SIMULATION OF OSCILLATIONS IN NEURONAL NETWORKS USING THE MORRIS-LECAR NEURON MODEL

CHAN SIOW CHENG

DOCTOR OF PHILOSOPHY IN ENGINEERING

LEE KONG CHIAN FACULTY OF ENGINEERING AND SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN
AUGUST 2015

SIMULATION OF OSCILLATIONS IN NEURONAL NETWORKS USING THE MORRIS-LECAR NEURON MODEL

By

CHAN SIOW CHENG

A thesis submitted to the Department of Mechatronics and
Biomedical Engineering,
Lee Kong Chian Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Engineering
August 2015

ABSTRACT

SIMULATION OF OSCILLATIONS IN NEURONAL NETWORKS USING THE MORRIS-LECAR NEURON MODEL

Chan Siow Cheng

Electrical recordings of brain activity show the presence of oscillations in different brain structures. Experimental and modeling studies of neuronal networks serve as the basis for exploration on the mechanisms of basic brain functions. In the first part of my work, a population density approach which is derived from the assumption of an infinite number of neurons is introduced to simulate the network activity of a large number of interconnected Morris-Lecar (ML) neurons. I demonstrated that the population density approach overcomes the limitation imposed by the large computation time required for direct simulations of a network of individual neurons when the number of neurons is very large. The use of the ML neuron model also enables more realistic simulations of the behavior of real neurons such as the activities of type I and type II neurons (integrators and resonators) that are not possible using the simple integrate-and-fire neuron model. For the second part of my work, I successfully simulated the ultra-slow oscillations that were observed experimentally in cortical cultures of rat brain neurons using a modified ML neuron model that takes into consideration the interaction between the neurons and the astrocytes in the network.

ACKNOWLEDGEMENTS

It is a great pleasure for me to acknowledge and appreciate the contribution of many individuals in completion of this study. First and foremost among these, I would like to express my deepest gratitude and thanks to my supervisor, Prof. Dato' Dr. Goh Sing Yau for his guidance, patience and continuous support of my PhD study and research. Being a great supervisor, he has provided me with an excellent atmosphere to develop my own idea and allow me to grow as a researcher. Without his valuable help and advice, this project would not be completed.

I would like to express my sincere appreciation to Prof. Roman R. Poznanski for his expertise, valuable ideas and inputs to this study. My research with him has exposed me to the valuable perspective of computational neuroscience.

Next my sincere thanks go to my research group members: Dr. Mok Siew Ying, Danny Ng Wee Kiat, Dr. Tan Lee Fan and Tan Yin Qing for their constructive comments and suggestions in our numerous discussions and interactions. I am also thankful to my friends and colleagues in UTAR for their assistance and friendship.

Last but not least, my special appreciation goes to my family members for their moral support and endless encouragement during my study. And also to my husband, who always cheer me up and stood by me all the times.

This research was supported by the Fundamental Research Grant Scheme under the Ministry of Higher education Malaysia and UTAR scholarship.

APPROVAL SHEET

This dissertation/thesis entitled "**SIMULATION OF OSCILLATIONS IN NEURONAL NETWORKS USING THE MORRIS-LECAR NEURON MODEL"** was prepared by CHAN SIOW CHENG and submitted as partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering at Universiti Tunku Abdul Rahman.

Approved by:

_____

(Prof. Dato' Dr. Goh Sing Yau)

Date:……..

Professor/Supervisor

Department of Mechanical and Material Engineering

Lee Kong Chian Faculty of Engineering and Science

Universiti Tunku Abdul Rahman

LEE KONG CHIAN FACULTY OF ENGINEERING AND SCIENCE

UNIVERSITI TUNKU ABDUL RAHMAN

Date: _____

SUBMISSION OF THESIS

It is hereby certified that **CHAN SIOW CHENG** (ID No: **08UEDO8107**) has completed this thesis entitled "**SIMULATION OF OSCILLATIONS IN NEURONAL NETWORKS USING THE MORRIS-LECAR NEURON MODEL**" under the supervision of **Prof. Dato' Dr. GOH SING YAU** (Supervisor) from the Department of Mechanical and Material Engineering, Lee Kong Chian Faculty of Engineering and Science.

I understand that University will upload softcopy of my thesis in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

_____

(CHAN SIOW CHENG)

DECLARATION

I hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

Name_____CHAN SIOW CHENG_____

Date _____

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

EEG	Electroencephalographic
PDA	Population density approach
ML	Morris-Lecar
IF	Integrate-and-fire
HH	Hodgkin-Huxley
LIF	Leaky integrate-and-fire
IFB	Integrate-and-fire-or-burst
PSP	Postsynaptic potential
ODE	Order differential equation
RMSE	Root-mean-square error
Ca^{2+}	Calcium ion
IP_3	Inositol 1, 4, 5-trisphosphate
ER	Endoplasmic reticulum
ATP	Adenosine triphosphate
AHP	Afterhyperpolarization
PLC	Phospholipase C

CHAPTER 1

INTRODUCTION

It is estimated that the human brain contains more than 10^{10} densely packed neurons that are interconnected to an intricate network (Gerstner and Kistler, 2002). Gray in his review (Gray, 1994) reported that a wide range of oscillatory patterns expressed at the level of individual cells as well as networks of cells is often exhibited in the nervous system. Brunel in his analytical study (Brunel and Hakim, 1999) described neuronal network activities of sparsely connected spiking neurons that displayed a rich repertoire of states including global activity that oscillates. It has also been reported that neuronal network oscillations with various rhythms serve different functional roles in many brain processes such as those involving cognitive functions, consciousness, behavioural activities, processing of sensory information and signal transmission (Wang, 2003). Therefore, the study of oscillatory mechanisms is a field of growing interest, both experimentally (Steriade et al., 1993; Buzsáki and Draguhn, 2004; Chen et al., 2006; Zhu et al., 2006; Compte et al., 2008) and theoretically (Destexhe et al., 1993; Wang and Buzsáki, 1996; Brunel and Hakim, 1999; Brunel, 2000; Tsodyks et al., 2000; Kudela et al., 2003; Melamed et al., 2008; Gielen et al., 2010; Buzsáki and Wang, 2012).

For modelling studies that approximate to the real situation where the neuronal network consists of a large number of neurons, computational efficiency and time become important (Nykamp and Tranchina, 2000; Reutimann et al., 2003; Marpeau et al., 2009). Also, it has been hypothesized that glial cells that are normally assumed to play only a supporting role to the neurons may actually play an important role in regulating the network global oscillations (Fellin et al., 2004; Volman et al., 2004; De Pittà et al., 2012). The above two issues are explored in the current study.

1.1 The Neuron Model

I have to choose a neuron model for my simulation studies. The integrate-and-fire (IF) neuron models that require the solution of only the membrane potential equation is one of the simplest and efficient spiking neuron models that is widely used in computational neuroscience. However, the model cannot reproduce the more complex properties and features of biological neurons. Hodgkin and Huxley (1952) developed the neuron model that is named after them by conducting experiments on the axon of a squid. Although the Hodgkin and Huxley (HH) model enables the simulation of more complex neuronal behaviours, it requires the solution of four simultaneous equations describing three ion channels and a leakage channel. Due to its complexity, HH model is computationally expensive especially when simulating a large number of interconnected neurons. Morris and Lecar (1981) reduced the HH model to a more manageable two dimensional model that only requires the

solution of two simultaneous equations - the membrane potential and a activation variable for the potassium current. The Morris-Lecar (ML) model has been reported to capture the essential features of neuronal dynamics and plays an important role in the study of neuronal rhythms and oscillations (Lim and Kim, 2007; Lim and Kim, 2011). This model is adopted in the current study.

1.2 Population Density Approach

To improve the computational efficiency of neuronal networks that consist of a large number of neurons, the population density approach (PDA) was introduced. Previous studies of PDA have mostly been developed for the IF models such as leaky integrate-and-fire (LIF) model (Omurtag et al., 2000), integrate-and-fire conductance based model (Nykamp and Tranchina, 2000; Haskell et al., 2001; Nykamp and Tranchina, 2001), and integrate-and-fire-or-burst (IFB) model (Casti et al., 2002; Apfaltrer et al., 2006; Huertas and Smith, 2006a; Huertas and Smith, 2006b). The first part of this thesis describes the PDA that is developed to simulate the network activity of an infinite number of ML neurons.

1.3 Simulation of Ultra-Slow Oscillations

The frequency of network oscillations can be manifested in three main categories: higher frequency band (>40Hz), slow oscillations (< 1Hz) during

sleep-wave sleep, and ultra-slow oscillations in between 0.001Hz and 0.01Hz. Most previous studies have focused on network oscillations in the range of traditional EEG frequency bands (>0.5 Hz). These studies explored many significant factors in these network oscillations including the presence of noise (Reinker et al., 2006; Nesse et al., 2008; Kilpatrick and Bressloff, 2010), the role of endogenously active cells (Latham et al., 2000), shunting inhibition (Vida et al., 2006; Talathi et al., 2010), as well as the balance of excitation and inhibition in the networks (Bazhenov et al., 2008; Liu et al., 2010).

In recent years, however, many researchers start to focus on brain activities in a much slower time scale such as ultra-slow oscillations that have been identified in various brain regions such as in the cortex (Picchioni et al., 2011), hippocampus (Penttonen et al., 1999; Zhu et al., 2010), and thalamus (Lőrincz et al., 2009). The potential role of ultra-slow oscillations in neural activity is discussed in relation to the communication between the brain, spinal cord and vegetative system (Başar, 2011), the establishment of activity-dependent synaptic connectivity (Feller, 1999) and synaptic plasticity(Allers et al., 2002).

Such slow oscillatory dynamics in neuronal networks have also been investigated in recent computational studies. A mean field model in cortical default network was proposed to predict the naturally arising of these slow cycling of cortical activity (Steyn-Ross et al., 2011). Ng et al. (2013) simulated the ultra-slow oscillations similar to those observed by Mok et al. (2012) in a study of rat brain cortical cells cultured in multi-electrode arrays

using the IF neuron model. However, the neural mechanisms underlying the ultra-slow oscillations are still under-investigated.

It has recently been revealed that glial cells may be involved in processing information and modulating neuronal dynamics in the brain (Hirase et al., 2004; Poskanzer and Yuste, 2011). A few experimental studies suggested the possible involvement of glial cells in producing slow oscillatory phenomena (Hughes et al., 2011; Krueger et al., 2011). In the second part of this thesis, I present a computational model that takes into consideration the bi-directional interaction between neurons and glial cells in the network that successfully simulated the ultra-slow oscillations that was observed in cortical cultures of rat brain neurons (Mok et al., 2012).

The remaining chapters of this thesis are organized as follow:

Chapter 2 presents the description and corresponding equations of the PDA using the ML neuron model. To test the accuracy and computational efficiency of PDA, the simulation results are compared against conventional direct simulation for groups of individual neurons in a few network examples.

Chapter 3 describes a modified ML neuron model that takes into consideration the bi-directional communication between neurons and glial cells. The role of glial cells in modulating the frequency of neuronal network activities is investigated. The simulation results are compared with the experimental results that are observed in cortical cultures of rat brain neurons.

Chapter 4 presents the overall conclusions. Some future works are also suggested in this chapter.

CHAPTER 2

SIMULATION OF NETWORK OSCILLATIONS USING THE POPULATION DENSITY APPROACH

A population density approach (PDA) is presented to simulate the global activity of a network of Morris-Lecar (ML) neurons. The network is composed of identical excitatory and inhibitory ML neurons. Each neuron randomly receives excitatory and inhibitory connections from other neurons in the network and an excitatory external input which is described by an independent Poisson process from neurons outside the network. I solved the evolution equations for the population density approach numerically. The results were compared against those obtained from conventional computation for groups of individual neurons in a few example networks. I found that when the neuronal network comprises a large number of identical excitatory ML neurons that are sparsely connected, the population density approach gives a closer approximation to the network activity. I also demonstrated that the population density approach using the ML neuron model can be used to simulate the activities of type I and type II neurons (integrators and resonators) in a network of sparsely connected excitatory and inhibitory neurons that was not possible using the integrate-and-fire neuron model.

---

* Published as: Chan, S.C., Poznanski, R.R. and Goh, S.Y., 2014. Network activity in a Morris-Lecar population density model. *Neurocomputing*, 138, pp. 332-338.

2.1 Introduction

Previous studies of somatosensory, visual cortex (Mountcastle, 1957; Hubel and Wiesel, 1962) and pools of motor neurons (Eric R. Kandel, 1991) showed that in many parts of the brain, neurons are structured in units with similar properties. Therefore it is convenient to describe the mean population activity rather than the spiking of single neurons. For networks that consist of a large number of neurons, the fraction of neurons i with membrane potential $v < V(t) < v + dv$ is approximated as

$$\lim_{N\to\infty}\left(\frac{neurons\ with\ v < V(t) < v + dv}{N}\right) = \int_{v}^{v+dv} \rho(v,t)dv$$

where $\rho(v,t)$ is the membrane potential probability density.

The PDA overcomes the limitation imposed by the large computation time required for simulations of a network of neurons when the number of neurons becomes very large. The computation time for PDA is dependent on the number of interacting populations rather than the number of neurons (Apfaltrer et al., 2006).

This chapter is organized as follows. The literature review of PDA, Type I and Type II neurons are discussed in section 2.2. The methodology is given in section 2.3. The network architecture is described in section 2.3.1 and the direct simulation of the conductance-based ML neuron model in section 2.3.2. In section 2.3.3, I introduced the PDA for the ML neuron model and derived the corresponding PDA equations. The numerical algorithm for

solving the PDA equations is presented in section 2.3.4. I presented the results of a single uncoupled population of type II neurons in section 2.4.1. The effects of the number of connections on the network behaviour is investigated and discussed in section 2.4.2. The performance of PDA is also tested by varying the postsynaptic potential (PSP) for inhibitory synapses in section 2.4.3.1. The simulation for Type I and Type II neurons are presented in section 2.4.3.2. Section 2.4.4 shows the comparison of computation time between the PDA and the direct simulation of a network of ML neurons. New implications and advances in the study of neural systems are stated in section 2.5. Section 2.6 is the conclusion of this chapter.

2.2 Literature Review

2.2.1 Population Density Approach (PDA)

The population density approach (PDA) has been used to study the network behaviour of a large number of identical IF neurons which have similar biophysical properties. Most of the previous studies focused one-dimensional PDA for the leaky integrate-and-fire (LIF) model (Omurtag et al., 2000) and the integrate-and-fire conductance based model (Nykamp and Tranchina, 2000; Haskell et al., 2001; Nykamp and Tranchina, 2001). These were extended to studies of two-dimensional PDA for the integrate-and-fire-or-burst (IFB) model (Casti et al., 2002; Apfaltrer et al., 2006; Huertas and Smith, 2006a; Huertas and Smith, 2006b). In the limit of a small voltage jump, the

PDA can be reduced to the Fokker-Planck (diffusion) approximation to analyze the dynamics of the distribution of neuron potentials (Brunel and Hakim, 1999; Brunel, 2000; Mongillo and Amit, 2001; Reutimann et al., 2003; Wang and Jiao, 2006; Marpeau et al., 2009; Jiao and Wang, 2010). In order to describe the neuron dynamics precisely, a PDA that takes into consideration the effects of slow ionic currents was proposed by Chiznov et al. (Chizhov et al., 2006). They simulated the activity of a recurrent inhibitory neuron network with a constant current step input.

I am not aware of any previous computational study that has applied the PDA to a network that comprises a large number of identical ML neurons. The current study is expected to provide simulations of more complex properties of large networks of neurons.

2.2.2 Type I and Type II Neurons

The ML model can reproduce the integrator or the resonator neurons (hereafter termed as type I and type II neurons respectively) depending on the parameter of voltage-dependent potassium current (Tonnelier, 2005) while the IF neuron is an integrator. The integrator neuron exhibits saddle-node bifurcation when it transits between a rest state and repetitive firing state. In contrast, the resonator neuron exhibits Andronov-Hopf bifurcation (Izhikevich, 2000). With a higher frequency of input spike train, the integrators are more likely to fire whereas resonators tend to fire when the frequency of input spike train is similar with the frequency of sub-threshold membrane potential oscillations

(rhythmic fluctuations of the voltage difference between interior and exterior of neurons). Many cortical neurons are integrators while the resonator neuron behaviour has been formed in the thalamic (Puil et al., 1994) and cortical regions (Hutcheon et al., 1996b; Hutcheon et al., 1996a).

2.3 Methodology

2.3.1 Network Architecture

A network of interconnected excitatory and inhibitory populations is shown in Figure 2.1. There are N_{exc} and N_{inh} identical ML neurons in the excitatory and inhibitory populations respectively. Each population randomly receives q_{exc} excitatory connections and q_{inh} inhibitory connections from other neurons inside the network. It also receives q_{ext} external excitatory inputs with rate ν_{ext} from neurons outside the network. The total effect of the external network is treated as an external Poisson input. External spikes are statistically independent and approximated by a Poisson distribution (Brunel and Hakim, 1999; Brunel, 2000)

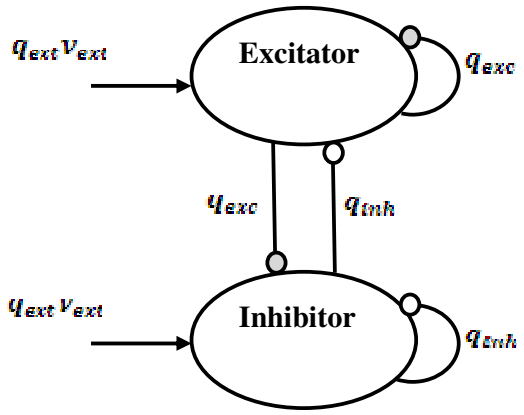

Figure 2.1: Schematic diagram of the network architecture. The network is composed of an excitatory population and an inhibitory population that interconnect with each other. Each population receives q_{ext} excitatory external input from neurons outside the network with rate ν_{ext}. The symbols q_{exc} and q_{inh} are the number of excitatory and inhibitory connections from neurons inside the network.

2.3.2 The Direct Simulation of the Conductance-based ML Neuron Model

The set of differential equations that governs the dynamics of the membrane potential for ML neuron i ($i = 1, 2, 3, \ldots, N_{exc} + N_{inh}$) is written as follows (Morris and Lecar, 1981):

$$C\frac{dV_i}{dt} = -I_{ion,i} + I_{syn,i} \tag{2.1}$$

$$\frac{dW_i}{dt} = \varphi\frac{(W_\infty(V_i) - W_i)}{\tau_R(V_i)} \tag{2.2}$$

where

$$I_{ion,i} = I_{Ca,i} + I_{K,i} + I_{L,i} \tag{2.3}$$

$$= g_{Ca}m_\infty(V_i - E_{Ca}) + g_K W_i(V_i - E_K) + g_L(V_i - E_L)$$

$$m_\infty(V_i) = 0.5[1 + tanh\{(V_i - V_1)/V_2\}] \tag{2.4}$$

$$W_\infty(V_i) = 0.5[1 + tanh\{(V_i - V_3)/V_4\}] \tag{2.5}$$

$$\tau_R(V_i) = \frac{1}{cosh\{(V_i - V_3)/(2V_4)\}} \tag{2.6}$$

Here, V_i is the membrane potential and W_i is the activation variable for the potassium current. There are two kinds of source currents to each neuron, $I_{ion,i}$ and $I_{syn,i}$. $I_{ion,i}$ is the total ionic current that consists of K^+ current, $I_{K,i}$, Ca^{2+} current, $I_{Ca,i}$ and a leakage current, $I_{L,i}$ and C is the membrane capacitance. The maximum conductance for the ions and the leakage channels are denoted by g_{Ca}, g_K and $g_L$whereas E_{Ca}, E_K and E_L represent the reversal potentials for the ions and the leakage channels. W_i tends to the saturation value $W_\infty(V_i)$ with a characteristic time scale of $(\tau_R(V_i)/\phi)$ where $\tau_R(V_i)$ is associated with the relative time scales of firing dynamics, which varies broadly from cell to cell and exhibits significant of temperature dependency. Fast changes of the calcium current take the gate variable m_i as the saturation value $m_\infty(V_i)$.

When the pre-synaptic neuron j $(j = 1, 2, 3, \dots, q_{ext} + q_{exc} + q_{inh})$ fires at time t, the potential of the connected postsynaptic neuron i is increased or decreased by potential (PSP) amplitude J_{ij}. For simplicity, I assume that $J_{ij} = J_{exc}$ for excitatory synapses and $J_{ij} = J_{inh}$ for inhibitory synapses. The synaptic current of the ith neuron is described as follows:

$$RI_{syn,i} = \tau_{syn} \sum_j J_{ij} \sum_k \delta\,(t - t_j^k) \tag{2.7}$$

where $\tau_{syn} = RC$ is the synaptic time constant, R is membrane resistance and t_j^k is the time of the kth spike on neuron j. When V_i crosses the threshold value V_{th}, neuron i emits a spike.

2.3.3 The PDA for ML Neuron Model

A PDA is introduced to represent the membrane behaviour of a large number of identical ML neurons described in the previous section (Nykamp and Tranchina, 2000; Huertas and Smith, 2006a; Huertas and Smith, 2006b),

$$\rho(v,w,t)dvdw = Pr\{V(t)\epsilon(v, v+dv)\ and\ W(t)\epsilon(w, w+dw)\} \tag{2.8}$$

for $v\ \epsilon\ (V_{min}, V_{max})$ and $w\ \epsilon\ (0,1)$.

