Joint Institute of Computational Science (JICS)
CSURE Internship

Submitted by
Ciara Thompson

Submitted to
Dr. Kwai Wong

In requirements of the JICS CSURE Internship
June 2013

Introduction

The Joint Institute of Computational Sciences (JICS) was established by the Oak ridge National Laboratories and the University of Tennessee to advance scientific discovery and state-of-the-art engineering and to further knowledge of computational modeling and simulation by contributing to the education of a new generation of scientists and engineers well-versed in the application of computational modeling and simulation for solving the most challenging scientific and engineering problems. The Computational Science for Undergraduate Research Experiences (CSURE) REU program is a student Internship program organized by JICS which serves to expose undergraduate students to emergent computational science fields in five domain sciences: chemistry, material sciences, systems biology, engineering mechanics, atmospheric sciences, and parallel solvers on emergent platforms

Goal

My project for this summer is entitled the Multi-physics Simulation of the Biomedical Processes – Heart Electrophysiology. The goal of this project is to utilize computer programs to study and simulate the multi-physics phenomena of the heart. These phenomena are the interactions between the electrical, physiological, mechanical, and fluid interactions of the heart. My focus during this summer's internship is to program the electrical and physiological interactions of the heart and simulate them using parallel computing.

Steps to Achieve Goal

The steps that I am going to take to achieve my goal during this internship are as follows

- Study and understand the governing equations for simulating the heart: Monodomain Model and Beeler-Reuter Model.
- Develop the geometry and mesh of the heart.
- Program the electrical models: Beeler-Reuter and Monodomain model.
- Examine the interaction between the electrical and physiological effects of the heart.

Training

In order to utilize the computer programs I was trained in some programming software. I was trained on how to use the LINUX Operating system. This OS will be used throughout this internship because it allows the user to remotely access another computer. I also received training on the VI editor, C, C++, FORTRAN, Python and Bash shell, Makefile, Open MP, and Parallel programming. In terms of preprocessing and post processing tools I was trained to use Cubit, Visit and Paraview.

Project

Overview

The electrical stimulus of the heart begins in the Sino atrial node. The Sino atrial node spontaneously produces electrical current without being simulated. The gates of the cells are bias and allow certain ions that are allowed to flow in or out. The difference in ion concentration produces a potential difference. The potential difference developed in the surrounding cells around the Sino atrial node causes the electric current to be propagated. This current is also sent to the Atrioventricular node. The Atrioventricular node is the connection between the atrial and ventricular cells. The electrical current is then passed on to the ventricular cells as a result of the potential difference developed in them. The resultant of this current causes the beating of the heart. There are two main models that describe the electrical processes at the tissue level: the Bi-domain and the Monodomain model.

Bidomain Model

The Bidomain model is based on the tissue of the heart and consists of two components: intracellular and extracellular.

$$\nabla \cdot (M_i \nabla v) + \nabla \cdot (M_i \nabla u_e) = \frac{\partial v}{\partial t} + I_{ion}(v, s), \qquad (1)$$

Equation 1 is the equation of the Bidomain model.

The Bidomain model is difficult to solve and computationally demanding. As a result, assumptions are made which simplifies the model into the Monodomain model.

Monodomain Model

The Monodomain model will be used during this project because it is easier it solve and computationally less demanding. Using the operator splitting scheme, the Monodomain model can be split into three equations.

$$\frac{\partial v}{\partial t} = -I_{ion}(v, s), \qquad v(t_n) = v^n \ (2)$$

$$\frac{\partial s}{\partial t} = f(v, s), \qquad s(t_n) = s^n \ (3)$$

$$\frac{\partial v}{\partial t} = \frac{\Lambda}{1 + \Lambda} \nabla \cdot (M_i \nabla v), \quad v(t_n) = v_\theta^n \ (4)$$

Equations 2 and 3 are solved simultaneously at different time steps and equation 4 is solved linearly at different time steps. The solutions to these equations describes the electrical processes at the tissue level

Beeler-Reuter Model

The equations to model the electrical currents have been developed and include the Hodgkin- Huxley, the Beeler-Reuter and the Luo Rudy model amongst others.

The Beeler-Reuter Model describes the electrical stimulus in the cells (the action potential). The model is based on experimental data from a guinea pig. It is based on the concentration level of Potassium, Calcium, and Sodium in the cells of the heart and their surroundings. This difference in concentration level in the cells and the surroundings causes an electrical current to develop.

The Beeler-Reuter model is still used in spite of being the first model depicting the transmembrane potential of the ventricular cells because it is easier to compute than some of the newer models. The Taylor Series method is used to compute the new values of the potential across the membrane as well as the new values of Calcium in the cell. The equations representing the gates are ordinary differential equations which can be solved to obtain the general solution and hence the values at the next time step.

Programming the Beeler-Reuter Model

The ordinary differential equations associated with the Beeler-Reuter Model can be solved easily. The Beeler-Reuter model is solved using the explicit time marching scheme called the Finite Element Euler Method. By using this scheme the only information needed are the initial conditions and the equations.

The time range for the Beeler- Reuter model is 0-400ms which was determined through experimentation. The Taylor series for the potential for the model is

$$V^{n+1} = V^n + \frac{dV}{dt}\,\Delta t + O(\Delta t)^2 \qquad (1)$$

where $O(\Delta t)^2$ is the error

The smaller (Δt) is the smaller the error from the computations will be. (Δt) is the time step.

$$\frac{dV}{dt} = -\frac{(i_{k1} + i_{x1} + iNa + iCa - i_{external})}{Cm} \qquad (2)$$

i_{k1} -Potassium current

i_{x1}- Another Potassium current

iNa- Sodium current

iCa- Calcium current

$i_{external}$- External stimulus that is generated by the Sino atrial node and passed to the Atrioventricular node

