# COM 6009 ASSIGNMENT 2

# MODELLING REAL NEURONS


Aditya Chauhan

Student Id : acp17ac

*Supervisor:* Eleni Vasilaki

*Supervisor:* Dawn Walker


May 17, 2018

# Contents

# List of Figures

# Chapter 1

# INTEGRATE AND FIRE MODEL

The basic simulation of the neuron for *integrate-and-fire model* consist of a capacitor **C** in parallel with a Resistor **R** with an injected current **I(t)**, where current is a time function describing the variable spikes in neurons based on the firing rate depending on different situations brain is subjected to.

The driving current **I(t)** can be split up into two different components (resistive current) $I_R$ and (capacitive current) $I_C$.

$$I(t) = I_R + I_C \tag{1.1}$$

The resistive current can be derived by Ohms Law as $I_R = u/R$ where $u$ is the voltage across resistor R. The capacitive current can be defined as $I_C = Cdu/dt$.

$$I(t) = \frac{u(t)}{R} + C\frac{du}{dt} \tag{1.2}$$

Rearranging the equation to get the differential form of voltage we get a standard form of leaky integration:

$$RC\frac{du}{dt} = -u(t) + RI(t) \tag{1.3}$$

Replace RC with membrane time constant

$$\tau_m \frac{du}{dt} = -u(t) + RI(t) \tag{1.4}$$

In this model spikes are formed based on firing time ($t^f$) and this firing time is defined by threshold criterion.

$$t^f : u(t^f) = \vartheta \tag{1.5}$$

As soon as the firing time is over, the potential is reset to a new value $u_r$ which is less than $\vartheta$.

$$\lim_{t \to \; t^f, t > t^f} u(t) = u_r \tag{1.6}$$

**The combination of equation 1.4 and 1.6 defines the basic integrate-and-fire model.**[5]

# Chapter 2

# SIMULATE INF MODEL

## 2.1 Under Constant Input

Simulating integrate-and-fire model defined by (1.4) and (1.6) over the constant current I(t) = $I_0$ and setting reset potential $u_r = 0$. Let assume that spike occurred at $t = t^1$ and the initial condition for potential is $u(t^1) = u_r = 0$. Integrating the above integrate-and-fire equation with these initial conditions results in:

$$u(t) = RI_0[1 - \exp(-\frac{t - t^1}{\tau_m})] \tag{2.1}$$

While $RI_0 < \vartheta$ no further spikes occur and while $RI0 > \vartheta$ the membrane potential reaches the threshold and stays there for the total duration left on the initial time span thus setting $u(t^2) = \vartheta$. Thus making equation (2.1):

$$\vartheta = RI_0[1 - \exp(-\frac{t^2 - t^1}{\tau_m})] \tag{2.2}$$

Rearranging and calculating time period $T = t^2 - t^1$results in:

$$T = \tau_m \ln \frac{RI_0}{RI_0 - \vartheta} \tag{2.3}$$

Above equation(2.3) describes the constant input integrate-and-fire model and time interval between spikes.

In the Figure 2.2 the spike occurs at the time the constant current is injected and continues to stay the same voltage until the current injection is stopped after which it declines gradually for the duration of the remaining time without any injected current.

```
dt = 0.1;                                                   %time step
t_total = 500;                                              %total time to run
inject = 100;                                     %time to start injecting current
stop = 400;                                       %time to end injecting current
Resting_pot = -70;                                               %resting membrane potential
Resistance = 10;                                            %membrane resistance
tau = 10;                                                   %membrane time constant

time = 0:dt:t_total;                                   %vector for time
V = zeros(1,length(time));                        %voltage vector
i = 1;
V(i)=Resting_pot;                                           %first element of V, i.e V at t = 0

I_Stim = 1;                                                 %magnitude of pulse of injected current
Ivect = zeros(1,inject/dt);                    %portion of current from t=0 to t=t_current_inject
Ivect = [Ivect I_Stim*ones(1,1+((stop-inject)/dt))];
%add portion from t=inject to t=stop
Ivect = [Ivect zeros(1,(t_total-stop)/dt)];
%add portion from t=stop to t=t_end
for t=dt:dt:t_total                                         %loop through values of t in steps of dt ms
 V_inf = Resting_pot + Ivect(i)*Resistance;                   %value that V is exponentially
 V(i+1) = V_inf + (V(i) - V_inf)*exp(-dt/tau);
 i = i + 1;
end
%MAKE PLOTS
figure(1)
plot(time, V);
title('Voltage vs Time');
xlabel('Time');
ylabel('Voltage');
```

