Simulating the Electrical Activity in the Heart

Lecture at the Winter School in Computational Mathematics

Glenn Terje Lines

Department of Informatics, University of Oslo, Norway

Mars 2001

Overview

- Motivation
- Some physiology
- The mathematical model
- Numerical aspects and Implementation
- Simulations and CPU usage

Motivation

- The contraction of the heart is preceded by an electro-chemical reaction
- ECG measurements records the body surface potential generated by this reaction
- The ECG changes characteristically under pathological conditions.
- By performing simulation a better understanding of the relationship between the signal and anomalies can be obtained

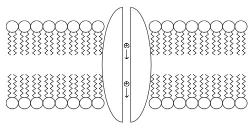
Challenges

- Complex mathematical model
  - PDEs coupled to ODEs with many state variables
- Stiff problem,
  - high spatial and temporal resolution needed
- Complicated geometry
  - Irregular surface
  - Anisotropic tissue
  - Inhomogeneities

4

The Cell Membrane

- Consist of a bilipid layer
- Embedded proteins for transport control
- Selectively permeable
- Maintains concentration gradients
- Has a transmembrane potential



5

Flow through the membrane

Two types of ionic current:

Passive: Diffusion along the concentration gradient

- Through the membrane (H_2O, O_2, CO_2)
- Through specialized channels (Na^+, K^+, Cl^-)
- Carrier mediated transport

Active: Energy driven flow against the gradients

- ATP driven pumps ($Na^+ - K^+$, Ca^{2+})
- Exchangers driven by concentration gradients ($Na^+ - Ca^{2+}$)

6

Accumulation around the membrane

The membrane has properties similar to a capacitor

- Consists of two conducting medias (intra- and extra cellular space)
- These are separated by a insulating material (the membrane)

The potential over a capacitor is proportional to the separated charge (q):

$$v = q/c$$

The factor c is called the capacitance of the capacitor.

7

The cell membrane modeled as a leaky capacitor

As any real capacitor the membrane conducts some current. The flux of ions (I_{ion}) will cause a change in q and thus v.

Consider the change over a time interval Δt. It follows that $\frac{\Delta v}{\Delta t} = \frac{1}{C_m}\frac{\Delta q}{\Delta t}$ and in the limit we get:

$$\frac{dv}{dt} = \frac{1}{C_m}\frac{dq}{dt}$$

The term $\frac{dq}{dt}$ is called the capacitive current and is denoted I_c.

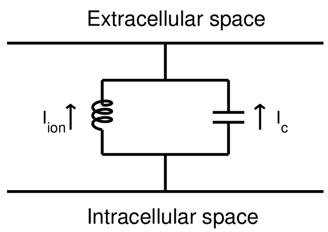
8

Electrical circuit model of the cell membrane

The membrane behaves like resistor and capacitor in parallel:

$$I_t = I_{\text{ion}} + I_c$$

If the loop is closed then $I_t = 0$. In that case all the ions passing the membrane accumulate and change the membrane potential accordingly.



9

Model for (I_{ion})

The appropriate model for the ionic current will depend upon several factors:

- Species
- Location within the heart
- Normal or pathological conditions

The passive part of the current must under any circumstances respect the Nernst Equilibrium Potential. It follows from Fick's Law and Plank's equation

10

Fick's Law

Flux of ions due to a concentration gradient

$$\text{J} = -D\nabla c$$

The diffusion coefficient D depends upon the solvent and the temperature of the embedding fluid:

$$D = \frac{kT}{f}$$

T is the temperature measured in Kelvin, f is a frictional constant and k is the Boltzmann's constant.

11

Plank's equation

Models the ion-flux caused by an electrical field:

$$J = -m\frac{z}{|z|}c\nabla u$$

with
m - mobility of the ions in the liquid
$z/|z|$ - sign of the charge of the ion
c - the concentration of the ion
∇u - the electrical field

Relationship between D and m:

$$m = D\frac{|z|F}{RT}$$

here R is the gas constant and and F is Faraday's constant.
Combined effect of concentration gradient and an E-field:

$$J = -D(\nabla c + \frac{zF}{RT}c\nabla u)$$

Nernst Equilibrium Potential

Consider equilibrium in 1D flow:

$$\frac{dc}{dx} + \frac{zF}{RT} c \frac{du}{dx} = 0$$
$$\frac{1}{c}\frac{dc}{dx} + \frac{zF}{RT}\frac{du}{dx} = 0$$

Integrating from inside (x=0) to outside (x=L) yields:

$$\ln(c)|_{c(0)}^{c(L)} = \frac{zF}{RT}(u(L) - u(0))$$

We define the transmembrane potential to be $v = u_i - u_e$ The value of the transmembrane potential at zero flux is then

$$v = \frac{RT}{zF} \ln(\frac{c_e}{c_i}) \tag{1}$$

Equation (1) is referred to as the Nernst Equilibrium Potential.

The Hodgkin-Huxley Model

Simple model for I_{ion}

$$\begin{aligned} I_{\text{ion}} = -I_c = C_m \tfrac{dv}{dt} &= g_{\text{Na}}(v - v_{\text{Na}}) + g_{\text{K}}(v - v_{\text{K}}) + g_{\text{L}}(v - v_{\text{L}}) \\ &= g_{\text{eff}}(v - v_{\text{eq}}) \end{aligned} \tag{2}$$

where

$$g_{\text{eff}} = g_{\text{Na}} + g_{\text{K}} + g_{\text{L}} \tag{3}$$

and

$$v_{\text{eq}} = \frac{g_{\text{Na}} v_{\text{Na}} + g_{\text{K}} v_{\text{K}} + g_{\text{L}} v_{\text{L}}}{g_{\text{eff}}} \tag{4}$$

v_{eq} is a weighted average of the individual equilibrium potentials. The weighing factors are time and voltage dependent.