$\frac{dV}{dt}$- Change in voltage with time

The equations representing the currents are:

$$i_{k1} = \left\{\frac{4\{\exp[0.04(V+85)] - 1\}}{\exp[0.08(V+53)] + \exp[0.04(V+53)]} + \frac{0.2(V+23)}{1 - \exp[-0.04(V+23)]}\right\} \qquad (3)$$

$$i_{x1} = x_1 \cdot 0.8 \cdot \frac{\{\exp[0.04(V+77)] - 1\}}{\exp[0.04(V+35)]} \qquad (4)$$

$$\frac{dx_1}{dt} = \alpha_{x1}(1 - x_1) - \beta_{x1}x_1 \quad (5)$$

$$iNa = (gNa \cdot m^3 \cdot h \cdot j + gNaCa)(V - ENa) \quad (6)$$

$$\frac{dm_1}{dt} = \alpha_{m1}(1 - m_1) - \beta_{m1}m_1 \quad (7)$$

$$\frac{dh_1}{dt} = \alpha_{h1}(1 - h_1) - \beta_{h1}h_1 \quad (8)$$

$$\frac{dj_1}{dt} = \alpha_{j1}(1 - j_1) - \beta_{j1}j_1 \quad (9)$$

$$iCa = gCa \cdot d \cdot f \cdot (V - ECa) \quad (10)$$

$$\frac{dd_1}{dt} = \alpha_{d1}(1 - d_1) - \beta_{d1}d_1 \quad (11)$$

$$\frac{df_1}{dt} = \alpha_{f1}(1 - f_1) - \beta_{f1}f_1 \quad (12)$$

$$ECa = -82.3 - 13.0287 \ln[Ca^{2+}] \quad (13)$$

$$Ca^{n+1} = Ca^n + \frac{dCa}{dt} \cdot \Delta h \quad (14)$$

$$\frac{dCa}{dt} = -10^{-7} \cdot iCa + 0.07 \cdot (10^{-7} - [Ca]) \quad (15)$$

$$\alpha \; or \; \beta = \frac{C1\exp\{[C2 \cdot (V + C3)] + C4 \cdot (V + C5)}{\exp[C6 \cdot (V + C3)] + C7} \quad (16)$$

C1-C7 are constants

$\Delta h = \frac{\Delta t}{10}$

V- Voltage of the previous time

$gNa = 4$

$gNaCa = 0.003$

$ENa = 50$

$gCa = 0.09$

The ordinary differential equations 5, 7, 8, 9, 11, and 12 are solved for their general solution. The general solution of all the ODE's is

$$g(t) = Ce^{\Delta t(-\alpha-\beta)} + \frac{\alpha}{\alpha+\beta}$$

g is used to represent x, j, h, m, f, and d.

The value of the next time step is solved using the current value as the initial condition therefore the time is always equal to Δt.

Hence

$$C = g(0) + \frac{\alpha}{\alpha+\beta}$$

$g(0)$-Initial value or previous value

$$g(t) = \left(g(0) + \frac{\alpha}{\alpha+\beta}\right) \cdot e^{\Delta t(-\alpha-\beta)} + \frac{\alpha}{\alpha+\beta}$$

After solving for x, j, h, m, f, d the equations below can be solved for

$$i_{x1} = x_1 \cdot 0.8 \cdot \frac{\{\exp[0.04(V+77)] - 1\}}{\exp[0.04(V+35)]} \quad (4)$$

$$iNa = (gNa \cdot m^3 \cdot h \cdot j + gNaCa)(V - ENa) \quad (6)$$

Solving for iCa involves solving for the concentration of Calcium in the next time step. ECa is dependent on the Calcium of the next time step.

Hence using the Taylor series expansion the next value for Calcium can be solved for using the equation

$$Ca^{n+1} = Ca^n + \frac{dCa}{dt} \cdot \Delta h$$

$\frac{dCa}{dt}$ is a function of Calcium. The calcium value used is the calcium of the current time step.

$$\frac{dCa}{dt} = -10^{-7} \cdot gCa \cdot d \cdot f \cdot (V + 82.3 + 13.0287 \ln[Ca^{2+}]) + 0.07 \cdot (10^{-7} - [Ca])$$

Using the current time step value of Calcium and the calculated f and d which is the value for the next time step the value of $\frac{dCa}{dt}$ can be calculated for.

After $\frac{dCa}{dt}$ has been calculated for, the value of Calcium in the next time step is calculated for and then the value of ECa and iCa can be calculated.

After obtaining all the values $\frac{dV}{dt}$ can be calculated for using the equation below

$$\frac{dV}{dt} = -\frac{(i_{k1} + i_{x1} + iNa + iCa - i_{external})}{Cm}$$

$i_{external}$ is the initial stimulus current of $30\mu A/cm^2$ which is from the atrioventricular node. Finally the potential voltage across the membrane can be calculated for using the equation below

$$V^{n+1} = V^n + \frac{dV}{dt}\,\Delta t + O(\Delta t)^2$$

In Matlab these equation are calculated for at each time step using the For Loop. The number of iterations is dependent on Δt and the results are plotted using the plot function.

The Matlab code, the results of the plot and the constants C are present in the Appendix.

Meshing

There were three model used during this project, the heart model, the thin membrane model and the cubic model. A step-by-step procedure was used to develop the heart model. Geometry and mesh information can be found in the table below

Model	Dimensions(cm)	Node interval (cm)	Mesh Type	No of Nodes	No of elements
Thin membrane	10x10x1	0.281	Hexahedral	7220	5476
Cubic	10x10x10	0.281	Hexahedral	5202	4352
Heart	Shown on model	6.51	Tetrahedral	1047	4356

The models were created using the CUBIT software. CUBIT is a full-featured software toolkit for robust generation of two- and three-dimensional finite element meshes (grids) and geometry preparation (Sandia).