The evolution equation for the probability of finding the membrane potential of a randomly chosen neuron in population of $a = exc$ (excitatory), inh(inhibitory) at v over all possible states at time t is based on conservation of probability (Nykamp and Tranchina, 2000):

$$\frac{\partial}{\partial t}\rho(v,w,t) = -\nabla \cdot \vec{J}(v,w,t) \tag{2.9}$$

where $\nabla = \hat{e}_v(\partial/\partial v) + \hat{e}_w(\partial/\partial w)$ and $\vec{J}(v,w,t)$ is the total probability flux crossing v and w at time t. The total probability flux consists of two components:

$$\vec{J}(v,w,t) = \vec{J}_{intrinsic}\,(v,w,t) + \vec{J}_{extrinsic}\,(v,w,t) \tag{2.10}$$

$\vec{J}_{intrinsic}(v,w,t)$ is the flux due to the intrinsic membrane dynamics:

$$\vec{J}_{intrinsic}(v,w,t) = [F_v(v,w)\hat{e}_v + F_w(v,w)\hat{e}_w]\rho(v,w,t) \tag{2.11}$$

where

$$F_v(v,w) = -\frac{I_{ion}}{C} \tag{2.12}$$

$$F_w(v,w) = -\varphi\frac{(w - w_\infty)}{\tau_R} \tag{2.13}$$

The flux, $\vec{J}_{extrinsic}(v,w,t)$ due to the synaptic input from external network and the connected neurons in the network is written in the form,

$$\vec{J}_{extrinsic}(v,w,t) = \vec{J}_{ext}(v,w,t) + \vec{J}_{exc}(v,w,t) - \vec{J}_{inh}(v,w,t) \tag{2.14}$$

As shown in Figure 2.2, when a neuron with voltage, $v = v'$ receives q_{ext} excitatory external input at rate of ν_{ext}, it could push v to higher voltages from any voltage $v'\epsilon(v - J_{ext}, v)$ and create a positive excitation flux:

$$\vec{J}_{ext}(v,w,t) = q_{ext}\nu_{ext}(t)\int_{v-J_{ext}}^{v}\rho(v',w,t)dv' \tag{2.15}$$

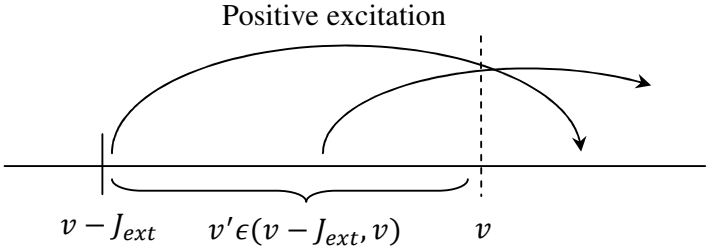

Figure 2.2: Positive excitation flux due to excitatory external input.

A neuron with voltage $v'\epsilon(v - J_{exc}, v)$ could generate another positive excitation flux across voltage v upon the arriving of q_{exc} excitatory internal with input rate of ν_{exc} from the connected neurons within the network (Figure 2.3):

$$\vec{J}_{exc}(v,w,t) = q_{exc}\nu_{exc}(t)\int_{v-J_{exc}}^{v}\rho(v',w,t)dv' \tag{2.16}$$

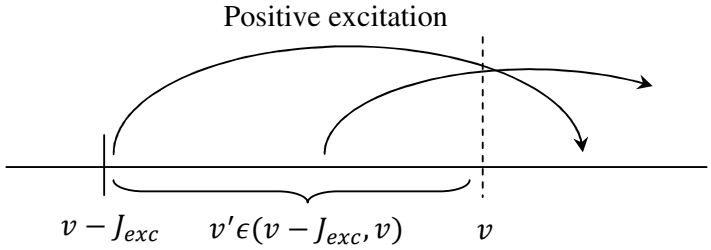

Figure 2.3: Positive excitation flux due to excitatory internal input.

Conversely if a neuron with voltage $v'\epsilon(v, v + J_{inh})$ receives q_{inh} inhibitory internal input with rate of ν_{inh} and crosses v to lower voltages, it could create a negative inhibition flux (Figure 2.4):

$$\vec{J}_{inh}(v,w,t) = q_{inh}\nu_{inh}(t)\int_{v}^{v+J_{inh}}\rho(v',w,t)dv' \tag{2.17}$$

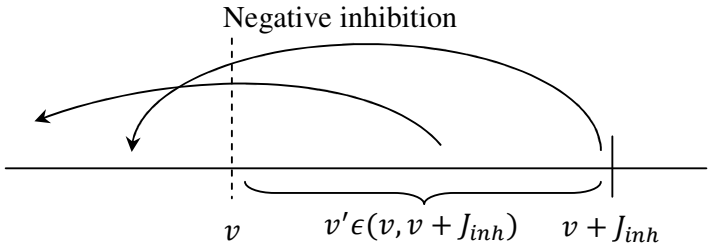

Figure 2.4: Negative inhibition flux due to inhibitory internal input.

When the upward movement of total probability flux, $\vec{J}$ crosses $v = V_{th}$, it corresponds to the fraction of neurons firing per unit time. Thus, the population firing rate that describes the average firing rate across all neurons in the population is obtained by integrating over all slow recovery

variables of the action of the potassium current, w':

$$v(t) = \int_0^1 \vec{J}_{spike}(w', t) \cdot \hat{e}_v \, dw' \tag{2.18}$$

where $\vec{J}_{spike}(w, t) = \vec{J}(v = V_{th}, w, t)$ when there are positive fluxes of $\vec{J}_{extrinsic}(v = V_{th}, w, t)$ and $\vec{J}_{intrinsic}(v = V_{th}, w, t)$.

The assumption of no probability flux across the boundaries at $w = 0, w = 1$ $v = V_{min}$ and $v = V_{max}$ leads us to the following boundary conditions associated with Equation 2.9,

$$\begin{aligned} \rho(v = V_{min}, 0 \le w \le 1, t) &= 0 \\ \rho(v = V_{max}, 0 \le w \le 1, t) &= 0 \\ \rho(V_{min} \le v \le V_{max}, w = 0, t) &= 0 \\ \rho(V_{min} \le v \le V_{max}, w = 1, t) &= 0 \end{aligned} \tag{2.19}$$

At any time t, the population density functions in Equation 2.9 with the boundary conditions above must satisfy the conservation of probability law,

$$\int_{V_{min}}^{V_{max}} \int_0^1 \rho(v', w', t) dv' dw' = 1 \tag{2.20}$$

2.3.4 Numerical Algorithm for PDA

To solve the model equations in the PDA, I discretize $(\partial \vec{J}/\partial v)$, $(\partial \vec{J}/\partial w)$ and solve the resulting set of ODEs using Runge-Kutta 4th order. For $k = 1, 2, \ldots, N_v$, the membrane voltage is discretized as

$$v_k = k\Delta v + V_{min} \tag{2.21}$$

where $\Delta v = \frac{(V_{max} - V_{min})}{N_v}$.

Whereas for $n = 1, 2, \ldots, N_w$, the discretization of gating variable for potassium is shown as

$$w_n = n\Delta w \tag{2.22}$$

where $\Delta w = \frac{1}{N_w}$

Using the Equation 2.14, the spatial derivatives of $\vec{J}_{extrinsic}$ at the grid points is given as

$$\begin{aligned} \frac{\partial \vec{J}_{extrinsic}}{\partial v} &= \frac{\partial\{\vec{J}_{ext}(v,w,t) + \vec{J}_{exc}(v,w,t) - \vec{J}_{inh}(v,w,t)\}}{\partial v} \\ &= q_{ext}\nu_{ext}(t)\{\rho(v,w,t) - \rho(v - J_{ext}, w, t)\} \\ &\quad + q_{exc}\nu_{exc}(t)\{\rho(v,w,t) - \rho(v - J_{exc}, w, t)\} \\ &\quad - q_{inh}\nu_{inh}(t)\{\rho(v + J_{inh}, w, t) - \rho(v,w,t)\} \end{aligned} \tag{2.23}$$

To improve stability, a downward and upward scheme is employed to discretize the flux due to the intrinsic membrane dynamics in the PDA. $(\partial \vec{J}_{intrinsic}/\partial v)$ is discretized as

$$\frac{\partial \vec{J}_{intrinsic}}{\partial v} = \frac{\partial}{\partial v}\left[-\frac{I_{Ca} + I_K + I_L}{C}\rho(v,w,t)\right] \tag{2.24}$$

$$= \frac{\partial}{\partial v}\left[-\frac{g_{Ca}m_\infty(v - E_{Ca}) + g_K w(v - E_K) + g_L(v - E_L)}{C}\rho(v,w,t)\right]$$

$$= \frac{\partial}{\partial v}\left[-\frac{1}{C}(av - b)\rho(v,w,t)\right]$$

$$= \frac{\partial}{\partial v}\left[-\frac{a}{C}\left(v - \frac{b}{a}\right)\rho(v,w,t)\right]$$

where

$$a = g_{Ca}m_\infty + g_K w + g_L$$

$$b = g_{Ca}m_\infty E_{Ca} + g_K w E_K + g_L E_L \tag{2.25}$$

A neuron with $v < (b/a)$ moves probability upwards creating positive fluxes while a neuron with $v > (b/a)$ moves probability downwards, creating negative fluxes (Figure 2.5). To ensure stability, a downward first order approximation is used for the derivatives for $v < (b/a)$,

$$\frac{\partial \vec{J}_{intrinsic}}{\partial v} = \frac{F_v(v,w,t)\rho(v,w,t) - F_v(v - \Delta v,w,t)\rho(v - \Delta v,w,t)}{\Delta v} \tag{2.26}$$

and an upward first order approximation for $v > (b/a)$,

$$\frac{\partial \vec{J}_{intrinsic}}{\partial v} = \frac{F_v(v + \Delta v,w,t)\rho(v + \Delta v,w,t) - F_v(v,w,t)\rho(v,w,t)}{\Delta v} \tag{2.27}$$

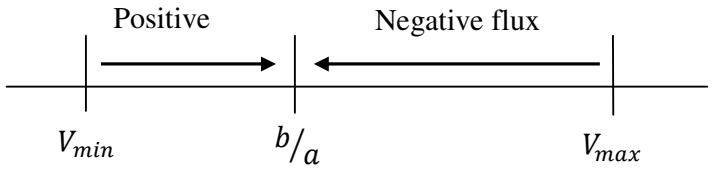

Figure 2.5: The positive and negative fluxes due to the intrinsic membrane dynamics, $\partial \vec{J}_{intrinsic}/\partial v$.

The discretization of the flux, $(\partial \vec{J}_{intrinsic}/\partial w)$ is denoted by

$$\frac{\partial \vec{J}_{intrinsic}}{\partial w} = \frac{\partial}{\partial w}[F_w(v,w,t)\rho(v,w,t)] \\ = \frac{\partial}{\partial w}\left[-\frac{\varphi}{\tau_R}\big(w - w_\infty(v)\big)\rho(v,w,t)\right] \tag{2.28}$$

For a neuron with $w < w_\infty(v)$, positive fluxes are created and a downward first order approximation is used:

$$\frac{\partial \vec{J}_{intrinsic}}{\partial w} = \frac{F_w(v,w,t)\rho(v,w,t) - F_w(v,w-\Delta w,t)\rho(v,w-\Delta w,t)}{\Delta w} \tag{2.29}$$

while an upward first order approximation is used for $w > w_\infty(v)$ (Figure 2.6),

$$\frac{\partial \vec{J}_{intrinsic}}{\partial w} = \frac{F_w(v,w+\Delta w,t)\rho(v,w+\Delta w,t) - F_w(v,w,t)\rho(v,w,t)}{\Delta w} \tag{2.30}$$

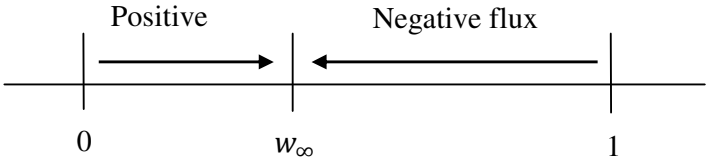

Figure 2.6: The positive and negative fluxes due to the intrinsic membrane dynamics, $\partial \vec{J}_{intrinsic}/\partial w$.

2.4 Results and Discussion

2.4.1 Single Uncoupled Population of Neurons Results

The root-mean-square error (RMSE) was computed to measure the difference between individual neuron histograms and the corresponding regions under the population density curve. Let $\hat{y}$ denotes the population of individual neuron results and y denotes the PDA results, then the RMSE is given as

$$RMSE(\hat{y}, y) = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y} - y)^2}{n}} \tag{2.31}$$

where n is the total number of data.

The response of a single excitatory population of ML neurons to the external Poisson input is simulated. Each neuron receives an excitatory external input, $q_{ext} = 1$ from neurons outside the network with mean firing rate, $\nu_{ext} = 200\ arrivals/ms$ and external postsynaptic potential, $J_{ext} = 0.1mV$. The other network parameters of type II neurons are shown in Table 2.1 (Balenzuela et al., 2006).

Table 2.1: Parameters for conductance based ML neuron model (Lim and Kim, 2007)

Parameter	**Description**	**Value for type II (type I)**
$\mathbf{V_1}$	Threshold value for m_∞	$-1.2\ mV$
$\mathbf{V_2}$	Steepness parameter for m_∞	$18\ mV$
$\mathbf{V_3}$	Threshold value for W_∞	$2\ mV\ (12mV)$
$\mathbf{V_4}$	Steepness parameter for W_∞	$30\ mV$
$\mathbf{E_{Ca}}$	Reversal potential for Ca^{2+} channels	$120\ mV$
$\mathbf{E_L}$	Reversal potential for leakage channels	$-60\ mV$
$\mathbf{E_K}$	Reversal potential for K^+ channels	$-84\ mV$
$\boldsymbol{C}$	Capacitance of membrane	$5\ \mu F/cm^2$
$\boldsymbol{\varphi}$	Temperature time scale factor	0.04
$\boldsymbol{g_{Ca}}$	Maximum conductance for Ca^{2+} channels	$4.4\ mS/cm^2$

$\boldsymbol{g_K}$	Maximum conductance for K^+ channels	8 mS/cm^2
$\boldsymbol{g_L}$	Maximum conductance for leakage channels	2 mS/cm^2
$\boldsymbol{\tau_{syn}}$	Synaptic time constant	1 ms
$\boldsymbol{V_{th}}$	Threshold value for spiking state	0 mV

The results obtained from the population density model are compared with those obtained from computations of populations of individual neuron for populations of 100, 1000 and 10,000 neurons in Figure 2.7. Figure 2.7(a-c) compares the firing rates during 50ms time period while Figure 2.7(d-f) compares the snapshots of the probability density across the membrane potential at time, $t = 25ms$. For the results of computations of populations of individual neurons, histograms of fixed bin size, 0.8mV and 0.02ms are used for the membrane potential distribution and firing rate respectively. Both the probability density across membrane potential and firing rates show that the error of the PDA decreases when the network comprises a large number of identical neurons. For an individual neuron population of 100 neurons, the membrane potential distribution and firing rate are sparse compared to the results obtained from the PDA with the average RMSE=0.0056 and 0.0486 respectively. Similar results were obtained for network size of 1000 neurons with the average RMSE=0.0031 and 0.0464. The membrane potential distribution and firing rate for 10,000 neurons compares well with the results obtained from the PDA with the average RMSE=0.0023 and 0.0279. In Figure 2.8, the corresponding temporal evolution of the PDA for membrane potential is shown with these three different sizes of population neurons. Figure 2.8(a)

shows the result of PDA whereas the results of individual neuron simulation with 10,000, 1000 and 100 neurons are shown in Figure 2.8(b-d).

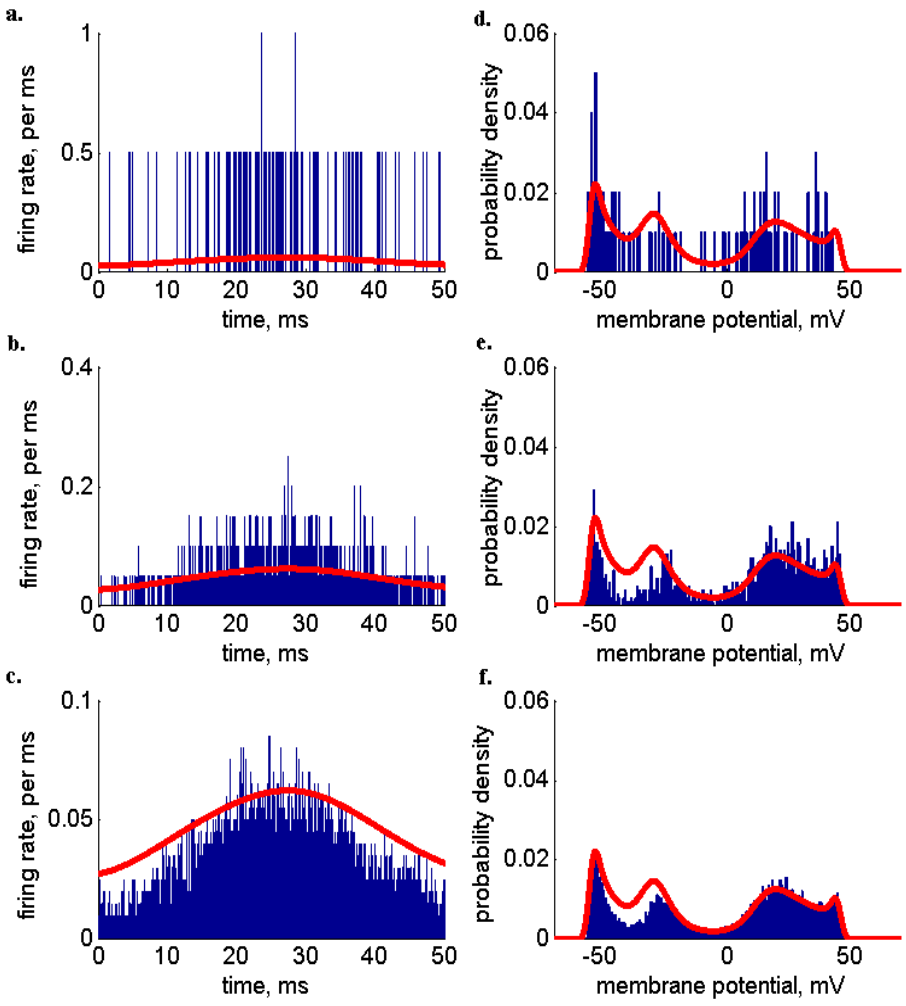

Figure 2.7: Comparison of the PDA with populations of individual neurons with three different population sizes: (a, d) 100 neurons, (b, e) 1000 neurons and (c, f) 10,000 neurons. The firing rates are shown in figures (a-c) and snapshots of probability density across membrane potential at time $t = 25\ ms$ are shown in figures (d-f). The red solid lines show the firing rates of the PDA results whereas histograms show the individual neuron results.

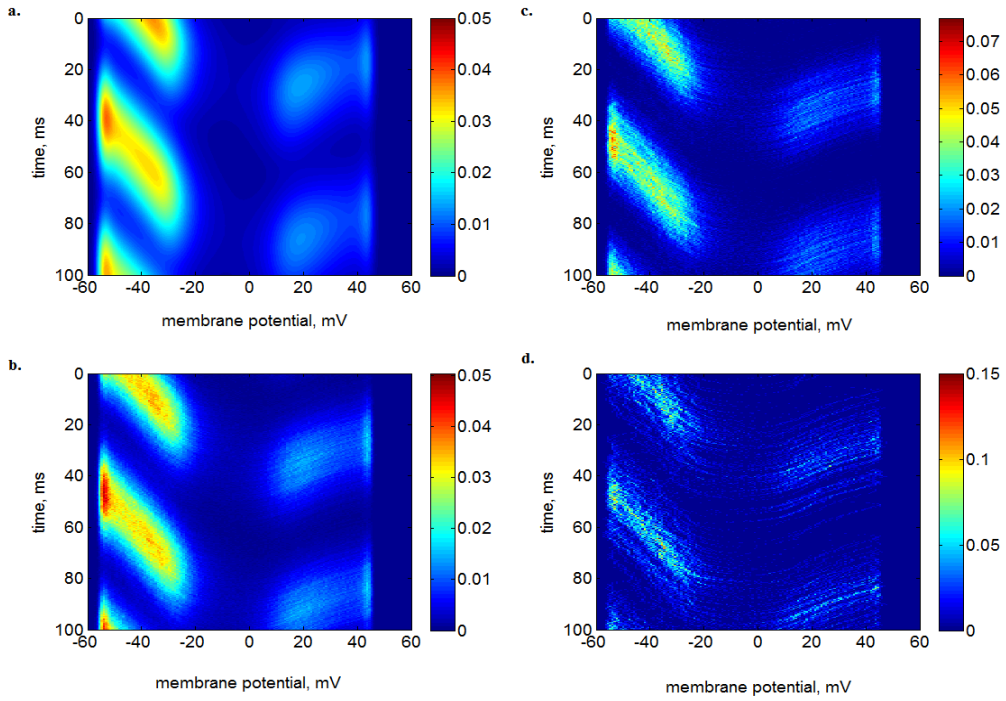

Figure 2.8: Temporal evolution of the membrane potential simulated with (a): PDA, (b): individual population of 10,000 individual neurons, (c): population of 1,000 individual neurons and (d): population of 100 individual neurons.

2.4.2 Single Coupled Population of Excitatory Neurons

I compared the results of the PDA with those computed using individual neuron populations for two different numbers of connectivity, $q_{exc} = 2500$ and 500 in Figure 2.9. Parameters used in the simulations were $N_{exc} = 10000$, $J_{exc} = 0.01mV$, $J_{ext} = 0.15mV$, $q_{ext} = 1$ and $\nu_{ext} = 120 arrivals/ms$. Other parameters are given in Table 2.1. In our simulation results, the smallest RMSE was obtained (0.0212 for RMSE of firing rate) in low connectivity of $q_{exc} = 500$ (Figure 2.9(b)). The PDA results appear closer to the results from those computed using individual neuron populations when the number of connections is small. When the number of connections is 2500, the results computed using individual neuron populations diverge from the PDA simulation (Figure 2.9(a)) with RMSE=0.0304. The sparse coupling lowers the probability of neurons sharing common inputs. This is justified in the PDA model when the input spike trains to each neuron are independent. Therefore, a large error is obtained for densely connected networks. The corresponding temporal evolution of the probability density for membrane potential is illustrated in Figure. 2.10.

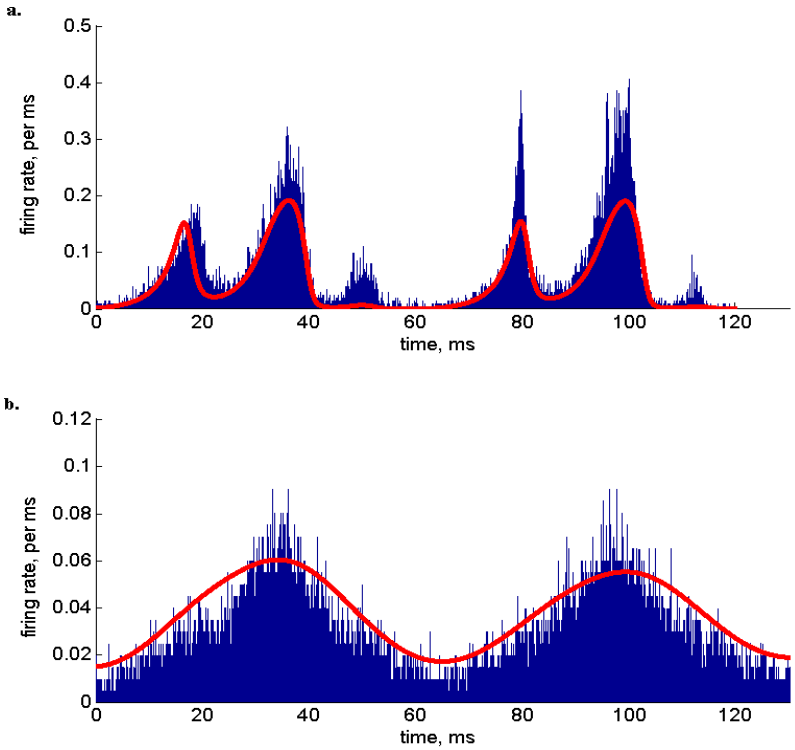

Figure 2.9: Comparison of the firing rate for the PDA with populations of individual neurons for two different connectivity: (a) $q_{exc} = 2500$ and (b) $q_{exc} = 500$. The red solid lines show the firing rates of the PDA results whereas histograms show the individual neuron results.