The Luo-Rudy model

The state vector

$$s = (v, [\text{Na}^+]_\text{i}, [\text{K}^+]_\text{i}, [\text{Ca}^{2+}]_\text{i}, [\text{Ca}^{2+}]_{\text{JSR}}, [\text{Ca}^{2+}]_{\text{NSR}}, m, h, j, x, d, f)$$

Gating variables m, h, j, x, d, f governed by

$$\dot{g} = \alpha_g(v)(1 - g) - \beta_g(v) g \qquad g = m, h, j, x, d, f$$

Concentrations $[\text{Na}^+]_\text{i}$, $[\text{K}^+]_\text{i}$, $[\text{Ca}^{2+}]_\text{i}$ governed by

$$\frac{d[B]}{dt} = -I_B \cdot \frac{A_{Cap}}{V_C z_B F} \qquad B = [\text{Na}^+]_\text{i}, [\text{K}^+]_\text{i}, [\text{Ca}^{2+}]_\text{i}$$

Ionic current sum 14 sub-current, schematically:

$$I_{\text{ion}}(s) = I_{\text{Na,tot}}(s) + I_{\text{K,tot}}(s) + I_{\text{Ca,tot}}(s)$$

The Bidomain Model

The tissue is divided in two domains

- The intracelluar space
- The extracelluar space

The potentials are denoted u_i and u_e.

Volume averaged quantities.

The two fields are defined over the same domain, H, which is the heart muscle.

The Bidomain Model

Each domain is assumed to have Ohmian resistance, e.g:

$$J_i = M_i \nabla u_i$$

$$J_e = M_e \nabla u_e$$

where M_i and M_e are conductivity tensors.

The conductivity tensors

Fiber axis coordinate system:
a_l = along muscle fiber
a_t = orthogonal to a_l within sheet
a_n = sheet normal
All vectors of unit length so $A^{-1} = A^T$ if the columns of A are $A = (a_l, a_t, a_n)$
Conductivity tensor expressed in local coordinates:

$$M^* = \begin{bmatrix} \sigma_l & 0 & 0 \\ 0 & \sigma_t & 0 \\ 0 & 0 & \sigma_n \end{bmatrix}$$

Global coordinates

A potential gradient (p) must first be mapped to local coordinates before applying M^*. Finally, the resulting current vector (i) must be mapped back to global coordinates:

$$i = A(M^*(A^T p))$$

We define the global conductivity tensors by

$$M_i = AM_i^*A^T \text{ and } \quad M_e = AM_e^*A^T$$

Discrete information, continuous representation

The axis directions a^i are known at finite number of locations $x_1, x_2 \ldots x_n$. The direction at an arbitrary point x can be computed by a weighted average:

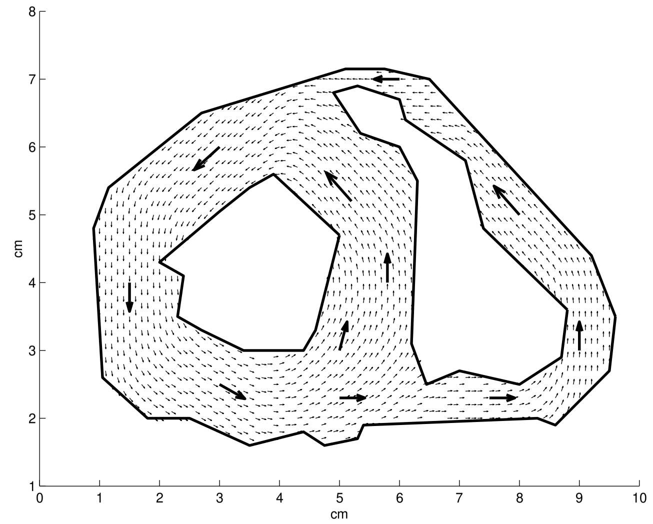
$$a(x) = \sum_{j=1}^{n} \gamma_j a^j$$

with

$$\gamma_j = \frac{e^{-\lambda ||x-x_j||^2}}{\sum_{j=1}^{n} e^{-\lambda ||x-x_j||^2}}$$

where λ is a parameter controlling the relative weight of locations with different distances. The computed vector $a(x)$ is normalized to have length one.

Example in 2D



The Bidomain Model, continued

The transmembrane current (per volume):

$$I_m = -\nabla \cdot J_i$$

By conservation of current we have:

$$\nabla \cdot J_i = -\nabla \cdot J_e$$

From the membrane model we have transmembrane current (per membrane area)

$$I_t = I_{\text{ion}} + I_c$$

So

$$\begin{aligned} I_m = \chi I_t &= \chi(I_{\text{ion}} + I_c) \\ &= \chi(I_{\text{ion}} + C_m \tfrac{dv}{dt}) \end{aligned}$$

where χ is the membrane surface area to volume ratio.