A. Thin membrane model

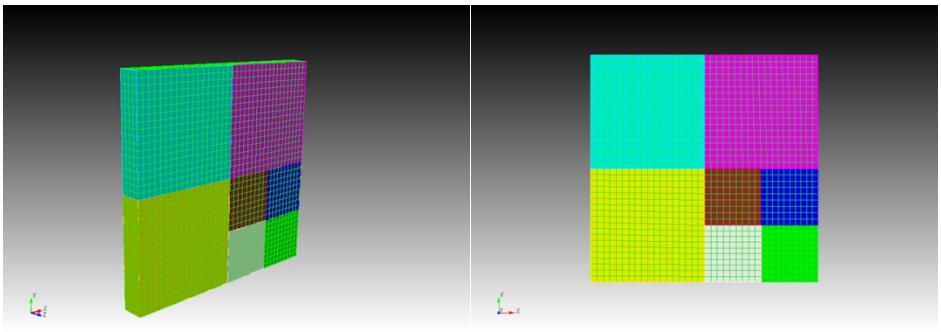


Figure 1- Thin membrane model meshed with hexahedral mesh

B. Cubic Model

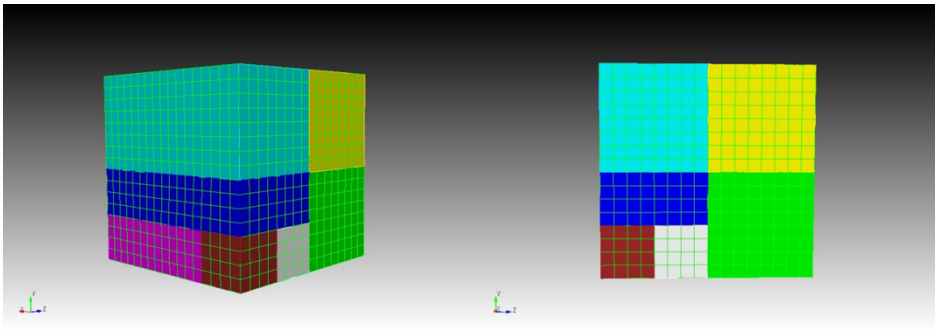


Figure 2- Cubic membrane model meshed with hexahedral mesh

C. Heart Model

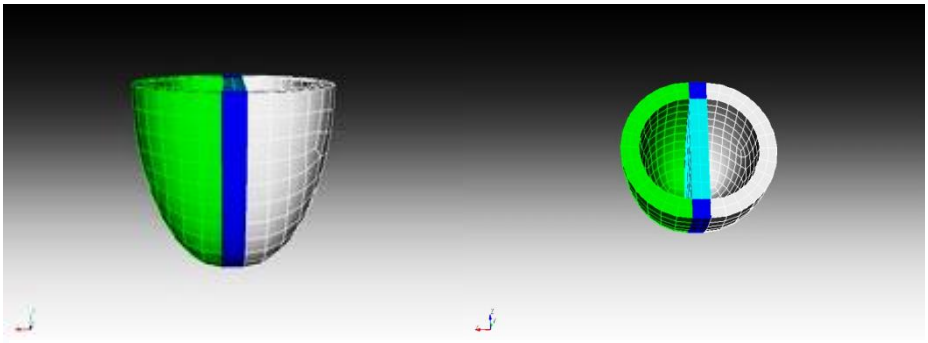


Figure 3- Model of heart with Hex mesh

Figur3 consists of three parts created separately and merged together to form a very simple model of the heart. The mesh type used on this figure is the hexahedral mesh.

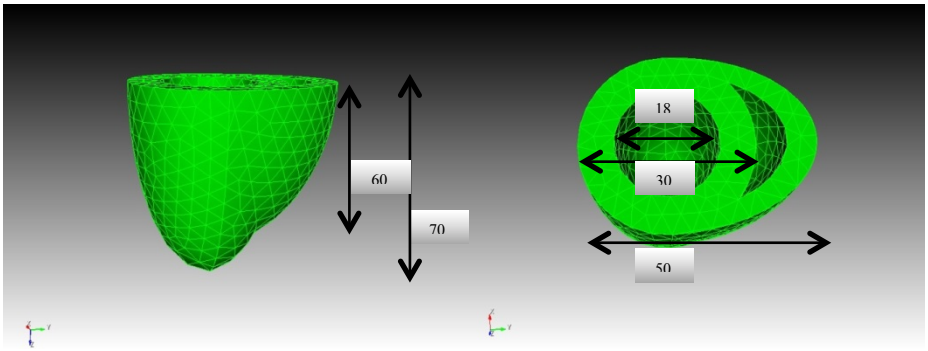


Figure 4- Model 2 with Tet mesh

Figure4 is a more complex model of the heart made up of two spheres that were constructed according to dimensions and merged to form the model above. The mesh type used is the Tet Mesh.

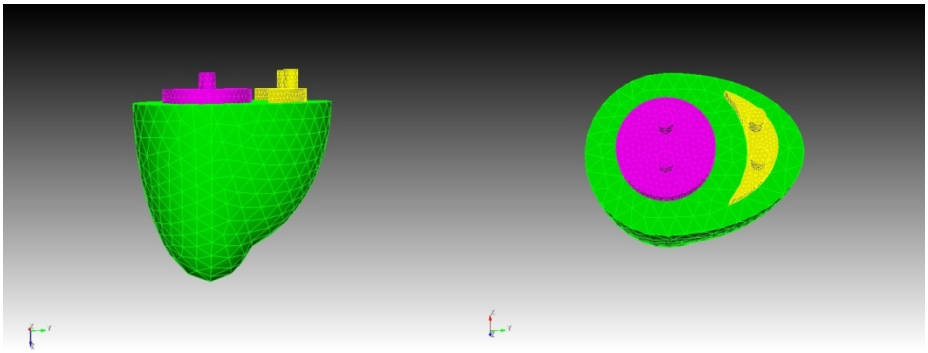


Figure 5- Model 3 with Tet mesh

Figure5 is an adaptation of **Figure4**. It was made by adding surfaces to cover the holes and extending those surfaces to form end caps. After the formation of the end caps two cylinders were added to portray inlets and outlets.

Figure 4 was the heart model used to run the cases.

Program

The program used during this project was coded by Andrew Kail, a graduate student at the University of Tennessee Knoxville and one of my mentors. The program is utilized to run cases meshed with tetrahedral, hexahedral, triangle or quadrilateral meshes. The basic structure of the program is displayed in the image below.

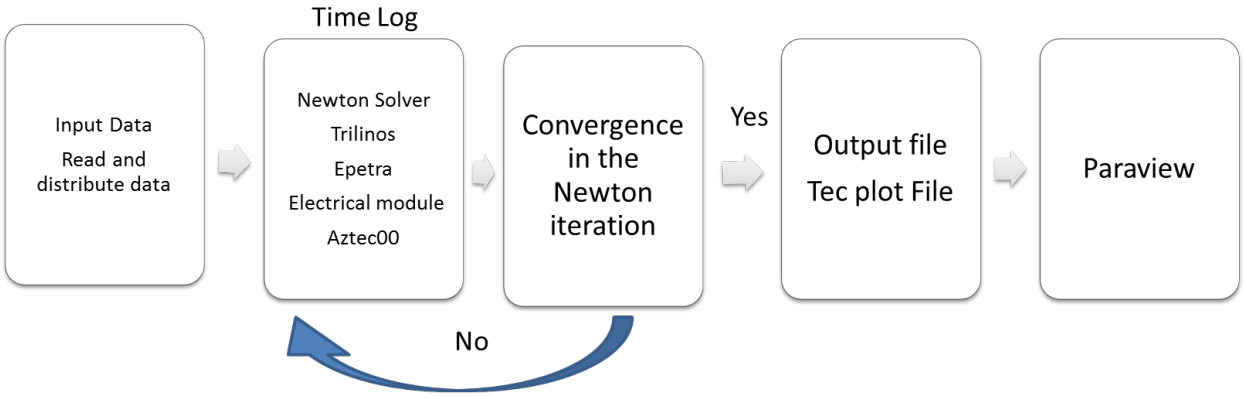


Figure 6- Flow Chart demonstrating how the program runs

Data is input as a Patran file into the program. The program also allows certain parameters to be manually input into the program. Some of these parameters are runtime, time step and diffusion.. The program runs the first time log or time step. In each time log the data is partitioned, distributed and calculated. The Forward Euler numerical analysis method is used so the program runs the calculation at each time step until convergence is reached. The output is saved as a Tec plot file and uploaded into Paraview for visualization.