Figure 2.1: Code snippet for constant input Simulation of Integrate and fire model

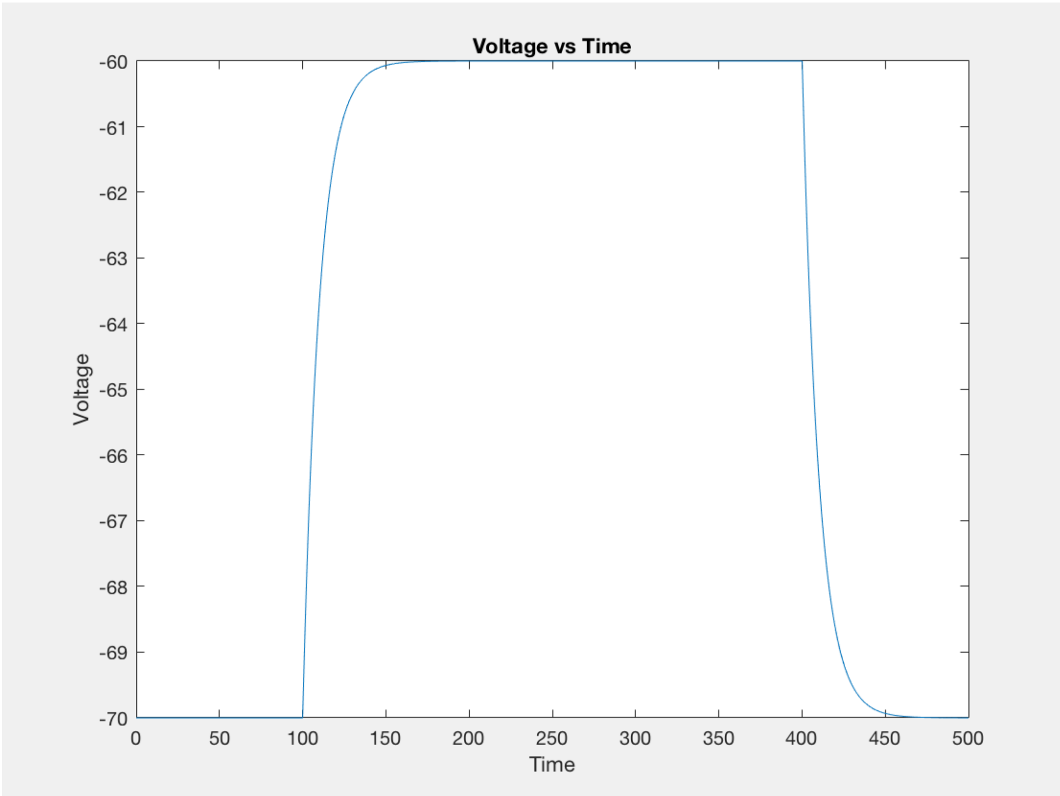


Figure 2.2: Result of constant input Integrate and fire model

## 2.2 Spike Triggering constant current

```
dt = 0.1;                                          %time step
t_total = 500;                                     %total time to run
inject = 100;                                      %time to start injecting current
stop = 400;                                        %time to end injecting current
Resting_pot = -70;
Threshold = -55;                                   %spike threshold
reset = -75;                                       %value to reset voltage to after a spike
spike = 20;                                        %resting membrane potential
Resistance = 10;                                   %membrane resistance
tau = 10;                                          %membrane time constant
time = 0:dt:t_total;                               %vector for time
V = zeros(1,length(time));
V_plot = zeros(1,length(time));  i = 1;
V(i)=Resting_pot;
V_plot(i) = V(i);
I_Stim = 1.55;                                     %magnitude of pulse of injected current
Ivect = zeros(1,inject/dt);                        %portion of current from t=0 to t=t_current_inject
Ivect = [Ivect I_Stim*ones(1,1+((stop-inject)/dt))];
Ivect = [Ivect zeros(1,(t_total-stop)/dt)]; Num = 0;
for t=dt:dt:t_total                                %loop through values of t in steps of dt ms
 V_inf = Resting_pot + Ivect(i)*Resistance;        %value that V is exponentially
 V(i+1) = V_inf + (V(i) - V_inf)*exp(-dt/tau);
 if (V(i+1) > Threshold)
 V(i+1) = reset;
 V_plot(i+1) = spike;
 Num = Num + 1;
 else
 V_plot(i+1) = V(i+1);
 end
 i = i + 1;
end
plot(time, V_plot);
```

