    -1.7087120,    -4.0294250
                                  ...
*ELEMENT, TYPE=C3D8R, ELSET=EB1
       1,        1,        2,        3,        4,        5,        6,        7,        8
       2,        9,       10,       11,       12,       13,       14,       15,       16
       3,       17,       18,       14,       13,       19,       20,       21,       22
       4,       13,       14,       15,       16,       22,       21,       23,       24
       5,       25,       26,       27,       28,       29,       30,       31,       32
                                  ...
```

Figure 1: An example of all hexahedra grid in Abaqus format. Only 4 nodes and 5 connectivities are shown.

The definition of different volume blocks allows to account for piecewise constant cardiac tissue (feature not included in this release). For the sake of simplicity, boundary conditions and source regions are set via scalar cut planes (see section 3.2.2) rather than through mesh elements. The same considerations apply to grids in Chaste format: in that case, only .node and .ele files are needed for each grid[1].

3.2 Analysis setup

The setup of the numerical simulation is provided by means of several input files, which should be edited and customized by the user.

3.2.1 File “CardioELSE.input”

An example of the file *CardioELSE.input* is displayed in figure 2.

```
./MODELS/UPFNoTop_0.1cm.model
1                              ! spectral degree
xdmf                           ! graphic files format (vtk/xdmf)
.false.                        ! write iteration history on a file
.false.                        ! write SE grid (Abaqus format)
.false.                        ! write non-zero-entries-per-row histogram
```

Figure 2: Structure of the input file "CardioELSE.input"

The first line is the name of the model file (described later): then comes the spectral degree. At line 3, the choice between VTK or XDMF format (feature not included in this release) for the snapshots of the transmembrane potential at given times is set. Other lines concern the possibility to: write the iteration

1 In our opinion, the best all-hexahedra grid generator currently available is Cubit (cubit.sandia.gov), which also comes with the Tetmesh engine by Distene/INRIA for tetrahedral grid generation. Cubit allows to save grids in the format illustrated in figure 1 by selecting the options *Abaqus format.* A former invocation of the command *compress* eliminates gaps in the numbering of grid points. Grids in the format at hand can also be imported by Cubit, modified (for instance to refine or smooth the mesh) and then exported again. Of course, mesh generators other than Cubit can be used as well, provided the structure shown in figure 1 is preserved.

history on a file; generate a file containing the whole spectral grid, including the newly added nodes for spectral degrees > 1, in Abaqus format (feature not included in this release); provide a text file with the non-zero-entries-per-row distribution, useful for particular tasks (e.g. the choice of the proper matrix format for the implementation of the matrix-vector product on GPUs).

3.2.2 The model file

An example of model file is displayed in figure 3. The first line contains the name of the grid file. The extension *.inp* indicates a file written in Abaqus format. When using the Chaste format, the user should provide the name of the files without extension. For instance, if the following is specified:

```
./GRIDS/UPFNoTop_0.1cm          ! grid file
```

CardioELSE will look for the files UPFNoTop_0.1cm.node and UPFNoTop_0.1cm.ele. Line 2 contains the initial and final times of the simulation, along with the time-step to be used for the solution of the PDE and the ODE, respectively. All times are expected in milliseconds. A further, optional argument is the output time-step, described later. Line 3 refers to stimulus parameters: start, end, duration, period. Refer to figure 5 for the interpretation of such parameters. Line 4 deals with boundary conditions (BCs). In general, CardioELSE expects BCs to be of homogeneous Neumann type on the whole boundary except for a specific region defined by the user. In that region BCs are set independently of the internal and external potential u_i , u_e: for instance:

```
DN                              ! boundary conditions on u_i, u_e (Dir/Neu)
```

means that, on that region, a Dirichlet BC for u_i and a Neumann BC for u_e have been set.