Case 1: Single Beat

All three models were run for one beat and visualized in Paraview. The run time for all models in this case was 400ms and was chosen because the plot of the Beeler-Reuter model shows that voltage returns to its initial conditions before that time.

A. Thin Membrane Model
   - Time step: 1ms
   - Diffusion: 0.01
   - Boundary condition applied to $1/16^{th}$ of nodes.

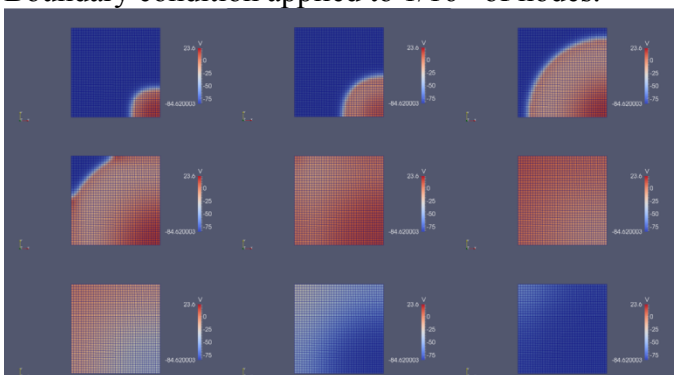


Figure 7- Simulation of thin membrane for single beat case

From the figure we deduced that the wave front and wave back propagated as expected.

B. Cubic Model
   - Time step: 1ms
   - Diffusion: 0.1
   - Boundary condition applied to $1/32^{th}$ of nodes

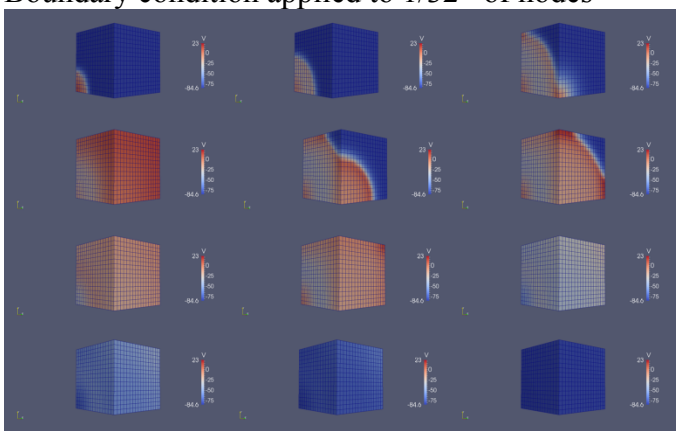


Figure 8- Simulation of cubic membrane for single beat case

The cubic model also displayed both the wave front and the wave back of the wave although the wave back was not as pronounce as the wave front. A range of diffusion values where tested and it was determined that the range of diffusion values that worked for this model was between 0.07 and 0.13 with 0.07 producing the best results.

C. Heart model
   - Time step: 1ms
   - Diffusion: 10
   - Boundary condition applied to edge of large opening

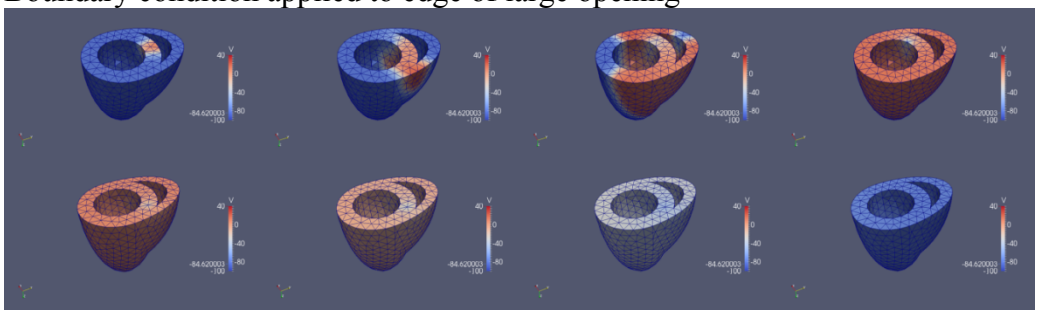


Figure 9- Simulation of heart membrane for single beat case

The results of the heart simulation displayed a discrepancy in the wave back. The wave front propagated as expected with the voltage increasing sequentially with time. However, the wave back seemed to diffuse across the whole model instead of resembling the wave front propagation.

Case2: Running Multiple Beats

Both the thin membrane and cubic model worked very well for the single beat. The next step was to attempt multiple beats. Multiple beats on the thin membrane was successful. We could see both the wave front and the wave back in both beats with the second beat depicting a better wave propagation that the first beat.

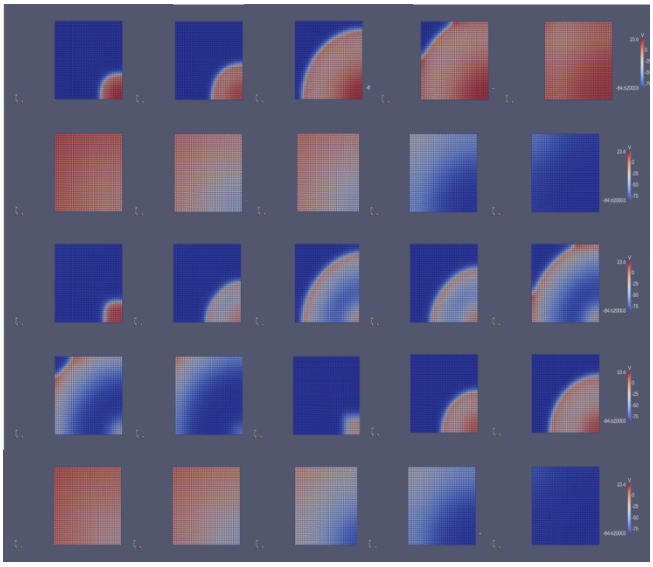


Figure 10- Simulation of thin membrane for 2 beats

The cubic model on the other hand did not produce the same results. Propagation of the second beat was seen only in the region where the boundary conditions where applied.

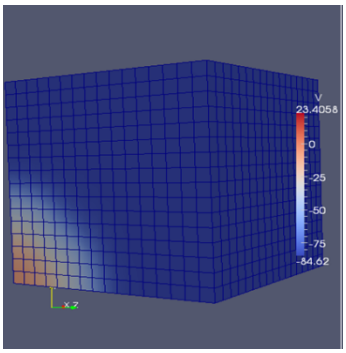


Figure11 – Incomplete propagation of second beat in cubic model

The same case was run on the heart model . The only evidence of a multiple beat was a slight simulation lasting for about a second. With the failure of both the cubic and heart model we proceeded to run test to discover what the issue with the code was.