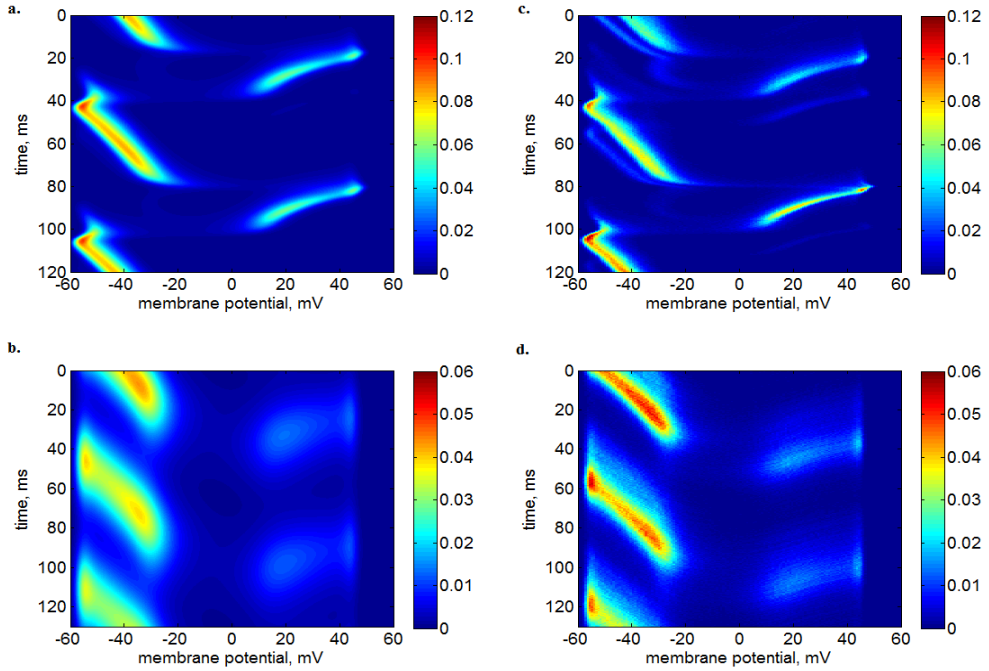

Figure 2.10: Temporal evolution of the probability density for membrane potential that simulated with connectivity of (a, c): $q_{exc} = 2500$, (b, d): $q_{exc} = 500$. The population density model results are shown in (a, b) whereas the individual neuron results are shown in (c, d).

2.4.3 Network of Excitatory and Inhibitory Neurons

2.4.3.1 Varying PSP Amplitude for Inhibitory Synapses

In this section, the simulations for a network of coupled excitatory and inhibitory neurons are presented. The PDA results are compared with those obtained from the direct simulation of a population of individual neurons by varying the postsynaptic potentials (PSP) amplitude for inhibitory synapses, J_{inh}. Parameters used in the simulations were $N_{exc} = 8000$, $N_{inh} = 2000$, $q_{exc} = q_{inh} = 500$, $J_{exc} = 0.01mV$, $J_{ext} = 0.15mV$, $q_{ext} = 1$ and $\nu_{ext} = 120\ arrivals/ms$. Other parameters are shown in Table 2.1. Both simulation results of firing rate calculated by the PDA and direct simulation show similar response as shown in Figure 2.11. With a higher level of J_{inh}, the network activity tends to become more disordered. At $J_{inh} = 0.01mV$, a slow oscillation with frequency around 15Hz was obtained (Figure 2.11(a)). When J_{inh} is increased to $0.1mV$, the network activity is damped and loses its synchronization as shown in Figure 2.11(b).

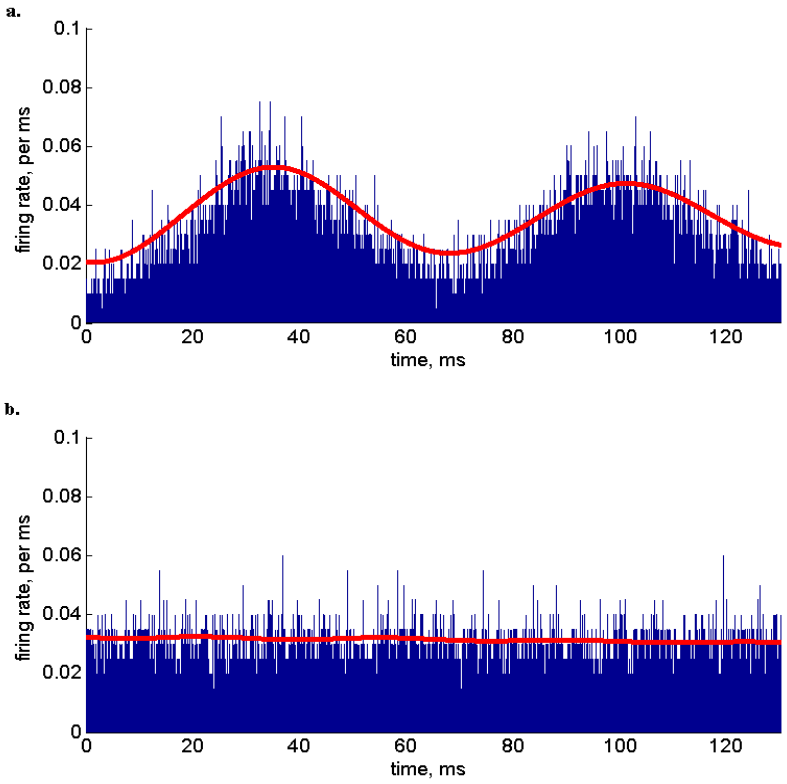

Figure 2.11: Comparison of the firing rate for PDA with populations of individual neurons for two different PSP for inhibitory synapses, (a) $J_{inh} = 0.01mV$ and (b) $J_{inh} = 0.1mV$. The red solid lines presented the firing rates of the PDA results whereas histograms presented the individual neuron results.

2.4.3.2 Simulation for Type I and Type II Neurons

The performance of the PDA is also tested for simulating type I and type II neurons. Parameters used in the following simulations were $J_{exc} = J_{inh} = 0.01mV, J_{ext} = 0.1mV, N_{exc} = 8000, N_{inh} = 2000, q_{ext} = 1, q_{exc} = 500\ and\ q_{inh} = 200$. Other network parameters are specified in Table 2.1. By changing the potassium activation curve, the ML model can reproduce type I and type II neurons behavior. The comparison of the results of firing rate from the PDA with those obtained from direct simulation of a population of individual neurons for network of type I and II neurons is illustrated in Figure 2.12. Type I and type II neurons are driven by external synaptic input with $\nu_{ext} = 120\ arrivals/ms$ (Figure 2.12(a)) and $180\ arrivals/ms$ (Figure 2.12(b)) respectively. Both showing good similarity between the results with average RMSE=0.0108 and 0.0199.

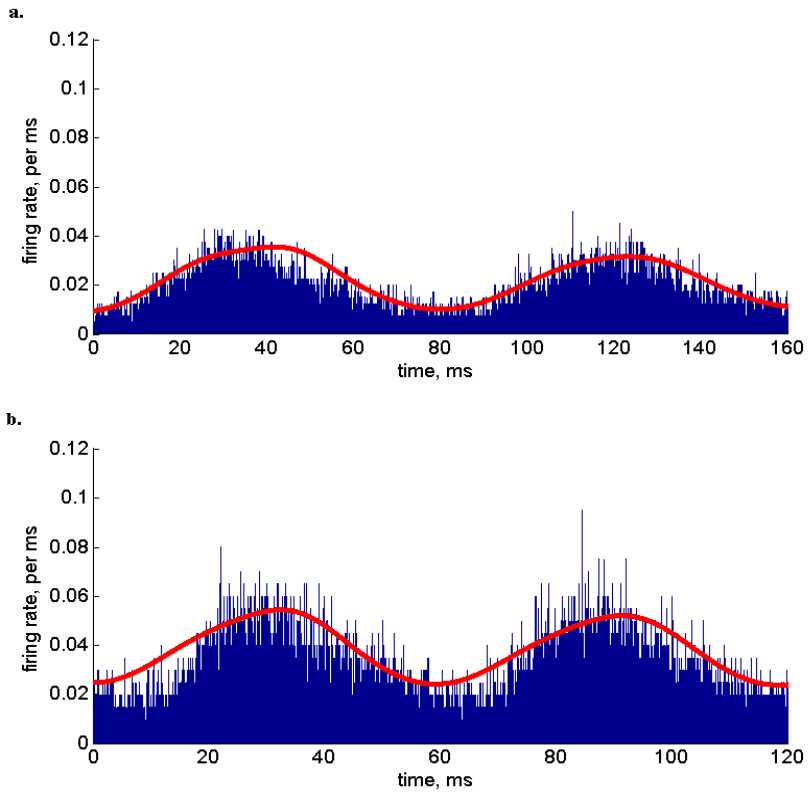

Figure 2.12: Comparison of the firing rate for the PDA with populations of individual neurons for networks of (a) type I and (b) type II neurons. The red solid lines show the firing rates of the PDA results whereas histograms show the individual neuron results.

2.4.4 Comparison of Computation Time

The comparison of computation time for the PDA with direct simulation of a network of ML neurons is shown in Figure 2.13. The network consists of M subnetworks. Each subnetwork has 8000 and 2000 ML neurons in excitatory and inhibitory populations respectively. Each neuron in the population receives synaptic input from randomly chosen neurons inside the subnetwork with probability 0.1 and an external Poisson synaptic input. The computation time shown in the $y-axis$ is relative to the computation time of a single subnetwork of direct simulation with time step of 0.01ms. All the neural network simulations are computed by a quad 2.8GHz processor workstation. The PDA outperforms direct simulation especially when the number of subnetworks is increased.

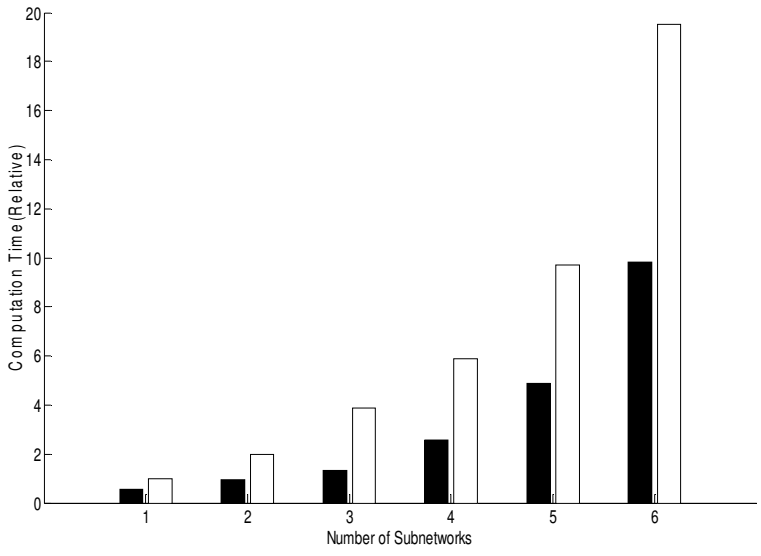

Figure 2.13: Comparison of computation time for the PDA (black bars) with populations of individual neurons (white bars).

2.5 New Implications and Advances in the Study of Neural Systems

The present approach can be applied to study more realistic models that consist of complex combinations of subnetworks such as the neocortex of human brain. The neocortex can contain up to 28×10^9 neurons and a huge number of synapses of the order of 10^{12}. The cortical neurons are organized vertically into cortical columns. Each cortical column contains approximately 60,000 neurons which have certain sets of common static and physiological dynamic properties (Mountcastle, 1997). The present PDA is suitable for simulating the mean activity of each column using the interactions with adjacent columns as external inputs to the column. Also with the participation of a large network of biologically plausible type I and type II neurons (integrators and resonators respectively), the present PDA can be applied to study complex phenomena such as resonance and oscillations observed in many biological neurons in thalamic and cortical regions. Resonator neurons with low and high-frequency resonances support the thalamocortical delta-wave oscillations during deep sleep and high frequency rhythm during cognition (Hutcheon and Yarom, 2000). Furthermore resonator neurons also exhibit damped subthreshold oscillations of membrane potential that are significant in sustaining synchronized rhythmic activity (Lampl and Yarom, 1997).

2.6 Conclusion

I have introduced a PDA for modelling a network of ML neurons. In order to assess the accuracy, the simulations of the PDA and the direct simulation of individual neurons were compared based on the distributions of neurons across the membrane potential and population firing rates. For neuronal networks with a large number of sparsely connected neurons, the PDA provides a good approximation to the behaviour of networks. For large populations/subnetworks, the computational efficiency of the PDA is better than direct simulation of individual neurons.

CHAPTER 3

SIMULATION OF ULTRA-SLOW OSCILLATIONS IN CORTICAL NETWORK

In this second part of my thesis, I simulated the ultra-slow oscillatory activity using a modified Morris-Lecar neuron model that takes into consideration the interaction between the neurons and the glial cells in the networks. Based on the model, the frequency of the network activity is affected by variations in the calcium level of the glial cells and ultra-slow oscillations are induced when the recovery time for inositol 1, 4, 5-trisphosphate (IP_3) receptor in the endoplasmic reticulum (ER) is slow. My results of simulations of ultra-slow oscillatory activity are comparable with those obtained from the in-vitro recordings in dissociated cortical cultures. I further proposed that variations in the interaction between neurons and glial cells have a primary role in changing the degree of synchrony among neurons in the ultra-slow oscillatory activity.

3.1 Introduction

In the recent studies, many researchers started to focus on brain activity that is not displayed in the range of traditional EEG frequency bands but in a much slower time scale with frequencies ranging from 0.001-0.01Hz (Drew et al., 2008; Lőrincz et al., 2009; Zhu et al., 2010; Picchioni et al., 2011). A similar slow rhythm was reported in networks of rat brain cortical cells cultured on multi-electrode arrays (Wagenaar et al., 2005; Mok et al., 2012). Ng et al. (2013) simulated the ultra-slow oscillations by introducing generation and dissipation terms of a hypothetical inhibitory property into the network equations of a network of Integrate and Fire (IF) neurons. While the underlying mechanisms of these ultra-slow oscillations were not completely understood, a few studies suggested the possible involvement of glial cells in the generation of these ultra-slow oscillations (Hughes et al., 2011; Krueger et al., 2011). In the present work, a network model that takes into consideration the interaction between neurons and glial cells is used to simulate the ultra-slow oscillations.

This chapter is organized as follows. The literature review of glial cells, communication between neurons and astrocytes, astrocytic excitation: neuron dependent excitation and spontaneous excitation, as well as computation models for interaction between neurons and astrocytes are presented in section 3.2. In section 3.3, the methodology is presented. The network connectivity among neurons and astrocytes are developed in section 3.3.1. The modified conductance-based ML neuron model is described in section 3.3.2. The astrocyte model is presented in section 3.3.3 and the results of bursting

activities in a network of ML neurons, in section 3.4.1. The results of ultra-slow oscillations in a network of ML neurons are shown in section 3.4.2. Emergence of synchronous/asynchronous pattern in neuronal network activities is studied in section 3.4.3. The effect of $\tau_{Ca^{2+}}$ on the rise time of up states in ultra-slow oscillations is presented in section 3.4.4. Section 3.5 is the discussion and section 3.6, the conclusion of this chapter.

3.2 Literature Review

3.2.1 Glial Cells

In the human brain, glial cells outnumbered neurons with a ratio of approximately 10:1 (Magistretti, 1996). Because glial cells cannot generate action potentials, they were conventionally considered as structural and chemical supporting elements for the neurons. However, recent studies have provided increasing evidence to suggest that astrocytes (the most numerous type of glial cells) play an essential role in processing information and modulating neuronal dynamics in the brain (Hirase et al., 2004; Nimmerjahn et al., 2004; Poskanzer and Yuste, 2011; Wade et al., 2012)

3.2.2 Communication between Neurons and Astrocytes

Astrocytes can listen and respond to neurons. Neurotransmitters released by neurons interact with the receptors on the astrocytes and induce intracellular calcium elevation (Araque et al., 2000; Fellin et al., 2006). The intracellular

calcium elevation in astrocytes in turn causes the release of gliotransmitter such as glutamate and ATP that regulate the synaptic neurotransmission (Newman, 2003; Perea and Araque, 2005; Di Castro et al., 2011).

3.2.3 Astrocytic Excitations: Neuron-dependent Excitations and Spontaneous Excitations

Astrocytic excitations can be caused by neurons or occur without the participation of neurons, which is known as spontaneous excitation (Volterra and Meldolesi, 2005). Spontaneous calcium oscillations have been observed experimentally in the cortex (Fatatis and Russell, 1992; Charles, 1994), hippocampus (Harris-White et al., 1998) and thalamus (Parri and Crunelli, 2003). Calcium elevation can occur intrinsically and are not dependent on neuronal activities (Araque et al., 2000; Parri and Crunelli, 2003; Perea et al., 2009) Perea *et al.*, 2009). The spontaneous intracellular Ca^{2+} signals have different properties from those in neurons: long duration in tens of seconds, large amplitude and regular but rare occurrence from 0.5 to 5 min(Volterra and Meldolesi, 2005). There are some inconsistent results reported for the initiation of spontaneous calcium oscillations. Some of the investigators showed that extracellular calcium was required in the generation of spontaneous calcium oscillations (Aguado et al., 2002; Volterra and Meldolesi, 2005) while others reported that intracellular calcium transients trigger the spontaneous oscillations (Parri and Crunelli, 2003; Wang et al., 2006). Regardless of their origin, these spontaneous events are important in the bidirectional communication between neurons and astrocytes, which is able to influence neuronal excitability (Parri et al., 2001; Pasti et al., 2001; Nett et al.,

2002; Fellin et al., 2004).

Besides studies in the level of a single astrocyte, the collective dynamic behaviour of astrocytes has been widely investigated. Sasaki et al. (2011) showed that clusters of 2 to 5 astrocytes in the mouse hippocampus and neocortex spontaneously exhibited locally synchronized activity. In addition, Kuga et al. (2011) studied the larger scale behaviour of hippocampal astrocytes in the mouse and showed that almost all the hundreds of astrocytes exhibited synchronized calcium activity. Parri et al. (2001) reported regular spontaneous calcium oscillations in thalamic astrocytes with periodicity of around 0.019Hz and suggested that such oscillations might lead to rhythmic neuronal activity.

3.2.4 Computation Models for Interactions between Neurons and Astrocytes

Several computational studies have been developed to investigate the underlying biophysical mechanisms of interaction between neurons and astrocytes. Nadkarni and Jung (2004) introduced “dressed neuron model” that consists of a single neuron and astrocyte. The bidirectional communication in this simple neuron-glial circuit can provide long-term potentiation and can induce spontaneous oscillations in the dressed neuron. Furthermore, a mathematical model for the synaptic interactions between presynaptic neuron, postsynaptic neuron and astrocyte in tripartite synapse was also proposed (Nadkarni and Jung, 2007; Postnov et al., 2007). Postnov et al. (2009) extended the tripartite synapse model by incorporating the spatial branching

structure of coupled astrocytes. With the extended model, they reproduced the most typical glial cell responses and patterns of signal transmission. Sotero and Martínez-Cancino (2010) presented a dynamical mean field model that incorporates a large number of tripartite synapses. The model predicts that astrocytic activity can strongly influence neuronal electrical activities. A neural population model which considered the functional outcome of neuron-astrocyte interaction was also investigated to study the neural synchronization (Amiri et al., 2011; Amiri et al., 2012). Amiri et al. (2013) expanded the network size to 50 pyramidal neurons and 50 interneurons. The ratio of one astrocyte to two neurons has been used. They concluded that astrocytes play a primary role in synchronization of the neuronal network activities.

Except for Amiri et al. (2013), most of the previous computational studies focused on functional-based approach to model network that consists of only several neurons and astrocytes. However, modeling of a large and biophysically meaningful network of interacting neurons and astrocytes has yet to be carried out. In the present study, a network model that comprises 10,000 ML neurons interacting with 10,000 astrocytes is described. The ML neurons and astrocytes in the network are randomly connected. Besides of the primary neuron-neuron interactions, the synaptic transmission is also modulated by astrocyte-neuron interactions.

3.3 Methodology

3.3.1 Network Architecture

A network of interconnected excitatory neurons and inhibitory neurons as well as astrocytes is presented in Figure 3.1. There are randomly connected N_{exc} excitatory neurons, N_{inh} inhibitory neurons and N_{ast} astrocytes in the network. The connectivity between pre-synaptic neuron j and postsynaptic neuron i is chosen probabilistically and is denoted by P_{ij} (Latham et al., 2000). For infinite range of connectivity, P_{ij} is only dependent on the type of presynaptic and postsynaptic neurons,

$$P_{ij} = P^{\infty}_{Ty_i,Ty_j} \tag{3.1}$$

where Ty refers to excitatory (E) or inhibitory (I) type of neuron. In the terms of mean number of connections, K_{Ty_j} and connectivity bias, B_{Ty_j}, the connection probability can be written in the form of

$$P^{\infty}_{ETy_j} = \frac{K_{Ty_j}}{N_{exc} + N_{inh}B_{Ty_j}} \tag{3.2}$$

$$P^{\infty}_{ITy_j} = \frac{K_{Ty_j}B_{Ty_j}}{N_{exc} + N_{inh}B_{Ty_j}} \tag{3.3}$$

When there is a connection between neurons, the synaptic strength is controlled by parameters: JEE (excitatory neuron connects to excitatory neuron), JIE (excitatory neuron connects to inhibitory neuron), JEI (inhibitory neuron connects to excitatory neuron) and JII (inhibitory neuron connects to inhibitory neuron).

In this study, I assumed that each astrocyte is randomly connected to 2 neurons (Nedergaard et al., 2003). When the neuron emits a spike, neurotransmitter, T will be released and causes the elevation of intracellular Ca^{2+} level in the astrocytes. In turn, the activated astrocytes will release gliotransmitters which can regulate synaptic transmission via the connection strength of parameters γ_1 and γ_2.