Substitutes for J_i and J_e and eliminates u_i:

$$\begin{aligned} \chi(I_{\text{ion}} + C\tfrac{dv}{dt}) &= \nabla \cdot (M_i \nabla u_i) \\ \nabla \cdot (M_i \nabla u_i) &= -\nabla \cdot (M_e \nabla u_e) \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \chi(I_{\text{ion}} + C\tfrac{dv}{dt}) = \nabla \cdot (M_i \nabla u_i) &+ (-\nabla \cdot (M_i \nabla u_e) + \nabla \cdot (M_i \nabla u_e)) \\ \nabla \cdot (M_i \nabla u_i) - \nabla \cdot (M_i \nabla u_e) &= -\nabla \cdot (M_e \nabla u_e) - \nabla \cdot (M_i \nabla u_e) \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} \chi(I_{\text{ion}} + C\tfrac{dv}{dt}) &= \nabla \cdot (M_i \nabla v) + \nabla \cdot (M_i \nabla u_e) \\ \nabla \cdot (M_i \nabla v) &= -\nabla \cdot ((M_i + M_e) \nabla u_e) \end{aligned}$$

The bidomain model

$$\begin{aligned} &\chi C_m \tfrac{\partial v}{\partial t} + \chi I_{\text{ion}}(v, s) = \nabla \cdot (M_i \nabla v) + \nabla \cdot (M_i \nabla u_e) && x \in \textbf{Heart}(H) \\ &\nabla \cdot ((M_i + M_e) \nabla u_e) = -\nabla \cdot (M_i \nabla v) && x \in \textbf{Heart}(H) \end{aligned}$$

where

v	**Transmembrane potential, $v = u_i - u_e$**
s	**State vector (concentrations and gates)**
χ	**Membrane surface area to volume ratio**
C_m	**Membrane capacitance**
I_{ion}	**Total ionic current**

Monodomain reduction

In the special case

$$\frac{\sigma_e^l}{\sigma_i^l} = \frac{\sigma_e^t}{\sigma_i^t} = \frac{\sigma_e^n}{\sigma_i^n} = r \qquad (5)$$

we get that $M_e = rM_i$ and the elliptical equation reduces to:

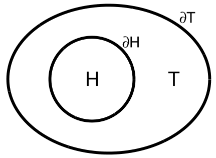
$$(1+r)\nabla \cdot (M_i \nabla u_e) = -\nabla \cdot (M_i \nabla v).$$

Using this equation can we eliminate u_e:

$$\begin{aligned} \chi C_m \frac{\partial v}{\partial t} + \chi I_{\text{ion}}(v,s) &= \nabla \cdot (M_i \nabla v) - \frac{1}{1+r} \nabla \cdot (M_i \nabla v) \\ &= \frac{r}{1+r} \nabla \cdot (M_i \nabla v) \end{aligned}$$

Measurements do not support the assumption in (5)

Extracardiac potential u_T



Conservation of current:

$$\begin{aligned} \nabla \cdot (M_T \nabla u_T) &= 0 \qquad x \in \text{Torso}(T) \\ (M_T \nabla u_T) \cdot n^T &= 0 \qquad x \in \partial T \end{aligned}$$

where M_T is the conductivity tensor and T the entire torso.

Interface conditions:

Transmembrane potential: $M_i \nabla v \cdot n^T = 0 \quad x \in \partial H$

Extracellular potential: $u_T = u_e \quad x \in \partial H$

$M_e \nabla u_e \cdot n^T = M_T \nabla u_T \cdot n^T \quad x \in \partial H$

Deriving a common equation for u_e and u_T

Set $\Omega = T \cup H$, and define $u(x) : \Omega \to R$ as

$$u(x) = \begin{cases} u_T & \text{if } x \in T \\ u_e & \text{if } x \in H \end{cases}$$

The equation for u becomes

$$\begin{aligned} \nabla \cdot (M \nabla u) &= f \qquad x \in \Omega \\ (M \nabla u) \cdot n^T &= 0 \qquad x \in \partial\Omega (= \partial T) \end{aligned}$$

where

$$f(x) = \begin{cases} -\nabla \cdot (M_i \nabla v) & \text{if } x \in H \\ 0 & \text{otherwise} \end{cases}$$

and

$$M(x) = \begin{cases} M_T & \text{if } x \in T \\ M_i + M_e & \text{if } x \in H \end{cases}$$

Time discretization

1. Assume u, v and s are known at time t_n, denoted u^n, v^n and s^n
2. Calculate s^{n+1} in H by solving $\frac{ds}{dt} = F(v^n, s), t_n < t \leq t_{n+1}$
3. Calculate v^{n+1} in H by solving semi-implicitly

   $$\chi C_m \frac{v^{n+1} - v^n}{\Delta t} + \chi I_{\text{ion}}(v^n, s^{n+1}) = \nabla \cdot (M_i \nabla v^{n+1}) + \nabla \cdot (M_i \nabla u^n), \ x \in$$

4. Calculate u^{n+1} in Ω by solving

   $$\nabla \cdot (M \nabla u^{n+1}) = f^{n+1}, \ x \in \Omega$$

   where f^n is given by

   $$f^n(x) = \begin{cases} -\nabla \cdot (M_i \nabla v^n) & \text{if } x \in H \\ 0 & \text{otherwise} \end{cases}$$

The weak formulation, elliptic equation

Multiplying equation is Step 4. with a test function $\phi \in H^1(\Omega)$ and integrating over Ω yields

$$\int_\Omega \phi \nabla \cdot (M \nabla u) dx = \int_\Omega \phi f dx. \tag{6}$$