Testing the Diffusion Code

The diffusion code was run on the thin membrane model. The results are shown below

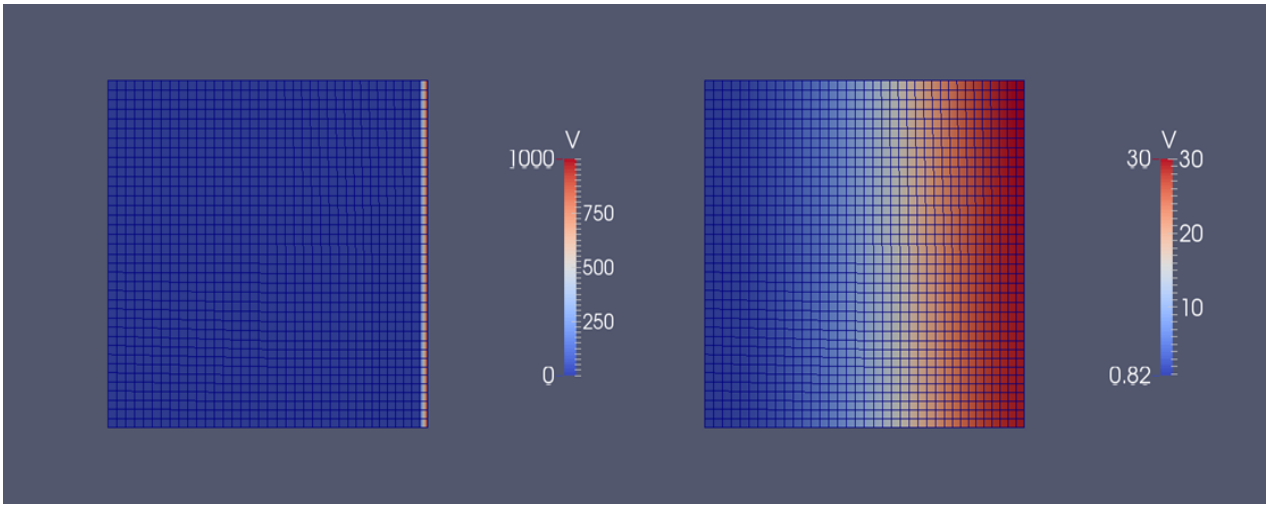


Figure12 – Simulation of diffusion model only

The diffusion reaction equations worked perfectly.

Testing the ODE Model

The ODE model was tested to determine if it was giving the right results. The Ode model was used to run two and ten beats and the plots from both of these beats are shown below. This is a plot of a node in the simulated region.

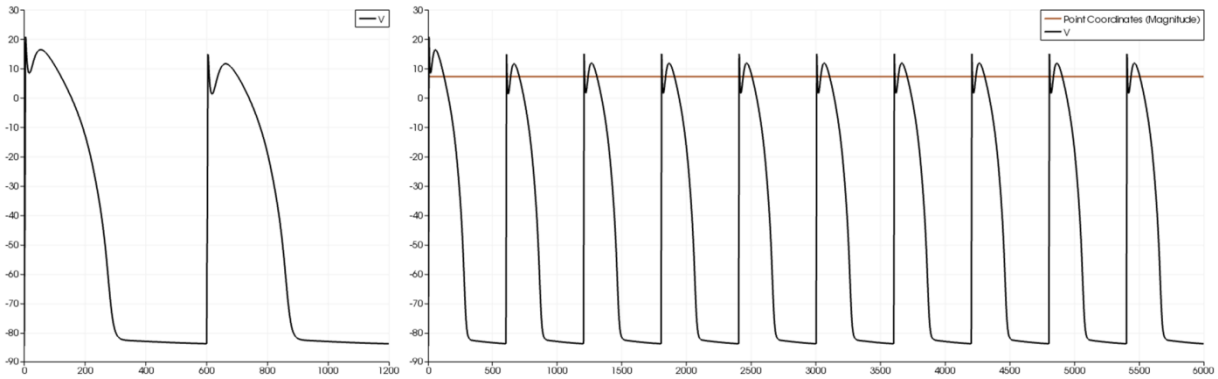


Figure13 – Plot of 2 beats and 10 beats using ODE model only

A plot of the nodes in the simulated region, the mid region and at the other end of the simulated region of the cubic model showed the results below.

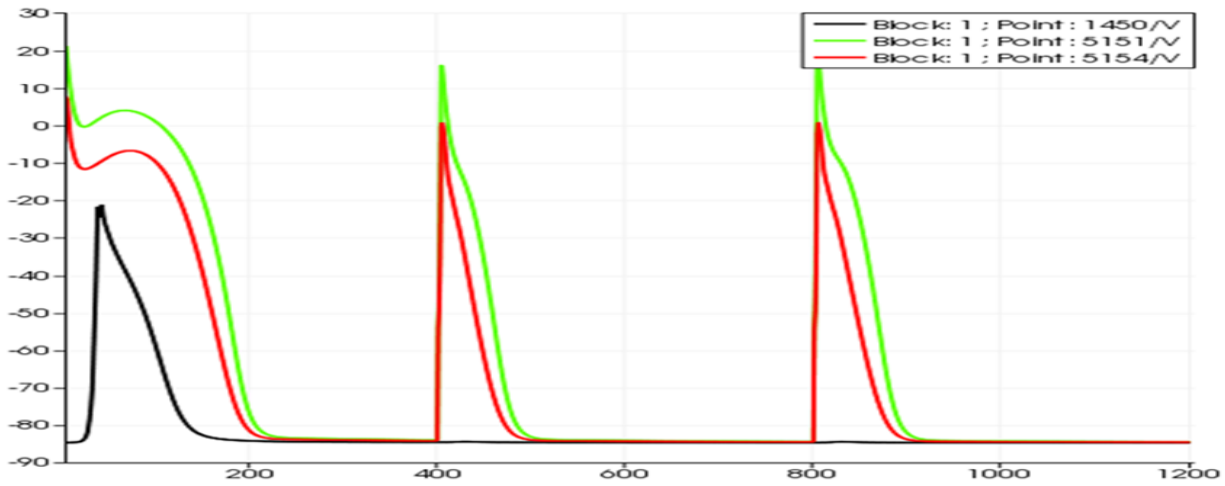


Figure 14 – Plot of 3 nodes in cubic model

The plot above revealed that the problem lay between the coupling of the PDE model of the first beat and the ODE model of the second beat. To test this conclusion, the ODE model was reinitialized during the second beat so that it had all the initial conditions of the first beat except the voltage which was the calculated result from the first beat. The cubic model was run with these changes and visualized in Paraview. The second beat propagated as expected.

