**BME 7310 Computational Laboratory #5**

Due: October 10, 2023

**PROBLEM #3. Nerve Transmission Simulations.** A firm has decided to award a contract to you for the development of a nerve stimulus transmission software package. Specifically, they want you to solve the **1-D Telegrapher’s equation** which is often used in impulse transmission models in nerves. The equation is:



where ‘c’ is the wave speed, ‘U’ is potential, and ‘’ is associated with damping. For the numerical scheme, use the following time-splitting scheme and assume second order center difference terms in time:



where ‘l+1, l, l-1’ refers to the future time step, the current time step, and the previous time step, respectively. You are going to solve this on a uniformly spaced 1-D grid, i.e. equal spatial steps in ‘x’.

1. To accomplish this, you will begin by reporting the *fully discretized molecule equations for a general* ***interior node*,** and *a node on each boundary* subject to the boundary conditions below:
2. 
3. 

*Be sure to explicitly write out how boundary conditions (i), and (ii) will be implemented. Be sure to address their spatial and temporal components.* For reference, boundary conditions (i) and (ii) are known as *open boundary conditions*. Their purpose is to let a propagating ***30 mv*** action potential waveform pass through the boundary as though it (i.e. the boundary) were not even there. This type of situation arises when studying physical problems which have domains that extend to infinity or are very long as in the case of nerve fibers. In practice, the mesh must be **truncated at some point which creates a fictitious boundary** with respect to the physical problem and it is desired to have the numerical solution “pass through” the **artificial termination of the computational domain**. In developing your molecules for these boundaries, you will need to extend beyond the standard Type III boundary condition for this case.

A diagram of mathematical equations

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

1. Now, code it up. For the sake of a sample problem on somewhat biological scales, let’s begin with Lx=100cm, c=9.905 cm/s, =0.05/s and the ***30 mv*** initial condition of

 

30

whereis the radius of a prescribed circle centered (xo) and , compute the solution to this problem when  and xo=50cm and with dx=2cm for the grid, and =0.5 with a time step of dt=0.005 sec. Solve this problem using whatever **direct or iterative method** you wish to compute the matrix solution needed to advance the solution at each time step (i.e. because the discretization scheme is **‘implicit’** you will need to solve a spatially coupled system to advance at each time step). With respect to output from your code:

**Answer:**

**(1)** Plot the solution at **every second over the interval 0< t <10 seconds**,

Here Jacobi Tol=1e-5:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | |

**2)** Plot the complete time history of your solution at x=0 cm and x=100 cm

|  |  |
| --- | --- |
|  |  |

1. What are the maximum magnitudes of the wave peaks reaching x=0 cm and x=100 cm? Does the value make sense – why or why not, explain using available information about problem? Also, report the time it takes for the maximum peak of the action potential to reach the boundary at x=0 cm and x=100 cm, does it make sense – why or why not, explain using the available information about problem?

**Answer:** Maximum magnitudes for wave peaks reaching x=0 cm and x=100 cm are both *14.016 mv* at *t=5.045 sec* due to the symmetric characteristic of the problem. This makes sense since *t=0* we pump in the potential with magnitude *30mv*, then the profile of our domain almost dissipates at *t=10sec* (from series of figure in b) )*.* Since it is symmetric, we get equal magnitudes on both sides of the boundary, and ~half of *10sec* for the lobes to travel to the edges.