Applying Greens theorem:

$$\int_\Omega \phi \nabla \cdot (M \nabla u) dx = \int_{\partial\Omega} \phi (M \nabla u) \cdot n ds - \int_\Omega \nabla \phi \cdot (M \nabla u) dx$$

the left hand side of (6) becomes

$$\int_\Omega \phi \nabla \cdot (M \nabla u) = - \int_\Omega \nabla \phi \cdot (M \nabla u) dx \tag{7}$$

because of the boundary condition on $\partial\Omega$. Inserting (7) on the left hand side of (6) yields

$$\int_\Omega \nabla \phi \cdot M \nabla u dx = - \int_\Omega \phi f dx. \tag{8}$$

For notational purposes it will be convenient to define

$$(\phi, \psi)_\Omega = \int_\Omega \phi \psi dx$$

and

$$a_\Omega(\phi, \psi) = \int_\Omega \nabla \phi \cdot (M \nabla \psi) dx.$$

Notice that $a_\Omega(\phi, \psi) = a_\Omega(\psi, \phi)$ since M is symmetric. Equation (8) can now be reformulated as

$$a_\Omega(\phi, u) = -(\phi, f)_\Omega \tag{9}$$

The problem equivalent to the original PDE is to find the $u \in H^1(\Omega)$ which satisfies (9) for all $\phi \in H^1(\Omega)$.

Weak formulation, parabolic equation

Collecting terms of v^{n+1} and v^n in Step 3 yields

$$v^{n+1} - \frac{\Delta t}{C_m \chi} \nabla \cdot (M_i \nabla v^{n+1}) = v^n + g^n$$

where

$$g^n = \frac{\Delta t}{C_m \chi} (\nabla \cdot (M_i \nabla u^n) - \chi I_{\text{ion}}(v^n, s^{n+1}))$$

Multiplying with a test function and integrating yields

$$\int_H \phi v^{n+1} dx - \frac{\Delta t}{C_m \chi} \int_H \phi \nabla \cdot (M_i \nabla v^{n+1}) dx = \int_H \phi v^n dx + \Delta t \int_H \phi g^n dx. \tag{10}$$

Using Greens theorem on the last term on the left hand side gives

$$\begin{aligned} \int_H \phi \nabla \cdot (M_i \nabla v^{n+1}) dx &= \int_{\partial H} \phi (M_i \nabla v^{n+1}) \cdot n ds - \int_H \nabla \phi \cdot M_i \nabla v^{n+1} \\ &= - \int_H \nabla \phi \cdot M_i \nabla v^{n+1} \end{aligned}$$

where we have used the boundary condition for v in the last transition.

By defining

$$(\phi, v)_H = \int_H \phi v dx$$

and

$$a_H(\phi, v) = \int_H [\phi v + \frac{\Delta t}{C_m \chi} \nabla \phi \cdot M_i \nabla v] dx$$

we can reformulate (10) as

$$a_H(\phi, v^{n+1}) = (\phi, v^n + g^n)_H$$

In evaluating $(\phi, g^n)_H$ the Greens theorem should be invoked again to avoid double derivatives.

$$\begin{aligned} (\phi, g^n)_H &= \tfrac{\Delta t}{C_m \chi} (\phi, \nabla \cdot (M_i \nabla u^n) - \chi I_{\text{ion}}(v^n, s^{n+1}))_H \\ &= \tfrac{\Delta t}{C_m \chi} (\phi, \nabla \cdot (M_i \nabla u^n))_H - \tfrac{\Delta t}{C_m} (\phi, I_{\text{ion}}(v^n, s^{n+1}))_H \\ &= \tfrac{\Delta t}{C_m \chi} [\int_{\partial H} \phi (M_i \nabla u^n) \cdot n^T ds - \int_H \nabla \phi \cdot M_i \nabla u^n dx] \\ &\quad - \tfrac{\Delta t}{C_m} (\phi, I_{\text{ion}}(v^n, s^{n+1}))_H \end{aligned}$$

The finite element method

FEM uses the weak formulation to find an approximate solution from a subspace V_h of H^1:

$$u_h = \sum_{i=1}^{n} c_i \phi_i(x). \tag{11}$$

The parameters are determined by the following discrete equivalent of the weak form

$$a_\Omega(v_h, u_h) = -(f, v_h)_\Omega, \quad \text{for all } \; v_h \in V_h \tag{12}$$

This implies

$$a_\Omega(\sum_{i=1}^{n} c_i \phi_i, \phi_j) = -(f, \phi_j)_\Omega, \quad \text{for } \; j = 1, \dots, n \tag{13}$$

Since a_Ω is linear we can write

$$\sum_{i=1}^{n} c_i a_\Omega(\phi_i, \phi_j) = -(f, \phi_j)_\Omega, \quad \text{for } \; j = 1, \dots, n \tag{14}$$

This is a linear system of equations

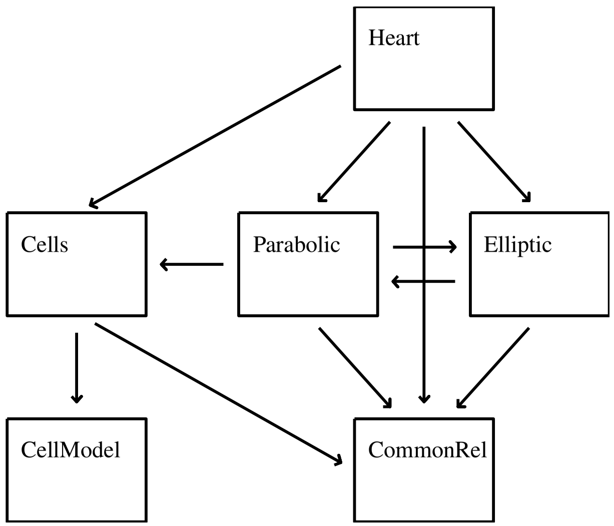
$$Ac = b$$

where $(A)_{ij} = a_\Omega(\phi_i, \phi_j)$, $(b)_j = -(f, \phi_j)_\Omega$ and $C = [c_1, \dots, c_n]$.