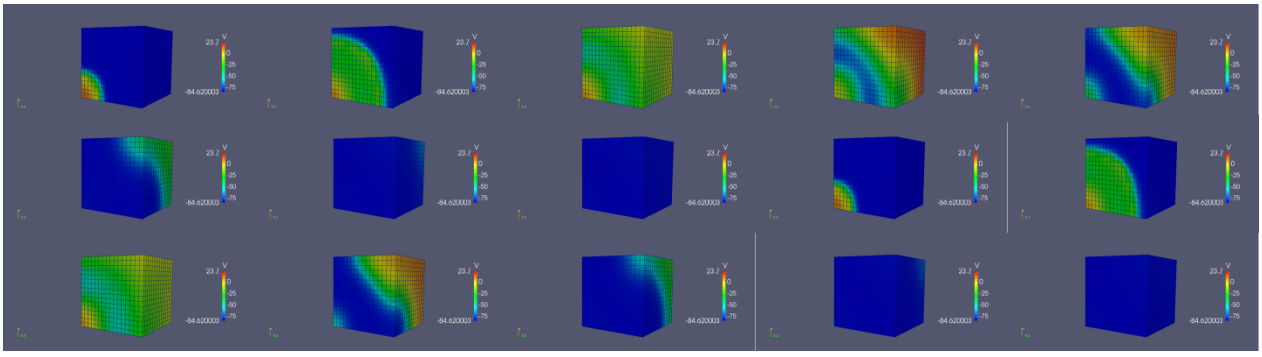


Figure 15 – Simulation of 2 beats with cubic model

The plot of the node in the stimulated region showed the results below

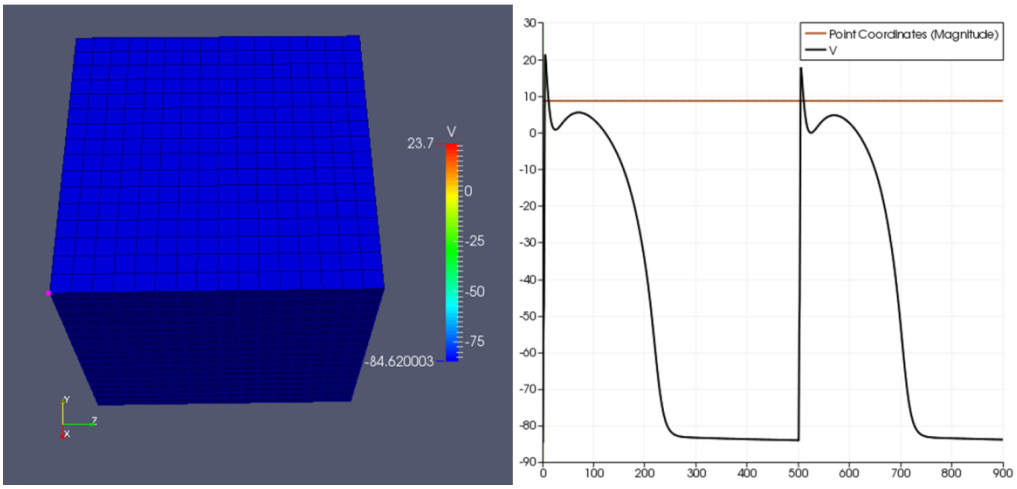


Figure 16 – Plot of node in simulated region

The results were similar to that obtained from the ODE model and the Beeler-Reuter model. The second beat had a lower peak that the first beat but this was because the calculated voltage potential, after the first beat, for the second beat was different from the initial voltage value. From the results it was determined that the ODE model was very sensitive to the vector potential returning from the first beat.

Reducing the Time step of the ODE Model

Since most numerical analysis methods depend on the Δt , this was decreased in the ODE model. One iteration in the PDE model consisted of ten iterations in the ODE model. After this change the model performed as expected. The results on the cubic and heart model are shown below.