Figure 2.3: Code to observe spikes in neuron model

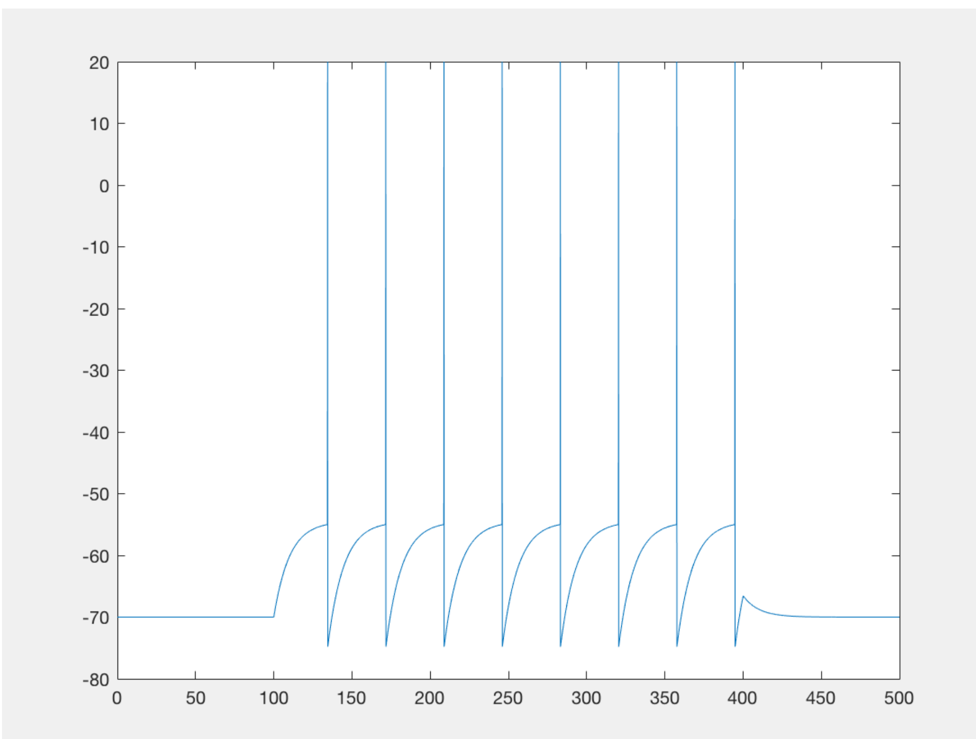


Figure 2.4: Resulting spikes in neuron model

# Chapter 3

# LINK BETWEEN INTEGRATE-AND-FIRE AND POISSON MODEL

The output of the Poisson neuron model is rate modulated Poisson process. The time varying rate parameter $\mathbf{r(t)}$ is a instantaneous function of the stimulus. Because of the instantaneous firing rate and instant probability of firing the output of this model is a stochastic function of the input.

In integrate and firing model the output is filtered based on the threshold function. The input is passed through the low-pass filter (this filter is based on $\tau$, the membrane time constant) and than integrated until the membrane potential $\nu(t)$ reaches the threshold and than it resets to the initial value of membrane potential $\nu(t)$. This in turn make the output of the integrate-and-fire model more deterministic in comparison to the stochastic nature of Poisson model. Due to this deterministic nature of this model it is usually used as the primary model for understanding the behaviour of single neurons.

A slightly modified version of Poisson Model can be derived from the integrate-and-fire model.

