Table of Contents

Core Conductor model: Propagation Simulation

```
function [time, V_membrane, I_total, I_s, I_C, I_Na, I_K, I_L, g_Na, g_K, g_L]
 = HHPropagate(simulation_time, stim1mag, ...
    stim1start, stim1dur, stim1location, stim2mag, ...
    stim2start, stim2dur, stim2location, axon_length, if_plot)
```

Hyperparameters

```
dt = 1e-2;
simulation_time_in_samples = simulation_time/dt;
dx = 1;
```

Stimulation vector checks

```
stim1start_in_samples = stim1start / dt;
stim1dur_in_samples = stim1dur / dt;
stim2start_in_samples = stim2start / dt;
stim2dur_in_samples = stim2dur / dt;

if stim1mag
    if stim1start_in_samples + stim1dur_in_samples >
 simulation_time_in_samples
        error('Invalid Parameters.')
```

```
    end
end
if stim2mag
    if stim2start_in_samples + stim2dur_in_samples >
 simulation_time_in_samples
        error('Invalid Parameters.')
    end
end
```

Cell related parameters

```
a = 10*10^-6;
ri = 2000;
re = 1;
```

Constants

Current will be in microamperes, time will be in msec, hence capacitance must be in uFarads

```
% mV %
E_Na = -115; %Sodium nernst is positive
E_K = 12;
E_l = -10.613;
% mS
g_na_bar = 120;
g_k_bar = 36;
g_l_bar = 0.3;
```

Resting referenced Nernst potential corrections

```
V_rest = -90; %mV
E_Na = V_rest - E_Na; %mV
E_l = V_rest - E_l; %mV
E_K = V_rest - E_K; %mV
```

Vector Initializations

```
vm = zeros(simulation_time_in_samples,axon_length/dx);
Delta_vm = zeros(size(vm));
I_s = zeros(size(vm));

stim1start_in_samples = stim1start / dt;
stim1dur_in_samples = stim1dur / dt;
stim2start_in_samples = stim2start / dt;
stim2dur_in_samples = stim2dur / dt;

I_s(stim1start_in_samples:(stim1start_in_samples
+stim1dur_in_samples),stim1location/dx) = stim1mag;
I_s(stim2start_in_samples:(stim2start_in_samples
+stim2dur_in_samples),stim2location/dx) = stim2mag;
```

Current vector initializations

```
I_Na = zeros(size(vm));
I_K = zeros(size(vm));
I_L = zeros(size(vm));
I_C = zeros(size(vm));
I_total = zeros(size(vm));
C_m = ones(size(vm));

[a_mi,b_mi,a_hi,b_hi,mi,tau_m,hi,tau_h] = calculate_na_params(0);
[a_ni,b_ni,ni,tau_n] = calculate_k_params(0);

n = ni*ones(size(vm));
m = mi*ones(size(vm));
h = hi*ones(size(vm));
```

Channel Conductances

```
g_Na = g_na_bar*mi^3*hi*ones(size(vm));
g_K = g_k_bar*ni^4*ones(size(vm));
g_L = g_l_bar*ones(size(vm));

a_n = a_ni*ones(size(vm));
a_m = a_mi*ones(size(vm));
a_h = a_hi*ones(size(vm));
b_n = b_ni*ones(size(vm));
b_h = b_hi*ones(size(vm));
b_m = b_mi*ones(size(vm));
```

Define membrane voltage

```
V_membrane = V_rest*ones(size(vm));
denominator_for_vx_update = ((2*pi*a)*(ri+re));
```

Action potential propagation loop

```
for t = 1:simulation_time_in_samples-1
    for x = 1:(axon_length/dx)
```

Calculate second derivative in all points for membrane current

```
        if x>1 & x<axon_length/dx % Second order derivative
            I_total(t,x) = I_s(t,x) + ((vm(t,x-1)-2*vm(t,x)+vm(t,x+1))/dx^2 +
 re*I_s(t,x))/denominator_for_vx_update;
        elseif x == 1 % (only right derivative)
            I_total(t,x) = I_s(t,x) + ((-vm(t,x)+vm(t,x+1))/dx + re*I_s(t,x))/
denominator_for_vx_update;
        elseif x == axon_length/dx % x at final position (only left
 derivative)
```

```
            I_total(t,x) = I_s(t,x) + ((vm(t,x-1)-vm(t,x))/dx + re*I_s(t,x))/
denominator_for_vx_update;
        end
```

Calculate channel conductances

```
        g_Na(t,x) = g_na_bar*m(t,x)^3*h(t,x);
        g_K(t,x) = g_k_bar*n(t,x)^4;
        g_L(t,x) = g_l_bar; % does not change
```

Update membrane voltage with respect to extracellular potential

```
        V_membrane(t,x) = vm(t,x) + V_rest;
```

Calculate channel currents

```
        I_Na(t,x) = g_Na(t,x) * (V_membrane(t,x)-E_Na);
        I_K(t,x) = g_K(t,x) * (V_membrane(t,x)-E_K);
        I_L(t,x) = g_L(t,x) * (V_membrane(t,x)-E_l);
        I_C(t,x) = I_total(t,x) - (I_Na(t,x) + I_K(t,x) + I_L(t,x));
```

Update deviation from resting

```
        Delta_vm(t,x) = dt * I_C(t,x) / C_m(t,x);
        vm(t+1,x) = vm(t,x) + Delta_vm(t,x);
```

Calculate activation and inactivation parameters

```
        [a_m(t,x),b_m(t,x),a_h(t,x),b_h(t,x),~,mi,~,hi] =
 calculate_na_params(vm(t,x));
        [a_n(t,x),b_n(t,x),ni,~] = calculate_k_params(vm(t,x));
```

Set m, n, h for the next iteration

```
        m(t+1,x) = m(t,x) + dt * (a_m(t,x)*(1 - m(t,x)) - b_m(t,x)*m(t,x));
        n(t+1,x) = n(t,x) + dt * (a_n(t,x)*(1 - n(t,x)) - b_n(t,x)*n(t,x));
        h(t+1,x) = h(t,x) + dt * (a_h(t,x)*(1 - h(t,x)) - b_h(t,x)*h(t,x));

    end
end
```

The final time step is adjusted so that it is now with respect to extracellular potential

```
V_membrane(end,:) = vm(end,:) + V_rest;
```

Return the time vector for plotting purposes.

```
time = [1:simulation_time_in_samples]*dt;
```

Plot the membrane voltage and excatation current.

```
if if_plot == 1
% figure
plot(time,V_membrane,'LineWidth',2)
ylabel('Voltage(mV)')
hold on
yyaxis right
plot(time,I_d,'LineWidth',3)
ylabel('Current({\mu}A)')
xlabel('Time(ms)')
legend('Membrane Potential','Excitation
 Current','Location','northeastoutside')
end

end
```

Published with MATLAB® R2022b