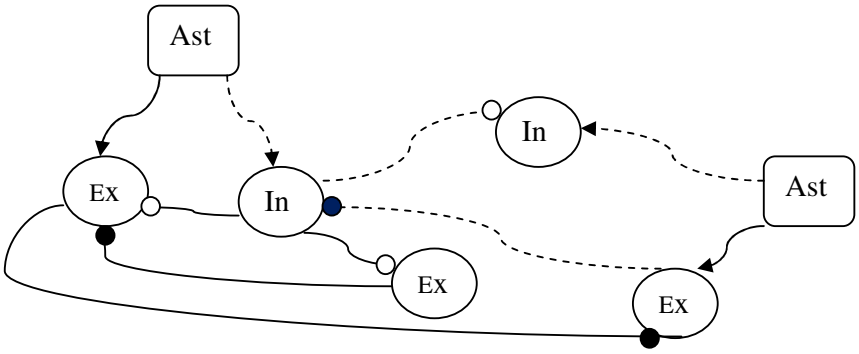

Figure 3.1: Schematic diagram of the network architecture. The network is composed of randomly connected excitatory (Ex) and inhibitory (In) neurons as well as astrocytes (Ast) which can regulate synaptic transmission by the strength connections represented by γ_1(⟶) and γ_2 (--▸). The coupling strength between neurons are controlled by JEE (⟶●), JEI (⟶○), JIE (---●) and JII (---○).

3.3.2 The Modified Conductance-based ML Neuron Model

A modified version of the ML neuron model (Morris and Lecar, 1981; Prescott et al., 2008) is used in the following simulations. The advantage of this model is that as long as the tangent-bifurcation structure is retained, the results for arbitrary long inter-spike intervals will not be altered (Volman et al., 2007) even if a different set of parameters for the model is chosen. The set of differential equations that governs the dynamics of the membrane potential for

neuron i $(i = 1, 2, 3, \dots, N_{exc} + N_{inh})$ are written as follows:

$$C\frac{dV_i}{dt} = -I_{Na,i} - I_{K,i} - I_{L,i} - I_{sAHP,i} - I_{syn,i} + I_{a,i} + I_{ast,i} \tag{3.4}$$

$$\frac{dW_i}{dt} = \varphi\frac{(W_\infty(V_i) - W_i)}{\tau_R(V_i)} \tag{3.5}$$

where

$$I_{Na,i} = g_{Na}m_\infty(V_i)(V_i - E_{Na}) \tag{3.6}$$

$$I_{K,i} = g_K W_i(V_i)(V_i - E_K) \tag{3.7}$$

$$I_{L,i} = g_L(V_i - E_L) \tag{3.8}$$

$$m_\infty(V_i) = 0.5(1 + tanh\{(V_i - V_1)/V_2\}) \tag{3.9}$$

$$W_\infty(V_i) = 0.5(1 + tanh\{(V_i - V_3)/V_4\}) \tag{3.10}$$

$$\tau_R(V_i) = \frac{1}{\cosh\{(V_i - V_3)/(2V_4)\}} \tag{3.11}$$

Here, V is the membrane potential and W is the activation variable for the potassium current. The internal ionic current consists of K^+ current, $I_{K,i}$, Na^+ current, $I_{Na,i}$ and a leakage current, $I_{L,i}$. C is the capacitance of the membrane. The maximum conductance for the ion and the leakage channels are denoted by g_{Na}, g_K and g_L where E_{Na}, E_K and E_L represent the reversal potentials for the ion and leakage channels. The fraction of open potassium channels W tends to the saturation value $W_\infty(V_i)$ with a characteristic time scale of $(\tau_R(V_i)/\phi)$. Fast changes of the calcium current take the gate variable m_i as the saturation value $m_\infty(V_i)$.

The original ML model is modified by including the additional slow adaptation-afterhyperpolarization (AHP) current. The slow

afterhyperpolarization current, $I_{sAHP,i}$, is activated following the action potential emitted by each neuron. This current is described as (Prescott et al., 2008):

$$I_{sAHP,i} = g_{sAHP} z_{sAHP,i} (V_i - E_K) \tag{3.12}$$

After each action potential, I_{sAHP}can persist up to 5s (Sah and Louise Faber, 2002). The activation of $I_{sAHP,i}$ is controlled by $z_{sAHP,i}$:

$$\frac{dz_{sAHP,i}}{dt} = \left[\frac{1}{1 + e^{(\beta_z - V_i)/\gamma_z}} - z_{sAHP,i}\right] / \tau_{z_{sAHP}} \tag{3.13}$$

where g_{sAHP} is the slow afterhyperpolarization conductance, $\tau_{z_{sAHP}}$ is the time constants. β_z and γ_z are set to $0mV$ and $1mV$ respectively. Exemplar voltage trace of neuron with and without AHP current is shown in Figure 3.2. The AHP current following each single action potential will suppress the depolarization for periods that is dependent on the time constant of $\tau_{z_{sAHP}}$.

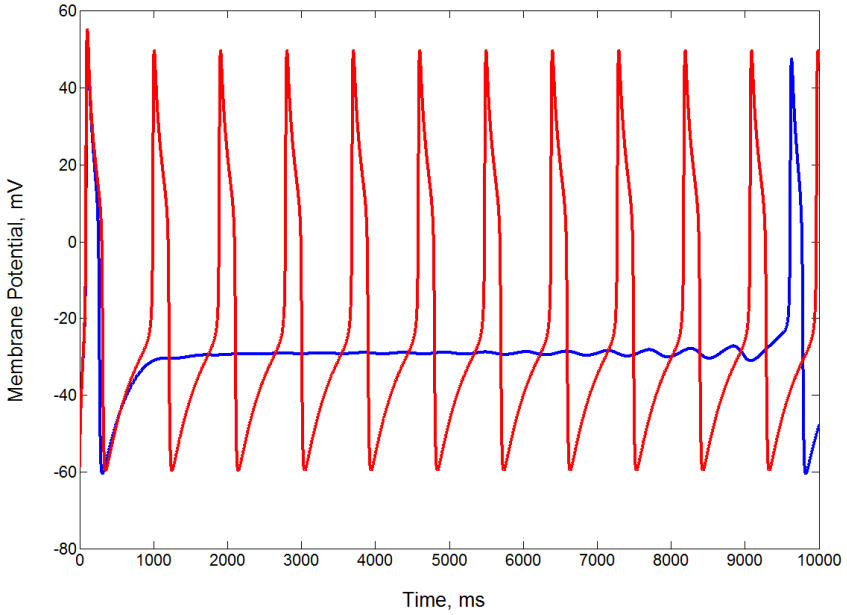

Figure 3.2: The membrane potential of neuron with AHP current (blue solid line) and without AHP current (red solid line) in the case of stimulation by constant input current of $60\mu A/cm^2$ and $\tau_{z_{sahp}} = 55ms$.

The synaptic current of the ith neuron is given as (Latham et al., 2000)

$$I_{syn,i} = \sum_{j(\neq i)} Q_{ij} s_{ij}(t)(V_i - E_{syn,j}) \tag{3.14}$$

The coupling strength from neuron j to neuron i is controlled by Q_{ij} which mainly depends on the type of neurons connected: $Q_{ij} = JEE, JEI, JIE \; or \; JII$. When there is no connection between neurons, $Q_{ij} = 0$. $E_{syn,j}$ is the reversal potential for excitatory synapse or inhibitory synapse. When neuron j fires, the the fraction of open channels, s_{ij} on i neuron will increase instantaneously and then decay exponentially:

$$\frac{ds_{ij}}{dt} = -\frac{s_{ij}}{\tau_s} + r_s \sum_k \delta(t - t_j^k) \tag{3.15}$$

t_j^k is time of the kth spike on neuron j. When V_j crosses the threshold value V_{th}, neuron j emits a spikes and r_s determines the number of closed channels

open each time neuron j fires. In order to reduce the computation time, Equation 3.14 is written in another form of

$$I_{syn,i} = V_i I_i - I_{\zeta,i} \tag{3.16}$$

where

$$I_i = \sum_j Q_{ij} s_{ij} \tag{3.16a}$$

$$I_{\zeta,i} = \sum_j Q_{ij} s_{ij}\, \zeta_j \tag{3.16b}$$

I_i and $I_{\zeta,i}$ are evolved based on the following differential equations

$$\frac{dI_i}{dt} = -\frac{I_i}{\tau_s} + r_s \sum\nolimits_{j(\neq i)k} Q_{ij}\delta(t - t_j^k) \tag{3.17}$$

$$\frac{dI_{\zeta,i}}{dt} = -\frac{I_{\zeta,i}}{\tau_s} + r_s \sum\nolimits_{j(\neq i)k} Q_{ij} E_{syn,i}\delta(t - t_j^k) \tag{3.18}$$

The applied current, $I_{a,i}$ for each neuron i was uniformly chosen between 0 and I_{max}. It defines how many neurons in the network are endogenously active that are close to the threshold value and tend to fire (Latham et al., 2000). The last term of Equation 3.4, $I_{ast,i}$ denotes the current induced by astrocytes' activities.

3.3.3 The Astrocyte Model

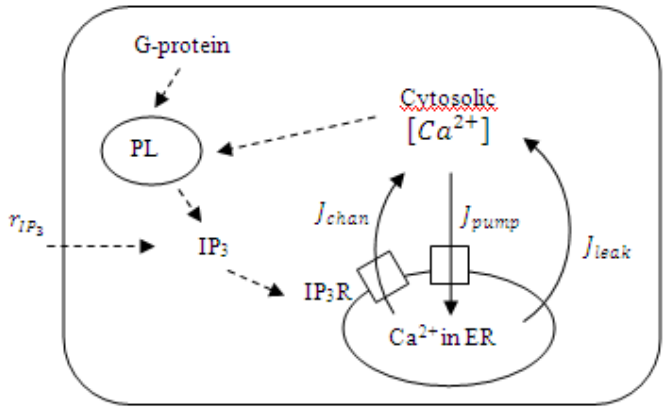

Figure 3.3: Schematic illustration of astrocyte model. The solid arrows indicate the movement of the calcium while the dashed arrows indicate the cooperative effects (e.g., the effect of cytosolic $[Ca^{2+}]$ on PLC).

The schematic illustration of the astrocyte model is presented in Figure 3.3. Astrocytes excitation is due to two main forms: neuron-dependant excitation and spontaneous excitation. For neuron-dependant excitation, neurotransmitter, T released from presynaptic neurons activates the receptors on the astrocytes and induces the production of $[IP_3]$ in the intracellular space. The IP_3 receptor (IP_3R) is opened, allowing Ca^{2+} to flow out from the endoplasmic reticulum (ER) to the cytosol. For spontaneous excitation in our model, the production of $[IP_3]$ depends on the effect of intracellular $[Ca^{2+}]$ stimulation on membrane-bound phospholipase C (PLC) or agonist stimulation by G-protein mechanism. When $[IP_3]$ is produced, Ca^{2+} is released into the cytosol rapidly, and lead to inactivation of the IP_3R channels. At this moment, cytosolic $[Ca^{2+}]$ is pumped back to ER. IP_3R channels are activated rapidly again when $[Ca^{2+}]$ has decreased sufficiently. This biphasic response of IP_3R causes the cycle to repeat itself and thereby inducing $[Ca^{2+}]$ oscillations (De Young and Keizer, 1992; Foskett et al., 2007). The biphasic response of IP_3R and the

corresponding concentration of calcium in the cytosolic and ER under the oscillatory regime is shown in Figure 3.4.

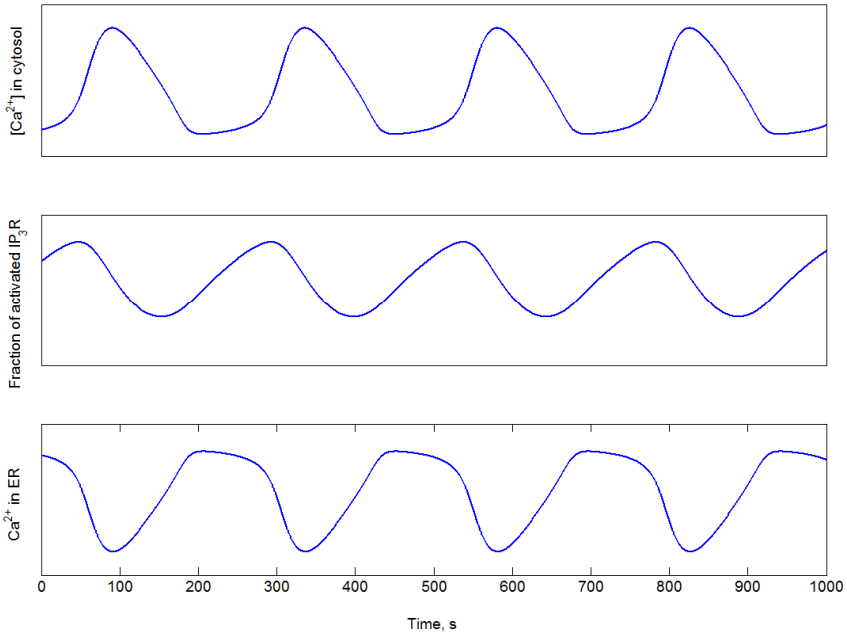

Figure 3.4: The relationship between the fraction of activated IP_3R, concentration of calcium in the cytosolic and ER under the oscillatory regime.

Accordingly, the generation of intracellular $[IP_3]$ in the astrocyte is provided by (De Young and Keizer, 1992; Nadkarni and Jung, 2003)

$$\frac{d[IP_3]}{dt} = v_4\left(\frac{[Ca_i^{2+}] + (1-\alpha)k_4}{[Ca_i^{2+}] + k_4}\right) - \frac{[IP_3]}{\tau_{IP_3}} + r_{IP_3}\Theta(V_i - V_{th}) \tag{3.19}$$

where $0 \leq \alpha \leq 1$. α indicates the relative effect of $[Ca^{2+}]$ activation of PLC on $[IP_3]$ production. The maximum production rate of $[IP_3]$, v_4 is independent of $[Ca^{2+}]$. In this study, I assumed that the effective v_4 can be increased via agonist stimulation by G-protein mechanism. k_4 is the dissociation constant for $[Ca^{2+}]$ activation of $[IP_3]$ production and $\frac{1}{\tau_{IP_3}}$ is the degradation rate. The parameter r_{IP_3} determines the production of $[IP_3]$ in response to a neuronal

action potential. The production term is activated when the membrane potential of the neuron is larger than V_{th} via the step function Θ.

The Li-Rinzel model (Li and Rinzel, 1994) is used as the dynamic model of the astrocytes. The set of differential equations that governs the dynamics of $[Ca^{2+}]$ concentration in the intracellular space is written as follows:

$$\frac{d[Ca^{2+}]}{dt} = J_{chan} - J_{pump} + J_{leak} \tag{3.20}$$

$$\frac{dq}{dt} = \alpha_q(1-q) - \beta_q q \tag{3.21}$$

where

$$\alpha_q = a_2 d_2 \frac{[IP_3] + d_1}{[IP_3] + d_3} \tag{3.22}$$

$$\beta_q = a_2[Ca^{2+}] \tag{3.23}$$

q denotes the fraction of activated IP_3R. α_q and β_q correspond to the IP_3R opening rate and closing rate respectively that are controlled by the parameter IP_3R inactivation binding rate, a_2 and $[Ca^{2+}]$ inactivation dissociation constant, d_2. The examples of intracellular calcium oscillations of different frequencies with respect to a_2 and d_2 are shown in Figure 3.5.

J_{chan},J_{leak} and J_{pump} are the calcium flux from ER through IP_3 release channels, the leakage flux from ER to cytosol and the pump flux from cytosol to ER respectively. The fluxes are described by following equations:

$$J_{chan} = c_1 v_1 p_\infty^3 n_\infty^3 q^3([Ca^{2+}]_{ER} - [Ca^{2+}]) \tag{3.24}$$

$$J_{leak} = c_1 v_2([Ca^{2+}]_{ER} - [Ca^{2+}]) \tag{3.25}$$

$$J_{pump} = \frac{v_3[Ca^{2+}]^2}{[Ca^{2+}]^2 + k_3^2} \tag{3.26}$$

where

$$p_\infty = \frac{[IP_3]}{[IP_3 + d_1]} \tag{3.27}$$

$$n_\infty = \frac{[Ca^{2+}]}{[Ca^{2+}] + d_5} \tag{3.28}$$

The Ca^{2+} concentration in ER is denoted as

$$[Ca^{2+}]_{ER} = \frac{c_0 - [Ca^{2+}]}{c_1} \tag{3.29}$$

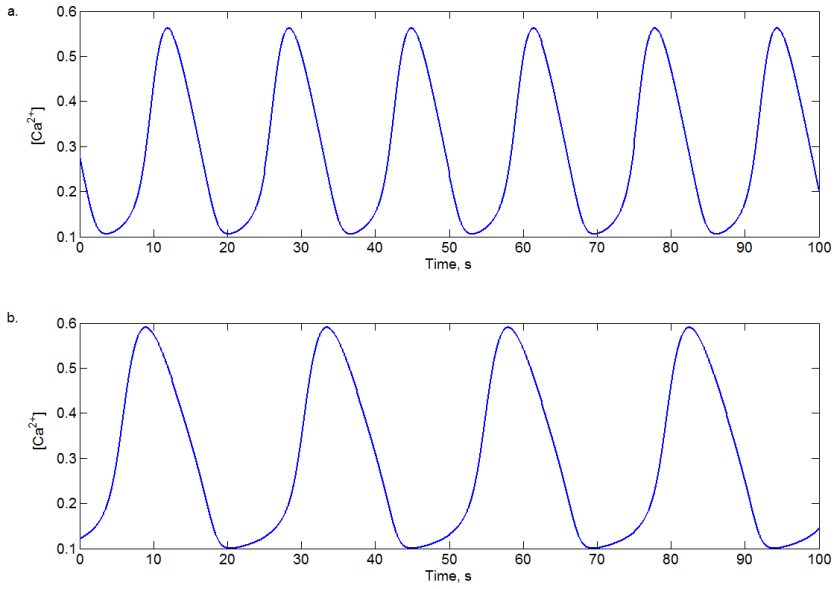

Figure 3.5: The periodic solution of intracellular calcium elevation simulated by the astrocyte dynamic model with $(a)\ a_2 = 0.1\mu Ms^{-1}, d_2 = 1.049\mu M$ and $(b)\ a_2 = 0.05\mu Ms^{-1}, d_2 = 1.049\mu M$.

The calcium signals are transmitted from cell to cell in the oscillatory regimes rather than stationary states. Among the physiological significant oscillatory behaviours that are included are (i) for the same second messengers, different processes could be switched depending on the changes in oscillations frequency, for instance, the releasing of different transcription factors in T-lymphocytes, (ii) a wide range of signal strengths could be accomplished, up to several orders of magnitude, and (iii) more types of distinct signals could be transmitted such as synchronization, phase locking and chaotic regimes (Schuster et al., 2002).

If the level of $[Ca^{2+}]$ exceeds the threshold, $[Ca^{2+}]_{th}$, the astrocyte will release a finite amount of gliotransmitters into the synaptic cleft (Volman et al., 2007). The interaction of astrocyte-neuron is modelled by

$$\frac{df}{dt} = \frac{-f}{\tau_{Ca^{2+}}} + (1-f)\kappa\Theta([Ca^{2+}] - [Ca^{2+}]_{th}) \tag{3.30}$$

where $\tau_{Ca^{2+}}$ is Ca^{2+} time constant, κ is a scaling factor and Θ is the Heaviside function. An increase of intracellular calcium, $[Ca^{2+}]$ leads to the release of ATP which can modulate neuronal excitability by suppressing or facilitating the synaptic transmission. Astrocytic ATP release activates metabotropic P2Y1 receptors and increases the excitation of inhibitory neurons that in turn enhances synaptic inhibition (Bowser and Khakh, 2004; Torres et al., 2012). On the other hand, the activation of different purinergic receptors by ATP suppresses the excitability of excitatory neurons (Fellin et al., 2006). The biological facts are functionally modelled by including the negative and positive signs to the output current from the astrocyte to the excitatory and inhibitory neuron respectively (Amiri et al., 2013):

$$I_{ast} = \begin{cases} -\gamma_1 f & for\ excitatory\ neuron \\ \gamma_2 f & for\ inhibitory\ neuron \end{cases}$$

γ_1 and γ_2 are the strength connections from the astrocyte to the excitatory neuron and the inhibitory neuron respectively.

3.4 Results

3.4.1 Neuronal Network without Interaction with Astrocytes

I solved the resulting set of ODEs in the model using Runge-Kutta 4th order with fixed time step of 0.1ms. For all the simulations, histograms of fixed bin size, 10ms is used. Firstly, the neuronal network without incorporating the influence of astrocyte's activities is investigated $(\gamma_1 = \gamma_2 = 0)$. Figure 3.6 demonstrates the intrinsic dynamics activities for a large network of neurons with different I_{max}: $(a)60\ \mu A/cm^2, (b)63.3\ \mu A/cm^2\ and\ (c)75\ \mu A/cm^2$. The other network parameters are shown in Table 3.1. When the number of endogenously active neurons is increased, the amplitude of firing rate is also increased. For a low number of endogenously active neurons, a low number of spikes occurred at very low rates (Figure 3.6(a)). The network activities for $I_{max} = 63.3\ \mu A/cm^2$ (Figure 3.6(b)) are comparable to the bursting activities within the peaks of the ultra-slow oscillations observed in experiment (unpublished data provided by S.Y. Mok) (Figure 3.6(d)). Further increasing I_{max} leads to neurons firing at very high rates (Figure 3.6(c)).

