Coding the Hodgkin-Huxley Model of a Neuronal Action Potential

Mathematical Description

$$C\frac{dV_m}{dt} = I_{ext} - (I_{Na} + I_K + I_L)$$

$$I_{Na} = G_{Na}m^3h(V_m - E_{Na})$$

$$I_k = G_K n^4(V_m - E_K)$$

$$I_L = G_L(V_m - E_L)$$

$$\frac{dx}{dt} = \alpha_x(1-x) - \beta_x x$$

$$x \in \{m, h, n\}$$

Euler's Forward Method of Integration

$$V_m(t+1) = V_m(t) + \frac{dt}{C}\left(I_{ext} - (I_{Na} + I_K + I_L)\right)$$

$$x(t+1) = x(t) + dt\left(\alpha_x\left(1 - x(t)\right) - \beta_x x(t)\right)$$

Set model constants

```
Vrest       = -70; % mV
dt          = 0.01; % ms
totalTime   = 180; % ms
startStim   = 50; % ms
endStim     = 150; % ms
startStim   = startStim / dt;
endStim     = endStim / dt;
t           = 0:dt:totalTime;
C       = 1; % uF/cm^2
E_Na    = 115 + Vrest; % mV
E_K     = -6 + Vrest; %mV
E_Leak  = 10.6 + Vrest; % mV
g_Na    = 120; % mS/cm^2
g_K     = 36; % mS/cm^2
g_Leak  = 0.3; % mS/cm^2
I_ext   = 13; % uA/cm^2

I_current = ones(1,length(t))*0.0;
I_current(startStim:endStim) = I_ext;
```

```
V(1) = Vrest; % membrane potential is starting at its resting state
% Separate functions to get the alpha and beta values
[alphaM, betaM] = m_equations(V(1), Vrest);
[alphaN, betaN] = n_equations(V(1), Vrest);
[alphaH, betaH] = h_equations(V(1), Vrest);
% Initializing gating variables to the asymptotic values
%  when membrane potential is set to the membrane resting value
m(1) = (alphaM / (alphaM + betaM));
n(1) = (alphaN / (alphaN + betaN));
h(1) = (alphaH / (alphaH + betaH));
```

Start simulation and calculate conductances

```
% Start simulation
for i = 1:length(t)
    % Calculate new alpha and beta based on last known
    [alphaN, betaN] = n_equations(V(i), Vrest);
    [alphaM, betaM] = m_equations(V(i), Vrest);
    [alphaH, betaH] = h_equations(V(i), Vrest);
    % Conductance variables
    % %(computed separately to show how this changes wi
    G_K(i) = g_K*(n(i)^4);
    G_Na(i)=g_Na*(m(i)^3)*h(i);
```

Calculate ionic currents

```
    % Calculating ionic currents
    I_Na(i) = G_Na(i)*(V(i)-E_Na);
    I_K(i) = G_K(i)*(V(i)-E_K);
    I_Leak(i) = g_Leak*(V(i)-E_Leak);
    % Calculating the total input
    Input = I_current(i) - (I_Na(i) + I_K(i) + I_Leak(i));
```

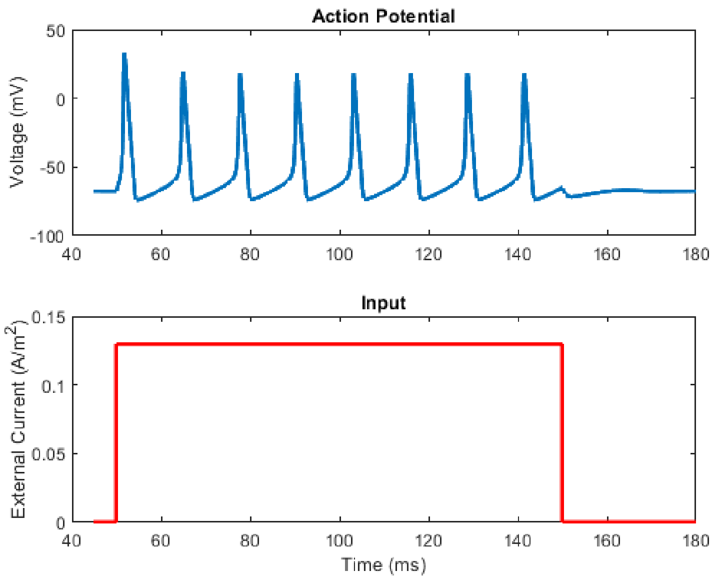
Use Euler's method for solving the system of differential equations

```
    % Calculating the new membrane potential
    V(i+1) = V(i) + Input* dt*(1/C);
    % Calculating new values for the gating variables
    m(i+1) = m(i) + (alphaM *(1-m(i)) - betaM * m(i))*dt;
    n(i+1) = n(i) + (alphaN *(1-n(i)) - betaN * n(i))*dt;
    h(i+1) = h(i) + (alphaH *(1-h(i)) - betaH * h(i))*dt;
end
```

Plot results

```
% Plot membrane potential
figure('Name', 'Membrane Potential vs input')
subplot(2,1,1)
plot(t,V)
ylabel('V_m (mV)')
title('Action Potential')
subplot(2,1,2)
plot(t,I_current*amp, 'r')
xlabel('Time (ms)')
ylabel('External Current (A/m^2)')
title('Input')
```

Results



Helper functions for getting *Alpha* and *Beta* values for *m*, *n* and *h*

Calculate alpha m and beta m for Na activation

```
function [alpha_m, beta_m] = m_equations(V, Vrest)
alpha_m = (2.5-0.1*(V-Vrest))/(exp(2.5-0.1*(V-Vrest))-1);
beta_m = 4*exp((Vrest-V)/18);
end
```

Calculate alpha n and beta n for K activation

```
function [alpha_n, beta_n] = n_equations(V, Vrest)
alpha_n = (0.1-0.01*(V-Vrest))/(exp(1-0.1*(V-Vrest))-1);
beta_n = 0.125*exp((Vrest-V)/80);
end
```

calculate alpha h and beta h for Na inactivation

```
function [alpha_h, beta_h] = h_equations(V, Vrest)
alpha_h = 0.07*exp((Vrest-V)/20);
beta_h = 1/(1+exp(3-0.1*(V-Vrest)));
end
```