Implementation

- The ODE systems are solved by 4. and 5. order Runge-Kutta methods.
- The PDEs are discretized with FEM
- Software developed with Diffpack — a C++ based environment for solving differential equations.
- Domain Decomposition used for simulation on parallel hardware

Class Hierarchy



Short description of classes

Heart: Initializes the classes, set up pointers between them, contains the main time loop

Elliptic: Solves the elliptic equation, needs the solution from Parabolic for its right hand side.

Parabolic: Solves the parabolic equation, needs the solution from both Elliptic and Cells

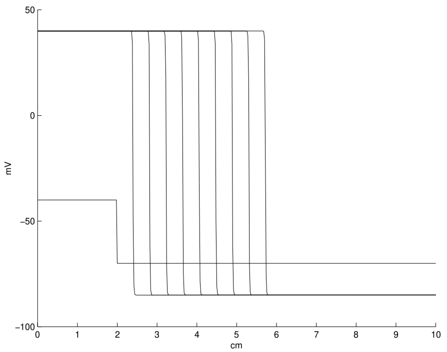
Cells: Used to solve the whole set ODEs, contains a field of state vectors

CellModel: Base class the different ODE systems

CommonRel: Contains data and method shared by several classes, e.g grid generation and conductivity tensors.

1D Simulation

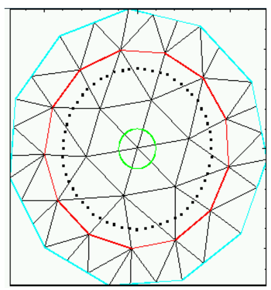
Simplified model with no repolarization, i.e. $I_{\text{ion}} = f(V)$.
Solution plotted at t = 0ms, 10ms, ...

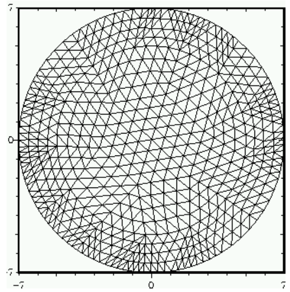


36

Resolution requirements

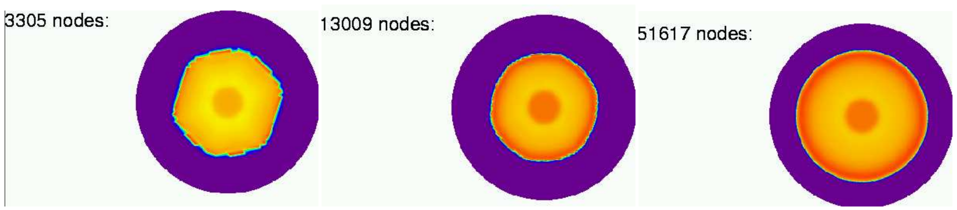
Simulations carried out on a circular 2D domain where the exact shape of the solution is known. Purpose: To find the number of nodes necessary to obtain match with the known solution.





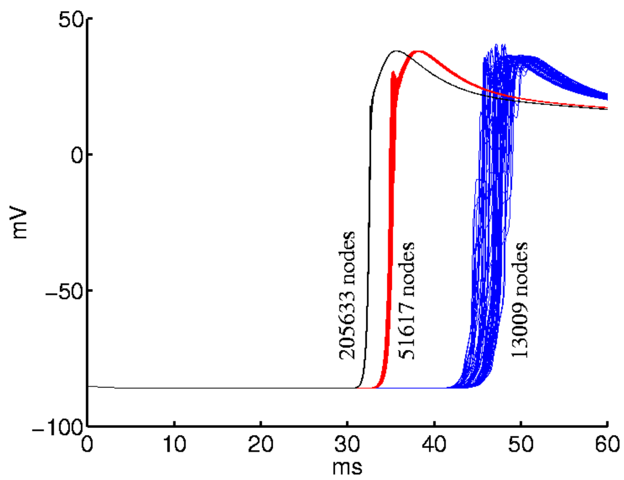
37

Solution at a fixed time, varying resolutions



38

Convergence of circular 2D problem



39

Convergence in numbers

Ref	Temporal			Spatial	
no	mean(τ)	$\max(\Delta\tau)$	sd(τ)	$\max(\Delta\hat{r})$	Δx
5	46.99	3.86	0.98	2.46	0.78
6	35.15	0.66	0.14	0.56	0.39
7	32.61	0.12	0.033	0.11	0.20
8	32.05	0.07	0.016	0.066	0.098

mean(τ) The average time of activation, measured as the moment where the potential becomes positive (ms).

$\max(\Delta\tau)$ The difference between the activation times of the earliest and the latest tracking point(ms).

sd(τ) The standard deviation of the 60 activation times(ms).