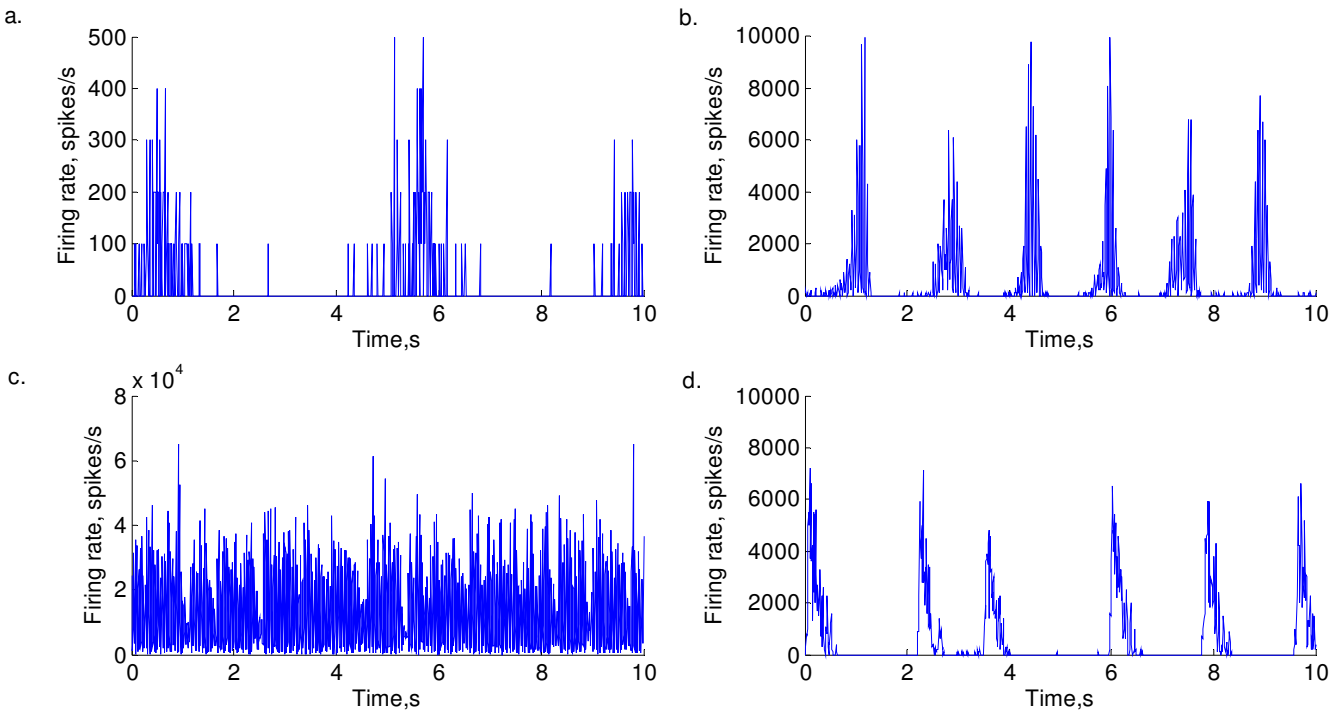

Figure 3.6: Network activities without astrocytes in the range of different I_{max} used in the network model: (a) 60 $\mu A/cm^2$, (b) 63.3$\mu A/cm^2$ and (c) 75 $\mu A/cm^2$. The experimental data is shown in (d).

Table 3.1: Parameters for modified ML neuron model (Latham et al., 2000; (Lim and Kim, 2007)

Parameter	Description	Value
$\mathbf{V_1}$	Threshold value for m_∞	$-1.2\ mV$
$\mathbf{V_2}$	Steepness parameter for m_∞	$18\ mV$
$\mathbf{V_3}$	Threshold value for W_∞	$12mV$
$\mathbf{V_4}$	Steepness parameter for W_∞	$30\ mV$
$\mathbf{E_{Na}}$	Reversal potential for Na^+ channels	$60\ mV$
$\mathbf{E_L}$	Reversal potential for leakage channels	$-60\ mV$
$\mathbf{E_K}$	Reversal potential for K^+ channels	$-84\ mV$
$\boldsymbol{C}$	Capacitance of membrane	$5\ \mu F/cm^2$
$\boldsymbol{\varphi}$	Temperature time scale factor	0.04
$\boldsymbol{g_{Na}}$	Maximum conductance for Na^+ channels	$7.5\ ms/cm^2$
$\boldsymbol{g_K}$	Maximum conductance for K^+ channels	$8\ ms/cm^2$
$\boldsymbol{g_L}$	Maximum conductance for leakage channels	$2\ ms/cm^2$
$\boldsymbol{V_{th}}$	Threshold value for spiking state	$0\ mV$
$\boldsymbol{g_{sAHP}}$	Slow afterhyperpolarization conductance	$1.8mS/cm^2$
$\boldsymbol{\tau_{z_{sAHP}}}$	Slow afterhyperpolarization time constant	$2000ms$
$\boldsymbol{E_{syn,E}}$	Reversal potential for excitatory neuron	$0mV$
$\boldsymbol{E_{syn,I}}$	Reversal potential for inhibitory neuron	$-80mV$
$\boldsymbol{JEE}$	Synaptic strength of excitatory neuron connects to excitatory neuron	$0.93mS/cm^2$
$\boldsymbol{JEI}$	Synaptic strength of inhibitory neuron connects to excitatory neuron	$4.19mS/cm^2$

Table 3.1 continued

$\boldsymbol{JII}$	Synaptic strength of inhibitory neuron connects to inhibitory neuron	$\boldsymbol{4.19 mS/cm^2}$
$\boldsymbol{JIE}$	Synaptic strength of excitatory neuron connects to inhibitory neuron	$0.93 mS/cm^2$
$\boldsymbol{\tau_s}$	Synaptic time constant	$3ms$
$\boldsymbol{r_s}$	Fraction of closed channels open	0.1
$\boldsymbol{N_{inh}}$	Number of inhibitory neurons	2000
$\boldsymbol{N_{exc}}$	Number of excitatory neurons	8000
$\boldsymbol{N_{ast}}$	Number of astrocytes	10000
$\boldsymbol{K_I}$	Mean number of inhibitory connections	2000
$\boldsymbol{K_E}$	Mean number of excitatory connections	2000
$\boldsymbol{B_E}$	Connectivity bias toward excitatory neurons	1.2
$\boldsymbol{B_I}$	Connectivity bias toward inhibitory neurons	0.8

3.4.2 Ultra-Slow Oscillations in a Network of ML Neurons

For this simulation, I included the role of the astrocytes, in modulating the frequency of neuronal network activities, into the model. Each astrocyte is randomly connected with two neurons in the network. With the participation of astrocytes, oscillatory activities as a succession of up and down states were obtained. To investigate the effect of astrocytes to the frequency of neuronal network activities, I varied the two parameters: IP_3R inactivation binding rate, a_2 and $[Ca^{2+}]$ inactivation dissociation constant, d_2 while keeping constant the coupling strength of astrocyte-neuron interactions, $\gamma_1 = 0.425$ and $\gamma_2 = 0.05$ (Figure 3.7). Note that the neuronal network frequencies are within

the broad range of ultra-slow oscillatory rhythm: 0.001-0.01Hz. With the same level of d_2, the frequency of neuronal network oscillations increased as the level of a_2 is increased. The neurons fire irregularly and the network oscillations are eliminated when d_2 is larger than 1.4 μM(not shown). We can observe from Figure 3.7, a transition in neuronal network frequency as a function of a_2 and d_2. This observation suggests that the astrocytes play a substantial role in regulating the frequency of ultra-slow oscillations.

Figure 3.8 shows some of the simulated results of ultra-slow oscillatory activities with slightly different frequencies. It is clear that the coupling strength from astrocytes to neurons, γ_1 determines the number of spikes contained within the burst during the down states of the oscillations. Figure 3.8(a) shows the simulated results of an ultra-slow oscillation of approximately 0.002Hz when $a_2 = 0.00141\mu Ms^{-1}$, $d_2 = 1.2\mu M$, $\gamma_2 = 0.05$ and $\gamma_1 = 0.425$. Figure 3.8(b) shows an ultra-slow oscillation of approximately 0.003Hz when $a_2 = 0.0018\mu Ms^{-1}$, $d_2 = 1.16\mu M$, $\gamma_2 = 0.05$ and $\gamma_1 = 0.625$. Figure 3.8(c) shows an ultra-slow oscillation of approximately 0.005Hz when $a_2 = 0.00234\mu Ms^{-1}$, $d_2 = 1.16\mu M$, $\gamma_2 = 0.05$ and $\gamma_1 = 0.8$. These simulation results are comparable to those observed in dissociated cortical networks (See Figures 3.9(a), 3.9(b) and 3.9(c)) in the experimental study (unpublished data provided by S.Y. Mok).

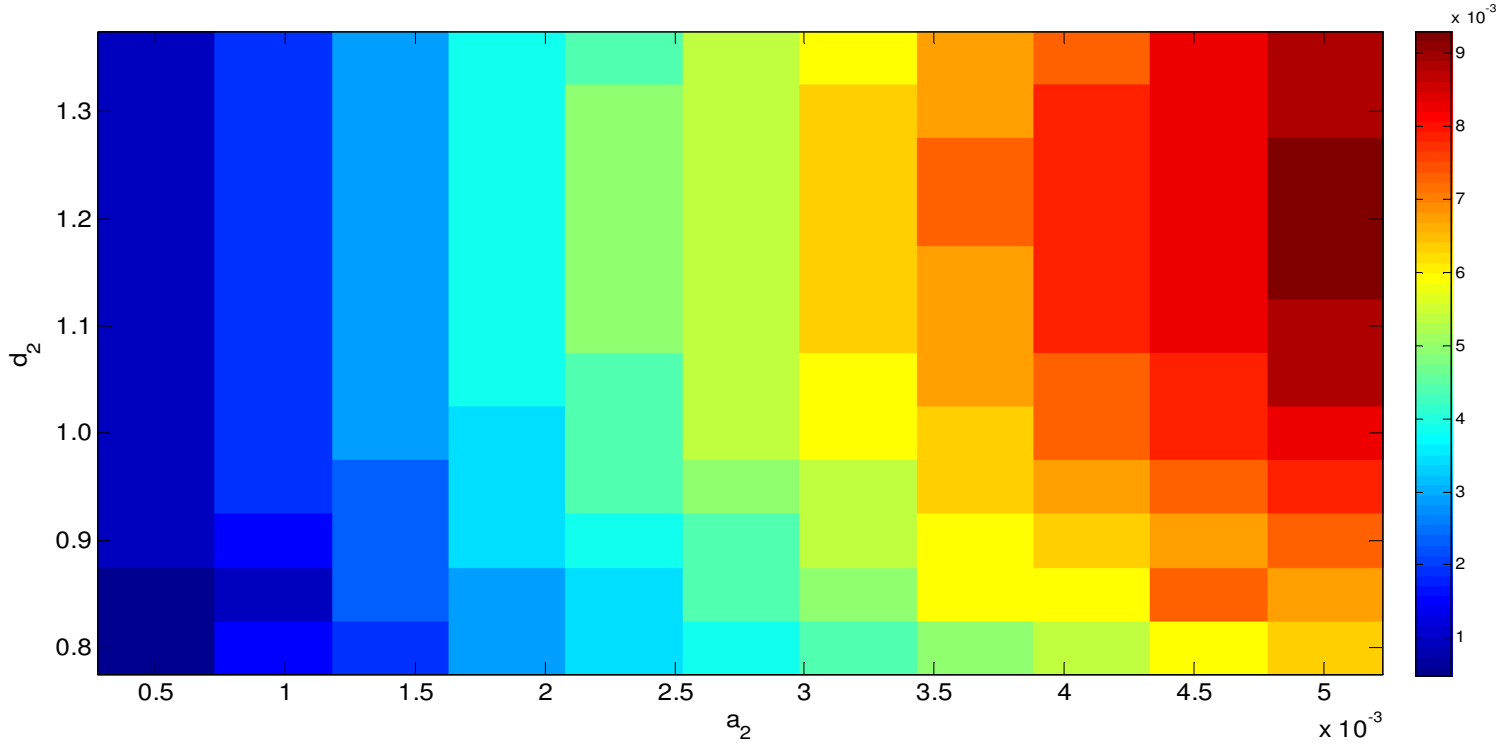

Figure 3.7**:** The frequency of neuronal network activities (represented by different colors) in unit Hz as the functions of $a_2(\mu M s^{-1})$ and $d_2(\mu M)$.

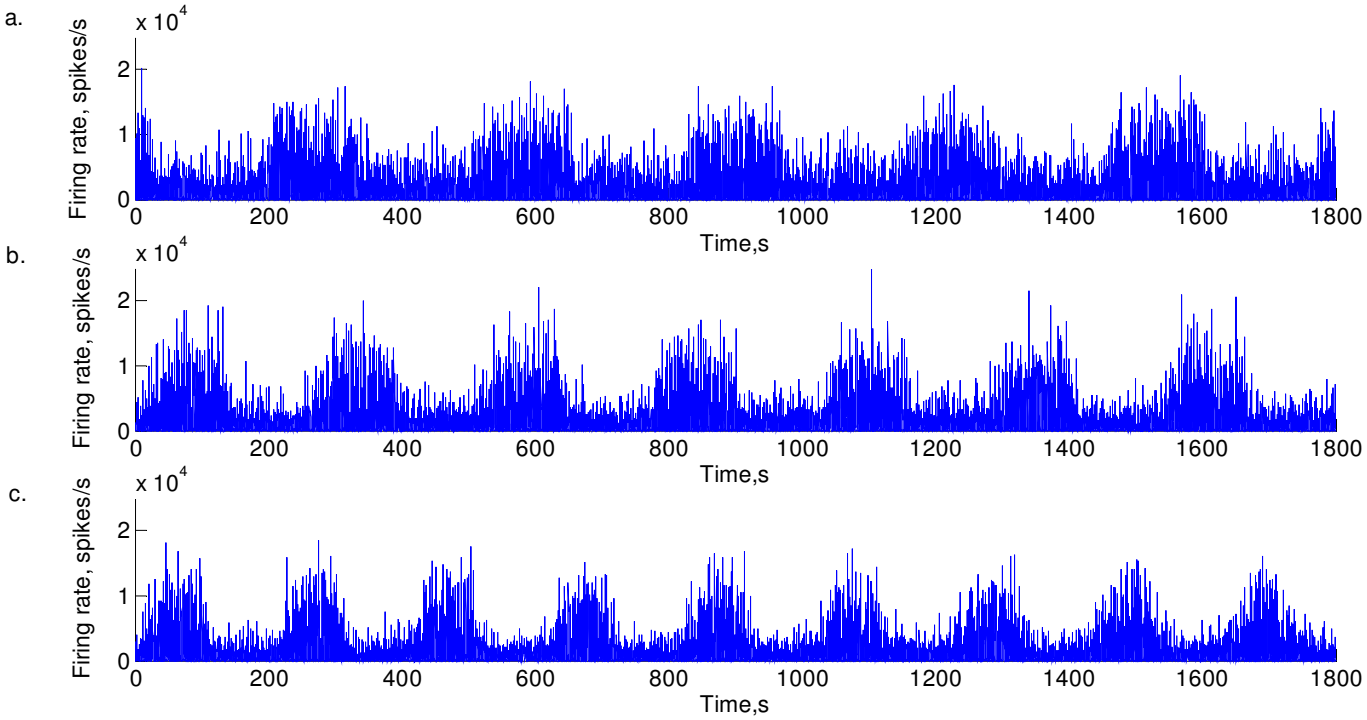

Figure 3.8: Simulation results of ultra-slow oscillations with (a) $a_2 = 0.00123\mu M s^{-1}, d_2 = 1.3\mu M$ and $\gamma_1 = 0.17$, (b) $a_2 = 0.00159\mu M, d_2 = 1.34\mu M$ and $\gamma_1 = 0.25$, (c) $a_2 = 0.00207\mu M s^{-1}, d_2 = 1.3\mu M$ and $\gamma_1 = 0.32$.

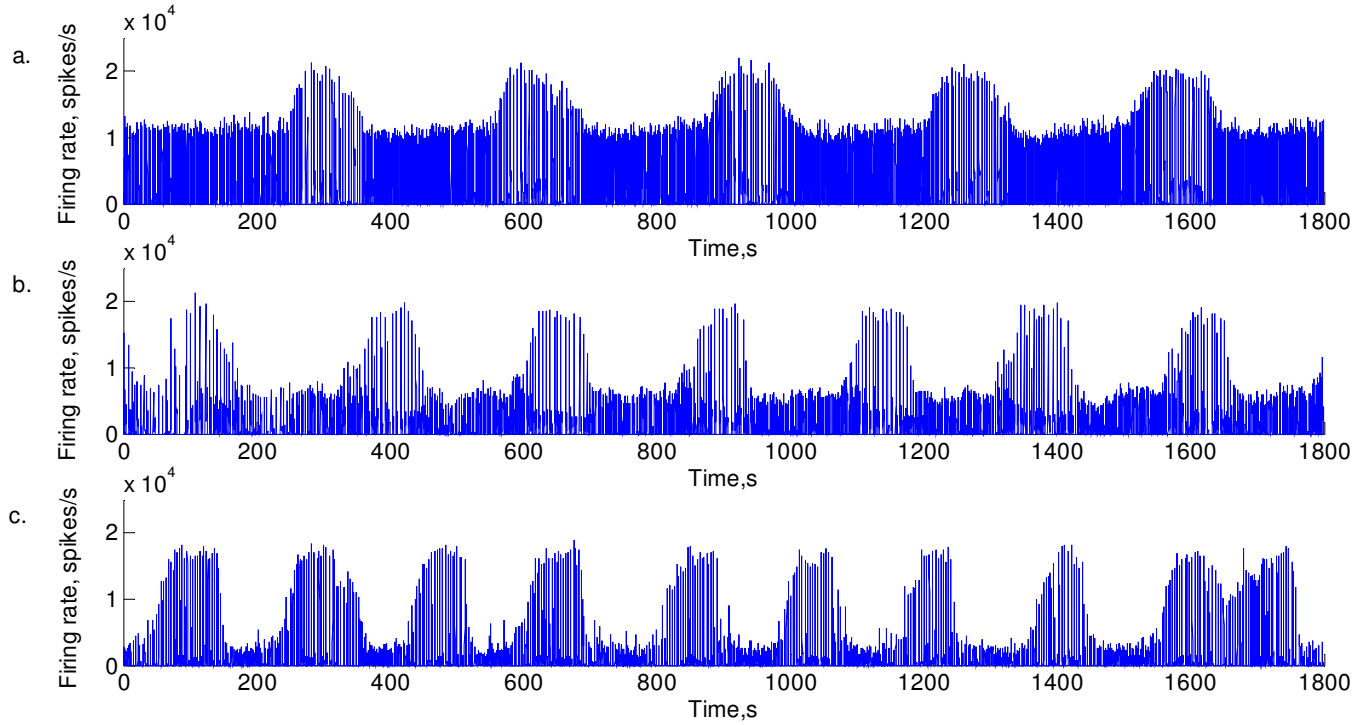

Figure 3.9: Experimental results of spontaneous ultra-slow oscillations from different culture.

Table 3.2: Parameters for astrocyte model (Amiri et al., 2013; De Young and Keizer, 1992)

Parameter	Description	Value
$\mathbf{c_0}$	Total free $[Ca^{2+}]$ concentration	$2\mu M$
$\mathbf{c_1}$	Ratio of ER volume to cytosol volume	0.185
$\mathbf{v_1}$	Maximum rate of CICR (calcium-induced calcium release)	$6s^{-1}$
$\mathbf{v_2}$	Ca^{2+} leakage rate from ER	$0.11s^{-1}$
$\mathbf{v_3}$	Maximum rate of pump intake	$0.9\mu Ms^{-1}$
$\mathbf{v_4}$	Maximum production rate of $[IP_3]$	$0.1s^{-1}$
$\mathbf{d_1}$	$[IP_3]$ dissociation constant	$0.13\mu M$
$\mathbf{d_3}$	$[IP_3]$ dissociation constant	$0.94\mu M$
$\mathbf{d_5}$	Ca^{2+} activation dissociation constant	$0.08234\mu M$
$\boldsymbol{\tau_{IP_3}}$	$[IP_3]$ degradation time constant	$7s$
$\mathbf{r_{IP_3}}$	Rate of $[IP_3]$ production	$0.01\mu Ms^{-1}$
$\mathbf{k_3}$	Pump activation constant	$0.1\mu M$
$\mathbf{k_4}$	Dissociation constant for $[Ca^{2+}]$ activation of $[IP_3]$ production	$1.1\mu M$
$\boldsymbol{\alpha}$	Relative effect of $[Ca^{2+}]$ activation of PLC on $[IP_3]$ production	0.2
$\boldsymbol{\kappa}$	Scaling factor	$0.5s^{-1}$
$\boldsymbol{\tau_{Ca^{2+}}}$	Decay rate of f	$10s$
$\boldsymbol{[Ca^{2+}]_{th}}$	Astrocyte Gliotransmitter release threshold	$0.2\mu M$

3.4.3 Emergence of Synchronous/Asynchronous Patterns

In this section, I further investigate the effect of interactions between astrocytes and neurons in neuronal synchronization. Different strengths of connection from astrocytes to excitatory neurons, γ_1 and from astrocytes to inhibitory neurons, γ_2 are varied in the following simulations. I kept constant the parameters $a_2 = 0.00123\mu M s^{-1}$ and $d_2 = 1.3\mu M$ while other parameters are as given in Table 1 and 2. By setting $\gamma_1 = 0.17$ and $\gamma_2 = 0.05$, the synchronized neural activities within the down states of ultra-slow oscillations are demonstrated in Figure 3.10(a). Further increasing the interactions between astrocyte-neuron to $\gamma_1 = 0.5, \gamma_2 = 0.35$ and $\gamma_1 = 1.0, \gamma_2 = 0.7$, the amplitude of neuronal firing synchronization is decreased (Figure 3.10(b) and Figure 3.10(c)). The simulation results reveal that increasing the coupling strength between the astocytes and the neurons alters the synaptic transmission and consequently changes the synchronization level in the neuronal network activities. Variation in the coupling strength of astrocyte-neuron interactions could be one of the plausible mechanisms in the emergence of synchronous/asynchronous pattern which is an important mechanism for neural information processing (Hamilton and Attwell 2010; Schummers et al. 2008; Pereira Jr and Furlan 2009).