```
./GRIDS/UPFNoTop_0.1cm.inp      ! grid file
0.   100.   0.005  0.01  20.     ! t0, t_fin, dt_ODE, dt_PDE [,dt_out] (ms)
0.   1200.   2.   375.           ! stimulus start, end, duration, period (ms)
DN                              ! boundary conditions on u_i, u_e (Dir/Neu)
-83.853      0.0                ! amplitudes Dirichlet solution (mV)
0.        -4000.0               ! amplitudes Neumann stimulus (microA/cm^2)
0.           0.0                ! amplitudes applied currents (microA/cm^2)
z < -5.                         ! location of BCs or of applied currents
3                               ! # receivers
0.       2.      -5.            ! receiver's coordinates (cm)
0.       2.      -2.5           ! receiver's coordinates (cm)
0.       2.       0.            ! receiver's coordinates (cm)
0                               ! # snapshots
```

Figure 3: Structure of the model file

Lines 5 and 6 refer to the right hand side of BCs: in this case u_i=*-83.853 mV,* while the Neumann stimulus on u_e equals *-4000.0* $\mu A/cm^2$.

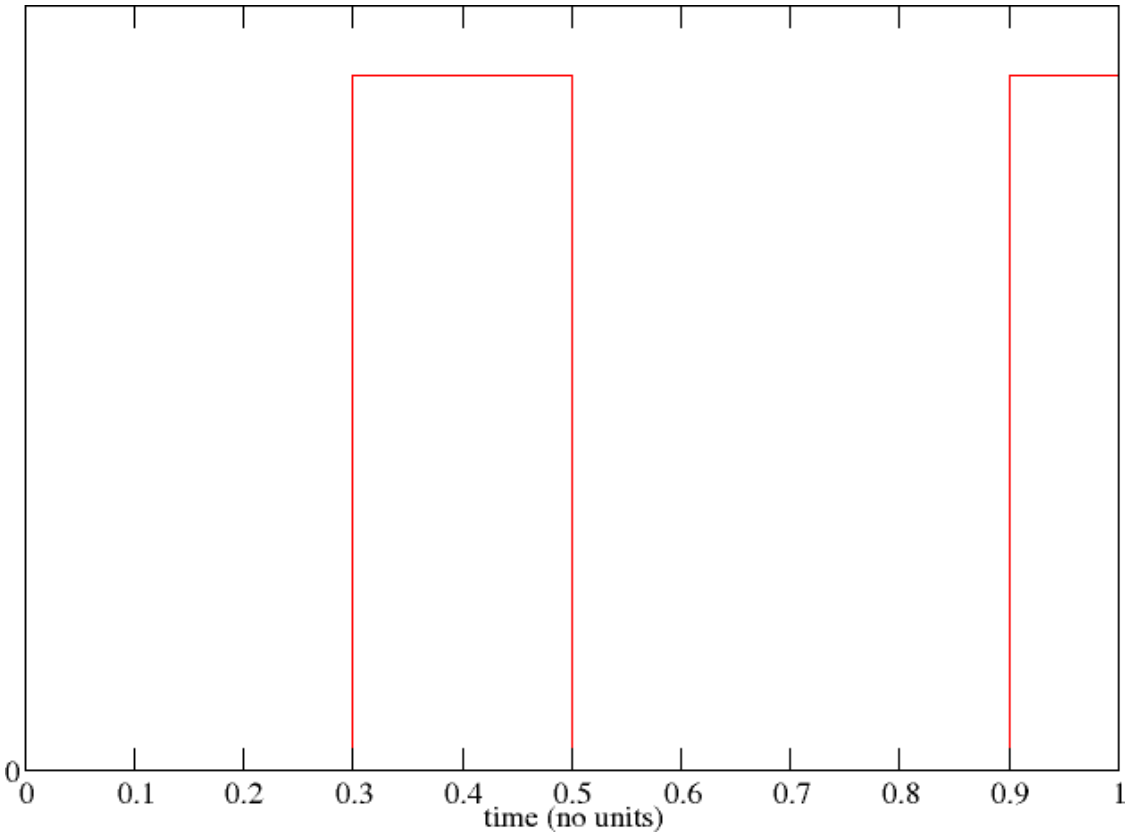

Figure 4: Example of stimulus with start=0.3, end=1., duration=0.2, period=0.6. No time units specified.

Line 7 contains information on amplitudes of (possibly) applied currents, respectively to internal and external cellular domain. Through line 8 the user can specify the region where BCs and possibly external currents are applied, using 1, 2 or 3 scalar cut planes (no more than one per dimension). In the case shown in figure 3, the condition identifies that part of the boundary such that $z<-5$; another example of admissible condition is the following:

```
z < -5. ,  x>0, y<3.14          ! location of BCs or applied currents
```

Note that conditions on different coordinates are separated by commas. Line 9 refers to `nrec`, the number of receivers, i.e. points of the cardiac tissue where the time-history of transmembrane potential should be recorded and written into a file (e.g. file "rec_023.d" contains the time-history of the 23rd receiver). Receivers' coordinates (in cm) follow from line 10 to 9 + `nrec`. Users should be aware that, within this code release, the solution is not interpolated on the exact position of receivers: instead, the closest grid point is used. When working with coarse grids, this can give rise to a small bias on the fast depolarization phase. Lastly, line 10 + `nrec` is used to specify `nsnap`, the number of snapshots, i.e. the records of the full 3D transmembrane potential in the whole domain at given time. There are two options for specifying snapshot times: explicitly, indicating a value of `nsnap`>0 and then `nsnap` values for the snapshot times (in ms); automatically, specifying `nsnap`=0 and a value for the optional output time-step `dt_out` (line 2). For instance, with the model file of figure 3, snapshots will be saved every 20 ms starting from t=0. In both cases, snapshots will be saved in a file called "V_nnn.vtk.gz" or "V_nnn.xdmf", depending on the format selected, where nnn is the number of the snapshot. If no snapshots are required, then `dt_out` should not be indicated, and `nsnap` should be set to 0.

3.2.3 File “membrane.input”

An example of the file “membrane.input” is displayed in figure 5.