If the firing rate is passed not on constant rate but with certain probability than the integrate-and-fire model can be changed to Poisson model. This will change the output of the neuron at particular input to a conditional distribution of firing rate time. This conditional distribution is represented as: $p(T/s(t))$ where $\mathbf{s(t)}$ is particular input and $\mathbf{T}$ is the firing time. This above assumption is true because while integrate and fire model produces the same spike train on different trial each time, the Poisson model produces the a different spike train. So in order to make this conversion work from integrate and fire to Poisson we have to consider the noise function that arises in the stimulus itself.

To summarize the integrate and fire model is a deterministic model but by including the noise function in the stimulus this model can lead to the derivation of the stochastic Poisson neuron model.

## 3.1 Simulation of Poisson Neuron Model

```
function poisson
timeStepS = 0.001;
spikesPerS = 50;
durationS = 1.000;
spikes = makeSpikes(timeStepS, spikesPerS, durationS, 25);
rasterPlot(spikes, timeStepS);
end
function rasterPlot(spikes, ~)
figure(1);
axes('position', [0.1, 0.1, 0.8, 0.8]);
axis([0, length(spikes) - 1, 0, 1]);
trains = size(spikes, 1);
ticMargin = 0.01;
ticHeight = (1.0 - (trains + 1) * ticMargin) / trains;
for train = 1:trains
    spikeTimes = find(spikes(train, :) == 1);
    yOffset = ticMargin + (train - 1) * (ticMargin + ticHeight);
    for i = 1:length(spikeTimes)
        line([spikeTimes(i), spikeTimes(i)], [yOffset, yOffset + ticHeight]);
    end
end
xlabel('Time (s)')
title('Raster plot of spikes');
end
function spikes = makeSpikes(timeStepS, spikesPerS, durationS, numTrains)...
```

Figure 3.1: Code to observe Poisson neuron model implementation using raster plots

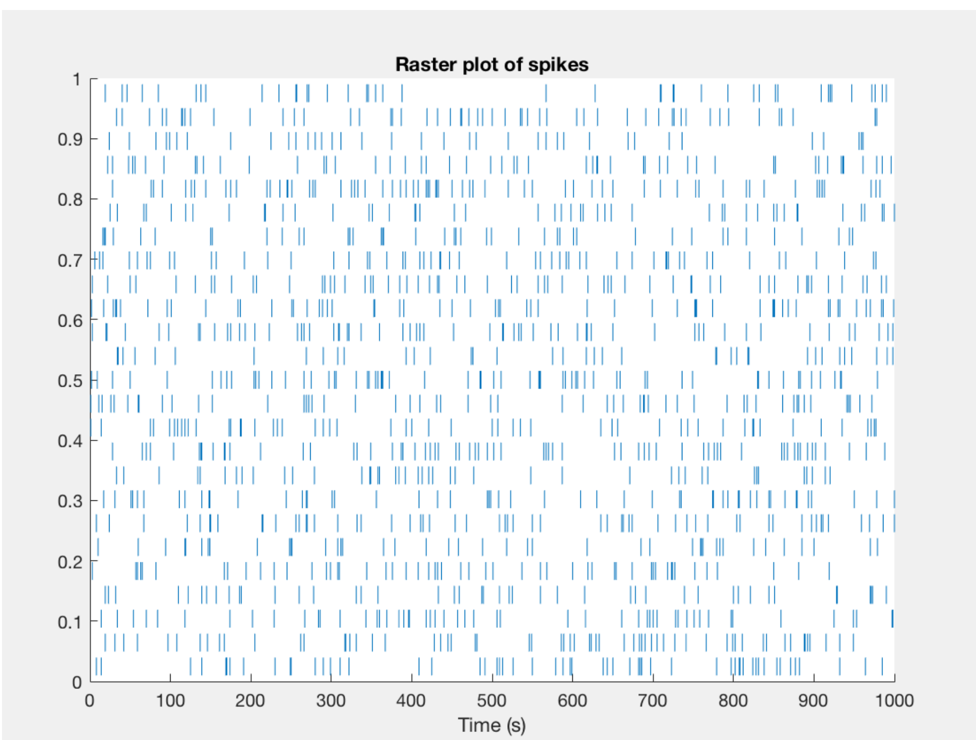


Figure 3.2: Raster plot of spikes using Poisson neuron model

# Chapter 4

# ADAPTIVE EXPONENTIAL INTEGRATE-AND-FIRE MODEL

This is a spiking neuron model with two relational equations. Equation(4.1) describes the dynamics of the membrane potential that includes the activation term that shows the exponential voltage dependence. Equation(4.2) describes the adaptive dependence of voltage.[4]