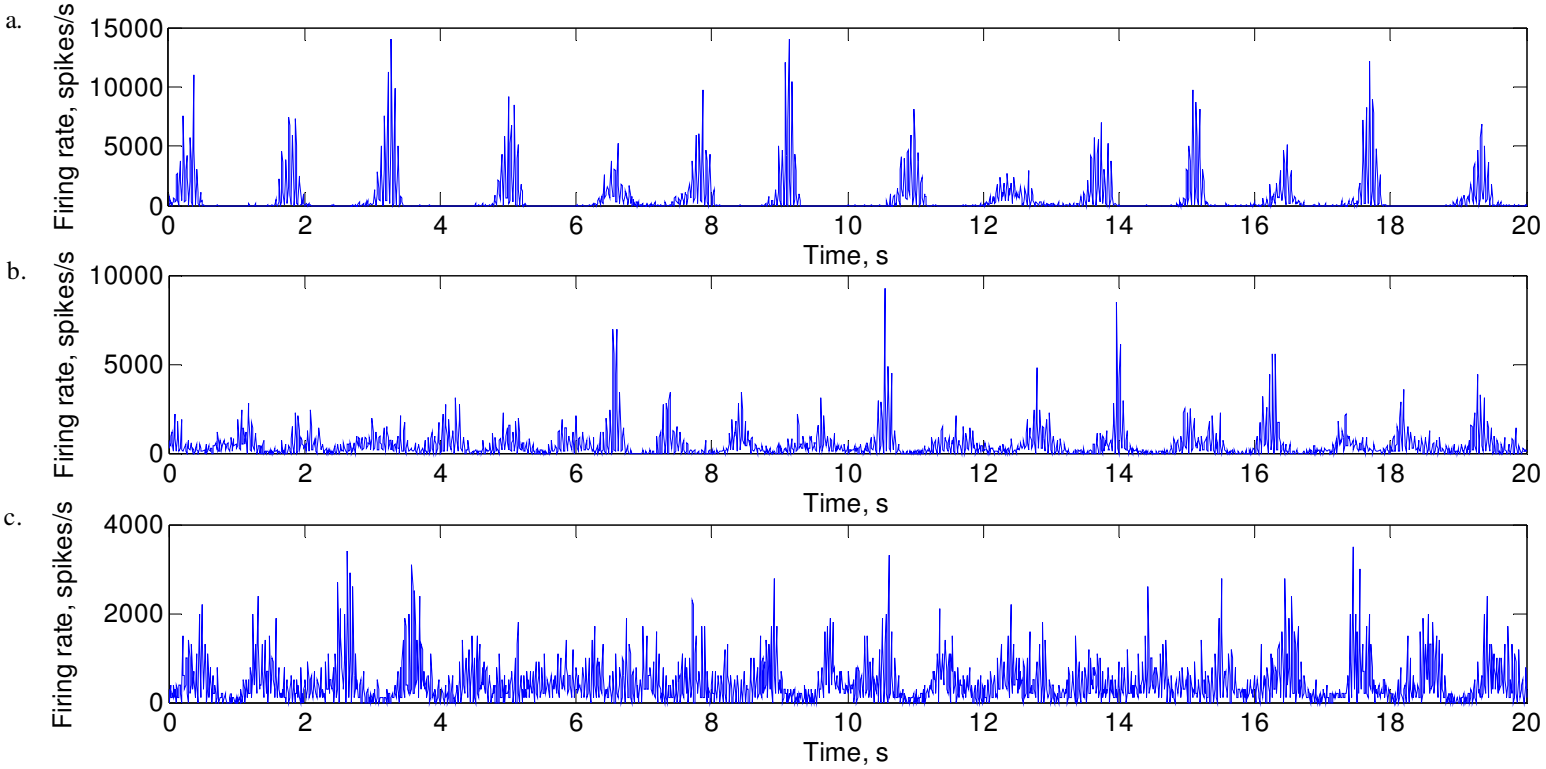

Figure 3.10**:** The effect of increasing astrocyte-neuron interactions within the down states of the ultra-slow oscillations: (a) $\gamma_1 = 0.17, \gamma_2 = 0.05$; (b) $\gamma_1 = 0.5,\ \gamma_2 = 0.35$ and (c) $\gamma_1 = 1.0, \gamma_2 = 0.7$.

3.4.4 Effect of $\tau_{Ca^{2+}}$ on the Rise Time of Up States in Ultra-Slow Oscillations

Experimental results revealed that the time required for the neuronal network activities to rise to the up states of ultra-slow oscillations varied in different cultures (Figure 3.11). The rise time is calculated as the average time taken for the neuronal network activities to change from the down states to 80% of the up states in each simulation. In our model, we can functionally reproduce this phenomenon. The critical parameter for eliciting this phenomenon is the decay time of astrocyte-neuron interaction function, $\boldsymbol{\tau_{Ca^{2+}}}$. For the following simulations, we kept constant the parameters $a_2 = 0.001\mu Ms^{-1}$, $d_2 = 1.0\mu M$ and histograms of fixed bin size=1s. Figure 3.12 shows the effect of, $\boldsymbol{\tau_{Ca^{2+}}}$ on the rise time. The astrocytes' influence on the synaptic terminal persists even after the intracellular calcium elevation in the astrocytes is lowered.

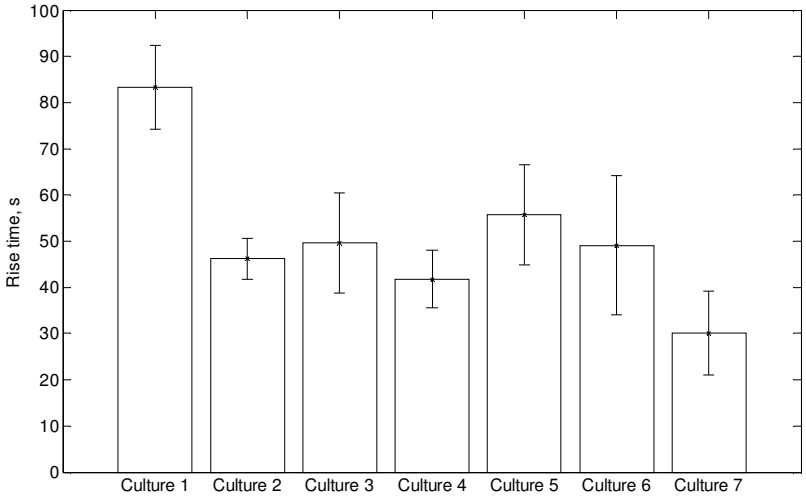

Figure 3.11: Experimental results of the rise time from different cultures.

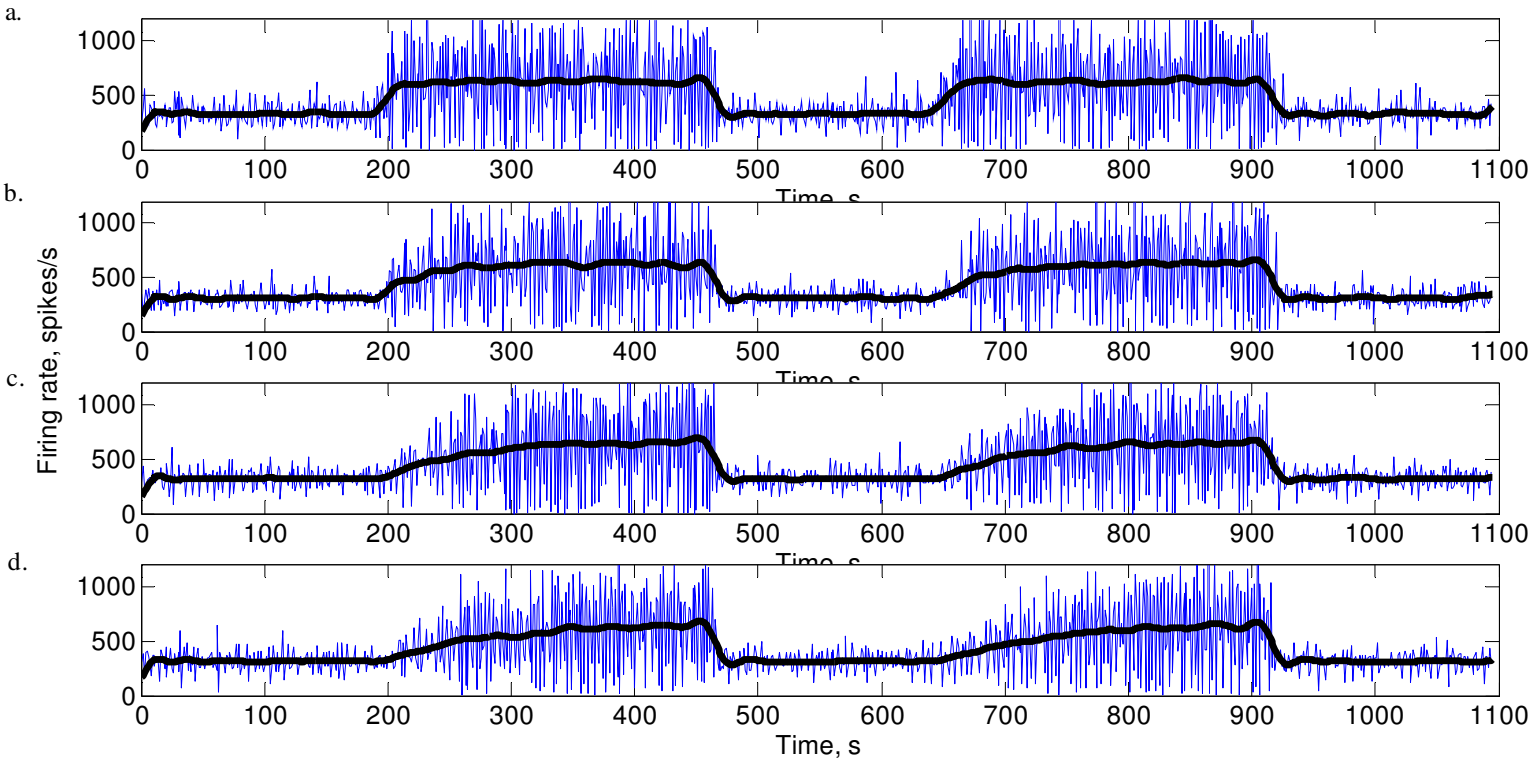

Figure 3.12: The effect of $\tau_{Ca^{2+}}$ on the rise time of up states in ultra-slow oscillations: (a)$\tau_{Ca^{2+}} = 10s$, rise time = 32.5s, (b) $\tau_{Ca^{2+}} = 20s$, rise time = 65.5s, (c)$\tau_{Ca^{2+}} = 30s, rise\ time = 101s$, and (d) $\tau_{Ca^{2+}} = 50s, rise\ time = 119s$.

3.5 Discussion

From my model, I suggest the possible involvement of astrocytes in neuronal network activities. My simulation results may provide a new insight into the putative role of astrocytes in generating the ultra-slow oscillations that were observed in cortical cultures of rat brain neurons.

There is increasing evidence to show that astrocytes play an active role in modulating neural network dynamics in the central nervous system (Parri et al., 2001; Wang et al., 2012). The bidirectional interactions between astrocytes and neurons are important in maintaining normal neuronal activities (Fellin, 2009; Amiri et al., 2013). In this study, I modelled the interactions of neuron-neuron as well as the interactions of astrocyte-neuron. The simulation results demonstrated that changes in the frequency of neuronal network oscillations depended mainly on the close rate and open rate of IP_3R, which were in turn controlled by two parameters: IP_3R inactivation binding rate, a_2 and $[Ca^{2+}]$ inactivation dissociation constant, d_2. This agrees with the findings by De Young and Keizer (1992) in which the frequency of the intracellular $[Ca^{2+}]$ oscillations is mostly affected by the recovery time for the IP_3R. These Ca^{2+} oscillations in astrocytes are closely linked to their environment including their interconnected neurons. Thus, I suggest that IP_3 causes changes in astrocytes calcium levels that oscillate within the physiological time scale range of $0.5 - 5mins$ (Volterra and Meldolesi, 2005) that then interact with the neurons to generate ultra-slow oscillations as observed in cortical cultures of rat brain neurons.

In accordance with experimental observations (Fellin, 2009; Pereira Jr and Furlan, 2009; Poskanzer and Yuste, 2011), the simulation results showed that astrocytes can regulate the neuronal excitability and play a causal role in regulating the synchronization level among neurons. The results showed that variation in the coupling strength of astrocyte-neuron interactions was the primary factor that influenced the synchronization level of ultra-slow oscillations. However, it should be pointed out that the synchronization of neuronal activities as well as network behavior also depended on the connectivity of neuron-glial networks that were utilized in the current model. This may be the reason why the ultra-slow oscillations were observed in some cultures but not in others.

In the current model, I have utilized a functional approach to model the astrocyte-neuron interactions. However, in order to enhance our understanding of ultra-slow oscillations, the fine details and the cellular signaling mechanisms underlying astrocyte-neuron interactions that alter the neuronal network activities should be further investigated using both experimental and computational approaches.

3.6 Conclusion

I propose a modified Morris-Lecar neuron model that takes into consideration the interaction between the neurons and the glial cells in the network. Using this model, I successfully simulated ultra-slow oscillations that are comparable to those observed in cortical cultures of rat brain neurons.

CHAPTER 4

CONCLUSION REMARKS AND FUTURE WORK

4.1 Summary of Major Results

4.1.1 Simulation of Network Oscillations using the Population Density Approach

A PDA is introduced to simulate a network of ML neurons. The results were compared against conventional computation for groups of individual neurons in a few example networks by varying (i) population sizes, (ii) different connectivity and (iii) different amplitude of PSP for inhibitory synapses. The PDA provided a closer approximation to the network activities for neuronal networks with a large number of sparsely connected neurons.

One of the advantages of this approach is saving in computation time for simulating large biophysically meaningful neuronal networks. For large populations/networks, the computational efficiency of the PDA is better than direct simulation of individual neurons. The incorporation of the ML neuron model in the PDA enables the approach to be applied to study more complex phenomena as demonstrated in the simulation of the behavior of type I and type II neurons.

4.1.2 Simulation of Ultra-Slow Oscillations in Cortical Networks

For a better understanding of ultra-slow oscillations on the time scale of minutes that was observed in dissociated cortical cultures of rat brain neurons, a modified ML network model that takes into consideration the interaction between neurons and astrocytes was developed. The current computational study provided a framework for improving understanding of the firing activity patterns within the oscillations as well as the bursts at the peaks and troughs.

Based on the simulation results of a large network of randomly connected neurons-astrocytes, the following observation may be made:

(i) Endogenously active cells: Without any external stimulation to the neurons, the intrinsic firing patterns within the ultra-slow oscillations of large neuronal network are significantly controlled by the number of endogenously active neurons.

(ii) Elevation of astrocytes calcium levels: By taking into consideration the interaction between neurons and astrocytes, a succession of peaks and troughs at different frequency can be generated. The frequency of the ultra-slow oscillatory activity is modulated by changes in the calcium levels of the astrocytes that interact with the ML neurons.

(iii) Strength of connection from astrocytes to neurons: Variations in the strength of the connection between the astocytes and the neurons can change the synchronous/asynchronous firing pattern of the neurons in the network.

4.2 Future Work

4.2.1 PDA studies

The PDA can be extended to simulate more complex combinations of subnetworks such as the neocortex of human brain that are organized vertically into cortical columns with each cortical column containing 60,000 or more neurons. Such simulations using the conventional approach to treat the network as a combination of individual neurons would require the computing power of a cluster of workstations to generate meaningful results. The PDA may enable more researchers with limited computing facilities to tackle more complex neural systems.

4.2.2 Ultra-Slow Oscillations

The current study is the first attempt at using a modified ML neuron model that takes into consideration the interaction between glial cells and neurons to simulate ultra-slow oscillations in a large network. The detailed mechanisms of the interaction of neuron-glial interactions are still not fully understood. Further experimental studies are necessary to investigate the effect of glial

cells (e.g. effect of calcium buffers) and their response to the neuronal network activities. Selective stimulation of glial cells via pharmacological or optogenetic stimulation can be devised to investigate the consequences of glial activities in the neuronal network. The new experimental evidence may provide useful hints on how to improve the analytical models. To enhance the computational efficiency, a two timescale mean field approach that glial cells operate on much slower timescales than neurons could be developed to study the network activities of interaction between neurons and glial cells.

REFERENCES

Aguado, F., Espinosa-Parrilla, J. F., Carmona, M. a. A. and Soriano, E., 2002. Neuronal activity regulates correlated network properties of spontaneous calcium transients in astrocytes in situ. *The Journal of neuroscience,* 22, pp. 9430-9444.

Allers, K. A., et al., 2002. Multisecond periodicities in basal ganglia firing rates correlate with theta bursts in transcortical and hippocampal EEG. *Journal of neurophysiology,* 87, pp. 1118-1122.

Amiri, M., Bahrami, F. and Janahmadi, M., 2012. Functional contributions of astrocytes in synchronization of a neuronal network model. *Journal of theoretical biology,* 292, pp. 60-70.

Amiri, M., Hosseinmardi, N., Bahrami, F. and Janahmadi, M., 2013. Astrocyte-neuron interaction as a mechanism responsible for generation of neural synchrony: a study based on modeling and experiments. *Journal of computational neuroscience,* 34, pp. 489-504.

Amiri, M., Montaseri, G. and Bahrami, F., 2011. On the role of astrocytes in synchronization of two coupled neurons: a mathematical perspective. *Biological cybernetics,* 105, pp. 153-166.

Apfaltrer, F., Ly, C. and Tranchina, D., 2006. Population density methods for stochastic neurons with realistic synaptic kinetics: Firing rate dynamics and fast computational methods. *Network: Computation in Neural Systems,* 17, pp. 373-418.

Araque, A., Li, N., Doyle, R. T. and Haydon, P. G., 2000. SNARE protein-dependent glutamate release from astrocytes. *The Journal of neuroscience,* 20, pp. 666-673.

Başar, E., 2011. Brain-Body-Mind in the nebulous Cartesian system: a holistic approach by oscillations.

Balenzuela, P., Buldú, J. M., Casanova, M. and García-Ojalvo, J., 2006. Episodic synchronization in dynamically driven neurons. *Physical Review E,* 74, pp. 061910.

Bazhenov, M., Rulkov, N. F. and Timofeev, I., 2008. Effect of synaptic connectivity on long-range synchronization of fast cortical oscillations.

Journal of neurophysiology, 100, pp. 1562-1575.

Bowser, D. N. and Khakh, B. S., 2004. ATP excites interneurons and astrocytes to increase synaptic inhibition in neuronal networks. *The Journal of neuroscience,* 24, pp. 8606-8620.

Brunel, N., 2000. Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of computational neuroscience,* 8, pp. 183-208.

Brunel, N. and Hakim, V., 1999. Fast global oscillations in networks of integrate-and-fire neurons with low firing rates. *Neural computation,* 11, pp. 1621-1671.

Buzsáki, G. and Draguhn, A., 2004. Neuronal oscillations in cortical networks. *Science,* 304, pp. 1926-1929.

Buzsáki, G. and Wang, X.-J., 2012. Mechanisms of gamma oscillations. *Annual review of neuroscience,* 35, pp. 203-225.

Casti, A., et al., 2002. A population study of integrate-and-fire-or-burst neurons. *Neural computation,* 14, pp. 957-986.

Charles, A., 1994. Glia-neuron intercellular calcium signaling. *Developmental neuroscience,* 16, pp. 196-206.

Chen, C., et al., 2006. The origin of spontaneous synchronized burst in cultured neuronal networks based on multi-electrode arrays. *BioSystems,* 85, pp. 137-143.

Chizhov, A. V., Graham, L. J. and Turbin, A. A., 2006. Simulation of neural population dynamics with a refractory density approach and a conductance-based threshold neuron model. *Neurocomputing,* 70, pp. 252-262.

Compte, A., et al., 2008. Spontaneous high-frequency (10–80 Hz) oscillations during up states in the cerebral cortex in vitro. *The Journal of neuroscience,* 28, pp. 13828-13844.

De Pittà, M., et al., 2012. Computational quest for understanding the role of astrocyte signaling in synaptic transmission and plasticity. *Frontiers in computational neuroscience,* 6.

De Young, G. W. and Keizer, J., 1992. A single-pool inositol 1, 4, 5-trisphosphate-receptor-based model for agonist-stimulated oscillations in Ca2+ concentration. *Proceedings of the National Academy of Sciences,* 89, pp. 9895-9899.

Destexhe, A., Babloyantz, A. and Sejnowski, T. J., 1993. Ionic mechanisms for intrinsic slow oscillations in thalamic relay neurons. *Biophysical journal,* 65, pp. 1538-1552.

Di Castro, M. A., et al., 2011. Local Ca2+ detection and modulation of synaptic release by astrocytes. *Nature neuroscience,* 14, pp. 1276-1284.

Drew, P. J., Duyn, J. H., Golanov, E. and Kleinfeld, D., 2008. Finding coherence in spontaneous oscillations. *Nature neuroscience,* 11, pp. 991-993.

Eric R. Kandel, J. H. S., Thomas M. Jessell 1991. *Principles of neural science, 3rd edition,* New York, Elsevier.

Fatatis, A. and Russell, J. T., 1992. Spontaneous changes in intracellular calcium concentration in type I astrocytes from rat cerebral cortex in primary culture. *Glia,* 5, pp. 95-104.

Feller, M. B., 1999. Spontaneous correlated activity in developing neural circuits. *Neuron,* 22, pp. 653-656.

Fellin, T., 2009. Communication between neurons and astrocytes: relevance to the modulation of synaptic and network activity. *Journal of neurochemistry,* 108, pp. 533-544.

Fellin, T., et al., 2004. Neuronal synchrony mediated by astrocytic glutamate through activation of extrasynaptic NMDA receptors. *Neuron,* 43, pp. 729-743.

Fellin, T., Pascual, O. and Haydon, P. G., 2006. Astrocytes coordinate synaptic networks: balanced excitation and inhibition. *Physiology,* 21, pp. 208-215.

Foskett, J. K., White, C., Cheung, K.-H. and Mak, D.-O. D., 2007. Inositol trisphosphate receptor Ca2+ release channels. *Physiological reviews,* 87, pp. 593-658.

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

Gielen, S., Krupa, M. and Zeitler, M., 2010. Gamma oscillations as a mechanism for selective information transmission. *Biological cybernetics,* 103, pp. 151-165.

Gray, C. M., 1994. Synchronous oscillations in neuronal systems: mechanisms and functions. *Journal of computational neuroscience,* 1, pp. 11-38.

Harris-White, M. E., Zanotti, S. A., Frautschy, S. A. and Charles, A. C., 1998. Spiral intercellular calcium waves in hippocampal slice cultures. *Journal of neurophysiology,* 79, pp. 1045-1052.

Haskell, E., Nykamp, D. Q. and Tranchina, D., 2001. Population density methods for large-scale modelling of neuronal networks with realistic synaptic kinetics: cutting the dimension down to size. *Network: Computation in Neural Systems,* 12, pp. 141-174.

Hirase, H., Qian, L., Barthó, P. and Buzsáki, G., 2004. Calcium dynamics of cortical astrocytic networks in vivo. *PLoS biology,* 2, pp. e96.

Hubel, D. H. and Wiesel, T. N., 1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology,* 160, pp. 106.

Huertas, M. A. and Smith, G. D., 2006a. A multivariate population density model of the dLGN/PGN relay. *Journal of computational neuroscience,* 21, pp. 171-189.

Huertas, M. A. and Smith, G. D., 2006b. A two-dimensional population density approach to modeling the dLGN/PGN network. *Neurocomputing,* 69, pp. 1286-1290.

Hughes, S. W., Lőrincz, M. L., Parri, H. R. and Crunelli, V., 2011. Infra-slow (< 0.1 Hz) oscillations in thalamic relay nuclei: basic mechanisms and significance to health and disease states. *Progress in brain research,* 193, pp. 145.