```
1400.                          ! Chi [1/cm] membrane area per tissue volume
1.0                            ! C_m [microF/cm^2] surface capacitance
1.75                           ! D^i [10^(-3)/(Ohm x cm)] conductivity tensor
7.0                            ! D^e [10^(-3)/(Ohm x cm)] conductivity tensor
```

Figure 5: Example of the file "membrane.input"

The file is self-descriptive. While the CardioELSE data structure has been designed for general conductivity tensors, in this release the tensors are supposed to be proportional to the identity matrix, with proportionality constants defined by `D^i` and `D^e` in figure 5.

3.2.4 File “lin_solv.input”

An example of the file “lin_solv.input” is displayed in figure 6.

```
1                         ! algebraic_solver (0-> CG + diagonal precond.; 1->PETSc)
!
! ----------- inputs for the CG + diagonal precond.  solver --------------
10000   10000             ! max # iterations, conditional max # iterations
1.e-06  1.e-06            ! tolerance, conditional tolerance,
1                         ! solver (1--->CG, 2--->CGS, 3--->SYMMLQ, 4--->symm_QMR)
1                         ! use diagonal preconditioner? (0--->no; 1--->yes)
! ------------------------------------------------------------------------
!
! ------------------------ inputs for PETSc -----------------------------
1.e-05                    ! rtol
1.e-50                    ! abstol
1.e+10                    ! dtol
10000                     ! maxits
! ------------------------------------------------------------------------
```

Figure 6: Example of the file "lin_solve.input"

Through this file it is possible to set the options for the solution of the algebraic system: this task is performed using the library PETSc[2], unless the label in the first line is set to 0: in this case a simple solver based on Conjugate Gradients (CG) with diagonal preconditioner is used[3] (available only for the sequential case). In this release of CardioELSE, `conditional max # iterations` and `conditional tolerance` are not used. As for PETSc, parameters `rtol`, `abstol`, `dtol`, `maxits`, may be specified, such that `maxits` is the maximum number of iterations, and convergence is detected at iteration `k` if `|| r_k || = || b - A*x_k || < max (rtol * ||b|| , abstol)` .
These settings may be overridden by run-time option specification (see the PETSc manual [1]).

2 S. Balay, J. Brown, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, PETSc Web page, www.mcs.anl.gov/petsc/, 2011

3 This has been done to prevent time-to-time PETSc warnings relating to a non-positive-definite preconditioner when using CG

3.3 Run

In this section we focus on analyses performed by PETSc: use of the sequential Conjugate Gradients with diagonal preconditioner is straightforward. Parallel analyses on either computer clusters or shared-memory machines can be launched by standard PETSc invocations, e.g.

```
mpirun -np 16 CardioELSE_V1.0_Intel.exe -pc_type bjacobi -ksp_type gmres -ksp_monitor_true_residual
```

for use of the block Jacobi preconditioner with the GMRES solver. These settings override those specified in the file “lin_solv.input”: see the PETSc manual [1] for a full list of possible options. Since the option `-ksp_monitor_true_residual` forces PETSc to print the true residual at each iteration, it can slow down the simulation and should be used only for testing or convergence studies, not for timing.

Once the simulation starts, a simple text report is visualized, including the mean features of the model and the status of the analysis, as shown in figure 7.

```
reading the gridfile './GRIDS/UPFNoTop_0.1cm.inp' ... done.

average dx,dy,dz = 0.101 , 0.113 , 0.123
X range: -4.21 to 3.96 ( range extension: 8.17 )
Y range: -4.11 to 7.12 ( range extension: 11.2 )
Z range: -5.93 to 1.22 ( range extension: 7.16 )

spectral degree:                                2
# macro-gridpoints:                           13691
# spectral gridpoints:                         95981
# non-Dirichlet spectral DOF :              189867
# nnz:                                                  6629891
# elements:                                        10328
# Dirichlet BC elements:                       484
# Neumann BC elements:                     484
t_0, t_fin, dt_ODE, dt_PDE (all [ms]) =  0.0     1.0    0.50E-02 0.10E-01