$\max(\Delta\hat{r})$ Estimate of the maximal spatial deviation from a circular solution(mm).

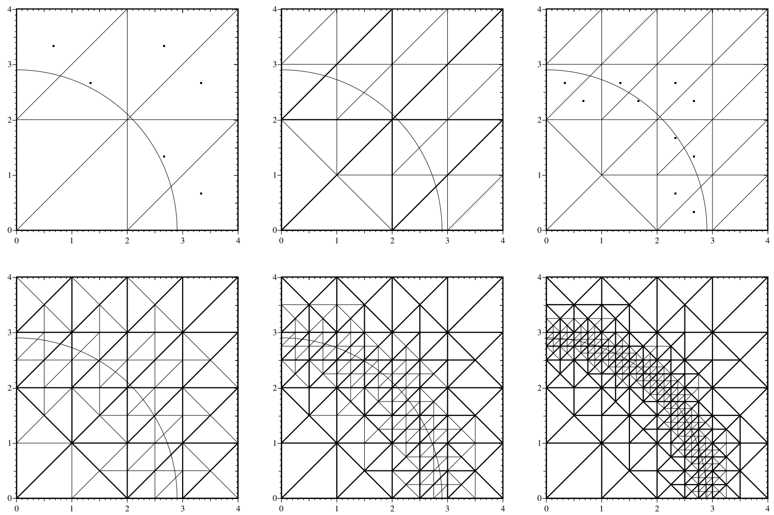
Δx The resolution of the grid (mm).

Resolution requirements, conclusion

- With the given ionic model sufficient accuracy was achieved at
  - Time step: $\Delta t = 0.125$ ms
  - Space resolution: $\Delta x = 0.02$ cm
- Manageable density in 2D
- Would require 65.000.000 nodes in a 3D sphere
- Adaptive scheme necessary in 3D

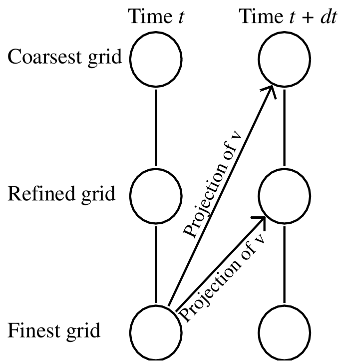
Adaptivity strategy

Refine only elements containing the wave front:

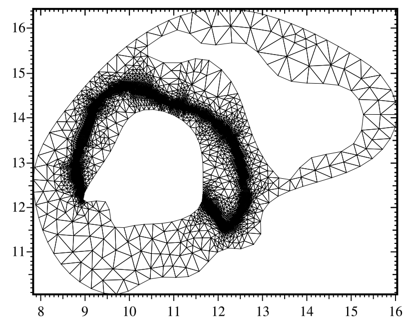


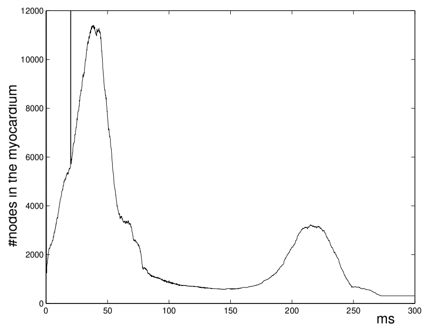
Where is the wave front?

- The solution from the previous time step is used to decide the location of the wave front.
- Elements containing the wave front is tagged an a refined grid is computed.
- The solution v^{n-1} is projected down to the refined grid
- The process is repeated until the finest grid level.

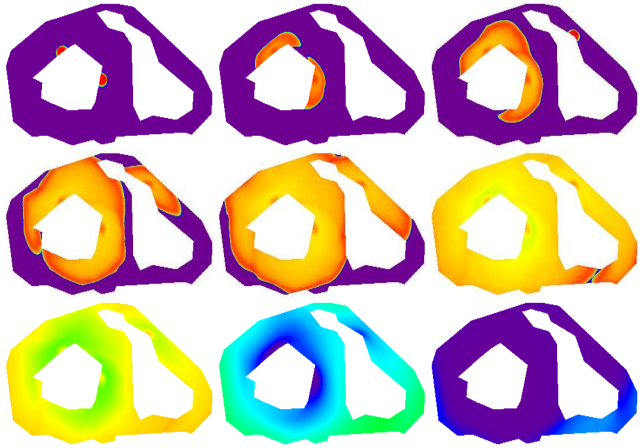


Snapshot of grid during a 2D simulation





2D simulation



CPU time consumption for varying problem sizes

N_P	N_E	Total	ODE	Make-P	Solve-P	Make-E	Solve-E
4239	16777	7.06	2.97	0.63	0.09	0.83	2.54
16383	66769	28.16	11.29	2.55	0.70	3.33	10.30
64383	266401	109.54	43.69	9.98	3.65	13.06	39.16
255231	1064257	433.09	175.38	40.55	28.12	57.13	131.92

Execution times per time step, reported in seconds.

N_P	N_E	ODE	Make-P	Solve-P	Make-E	Solve-E
4239	16777	0.70	0.15	0.02	0.05	0.15
16383	66769	0.69	0.16	0.04	0.05	0.15
64383	266401	0.68	0.15	0.06	0.05	0.15
255231	1064257	0.69	0.16	0.11	0.05	0.12

Execution times per time step and per unknown reported in milliseconds.