A. Cubic Model

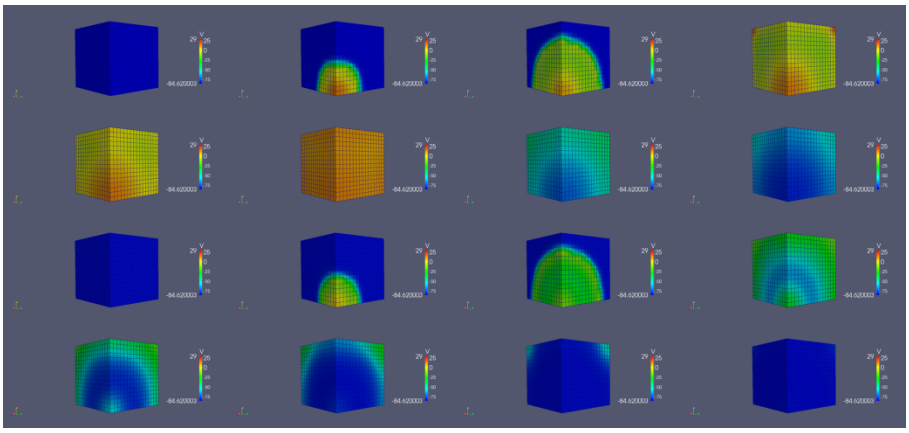


Figure 17– Simulation of 2 beats with cubic model without reinitializing

B. Heart Model

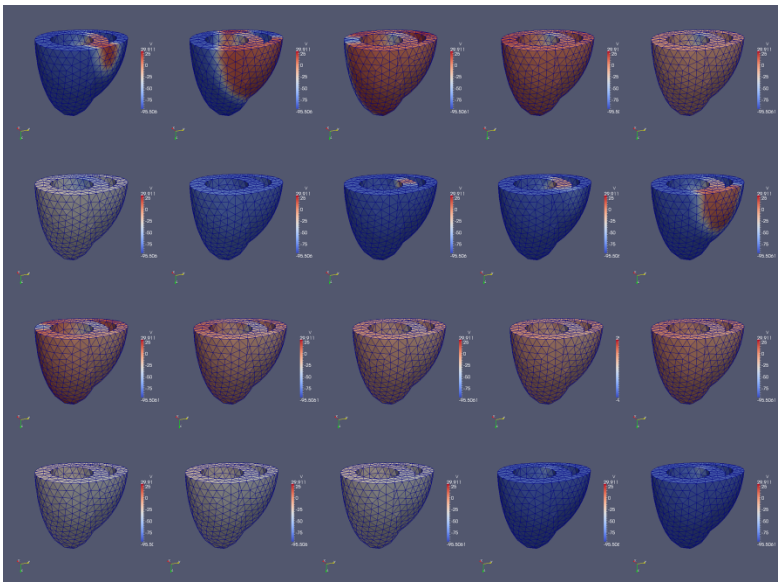


Figure 18– Simulation of 2 beats with heart model without reinitializing

Appendix

Matlab Code

```
close all

tic;
%[]Starts program timer.

clc;
%[]Clears command window.

clear;
%[]Clears variable workspace.

format long g;
%[]adjusts the format of the command window output.

tmax=400;
Vinit=-84.62;
dt=0.02;
% Parameters

tx=0:dt:400;
nt=tmax/dt+1;
V1(1:1:nt,1)=0;
V1(1)=Vinit;
Ca(1:1:nt,1)=0;
Ca(1)=10^-7;
f(1:1:nt,1)=0;
f(1)=0.983;
d(1:1:nt,1)=0;
d(1)=0.0001;
m(1:1:nt,1)=0;
m(1)=0.0011;
h(1:1:nt,1)=0;
h(1)=0.99869;
j(1:1:nt,1)=0;
j(1)=0.99887;
x(1:1:nt,1)=0;
x(1)=0.0074;
ix1(1:1:nt,1)=0;
ix1(1)=0;
ik1(1:1:nt,1)=0;
ik1(1)=0;
iNa(1:1:nt,1)=0;
iNa(1)=0;
iCa(1:1:nt,1)=0;
iCa(1)=0;
gNa=4;
gNaCa=0.003;
ENa=50;
gCa=0.09;
ECa(1:1:nt,1)=0;
dVdt(1:1:nt,1)=0;
Cm=1;
stmts=10; %[ms]
stmte=11; %[ms]
% [] Initial Values
```