Time loop has started (PETSc)...
25% - 50% - 75% - 100%            done
```

Figure 7: The screen report on model features and analysis status

Among them, the average grid size; the domain extension (useful to check if the proper units are used: CardioELSE needs grid point coordinates in cm); the spectral degree; the number of spectral grid points, including those added when the spectral degree is greater than 1; the size of the linear system after that the degrees of freedom corresponding to Dirichlet conditions have been dropped; the time-advancing stage of the simulation.

3.4 Output

CardioELSE produces a few output files by default: some others are requested on user demand.

3.4.1 Time-histories (optional)

File “rec_nnn.d” contains the time-history evaluated at the nnn-th receiver, in ASCII X-Y format, ready

to use with any simple 2D visualization tool, for instance Xmgrace *(plasma-gate.weizmann.ac.il/Grace/)*.

3.4.2 Snapshots (optional)

File “V_nnn.vtk” or “V_nnn.xmdf” contains the snapshot evaluated at the nnn-th time, ready to use with any tool for 3D visualization, like MayaVi *(mayavi.sourceforge.net/)* or Paraview (www.paraview.org/).

3.4.3 Spectral grid (optional)

If requested, CardioELSE generates an Abaqus file containing the whole spectral grid. This option is disabled in this release.

3.4.4 Convergence history (optional)

Upon request, the file “iterations_info.output” is generated, containing the history of convergence of the iterative method used, along with the main features of the method itself.

3.4.5 Stimulus (by default)

The file “stimulus.d” contains the time-history of the stimulus, in ASCII X-Y format, ready to use with any simple 2D visualization tool.

3.4.6 Execution times (by default)

Execution times of CardioELSE are measured and reported in the file “execution_times.d” which is incremented at each analysis and includes: the spectral degree; the time needed to read the grid, to carry out the preprocessing (building the spectral grid, assembling arrays, etc.), to perform the time-loop; the average number of iterations.

3.4.7 Non-zero-entries-per-row (optional)

CardioELSE may provide a text file named “non_zero_histogram.d” with the non-zero-entries-per-row distribution.

4 Referencing CardioELSE

The software is released as citeware. The following references are appropriate: [6], [7], [8].

5 Availability and portability

CardioELSE may be downloaded from www.comlab.ox.ac.uk/chaste/download.html in the following forms: source code (Fortran 90). It is released as open-source code under the GNU Lesser General Public License (either version 3.0 or, at the user’s option, any later version).
CardioELSE can be run on any Linux platform with access to the following libraries (version numbers indicate those for which regular testing is carried out – other versions may also work, but are not currently supported): MPI, PETSc (version 3.0 and 3.1), and BLAS/LAPACK or Intel MKL.
In order to compile from source, the following tools are required: GNU or Intel compiler.

6 Licence information

PARTICULAR PURPOSE. See the terms and conditions of the GNU Lesser General Public License for more details.

Acknowledgements
Authors are in debt with the participants to the project preDiCT for useful discussion and suggestions, in particular with R. Nobes and J. Southern from Fujitsu Laboratories of Europe, and with the researchers of the Computational Biology Group at the University of Oxford.

Bibliography

1. Balay S, Brown J, Buschelman K, Eijkhout V, Gropp WD, Kaushik D, Knepley MG, McInnes LC, Smith BF, and Zhang H. *PETSc Users Manual*, ANL-95/11 - Revision 3.1, Argonne National Laboratory, 2010

2. Bordas R, Carpentieri B, Fotia G, Maggio F, Nobes R, Pitt-Francis J, and Southern J. *Simulation of cardiac electrophysiology on next-generation high-performance computers.* Philosophical Transactions of the Royal Society A, special issue: The virtual physiological human: building a framework for computational medicine, 367:1951–1969, 2009.

3. Bueno A. Mathematical modeling and spectral simulation of genetic diseases in the human heart . Ph.D. Dissertation. University of Castilla-La Mancha . 2007

4. Casadei F, Fotia G, Gabellini E, Maggio F, and Quarteroni A. *A mortar spectral/finite element method for complex 2d and 3d elastodynamic problems.* Computer Methods in Applied Mechanics and Engineering, 191,45:5119–5148, 2002.

5. Jeannequin N, Whiteley J, and Gavaghan D. *Unstructured spectral elements applied to the bidomain model.* AIP Conference Proceedings, 936(1):292–295, 2007.

6. Maggio F, Carpentieri B, and Fotia G. *A 3D Solver for the Bidomain Equations Based on Unstructured All-hexahedra Spectral Elements*. Technical Report 09/50. CRS4, Center for Advanced Studies, Research and Development in Sardinia. Cagliari, Italy, Aug 2009.

7. Maggio F, Carpentieri B, and Fotia G. *Spectral Elements Applied to Cardiac Simulation: a Comparison with Finite Elements.* Technical Report 09/51. CRS4, Center for Advanced Studies, Research and Development in Sardinia. Cagliari, Italy, Sep 2009.

8. Maggio F, Southern J, Fotia G, and Cuccuru G. *A 3D Spectral Element Solver for the Bidomain Equations*. Poster presented at the VPH 2010 Annual Conference 30 September - 1 October 2010 Brussels, Belgium, 2010.