$$C\frac{dV}{dt} = -g_L(V - E_L) + g_L\Delta_T \exp(\frac{V - V_T}{\Delta_T}) - w + I \tag{4.1}$$

$$\tau_w\frac{dw}{dt} = a(V - E_L) - w \tag{4.2}$$

where,

V = Membrane Potential
w = Adaptation variable
I = Input current
C = Membrane Capacitance
$g_L$ = Leaky conductance
$E_L$ = Leaky reversal potential
$V_T$ = Threshold potential
$\Delta_T$ = Slope factor
a = Adaptation coupling parameter
$\tau_w$ = Adaptation time constant

According to equations(4.1 and 4.2) this is a nonlinear exponential model that generates spikes when membrane potential diverges towards infinity. Though this is not a very implementable outcome so the membrane potential is usually set to a finite value and data is recorded accordingly. The downswing in action potential is not described in the equations and is attained by resetting it to a fixed voltage simultaneously increasing the adaption value by certain amount(b). Every-time action potential is triggered both the equations resets.

$$At \quad t = t^f \quad reset \quad V \rightarrow\ V_r \quad and \quad w \rightarrow\ w + b \tag{4.3}$$

## 4.1 First Differential Equation

Equation 4.1 expresses the balance between the capacitive current $C\frac{dV}{dt}$, injected current $I$ and membrane currents. Membrane current composes of three parts:

1. **Leak Current :** $-g_L(V - E_L)$, This is linear in voltage and increases with distances from resting potential $E_L$.
2. **Exponential Current :** $g_L\Delta_T \exp(\frac{V-V_T}{\Delta_T})$, This describes the voltage dependent activation of sodium channel with assumption that activation is instantaneous.
3. **Adaptation Current :** $-w$, This describes the relation between the predictive and observed outcome.

## 4.2 Second Differential Equation

Equation 4.2 expresses the evolution of adaptation current with time constant $\tau_w$. This contains two parts:

1. **Linear Coupling :** $a(V - E_L)$, This coupling via $a$ can be described as a sub-threshold adaptation. For different values of $a$ the behaviour of coupling changes.
   - For ($a > 0$), this might hyper-polarize the membrane.
   - For ($a < 0$), this might de-polarize the membrane.
2. **Spiked Trigger Adaptation :** $-w$, This represents the reset of adaptation value $w \rightarrow w + b$ when action potential is triggered.

## 4.3 Biological interpretation of AdEx Model

- Without adaptation $V_T$ is the maximum voltage obtained at constant current injection but adding adaption factor this voltage can be shifted according to the desired output.
- The slope factor $\Delta_T$ defines the sharpness of spikes. If $\Delta_T \rightarrow 0$ than this model becomes the normal integrate-and-fire model with fixed threshold voltage $V_T$.
- Spike triggered adaptation(parameter $b$) summarizes the effect of calcium dependent potassium channels under the assumption that calcium influx occurs during action potential.
- Coupling or voltage and adaptation(parameter $a$) also contribute to spike triggered adaptation because of sharp rise of voltage during upswing of action potential.