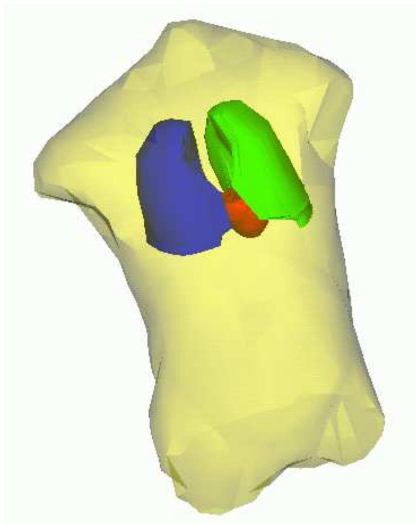
2D CPU consumption, observations

- ODE problem most CPU intensive
- Solve-E and Make-P follows, about equal per node
- Solve-P and Make-E easiest
- ODE and Make-routines scales linearly with problem size
- Solve-E also scales linearly due to multigrid
- Solve-P increases with size, but still small on largest problem

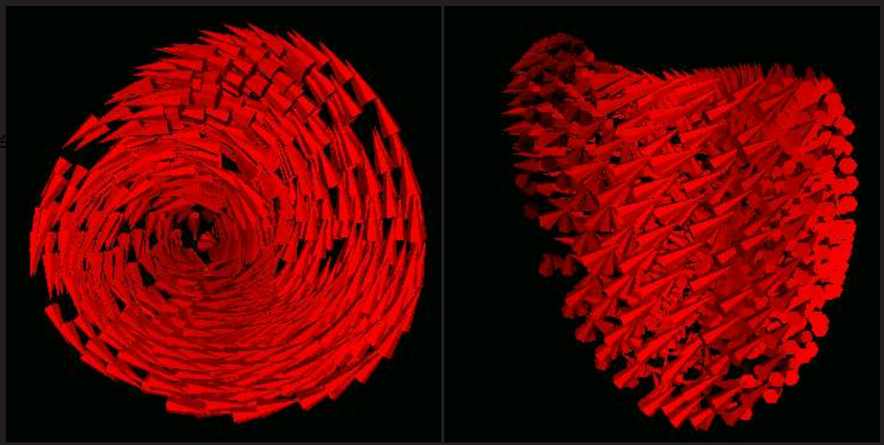
3D simulation, geometry

- Torso geometry from The Visible Human Project
- Traced contours → Surface → FEM grid of torso
- Heart geometry from The Bioengineering Research Group, Aukland
- Includes the orientation of the muscle fibers.

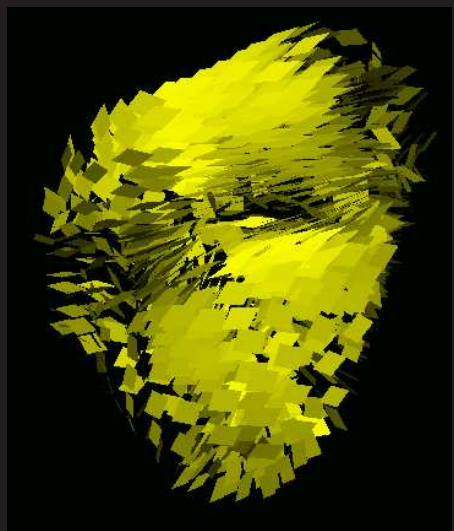
Geometry



Fiber direction



Sheet orientation



52

3D Simulation

Problem size:

- 82.792 nodes in the myocardium (230k elms)
- 219.398 nodes in the entire body (1.3M elms)
- 2400 steps at 0.125ms per step

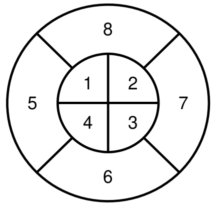
Hardware:

- Linux PC cluster
- 24 nodes
- Dual Pentium III 500MHz
- 512Mb RAM per node
- Connected with 100Mbit Ethernet

53

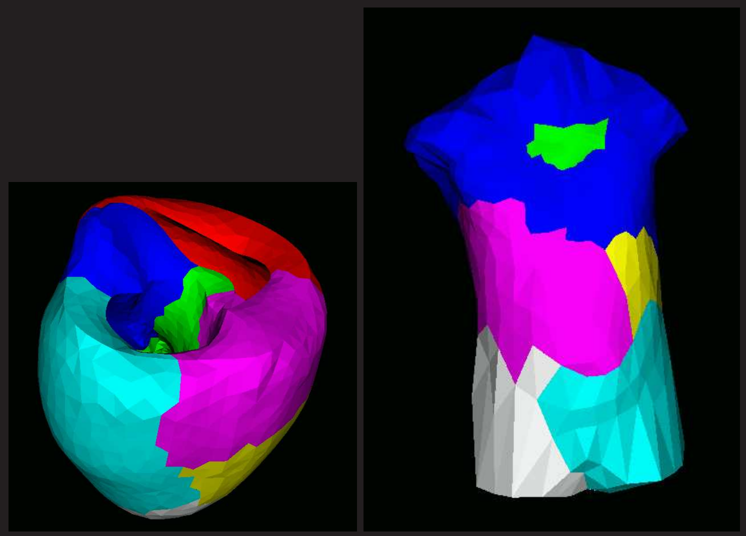
Parallel strategy

- At the linear algebra level
- Use existing tools implemented in Diffpack
- Use Metis as the grid partioner
- Non-overlapping domains
- Heart and torso domain partitioned separately
  - Problem: Different sized domains
  - Solution: Each node is assigned a domain from each grid