|  |
| --- |
| %% ---------------- Diffusion - Problem 3-----------------  clear all  r0 = 10; % cm  x0 = 50; % cm;  dx = 2; % cm  h=dx;  Lx = 100; %cm  c = 9.905; % cm/s  tau = 0.05; % sec  x=[0:dx:Lx];  dt = 0.005; %sec  theta = 0.5;  r = sqrt((x-x0).^2);  A = 30 \* cos(pi \* r / (2\*r0) ) ;  A(r>r0) = 0;  A = [0,A,0]; % extend PBC  Aprev = A;  Anext\_new = A;  Anext\_old = A;  time\_step = 0;  time\_points = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];  tolCheck = 10\*eps;  k = c^2\*dt^2 / (h^2);  figure(1), clf  plot(x,A(2:end-1),'DisplayName',['Time: '+ string(time\_step\*dt)+' sec'])  legend(gca,'show')  xlabel('x')  ylabel('C')  title(['Implicit FDM: # of Iterations '+ string(time\_step)+ '. Total time: ' + string(time\_step\*dt)+ ' sec. ' + 'Timestep: '+ string(dt)]);  x0 = [A(2)];  xL = [A(end-1)];  t = [0];  while (time\_step\*dt < 10)  time\_step=time\_step+1;  if time\_step > 1  Aprev = A;  A = Anext\_new;  elseif time\_step == 1  Aprev = A;  Anext\_new = A;  end  Anext\_old = Anext\_new;  itr=0;  pitr=0;  error=1;  while (error > 1e-5 & itr < 10000)  i=2;  Anext\_new(i) = -1 / (1 + tau\*dt/2 + k\*theta + k\*h/(c\*dt) + k\*theta\*tau\*h/(2\*c)) \* ...  ( ...  -k\*theta\* Anext\_old(i+1) + ...  (-2 + ( 2\*k + h\*k\*tau/c)\*(1-theta) )\* A(i) + (-2\*k\*(1-theta))\*A(i+1) + ...  (1 - tau\*dt/2 + k\*theta - k\*h/(c\*dt) + k\*theta\*h\*tau/(2\*c))\*Aprev(i) + (-k\*theta)\*Aprev(i+1) ...  );    for i=3:(length(A)-2)  Anext\_new(i) = -1 / (1 + tau\*dt/2 + k\*theta) \* ...  (...  (-k\*theta/2)\*Anext\_old(i-1) + (-k\*theta/2)\*Anext\_old(i+1) +...  (-k\*(1-theta))\*A(i-1) + (-2 + 2\*k\*(1-theta))\*A(i) + (-k\*(1-theta))\*A(i+1) +...  (-k\*theta/2)\*Aprev(i-1) + (1 - tau\*dt/2 + k\*theta)\*Aprev(i) + (-k\*theta/2)\*Aprev(i+1)...  );  end  i=length(A)-1;  Anext\_new(i) = -1 / (1 + tau\*dt/2 + k\*theta + k\*h/(c\*dt) + k\*theta\*tau\*h/(2\*c)) \* ...  ( ...  (-k\*theta)\* Anext\_old(i-1) + ...  (-2 + ( 2\*k + h\*k\*tau/c)\*(1-theta) )\* A(i) + (-2\*k\*(1-theta))\*A(i-1) + ...  (1 - tau\*dt/2 + k\*theta - k\*h/(c\*dt) + k\*theta\*h\*tau/(2\*c))\*Aprev(i) + (-k\*theta)\*Aprev(i-1) ...  );    error=max(max(abs(Anext\_new-Anext\_old)))/max(max(Anext\_old));  itr=itr+1;  Anext\_old = Anext\_new;  pitr=pitr+1;  end  % compute shadow nodes  i=2;  Anext\_new(i-1) = (-2\*h/(c\*theta\*dt)) \* ( ...  (Anext\_new(i) - Aprev(i)) ...  - c\*dt \* ( theta/(2\*h) \* (Anext\_new(i+1) + Aprev(i+1) - Aprev(i-1)) + (1-theta)/h\*(A(i+1) - A(i-1)) ) ...  + tau\*dt \*(theta\*( Anext\_new(i) + Aprev(i) )/2 + (1-theta)\*A(i)) ...  );    i=length(A)-1;  Anext\_new(i+1) = (-2\*h/(c\*dt\*theta)) \* ( ...  (Anext\_new(i) - Aprev(i)) ...  + c\*dt \* (theta/(2\*h) \* ( -Anext\_new(i-1) + Aprev(i+1) - Aprev(i-1)) + ((1-theta)/h)\*(A(i+1) - A(i-1)) ) ...  + tau\*dt \*(theta\*( Anext\_new(i) + Aprev(i) )/2 + (1-theta)\*A(i)) ...  );    t(end+1) = time\_step\*dt;  x0(end+1) = Anext\_new(2);  xL(end+1) = Anext\_new(end-1);    % plot  if any(abs(time\_points - time\_step\*dt) <= tolCheck)  figure(time\_step)  plot(x,Anext\_new(2:end-1), 'DisplayName',['Time: '+ string(time\_step\*dt)+' sec'])  legend(gca,'show')  xlabel('x')  ylabel('C')  title(['Implicit FDM: # of Iterations '+ string(time\_step)+ '. Total time: ' + string(time\_step\*dt)+ ' sec. ' + 'Timestep: '+ string(dt)]);  end  end  [PkAmp, PkTime] = findpeaks(x0);  [~,idx] = sort(PkAmp,'descend');  max\_x0 = PkAmp(idx(1)); %Amplitude of the peak  max\_time\_x0 = t(PkTime(idx(1))); %Time of the peak  [PkAmp, PkTime] = findpeaks(xL);  [~,idx] = sort(PkAmp,'descend');  max\_xL = PkAmp(idx(1)); %Amplitude of the peak  max\_time\_xL = t(PkTime(idx(1))); %Time of the peak  figure(1)  plot(t,x0)  xlabel('t')  ylabel('x0')  title(['x0 over time. Maximum peak at '+ string(max\_x0) + ' at '+ string(max\_time\_x0) + ' sec']);  figure(2)  plot(t, xL)  xlabel('t')  ylabel('xL')  title(['xL over time. Maximum peak at '+ string(max\_xL)+ ' at '+ string(max\_time\_xL) + ' sec']); |