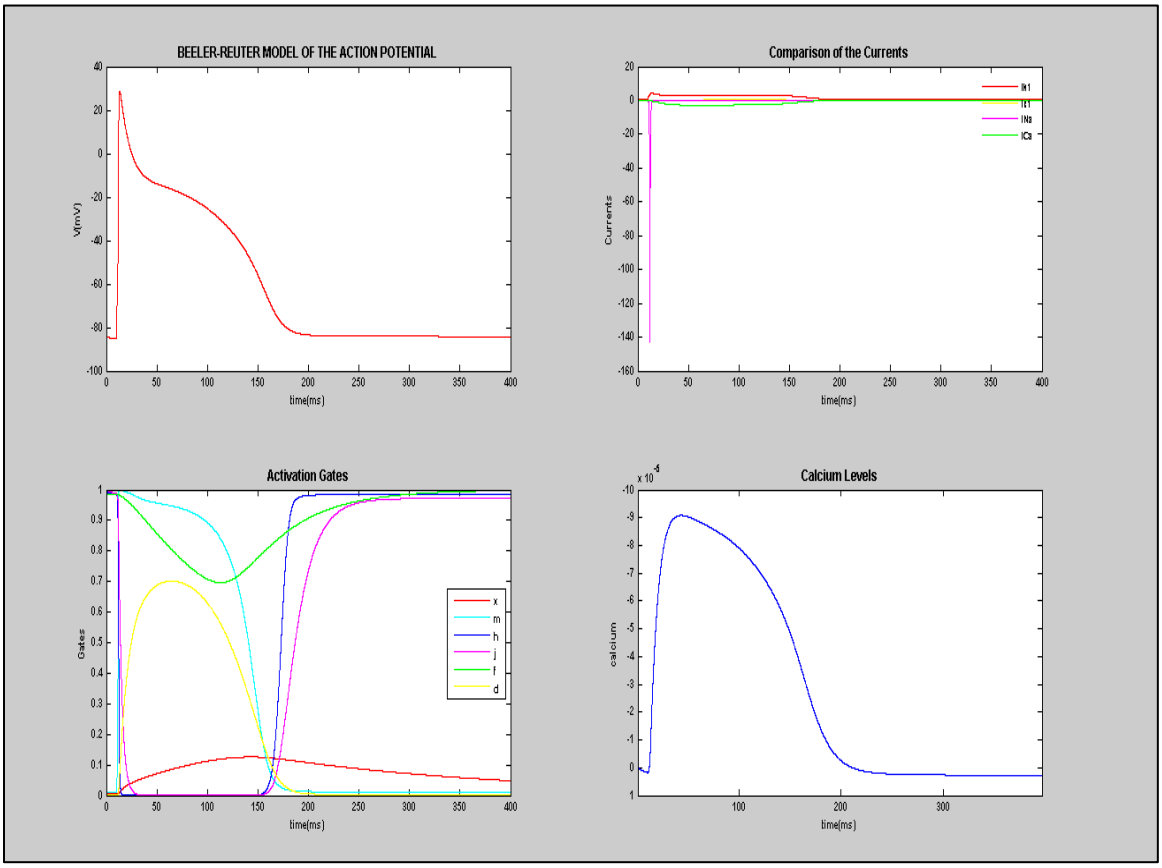
```
for n=1:nt-1

    if (tx(n)>= stmts && tx(n)<=stmte )
        ie= 30;
    else
        ie=0;
    end
    %[]Stimulation

    V=V1(n); % Initial Voltage

    % Potassium Current
    i1=4*(exp(0.04*(V+85))-1)/(exp(0.08*(V+53))+exp(0.04*(V+53)));
    i2=0.2*(V+23)/(1-exp(-0.04*(V+23)));
    ik1(n+1)=0.35*(i1+i2);

    %X Current
    %x

    c1=0.0005;     %[ms^-1]
    c2=0.083;     %[ms^-1]
    c3=50;     %[ms^-1]
    c4=0;     %[ms^-1]
    c5=0;     %[ms^-1]
    c6=0.057;     %[ms^-1]
    c7=1;     %[ms^-1]
    C1=0.0013;     %[ms^-1]
    C2=-0.06;     %[ms^-1]
    C3=20;     %[ms^-1]
    C4=0;     %[ms^-1]
    C5=0;     %[ms^-1]
    C6=-0.04;     %[ms^-1]
    C7=1;     %[ms^-1]

    a=(c1*exp(c2*(V+c3))+c4*(V+c5))/(exp(c6*(V+c3))+c7);
    b=(C1*exp(C2*(V+C3))+C4*(V+C5))/(exp(C6*(V+C3))+C7);
    x0=x(n)-(a/(a+ b));
    x(n+1)= x0* exp(-(a+b)*dt) + a/(a+b);
    ix1(n+1)=x(n+1)* 0.8*((exp(0.04*(V+77))-1)/exp(0.04*(V+35)));
```

```
% Sodium Current
    %m
    cm1=0;     %[ms^-1]
    cm2=0;     %[ms^-1]
    cm3=47;     %[ms^-1]
    cm4=-1;     %[ms^-1]
    cm5=47;     %[ms^-1]
    cm6=-0.1;     %[ms^-1]
    cm7=-1;     %[ms^-1]
    Cm1=40;     %[ms^-1]
    Cm2=-0.056;     %[ms^-1]
    Cm3=72;     %[ms^-1]
    Cm4=0;     %[ms^-1]
    Cm5=0;     %[ms^-1]
    Cm6=0;     %[ms^-1]
    Cm7=0;     %[ms^-1]

    am=(cm1*exp(cm2*(V+cm3))+cm4*(V+cm5))/(exp(cm6*(V+cm3))+cm7);
    bm=(Cm1*exp(Cm2*(V+Cm3))+Cm4*(V+Cm5))/(exp(Cm6*(V+Cm3))+Cm7);

    m0=m(n)-(am/(am+ bm));
    m(n+1)=m0* exp(-(am+bm)*dt) + am/(am+bm);

    ch1=0.126;     %[ms^-1]
    ch2=-0.25;     %[ms^-1]
    ch3=77;     %[ms^-1]
    ch4=0;     %[ms^-1]
    ch5=0;     %[ms^-1]
    ch6=0;     %[ms^-1]
    ch7=0;     %[ms^-1]
    Ch1=1.7;     %[ms^-1]
    Ch2=0;     %[ms^-1]
    Ch3=22.5;     %[ms^-1]
    Ch4=0;     %[ms^-1]
    Ch5=0;     %[ms^-1]
    Ch6=-0.082;     %[ms^-1]
    Ch7=1;     %[ms^-1]

    ah=(ch1*exp(ch2*(V+ch3))+ch4*(V+ch5))/(exp(ch6*(V+ch3))+ch7);
    bh=(Ch1*exp(Ch2*(V+Ch3))+Ch4*(V+Ch5))/(exp(Ch6*(V+Ch3))+Ch7);
    h0=h(n)-(ah/(ah+ bh));
    h(n+1)= h0* exp(-(ah+bh)*dt) + ah/(ah+bh);

    %j
    cj1=0.055;     %[ms^-1]
    cj2=-0.25;     %[ms^-1]
    cj3=78;     %[ms^-1]
    cj4=0;     %[ms^-1]
    cj5=0;     %[ms^-1]
    cj6=-0.2;     %[ms^-1]
    cj7=1;     %[ms^-1]
    Cj1=0.3;     %[ms^-1]
    Cj2=0;     %[ms^-1]
    Cj3=32;     %[ms^-1]
    Cj4=0;     %[ms^-1]
    Cj5=0;     %[ms^-1]
    Cj6=-0.1;     %[ms^-1]
    Cj7=1;     %[ms^-1]

    aj=(cj1*exp(cj2*(V+cj3))+cj4*(V+cj5))/(exp(cj6*(V+cj3))+cj7);
    bj=(Cj1*exp(Cj2*(V+Cj3))+Cj4*(V+Cj5))/(exp(Cj6*(V+Cj3))+Cj7);
    j0=j(n)-(aj/(aj+ bj));
    j(n+1)=j0* exp(-(aj+bj)*dt) + aj/(aj+bj);

    iNa(n+1)=(gNa*m(n+1)^3*h(n+1)*j(n+1)+gNaCa)*(V-ENa);
```

```
%Calcium Current
    %f
    cf1=0.012;    %[ms^-1]
    cf2=-0.008;    %[ms^-1]
    cf3=28;    %[ms^-1]
    cf4=0;    %[ms^-1]
    cf5=0;    %[ms^-1]
    cf6=0.15;    %[ms^-1]
    cf7=1;    %[ms^-1]d


    Cf1=0.0065;    %[ms^-1]
    Cf2=-0.02;    %[ms^-1]
    Cf3=30;    %[ms^-1]
    Cf4=0;    %[ms^-1]
    Cf5=0;    %[ms^-1]
    Cf6=-0.2;    %[ms^-1]
    Cf7=1;    %[ms^-1]


    af=(cf1*exp(cf2*(V+cf3))+cf4*(V+cf5))/(exp(cf6*(V+cf3))+cf7);
    bf=(Cf1*exp(Cf2*(V+Cf3))+Cf4*(V+Cf5))/(exp(Cf6*(V+Cf3))+Cf7);
    f0=f(n)-(af/(af+ bf));
    f(n+1)=  f0* exp(-(af+bf)*dt) + af/(af+bf);
    %d
    cd1=0.095;    %[ms^-1]
    cd2=-0.01;    %[ms^-1]
    cd3=-5;    %[ms^-1]
    cd4=0;    %[ms^-1]
    cd5=0;    %[ms^-1]
    cd6=-0.072;    %[ms^-1]
    cd7=1;    %[ms^-1]


    Cd1=0.07;    %[ms^-1]
    Cd2=-0.017;    %[ms^-1]
    Cd3=44;    %[ms^-1]
    Cd4=0;    %[ms^-1]
    Cd5=0;    %[ms^-1]
    Cd6=0.05;    %[ms^-1]
    Cd7=1;    %[ms^-1]


    ad=(cd1*exp(cd2*(V+cd3))+cd4*(V+cd5))/(exp(cd6*(V+cd3))+cd7);
    bd=(Cd1*exp(Cd2*(V+Cd3))+Cd4*(V+Cd5))/(exp(Cd6*(V+Cd3))+Cd7);


    d0=d(n)-(ad/(ad+ bd));
    d(n+1)=d0*exp(-(ad+bd)*dt)+ad/(ad+bd);


    %Ca
    dh=dt/10;
    p=-10^-7;
    c=0.07*(10^(-7)-Ca(n));
    k= (V+82.3+13.0287*log(Ca(n))*f(n+1)*gCa*d(n+1))*p;
    dCadt=k+c;


    for i=1:10
        Ca1(1)=Ca(n);
        Ca1(i+1)= Ca1(i)+ (dCadt*dh);
    end


    Ca(n+1)=Ca1(11);
    ECa(n+1)=-82.3-13.0287*log(Ca(n+1));


    iCa(n+1)= gCa*d(n+1)*f(n+1)*(V-ECa(n+1));
```

```
% Transmembrane Potential

    dVdt(n)=-(ik1(n+1)+ix1(n+1)+iNa(n+1)+iCa(n+1)-ie)/Cm;

    V1(n+1)= V1(n) + dVdt(n)*dt;
```

Graphs plotted with Matlab Code



References

1. "Cplusplus.com - The C Resources Network." *Cplusplus.com - The C Resources Network*. Cplusplus.com, n.d. Web. 10 July 2013.
2. Sundnes, Joakim, Glenn T. Lines, Xing Cai, Bjorn F. Nielsen, Kent-Andre Mardal, and Aslak Tveito. *Computing the Electrical Activity in the Heart*. Berlin: Springer, 2006. Print. Ser. 1.
3. https://cubit.sandia.gov/