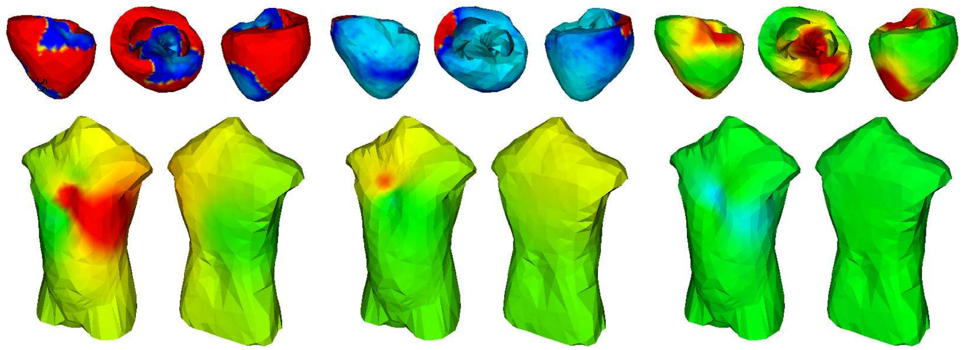


54

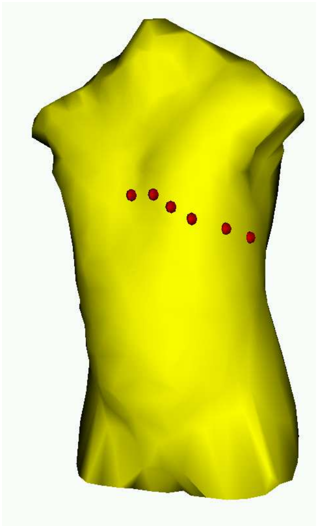
Grid partioning

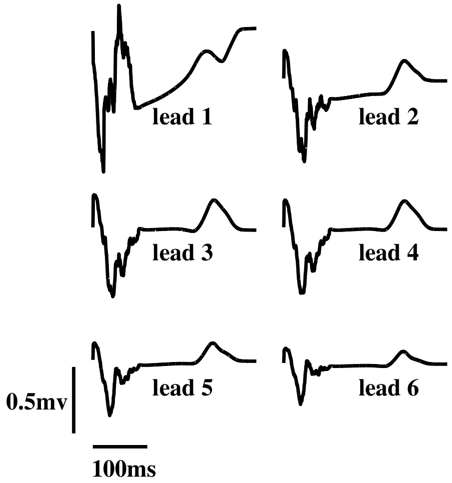


Snap shots of solution



ECG measuring points, standard 6 lead system





Performance measures

The speed-up of a parallel computation using p processors and with execution time T_p is

$$S_p = T_1/T_p$$

where T_1 is the execution time on a single CPU.

The efficiency of the computation is defined as

$$\eta_p = S_p/p$$

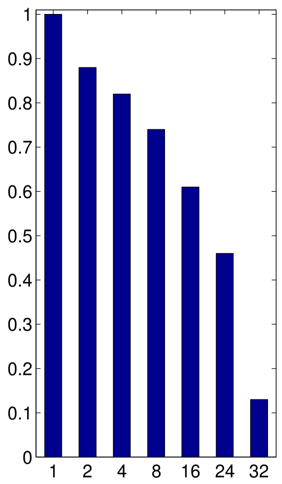
Execution time per time step (mm.ss)

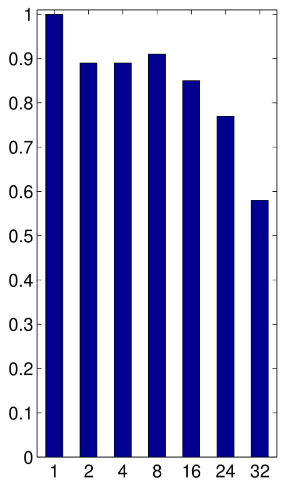
			Parabolic eq.		Elliptic eq.	
CPUs	Total	ODE	Make	Solve	Make	Solve
1	1370.04	75.01	4.38	24.27	3.42	1260.15
2	767.53	39.09	2.13	12.08	1.49	711.31
4	385.47	20.17	1.09	6.25	0.52	356.31
8	188.39	9.54	0.30	3.14	0.25	174.18
16	101.04	5.35	0.15	1.50	0.11	93.04
24	73.56	3.45	0.11	1.22	0.08	68.23
32	74.20	2.55	0.08	1.22	0.07	69.44

Speed-up

			Parabolic eq.		Elliptic eq.	
CPUs	Total	ODE	Make	Solve	Make	Solve
1	N/A	N/A	N/A	N/A	N/A	N/A
2	1.78	1.92	2.08	2.01	2.03	1.77
4	3.55	3.70	4.01	3.81	4.23	3.53
8	7.26	7.57	9.18	7.54	8.75	7.23
16	13.56	13.42	18.24	13.35	19.49	13.54
24	18.53	20.00	25.56	17.79	26.83	18.43
32	18.43	25.78	32.70	17.92	32.43	18.07

Efficiency, 28k and 220k nodes





Observations

- Close to optimal speed-up on sub-tasks with little communication
- Lower speed-up on more communication intensive parts
- Speed-up deteriorates with use of both CPUs on each node
- Better domain solvers needed, largest potential for efficiency gains here.
- Efficiency depends on problem size.