Hutcheon, B., Miura, R. M. and Puil, E., 1996a. Models of subthreshold membrane resonance in neocortical neurons. *Journal of neurophysiology,* 76, pp. 698-714.

Hutcheon, B., Miura, R. M. and Puil, E., 1996b. Subthreshold membrane resonance in neocortical neurons. *Journal of neurophysiology,* 76, pp. 683-697.

Hutcheon, B. and Yarom, Y., 2000. Resonance, oscillation and the intrinsic frequency preferences of neurons. *Trends in neurosciences,* 23, pp. 216-222.

Izhikevich, E. M., 2000. Neural excitability, spiking and bursting. *International Journal of Bifurcation and Chaos,* 10, pp. 1171-1266.

Jiao, X. and Wang, R., 2010. Synchronous firing patterns of neuronal population with excitatory and inhibitory connections. *International Journal of Non-linear Mechanics,* 45, pp. 647-651.

Kilpatrick, Z. P. and Bressloff, P. C., 2010. Effects of synaptic depression and adaptation on spatiotemporal dynamics of an excitatory neuronal network. *Physica D: Nonlinear Phenomena,* 239, pp. 547-560.

Krueger, J. M., et al., 2011. Involvement of cytokines in slow wave sleep. *Progress in brain research,* 193, pp. 39.

Kudela, P., Franaszczuk, P. J. and Bergey, G. K., 2003. Changing excitation and inhibition in simulated neural networks: effects on induced bursting behavior. *Biological cybernetics,* 88, pp. 276-285.

Kuga, N., et al., 2011. Large-scale calcium waves traveling through astrocytic networks in vivo. *The Journal of neuroscience,* 31, pp. 2607-2614.

Lampl, I. and Yarom, Y., 1997. Subthreshold oscillations and resonant behavior: two manifestations of the same mechanism. *Neuroscience,* 78, pp. 325-341.

Latham, P., Richmond, B., Nelson, P. and Nirenberg, S., 2000. Intrinsic dynamics in neuronal networks. I. Theory. *Journal of neurophysiology,* 83, pp. 808-827.

Li, Y.-X. and Rinzel, J., 1994. Equations for InsP< sub> 3</sub> Receptor-mediated [Ca< sup> 2+</sup>]< sub> i</sub> Oscillations Derived from a Detailed Kinetic Model: A Hodgkin-Huxley Like Formalism. *Journal of theoretical biology,* 166, pp. 461-473.

Lim, W. and Kim, S.-Y., 2011. Statistical-mechanical measure of stochastic spiking coherence in a population of inhibitory subthreshold neurons. *Journal of computational neuroscience,* 31, pp. 667-677.

Lim, W. and Kim, S., 2007. Characterization of stochastic spiking coherence in coupled neurons. *JOURNAL-KOREAN PHYSICAL SOCIETY,* 51, pp. 1427.

Liu, Y., Wang, R., Zhang, Z. and Jiao, X., 2010. Analysis of stability of neural network with inhibitory neurons. *Cognitive neurodynamics,* 4, pp. 61-68.

Lőrincz, M. L., et al., 2009. ATP-dependent infra-slow (< 0.1 Hz) oscillations in thalamic networks. *PloS one,* 4, pp. e4447.

Magistretti, P. J., 1996. Metabolic coupling between glia and neurons. *Journal of Neuroscience,* 16, pp. 877-885.

Marpeau, F., Barua, A. and Josić, K., 2009. A finite volume method for stochastic integrate-and-fire models. *Journal of computational neuroscience,* 26, pp. 445-457.

Melamed, O., et al., 2008. Slow oscillations in neural networks with facilitating synapses. *Journal of computational neuroscience,* 25, pp. 308-316.

Mok, S., Nadasdy, Z., Lim, Y. and Goh, S., 2012. Ultra-slow oscillations in cortical networks< i> in vitro</i>. *Neuroscience,* 206, pp. 17-24.

Mongillo, G. and Amit, D. J., 2001. Oscillations and irregular emission in networks of linear spiking neurons. *Journal of computational neuroscience,* 11, pp. 249-261.

Morris, C. and Lecar, H., 1981. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical journal,* 35, pp. 193-213.

Mountcastle, V. B., 1957. Modality and topographic properties of single neurons of cat's somatic sensory cortex. *J. neurophysiol,* 20, pp. 408-434.

Mountcastle, V. B., 1997. The columnar organization of the neocortex. *Brain,* 120, pp. 701-722.

Nadkarni, S. and Jung, P., 2003. Spontaneous oscillations of dressed neurons: a new mechanism for epilepsy? *Physical review letters,* 91, pp. 268101.

Nadkarni, S. and Jung, P., 2004. Dressed neurons: modeling neural–glial interactions. *Physical biology,* 1, pp. 35.

Nadkarni, S. and Jung, P., 2007. Modeling synaptic transmission of the tripartite synapse. *Physical biology,* 4, pp. 1.

Nedergaard, M., Ransom, B. and Goldman, S. A., 2003. < i> New roles for astrocytes</i>: Redefining the functional architecture of the brain. *Trends in neurosciences,* 26, pp. 523-530.

Nesse, W. H., Borisyuk, A. and Bressloff, P. C., 2008. Fluctuation-driven rhythmogenesis in an excitatory neuronal network with slow adaptation. *Journal of computational neuroscience,* 25, pp. 317-333.

Nett, W. J., Oloff, S. H. and Mccarthy, K. D., 2002. Hippocampal astrocytes in situ exhibit calcium oscillations that occur independent of neuronal activity. *Journal of neurophysiology,* 87, pp. 528-537.

Newman, E. A., 2003. Glial cell inhibition of neurons by release of ATP. *The Journal of neuroscience,* 23, pp. 1659-1666.

Ng, D., Siow-Cheng, C. and Sing-Yau, G., 2013. Simulation of Ultra-slow Oscillations Using the Integrate and Fire Neuron Model. *Engineering,* 4, pp. 65.

Nimmerjahn, A., Kirchhoff, F., Kerr, J. N. and Helmchen, F., 2004. Sulforhodamine 101 as a specific marker of astroglia in the neocortex in vivo. *Nature Methods,* 1, pp. 31-37.

Nykamp, D. Q. and Tranchina, D., 2000. A population density approach that facilitates large-scale modeling of neural networks: Analysis and an application to orientation tuning. *Journal of computational neuroscience,* 8, pp. 19-50.

Nykamp, D. Q. and Tranchina, D., 2001. A population density approach that facilitates large-scale modeling of neural networks: extension to slow inhibitory synapses. *Neural computation,* 13, pp. 511-546.

Omurtag, A., Knight, B. W. and Sirovich, L., 2000. On the simulation of large populations of neurons. *Journal of computational neuroscience,* 8, pp. 51-63.

Parri, H. and Crunelli, V., 2003. The role of Ca< sup> 2+</sup> in the generation of spontaneous astrocytic Ca< sup> 2+</sup> oscillations. *Neuroscience,* 120, pp. 979-992.

Parri, H., Gould, T. M. and Crunelli, V., 2001. Spontaneous astrocytic Ca2+ oscillations in situ drive NMDAR-mediated neuronal excitation. *Nature neuroscience,* 4, pp. 803-812.

Pasti, L., et al., 2001. Cytosolic calcium oscillations in astrocytes may regulate exocytotic release of glutamate. *The Journal of neuroscience,* 21, pp. 477-484.

Penttonen, M., et al., 1999. Ultra-slow oscillation (0.025 Hz) triggers hippocampal afterdischarges in Wistar rats. *Neuroscience,* 94, pp. 735-743.

Perea, G. and Araque, A., 2005. Properties of synaptically evoked astrocyte calcium signal reveal synaptic information processing by astrocytes. *The Journal of neuroscience,* 25, pp. 2192-2203.

Perea, G., Navarrete, M. and Araque, A., 2009. Tripartite synapses: astrocytes process and control synaptic information. *Trends in neurosciences,* 32, pp. 421-431.

Pereira Jr, A. and Furlan, F. A., 2009. On the role of synchrony for neuron–astrocyte interactions and perceptual conscious processing. *Journal of biological physics,* 35, pp. 465-480.

Picchioni, D., et al., 2011. Infraslow EEG oscillations organize large-scale cortical–subcortical interactions during sleep: a combined EEG/fMRI study. *Brain research,* 1374, pp. 63-72.

Poskanzer, K. E. and Yuste, R., 2011. Astrocytic regulation of cortical UP states. *Proceedings of the National Academy of Sciences,* 108, pp. 18453-18458.

Postnov, D., et al., 2009. Dynamical patterns of calcium signaling in a functional model of neuron–astrocyte networks. *Journal of biological physics,* 35, pp. 425-445.

Postnov, D., Ryazanova, L. S. and Sosnovtseva, O. V., 2007. Functional modeling of neural–glial interaction. *BioSystems,* 89, pp. 84-91.

Prescott, S. A., Ratté, S., De Koninck, Y. and Sejnowski, T. J., 2008. Pyramidal neurons switch from integrators in vitro to resonators under in vivo-like conditions. *Journal of neurophysiology,* 100, pp. 3030-3042.

Puil, E., Meiri, H. and Yarom, Y., 1994. Resonant behavior and frequency preferences of thalamic neurons. *Journal of neurophysiology,* 71, pp. 575-582.

Reinker, S., Li, Y.-X. and Kuske, R., 2006. Noise-induced coherence and network oscillations in a reduced bursting model. *Bulletin of mathematical biology,* 68, pp. 1401-1427.

Reutimann, J., Giugliano, M. and Fusi, S., 2003. Event-driven simulation of

spiking neurons with stochastic dynamics. *Neural computation,* 15, pp. 811-830.

Sah, P. and Louise Faber, E., 2002. Channels underlying neuronal calcium-activated potassium currents. *Progress in neurobiology,* 66, pp. 345-353.

Sasaki, T., et al., 2011. Locally synchronized astrocytes. *Cerebral Cortex,* 21, pp. 1889-1900.

Schuster, S., Marhl, M. and Höfer, T., 2002. Modelling of simple and complex calcium oscillations. *European Journal of Biochemistry,* 269, pp. 1333-1355.

Sotero, R. C. and Martínez-Cancino, R., 2010. Dynamical mean field model of a neural-glial mass. *Neural computation,* 22, pp. 969-997.

Steriade, M., Nunez, A. and Amzica, F., 1993. A novel slow (< 1 Hz) oscillation of neocortical neurons in vivo: depolarizing and hyperpolarizing components. *The Journal of neuroscience,* 13, pp. 3252-3265.

Steyn-Ross, M. L., Steyn-Ross, D. A., Sleigh, J. W. and Wilson, M. T., 2011. A mechanism for ultra-slow oscillations in the cortical default network. *Bulletin of mathematical biology,* 73, pp. 398-416.

Talathi, S. S., Hwang, D.-U., Carney, P. R. and Ditto, W. L., 2010. Synchrony with shunting inhibition in a feedforward inhibitory network. *Journal of computational neuroscience,* 28, pp. 305-321.

Tonnelier, A., 2005. Categorization of neural excitability using threshold models. *Neural computation,* 17, pp. 1447-1455.

Torres, A., et al., 2012. Extracellular Ca2+ acts as a mediator of communication from neurons to glia. *Science signaling,* 5, pp. ra8.

Tsodyks, M., Uziel, A. and Markram, H., 2000. Synchrony generation in recurrent networks with frequency-dependent synapses. *J Neurosci,* 20, pp. 825-835.

Vida, I., Bartos, M. and Jonas, P., 2006. Shunting inhibition improves robustness of gamma oscillations in hippocampal interneuron networks by homogenizing firing rates. *Neuron,* 49, pp. 107-117.

Volman, V., Baruchi, I., Persi, E. and Ben-Jacob, E., 2004. Generative modelling of regulated dynamical behavior in cultured neuronal networks. *Physica A: Statistical Mechanics and its Applications,* 335, pp. 249-278.

Volman, V., Ben-Jacob, E. and Levine, H., 2007. The astrocyte as a gatekeeper of synaptic information transfer. *Neural computation,* 19, pp. 303-326.

Volterra, A. and Meldolesi, J., 2005. Astrocytes, from brain glue to communication elements: the revolution continues. *Nature Reviews*

Neuroscience, 6, pp. 626-640.

Wade, J., et al., 2012. Self-repair in a bidirectionally coupled astrocyte-neuron (AN) system based on retrograde signaling. *Frontiers in computational neuroscience,* 6.

Wagenaar, D., Demarse, T. B. and Potter, S. M. MEABench: A toolset for multi-electrode data acquisition and on-line analysis. Neural Engineering, 2005. Conference Proceedings. 2nd International IEEE EMBS Conference on, 2005. IEEE, pp. 518-521.

Wang, F., et al., 2012. Astrocytes modulate neural network activity by Ca2+-dependent uptake of extracellular K+. *Science signaling,* 5, pp. ra26.

Wang, R. and Jiao, X., 2006. Stochastic model and neural coding of large-scale neuronal population with variable coupling strength. *Neurocomputing,* 69, pp. 778-785.

Wang, T.-F., et al., 2006. Cellular mechanism for spontaneous calcium oscillations in astrocytes. *Acta Pharmacologica Sinica,* 27, pp. 861-868.

Wang, X.-J. and Buzsáki, G., 1996. Gamma oscillation by synaptic inhibition in a hippocampal interneuronal network model. *The Journal of neuroscience,* 16, pp. 6402-6413.

Wang, X. J., 2003. Neural oscillations. *Encyclopedia of cognitive science*.

Zhu, G., et al., 2010. Transient alterations in slow oscillations of hippocampal networks by low-frequency stimulations on multi-electrode arrays. *Biomedical microdevices,* 12, pp. 153-158.

Zhu, L., et al., 2006. Nucleus-and species-specific properties of the slow (< 1 Hz) sleep oscillation in thalamocortical neurons. *Neuroscience,* 141, pp. 621-636.

APPENDIX A

PROGRAMMING CODE

A.1 The Direct Simulation of the Conductance-based ML Neuron Model

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <deque>
#include <tr1/random>
#include <vector>
#include <cmath>
#include <ctime>
#include <stdio.h>

using namespace std;
using namespace std::tr1;

class membrane{
	public:
		double C;
		double GCa;
		double m_inf;
		double ECa;
		double GK;
		double n;
		double EK;
		double GLeak;
		double ELeak;
		double q;
		double Iint;
		double Inoise;

		membrane():
			C(0),
			GCa(0),
			m_inf(0),
			ECa(0),
```

```
                GK(0),
                n(0),
                EK(0),
                GLeak(0),
                ELeak(0),
                Iint(Iint),
                Inoise(Inoise),
                q(0){}

        ~membrane(){}

                double operator () (double y, double t){
                        return( 1/C * ( -GCa*m_inf*(y-ECa) -
GK*n*(y-EK) - GLeak*(y-ELeak) ) + Inoise + Iint);
                }
        };

class potassium{
    public:
        double tau_R;
        double n_inf;
        double factor;
        double r;

        potassium():
                tau_R(0),
                r(0),
                factor(0),
                n_inf(0){}

        ~potassium(){}

                double operator () (double y, double t){
                        return( factor/tau_R*(n_inf-y) );
                }
        };

class Timing {
    public:
        Timing(){
                startt=time(NULL);
                endt=time(NULL);
        }
        ~Timing(){}
        void tic(){
        startt=time(NULL);
        }
        void toc(){
        endt=time(NULL);
        }
```

```
        time_t diff(){
        return endt-startt;
        }
    private:
        time_t startt, endt;
};


template <typename function> double runge_kutta_4th(function equation, double initial, double t, double dt){
    double k1 = equation(initial,t);
    double k2 = equation(initial + 0.5 * k1 * dt, t + 0.5 * dt);
    double k3 = equation(initial + 0.5 * k2 * dt, t + 0.5 * dt);
    double k4 = equation(initial + k3 * dt, t + dt);
    return initial + (k1 + 2*k2 +2*k3 + k4)/6 * dt;
}

int main(){
        Timing timing;
        ranlux4_01 r_seed(time(0));

        timing.tic();

        double C=5, GCa=4.4, GK=8, GLeak=2, ECa=120, EK=-84, ELeak=-60;
        double factor=0.04, V1=-1.2, V2=18, V3=2, V4=30, Vspk=0;
        double tend=1000, dt=0.01,t=0;

        // Allocate inhibitory and excitatory neurons
        int N=10000;
        double inb_fraction=0.2;
        int net_in=N*inb_fraction;
        int net_ex=N-net_in;


        //Generating connectivity
        double Connection_ex=800, Connection_inb=200;
        double con_probex=Connection_ex/net_ex;
        double con_probinb=Connection_inb/net_in;
        double JE=0.01, JI=0.01,Jpre;
        bernoulli_distribution r_netProbex(con_probex);
        bernoulli_distribution r_netProbinb(con_probinb);
        bernoulli_distribution *r_net;
        vector<vector<double> > net_con;

        for (int i=0;i<N;i++){
            vector<double> temp;
            for (int j=0;j<N;j++){
                if (j<net_in){
                    r_net=&r_netProbinb;
```

```
                        Jpre=JI;
                }
                else{
                        r_net=&r_netProbex;
                        Jpre=JE;
                }
                if (r_net->operator()(r_seed)==1 && j!=i)
                        temp.push_back(Jpre);
                else
                        temp.push_back(0);
        }
                net_con.push_back(temp);
}

//Neuron Initialization
vector<double>VV;
vector<double>V_temp;
vector<double>nn;
vector<double>spike;
uniform_real<double>r_real(-30,-10);
uniform_real<double>r_real2(0.0,1.0);

for (int k=0;k<N;k++){
        V_temp.push_back(r_real(r_seed));
        VV.push_back(r_real(r_seed));
        nn.push_back(r_real2(r_seed));
        spike.push_back(0);
}

//External input
double nuext=120;
double Jext=0.15;
poisson_distribution<int> ext_rate(nuext*dt);


// Generating voltage bin
double bin=0.8;
double Vmin=-90, Vmax=70;
int size_max=(Vmax-Vmin)/bin;
vector<double > array;

for(int j=0;j<=size_max;j++){
        array.push_back(0);
}

// Generating files
ofstream out_file3("voltage");
ofstream out_file4("spike");

membrane membrane1;
```

```
potassium potassium1;

while(t<tend){

    for (int k=0;k<N;k++){

        double m_inf=0.5*(1+tanh((VV[k]-V1)/V2));
        double tau_R=1/cosh((VV[k]-V3)/(2*V4));
        double n_inf=0.5*(1+tanh((VV[k]-V3)/V4));

        potassium1. tau_R=tau_R;
        potassium1. n_inf=n_inf;
        potassium1. factor=factor;
        potassium1.     r=runge_kutta_4th(potassium1,
nn[k], 0, dt);

        membrane1. C=C;
        membrane1. GCa=GCa;
        membrane1. m_inf=m_inf;
        membrane1. ECa=ECa;
        membrane1. GK=GK;
        membrane1. n=potassium1. r;
        membrane1. EK=EK;
        membrane1. GLeak=GLeak;
        membrane1. ELeak=ELeak;
        membrane1. Iint=spike[k]/dt;
        membrane1. Inoise=ext_rate(r_seed)*Jext/dt;
        membrane1.    q=runge_kutta_4th(membrane1,
VV[k], 0, dt);

        V_temp[k]=VV[k];
        VV[k]=membrane1. q;
        nn[k]=potassium1. r;

        // probability density
        int index=round((VV[k]-Vmin)/bin);
        array[index]++;
    }


    for (int j=0;j<=size_max;j++){
        out_file3<<array[j]/(N)<<",";
     }
     out_file3<<endl;

     for (int j=0;j<=size_max;j++){
        array[j]=0;
     }

    for(int k=0; k<N; k++){
```

```
                        spike[k]=0;
                }

                int r=0;
                for (int k=0; k<N;k++){
                        if (VV[k]>Vspk && V_temp[k]<Vspk){
                                r++;
                                for(int j=0;j<N;j++){
                                        if (net_con[j][k]!=0){
                                                spike[j]+=net_con[j][k];
                                        }
                                }
                        }
                }
                out_file4<<r<<endl;
        }
        out_file3.close();
        out_file4.close();
        V_temp.clear();
        VV.clear();
        nn.clear();
        spike.clear();
        net_con.clear();
        array.clear();

        timing.toc();
        cout << setw(10) << timing.diff()<< "s" << endl;
        return 0;
}
```

A.2 Population Density Approach (PDA)

```
# include<iostream>
#include <iomanip>
# include<cmath>
# include<fstream>
# include<vector>
# include<ctime>
# include<tr1/random>
#include<omp.h>

# define PI 3.141592654

using namespace std;
using namespace std::tr1;

void Simpson (double, int, vector<double>, double & );

class PopD{
        public:
                double dV;
                double ds;
                double fes, Fv1,Fv2,Fv3,Fn1,Fn2,Fn3,FVV,FNN;
                double N_inf;
                double SOD;
                double cc;
                vector<double>::iterator it_Vbegin;
                vector<double>::iterator it_Psbegin;
                vector<double>::iterator it_Qsbegin;
                vector<double>::iterator it_Rsbegin;

                int k;
                int j;
                int sod_max;
                int Vmax;
                double z;


                PopD(int sod_max, int Vmax,  int k, int j, double dV, double ds,
double cc, double N_inf, double SOD, double Fv1, double Fv2, double Fv3,
double Fn1, double Fn2, double Fn3, vector<double>::iterator it_Vbegin,
vector<double>::iterator it_PSbegin, vector<double>::iterator it_QSbegin,
vector<double>::iterator it_RSbegin):

                        dV(dV),
```

```
                ds(ds),
                fes(fes),
                Fv1(Fv1),
                Fv2(Fv2),
                Fv3(Fv3),
                Fn1(Fn1),
                Fn2(Fn2),
                Fn3(Fn3),
                N_inf(N_inf),
                SOD(SOD),
                cc(cc),
                it_Vbegin(it_Vbegin),
                it_Psbegin(it_PSbegin),
                it_Qsbegin(it_QSbegin),
                it_Rsbegin(it_RSbegin),
                j(j),
                sod_max(sod_max),
                Vmax(Vmax),
                k(k){}

        ~PopD(){}

        double operator()(double y, double t){


                if (cc>*(it_Vbegin+k))
                {
                        FVV=(Fv1*(*(it_Psbegin+k-1))-(Fv2*y))/dV;
                }
                else
                {
                         FVV=(Fv2*y-(Fv3*(*(it_Psbegin+k+1))))/dV;
                }

                if (N_inf>SOD)
                {
                        FNN=(Fn1*(*(it_Qsbegin+k))-(Fn2*y))/ds;
                }
                else
                {
                        FNN=(Fn2*y-(Fn3*(*(it_Rsbegin+k))))/ds;
                }

                return(-(fes+FVV+FNN));