# Chapter 5

# SIMULATE AdEx MODEL

The AdEx model described above consist of two equation and adjusting the parameters of that model gives different neuron firing patterns. In this simulation we have given a basic Matlab implementation with three different conditions.[9, 3, 2, 8]

```
function [V,w,St] = aEIF(t,I,C,gL,EL,Vt,Vp,Vr,Dt,tauw,a,b,initV,initw,pflag)
C     = C*1e-9;                                  %Membrane Capacitance
gL    = gL*1e-9;                                 %Leaky conductance
EL    = EL*1e-3;                                 %Leaky reversal potential
Vt    = Vt*1e-3;                                 %Threshold potential of spike
Vp    = Vp*1e-3;                                 %peak potential at spike
Vr    = Vr*1e-3;                                 %reset potential after spike
Dt    = Dt*1e-3;                                 %slope factor
a     = a*1e-9;                                  %adaptation coupling parameter
b     = b*1e-9;                                  %current increament parameter
tauw  = tauw*1e-3;                               %adaptation time constant
I     = I*1e-9;                                  %Injected current
initV = initV*1e-3;                              %initial membrane potential
initw = initw*1e-9;                              %initial adaptation current
dt    = t(2)-t(1);                               %time step
nNeurons = size(I,2);                            %number of neurons
[V,w,Sb] = deal(zeros([length(t) nNeurons]));
V(1,:)   = initV;                                %membrane potential vector
w(1,:)   = initw;                                %adaptation current vector
```

```
%loop over the time duration
for i = 1 : length(t)-1
    %updates membrane potential and adds to vector V
    V(i+1,:) = V(i,:) + dt/C*(-gL*(V(i,:)-EL) + gL*Dt*exp((V(i,:)-Vt)/Dt) - w(i,:) + I(i,:));
    %check for spikes
    spikes = V(i+1,:) > Vp;
    %update adaptation current
    w(i+1,:) = w(i,:) + dt/tauw*(a*(V(i,:)-EL) - w(i,:)) + spikes*b;
    V(i,spikes)   = Vp; %set spike potential
    V(i+1,spikes) = Vr; %reset membrane potential
    Sb(i,spikes)  = 1;  %set spike bits
end

%loop to check number of times spikes occured.
if nNeurons == 1
    St = find(Sb)*dt;
else
    St = cell([nNeurons 1]);
    for i = 1: nNeurons
        St{i} = find(Sb(:,i))*dt;
    end
end
```

```
%plot the results
if pflag
    figure, set(gcf,'Color',[1 1 1])
    subplot(6,1,1:3)
    plot(t,V,'-r')
    title('AdEx LIF neuron model')
    ylabel('Voltage(V))')
    axis([t([1 end]) -100 30])
    subplot(6,1,4:5)
    plot(t,w,'-k')
    ylabel('w')
    axis([t([1 end]) min(w(:))-0.02 max(w(:))+0.02])
    subplot(6,1,6)
    plot(t,I,'-b')
    xlabel('time'); ylabel('I')
    axis([t([1 end]) min(I(:))-0.02 max(I(:))+0.02])
end
```

Figure 5.1: Code snippets of AdEx model implementation

## 5.1 Condition 1 : Constant Current Injection

The neuron pattern observed with this input is called *tonic burst pattern.*

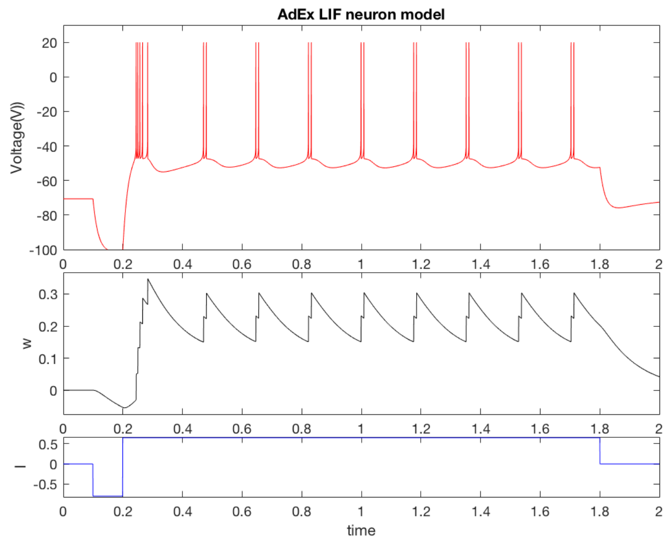


Figure 5.2: Voltage plot over time with constant injected current and adaptation current.

## 5.2 Condition 2 : Hyper-polarizing current injection

In this phenomenon spike is formed when a late arriving sub-threshold excitation input is assisted by an early inhibitory input. The inhibitory input causes a reduction of spike threshold at the time of arrival of the sub-threshold excitation. In this adaptation time constant $\tau_w$ is very high. This phenomena is also referred as *Post-inhibitory rebound*.

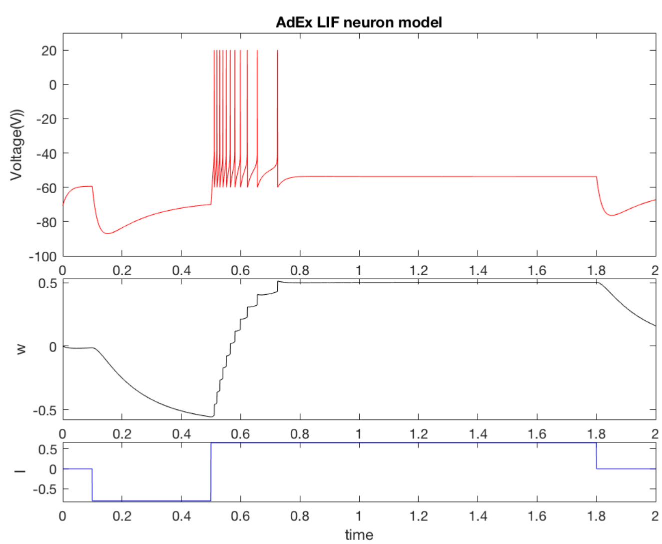


Figure 5.3: Voltage plot over time with hyper-polarizing current and adaptation current.

## 5.3 Condition 3 : Step-current Injection

In this phenomena the current is passed in step form and this leads to sub-threshold oscillation and resonant phenomena. Higher the value of $a$(adaptation coupling parameter) more sub-threshold oscillations occur. As the step size of current decrease so does the neuron spike threshold. This phenomena is also referred as *Damped oscillations.*

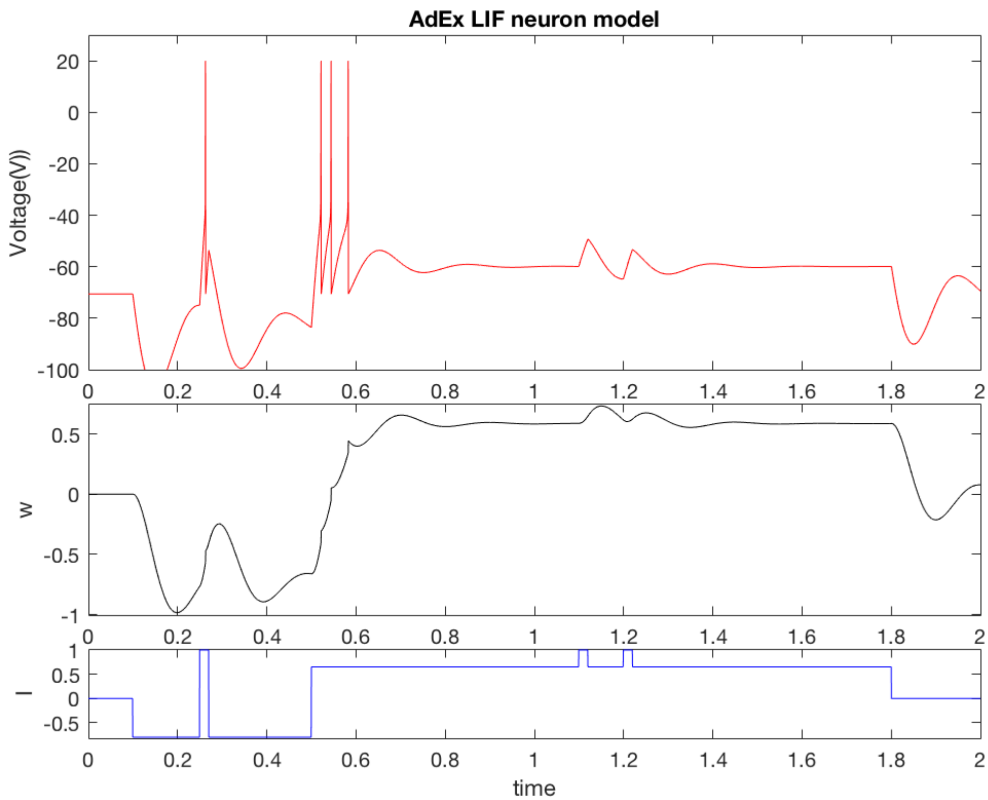


Figure 5.4: Voltage plot over time with step-current injection and adaptation current.

# Chapter 6

# ADVANTAGES AND DISADVANTAGES OF AdEx vs LEAKY INF

## 6.1 ADVANTAGES

- Upswing in the action potential is more realistic compared to the real neuron in AdEx model because of the exponential voltage dependence, while in leaky integrate-and-fire model action potential is not explicitly described and is more dependent on firing rate and threshold voltage with the reset factor in consideration.
- The choice of voltage threshold in AdEx model is determined with the slope factor $\Delta_T$ thus defining the sharpness of the spikes when action potential is triggered while in leaky integrate-and-fire model the threshold voltage plays the major role to reset the potential but does not affect the spike sharpness as $\Delta_T \rightarrow 0$.
- Multiple experiments showed that the voltage dependence in real neuron is actually a linear and exponential combination of terms which is represented in AdEx model while in leaky integrate-and-fire model shows only a linear voltage dependence.
- After every fire of a current in neuron AdEx model uses adaptation parameters to adapt and learn the potential growth thus the upswing in the curve while the leaky integrate-and-fire model resets the potential after each spike thus only the most recent spike is stored in memory making it in-adaptable model.

## 6.2 DISADVANTAGES

- In understanding the behaviour of a single neuron Leaky integrate-and -fire model proves more effective due to the sheer simplicity and linear nature while in the AdEx model due to its adaptive nature the model gives a complicated analysis to reach a same conclusion thus increasing a computation time significantly.

- Adding a simple feature of adaptation in the Leaky integrate-and-fire model gives a very accurate model for mapping the behaviour of complex neuron structure with simplified parameters compared to AdEx model.
- For large scale network simulations leaky integrate-and-fire model works better than AdEx because of its mathematical simplicity and computational efficiency.
- AdEx model itself is not able to capture the sudden decrease or non-monotonicity in firing rates as the driving current gets very high. On the other hand the firing rates plays the major role in the leaky integrate-and-fire model as the voltage dependence in linearly affected by firing rate.

**To summarize the AdEx model do allow fast fitting and better retention of data from previous neuron fire thus giving a very accurate result. But in case of leaky integrate-and-fire model due to sheer simplicity of the model the computational efficiency of this model for large networks is more preferred choice.[1, 9, 4, 7, 6]**

# Bibliography


[1] Brette, R., and Gerstner, W. Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of neurophysiology 94*, 5 (2005), 3637–3642.

[2] Dodla, R. Postinhibitory rebound and facilitation. *Encyclopedia of Computational Neuroscience* (2015), 2468–2471.

[3] Fourcaud-Trocmé, N., Hansel, D., Van Vreeswijk, C., and Brunel, N. How spike generation mechanisms determine the neuronal response to fluctuating inputs. *Journal of Neuroscience 23*, 37 (2003), 11628–11640.

[4] Gerstner, W., and Brette, R. Adaptive exponential integrate-and-fire model. *Scholarpedia 4*, 6 (2009), 8427.

[5] Gerstner, W., and Kistler, W. M. *Spiking neuron models: Single neurons, populations, plasticity.* Cambridge university press, 2002.

[6] Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. *Neuronal dynamics: From single neurons to networks and models of cognition.* Cambridge University Press, 2014.

[7] Hertäg, L., Hass, J., Golovko, T., and Durstewitz, D. An approximation to the adaptive exponential integrate-and-fire neuron model allows fast and predictive fitting to physiological data. *Frontiers in computational neuroscience 6* (2012), 62.

[8] Makhlooghpour, A., Soleimani, H., Ahmadi, A., Zwolinski, M., and Saif, M. High accuracy implementation of adaptive exponential integrated and fire neuron model. In *Neural Networks (IJCNN), 2016 International Joint Conference on* (2016), IEEE, pp. 192–197.

[9] Naud, R., Marcille, N., Clopath, C., and Gerstner, W. Firing patterns in the adaptive exponential integrate-and-fire model. *Biological cybernetics 99*, 4-5 (2008), 335.