        }
};

class Timing {
        public:
```

```
        Timing(){
                startt=time(NULL);
                endt=time(NULL);
        }
        ~Timing(){}
        void tic(){
        startt=time(NULL);
        }
        void toc(){
        endt=time(NULL);
        }
        time_t diff(){
        return endt-startt;
        }
    private:
        time_t startt, endt;
};

double interpolation(double x0, double x1, double x2, double y0, double y1, double y2, double x){
    return (x-x1)*(x-x2)*y0/((x0-x1)*(x0-x2))+(x-x0)*(x-x2)*y1/((x1-x0)*(x1-x2))+(x-x0)*(x-x1)*y2/((x2-x0)*(x2-x1));
}

template <typename function>double runge_kutta_4th(function equation, double initial, double t, double dt){
    double k1 = equation(initial, t);
    double k2 = equation(initial + 0.5 * k1 * dt, t + 0.5 *dt);
    double k3 = equation(initial + 0.5 * k2 * dt, t + 0.5 *dt);
    double k4 = equation(initial + k3 * dt, t + dt);
    return initial + (k1 + 2*k2 + 2*k3 + k4)/6 * dt;
}

int main ()
{
    Timing timing;
    ranlux4_01 r_seed(time(0));
    int time = 0;

    timing.tic();

    vector<double> V;
    vector<double> sod;
    vector<double> c_cons;
    vector<double> m_inf;
    vector<double> tau_R;
    vector<double> n_inf;
    vector<double> Jint_thr;
    vector<vector<double> > P;
    vector<vector<double> > Ps;
```

```
	// Parameter
	double C=5, GCa=4.4, GK=8, GLeak=2, ECa=120, EK=-84, ELeak=-
60;
	double dV=0.2, ds=0.001;
	double Vth=70, Vmin=-80, Vs=0;
	double Jext=0.15,nuext=120;
	double JE=0.01,JI=-0.01,Connection_ext=800, Connection_inb=200;
	double factor=0.04, V1=-1.2, V2=18, V3=2, V4=30;
	double Fn1, Fn2, Fn3, Fv1, Fv2, Fv3, fext, FE, FI, firing=0,
firing_temp=0,x, y, J_thr,b_cons;
	double sod_min=0, sod_th=1;
	double dt=0.001, tend=1000, t=0;
	int Vmax=int((Vth-Vmin)/dV);
	int Vspk=int((Vs-Vmin)/dV);
	int sod_max=int((sod_th-sod_min)/ds);


	for (int n=0; n<=sod_max; n++){
		P.push_back(vector<double>());
		Ps.push_back(vector<double>());
		c_cons.push_back(0);
		sod.push_back(n*ds+sod_min);
		Jint_thr.push_back(0);

		for (int i=0;i<=Vmax;i++){
			P[n].push_back(0);
			Ps[n].push_back(0);
		}
	}

	for (int i=0;i<=Vmax;i++){
		V.push_back(Vmin+i*dV);
		m_inf.push_back(0.5*(1+tanh((V[i]-V1)/V2)));
		n_inf.push_back(0.5*(1+tanh((V[i]-V3)/V4)));
		tau_R.push_back(1/cosh((V[i]-V3)/(2*V4)));
	}

	// Initial Condition
	double miu_ini=-20;
	double sigma_ini=5;
	for (int n=1;n<=sod_max-1;n++){
		for (int i=1;i<=Vmax-1;i++){
			Ps[n][i]=exp(-(pow((sod[n]-0.1874),2)))*exp(-
(pow((V[i]-miu_ini),2))/(2*pow(sigma_ini,2)))/(sigma_ini*sqrt(2*PI));

		}
	 }

	for (int k=0;k<=Vmax;k++){
```

```
		b_cons=0;
		for (int j=0; j<=sod_max-1;j++){
			b_cons=b_cons+0.5*ds*(Ps[j][k]+Ps[j+1][k]);
		}
		c_cons[k]=b_cons;
}

b_cons=0;
for (int k=0;k<=Vmax-1;k++){
		b_cons=b_cons+0.5*dV*(c_cons[k]+c_cons[k+1]);
}

for (int n=0;n<=sod_max;n++){
		for (int i=0; i<=Vmax;i++){
			Ps[n][i]=Ps[n][i]/b_cons;
		}
}

for (int k=0;k<=Vmax;k++){
		b_cons=0;
		for (int j=0; j<=sod_max-1;j++){
			b_cons=b_cons+0.5*ds*(Ps[j][k]+Ps[j+1][k]);
		}
		c_cons[k]=b_cons;
}

ofstream out_file1("firing");
ofstream out_file2("probability");

int kk=0;

while (t<=tend)
{
		// Flux across threhold, J_thr
		// a. Flux due to external input across threshold
		double Jes=0;
		double a = V[Vspk]-Jext;
		double b = V[Vspk];
		double hh=0.001;
		int n=int((b-a)/hh);
		int i=1;
		if (i>n){
			Jes=0;
		}
		else{
			while (i<=n){
				double Q;
				int index=0;
				double m=a+(i-0.5)*hh;
```

```
                        Q = interpolation(V[Vspk-1],V[Vspk],V[Vspk+1],c_cons[Vspk-1],c_cons[Vspk],c_cons[Vspk+1],m);
                        Jes=Jes+hh*Q;
                        i=i+1;
                }
        }

        //b. Flux due to excitatory neurons across threshold
        double Jess=0;
        a=V[Vspk]-JE;
        b=V[Vspk];
        n=int((b-a)/hh);
        i=1;
        if (i>n){
                Jess=0;
        }
        else{
                while (i<=n){
                        double Q;
                        int index=0;
                        double m=a+(i-0.5)*hh;

                        Q = interpolation(V[Vspk-1],V[Vspk],V[Vspk+1],c_cons[Vspk-1],c_cons[Vspk],c_cons[Vspk+1],m);
                        Jess=Jess+hh*Q;
                        i=i+1;
                }
        }


        // c. Flux due to intrinsic ion channel across threshold
        for (int j=0;j<=sod_max;j++){
                double at=GCa*m_inf[Vspk]+GK*sod[j]+GLeak;
                double bt=GCa*m_inf[Vspk]*ECa+GK*sod[j]*EK+GLeak*ELeak;
                double ct=bt/at;
                double yy;

                if (ct>V[Vspk]){
                        Jint_thr[j]=-at/C*(V[Vspk]-bt/at)*Ps[j][Vspk];
                }
                else{
                        Jint_thr[j]=0;
                }

        }

        Simpson( ds, sod_max/2, Jint_thr,J_thr);
        firing=J_thr+nuext*Jes +firing_temp*Connection_ext*Jess;
        out_file1<<firing<<endl;
```

```
firing_temp=firing;

#pragma omp parallel for
for (int j=1;j<=sod_max-1;j++){

        for (int k=1;k<=Vmax-1;k++){

                // a. Excitatory Flux due to external network
                double m=V[k]-Jext;
                double Q;
                int index=0;
                if (m<=V[0] || V[Vmax]<=m){
                        Q = 0;
                }
                else if (V[Vmax-1]<=m && m<V[Vmax]){
                        Q  =  interpolation(V[Vmax-2],V[Vmax-
1],V[Vmax],Ps[j][Vmax-2],Ps[j][Vmax-1],Ps[j][Vmax],m);
                }
                else if (V[0]<m && m<=V[1]){
                        Q                                       =
interpolation(V[0],V[1],V[2],Ps[j][0],Ps[j][1],Ps[j][2],m);
                }
                else{
                        Q               =               interpolation(V[k-
1],V[k],V[k+1],Ps[j][k-1],Ps[j][k],Ps[j][k+1],m);

                }
                fext=nuext*(Ps[j][k]-Q);

                //Excitatory Flux due to connected neurons
                double mm=V[k]-JE;
                double QQ;
                int indexx=0;
                if (mm<=V[0] || V[Vmax]<=mm){
                        QQ = 0;
                }
                else if (V[Vmax-1]<=mm && mm<V[Vmax]){
                        QQ          =          interpolation(V[Vmax-
2],V[Vmax-1],V[Vmax],Ps[j][Vmax-2],Ps[j][Vmax-1],Ps[j][Vmax],mm);
                }
                else if (V[0]<mm && mm<=V[1]){
                        QQ                                      =
interpolation(V[0],V[1],V[2],Ps[j][0],Ps[j][1],Ps[j][2],mm);
                }
                else{
                        QQ              =               interpolation(V[k-
1],V[k],V[k+1],Ps[j][k-1],Ps[j][k],Ps[j][k+1],mm);
                }
                FE=Connection_ext*firing*(Ps[j][k]-QQ);
```

```
                                //Inhibitory Flux due to connected neurons
                                double mmi=V[k]+JI;
                                double QQi;
                                if (mmi<=V[0] || V[Vmax]<=mmi){
                                        QQi = 0;
                                }
                                else if (V[Vmax-1]<=mmi &&
mmi<V[Vmax]){
                                        QQi = interpolation(V[Vmax-
2],V[Vmax-1],V[Vmax],Ps[j][Vmax-2],Ps[j][Vmax-1],Ps[j][Vmax],mmi);
                                }
                                else if (V[0]<mmi && mmi<=V[1]){
                                        QQi =
interpolation(V[0],V[1],V[2],Ps[j][0],Ps[j][1],Ps[j][2],mmi);
                                }
                                else{
                                        QQi = interpolation(V[k-
1],V[k],V[k+1],Ps[j][k-1],Ps[j][k],Ps[j][k+1],mmi);
                                }
                                FI=Connection_inb*firing*(QQi-Ps[j][k]);

                                //Flux due to ion channel

                                double aa=GCa*m_inf[k]+GK*sod[j]+GLeak;
                                double
bb=GCa*m_inf[k]*ECa+GK*sod[j]*EK+GLeak*ELeak;
                                double cc=bb/aa;

                                Fv1=(GCa*m_inf[k-1]*(V[k-1]-
ECa)+GLeak*(V[k-1]-ELeak))/C+GK*sod[j]*(V[k-1]-EK)/C;
                                Fv2=(GCa*m_inf[k]*(V[k]-
ECa)+GLeak*(V[k]-ELeak))/C+GK*sod[j]*(V[k]-EK)/C;
                                Fv3=(GCa*m_inf[k+1]*(V[k+1]-
ECa)+GLeak*(V[k+1]-ELeak))/C+GK*sod[j]*(V[k+1]-EK)/C;
                                Fn1=factor*(sod[j-1]-n_inf[k])/tau_R[k];
                                Fn2=factor*(sod[j]-n_inf[k])/tau_R[k];
                                Fn3=factor*(sod[j+1]-n_inf[k])/tau_R[k];

                                PopD PopD(sod_max, Vmax, k, j, dV, ds,cc,
n_inf[k], sod[j], Fv1, Fv2, Fv3, Fn1, Fn2, Fn3, V.begin(), Ps[j].begin(), Ps[j-
1].begin(), Ps[j+1].begin() );
                                PopD.fes=fext+FE-FI;
                                PopD.z=runge_kutta_4th(PopD,Ps[j][k],0,dt);
                                P[j][k]=PopD.z;
                        }

                }

                for (int k=0;k<=Vmax;k++){
                        b_cons=0;
```

```
                        for (int j=0; j<=sod_max-1;j++){
                                b_cons=b_cons+0.5*ds*(P[j][k]+P[j+1][k]);
                        }
                        c_cons[k]=b_cons;
                        out_file2<<c_cons[k]<<",";
                }
                out_file2<<endl;

                cout<<"t=       "<<t<<"\tintegration=       "<<b_cons<<"\tFiring=
"<<firing<<endl;
                timing.toc();
                cout << setw(10) << timing.diff()<< "s" << endl;
        }

        Ps=P;
        t=t+dt;
        cout << setw(10) << timing.diff()<< "s" << endl;
}

void Simpson(double dx, int m, vector<double>data, double & ans)
{
        double temp_f1=0;
        for (int k=1;k<=m-1;k++){
                temp_f1=temp_f1+data[2*k];
        }

        double temp_f2=0;
        for (int k=1;k<=m;k++){
                temp_f2=temp_f2+data[2*k-1];
        }

        ans = dx/3*(data[0] + data[2*m]) + 2*dx/3*temp_f1 +
4*dx/3*temp_f2;
}
```

A.3 The Astrocyte Model

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <deque>
#include <vector>
#include <cmath>

using namespace std;

template <typename function> double runge_kutta_4th(function equation,
double initial, double t, double dt){
        double k1 = equation(initial,t);
        double k2 = equation(initial + 0.5 * k1 * dt, t + 0.5 * dt);
        double k3 = equation(initial + 0.5 * k2 * dt, t + 0.5 * dt);
        double k4 = equation(initial + k3 * dt, t + dt);
        return initial + (k1 + 2*k2 +2*k3 + k4)/6 * dt;
}

class IP3temp{
                public:
                        double tau;
                        double star;
                        double z;
                        double k4;
                        double v4;
                        double Ca;
                        double alpha;
                        double rip3;

                IP3temp():
                        tau(7),
                        star(0),
                        k4(1.1),
                        v4(0),
                        Ca(0),
                        alpha(0),
                        rip3(0),
                        z(0){}

                ~IP3temp(){}
                        double operator () (double y, double t){
                                return(v4*((Ca+(1-alpha)*k4)/(Ca+k4))-
y/tau+2*rip3);
```

```
                }
};

class qtemp{
        public:
                double alpha;
                double beta;
                double z;

        qtemp():
                alpha(0),
                beta(0),
                z(0){}

        ~qtemp(){}
                double operator () (double y, double t){
                        return(alpha*(1-y)-beta*y);
                }
};

class Catemp{

        public:

                double c1;
                double v1;
                double ninf;
                double pinf;
                double q;
                double CaER;
                double v2;
                double v3;
                double k3;
                double Jchan;
                double Jleak;
                double Jpump;
                double z;

        Catemp():

                c1(0.185),
                v1(6),
                q(0),
                pinf(0),
                v2(0.11),
                v3(0.9),
                k3(0.1),
                Jpump(0),
                ninf(0),
                CaER(0),
```

```
                z(0){}

        ~Catemp(){}
                double operator () (double y, double t){

                        Jpump=0.9*pow(y,2)/(pow(y,2)+pow(k3,2));

    Jchan=c1*v1*pow(pinf,3)*pow(ninf,3)*pow(q,3)*(CaER-y);
                        Jleak=c1*v2*(CaER-y);


                        return (Jchan-Jpump+Jleak);
                }
};

class interaction{
        public:
                double tauCa;
                double factor;
                double Ca;
                double Cath;
                double q;
                double z;

        interaction():
                tauCa(0),
                factor(0.5),
                Ca(0),
                Cath(0.2),
                z(0){}

        ~interaction(){}
                double operator() (double y, double t){
                        if (Ca>Cath)
                                q=1;
                        else
                                q=0;
                        return (-y/tauCa+(1-y)*factor*q);
                }

};

int main(){

        //Constant parameter
        double          d1=0.13,        d3=0.94,d5=0.08234,             c0=2,
c1=0.185,a2=0.00123,d2=1.3,rip3=0;
        double pinf, Jchan, Jleak, Jpump,ninf,CaER;

        ofstream out_file("Ca");
```

```
ofstream out_file1("CaER");
ofstream out_file2("IP3R");

//Initialize
double IP3=0.01, Ca=0.15,q=0.65,f=0;

//Simulation time
double t=0, dt=0.1,tend=2000;

while (t<tend){

        IP3temp IP3temp1;
        Catemp Catemp1;
        qtemp qtemp1;
        interaction interaction1;

        //IP3 production
        IP3temp1.v4=0.1;
        IP3temp1.alpha=0.2;
        IP3temp1.Ca=Ca;
        IP3temp1.rip3=rip3;
        IP3temp1.z=runge_kutta_4th(IP3temp1,IP3,0,dt);

        qtemp1.alpha=a2*d2*(IP3+d1)/(IP3+d3);
        qtemp1.beta=a2*Ca;
        qtemp1.z=runge_kutta_4th(qtemp1,q,0,dt);

        pinf=IP3/(IP3+d1);
        ninf=Ca/(Ca+d5);
        CaER=(c0-Ca)/c1;

        Catemp1.pinf=pinf;
        Catemp1.ninf=ninf;
        Catemp1.CaER=CaER;
        Catemp1.q=q;
        Catemp1.z=runge_kutta_4th(Catemp1,Ca,0,dt);

        Ca=Catemp1.z;
        q=qtemp1.z;
        IP3=IP3temp1.z;

        interaction1.tauCa=10;
        interaction1.Ca=Ca;
        interaction1.z=runge_kutta_4th(interaction1,f,0,dt);
        f=interaction1.z;

        out_file<<Ca<<endl;
        out_file1<<CaER<<endl;
        out_file2<<q<<endl;
```

```
            t=t+dt;
        }
        return 0;
}
```

A.4 The Modified conductance-based ML Neuron Model

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <deque>
#include <tr1/random>
#include <vector>
#include <cmath>
#include <ctime>
#include <stdio.h>

using namespace std;
using namespace std::tr1;

template <typename function> double runge_kutta_4th(function equation,
double initial, double t, double dt){
        double k1 = equation(initial,t);
        double k2 = equation(initial + 0.5 * k1 * dt, t + 0.5 * dt);
        double k3 = equation(initial + 0.5 * k2 * dt, t + 0.5 * dt);
        double k4 = equation(initial + k3 * dt, t + dt);
        return initial + (k1 + 2*k2 +2*k3 + k4)/6 * dt;
}

class potassium{
        public:
                double tau_R;
                double n_inf;
                double factor;
                double r;

                potassium():
                        tau_R(0),
                        r(0),
                        factor(0),
                        n_inf(0){}

                ~potassium(){}

                        double operator () (double y, double t){
                                return( factor/tau_R*(n_inf-y) );
                        }
};

class slowahp{
```

```
        public:
                double beta;
                double gamma;
                double tau;
                double r_slow;
                double v;
                double sp;

        slowahp():
                beta(0),
                gamma(1),
                tau(2000),
                r_slow(0),
                v(0),
                sp(0){}

        ~slowahp(){}

                        double operator () (double y, double t){
                                return( (1/(1+exp((beta-v)/gamma))-y)/tau );
                        }
};

class synapse{
        public:
                double beta;
                double sy1;
                double sinf;
                double sp;

                synapse():
                        beta(3),
                        sinf(0),
                        sy1(0),
                        sp(0){}

                ~synapse(){}

                        double operator () (double y, double t){
                                return(-y/beta+0.1*sp);
                        }
        };

class synapses{
        public:
                double beta;
                double sy2;
                double sinf;
                double sp2;
```

```
		synapses():
			beta(3),
			sinf(0),
			sy2(0),
			sp2(0){}

		~synapses(){}

			double operator () (double y, double t){
				return(-y/beta+0.1*sp2);
			}
	};

class membrane{
	public:
		double C;
		double GNa;
		double m_inf;
		double ENa;
		double GK;
		double Gahpslow;
		double EK;
		double GLeak;
		double ELeak;
		double q;
		double Is1;
		double Is2;
		double IDC;
		double w;

		membrane():
			C(0),
			GNa(0),
			m_inf(0),
			ENa(0),
			GK(0),
			EK(0),
			GLeak(0),
			Gahpslow(0),
			ELeak(0),
			Is1(0),
			Is2(0),
			IDC(0),
			w(0),
			q(0){}

		~membrane(){}

			double operator () (double y, double t){
```

```
                                return( 1/C * ( -GNa*m_inf*(y-ENa) -
GK*w*(y-EK) - Gahpslow*(y-EK)- GLeak*(y-ELeak)  - (y*Is1-Is2) + IDC));

                        }
                };


class Timing {
        public:
                Timing(){
                        startt=time(NULL);
                        endt=time(NULL);
                }
                ~Timing(){}
                void tic(){
                startt=time(NULL);
                }
                void toc(){
                endt=time(NULL);
                }
                time_t diff(){
                return endt-startt;
                }
        private:
                time_t startt, endt;
};

int main(){
                Timing timing;
                ranlux4_01 r_seed(time(0));

                timing.tic();

                //~Constant parameter
                double C=5, GNa=7.5, GK=8, GLeak=2, ENa=60, EK=-84,
ELeak=-60;
                double factor=0.04,V1=-1.2, V2=18, V3=12, V4=30, Vspk=0,t;
                double delta=0.067,ss;
                double tend=500000,dt=0.1;

                //~Excitatory neurons parameters
                double VsynE=0,EPSP=1;

                //~Inhibitory neurons parameters
                double  VsynI=-80,IPSP=-1.5;

                //Neuron Initialization
                vector<double>VV;
                vector<double>V_temp;
                vector<double>nn;
```

```
		vector<double>nn_slow;
		vector<double>nn_fast;
		vector<double>s1;
		vector<double>s2;
		vector<double>Vsyn;
		vector<double>IDC;
		vector<double>spike;
		vector<double>sum_spikes;
		vector<double>sum_spikes2;
		vector<vector<double> >J;
		double sy, s_inf,f;
		double Isyn_Ex=0, Isyn_Inb=0,Inoise=0, VG_Ex=0,
VG_Inb=0,check;

		uniform_real<double>r_real(-70,-65);
		uniform_real<double>r_real2(0.0,0.6);
		uniform_real<double>r_real3(0.0,1.0);

		//~Allocate inhibitory and excitatory neurons
		int N=10000;
		double inh_fraction=0.2;
		int net_in=N*inh_fraction;
		int net_ex=N-net_in;

		//~Generating connectivity
		double net_mean=2000;
		double net_bias_in=0.8;
		double net_bias_ex=1.2;
		double
net_probInIn=(net_mean*net_bias_in)/(net_ex+net_in*net_bias_in);
		double
net_probInEx=(net_mean)/(net_ex+net_in*net_bias_in);
		double
net_probExEx=(net_mean)/(net_ex+net_in*net_bias_ex);
		double
net_probExIn=(net_mean*net_bias_ex)/(net_ex+net_in*net_bias_ex);
		bernoulli_distribution r_netprobExEx(net_probExEx);
		bernoulli_distribution r_netprobInIn(net_probInIn);
		bernoulli_distribution r_netprobExIn(net_probExIn);
		bernoulli_distribution r_netprobInEx(net_probInEx);
		bernoulli_distribution *r_net;
		double Jpre;

		for (int i=0;i<N;i++){
			J.push_back(vector<double>());
			Vsyn.push_back(0);
			for (int j=0;j<N;j++){
				J[i].push_back(0);
			}
```

```
		if (i<net_in){
			Vsyn[i]=VsynI;
		}
		else{
			Vsyn[i]=VsynE;
		}
	}


	for (int i=0;i<N;i++){

		for (int j=0;j<N;j++){

			if (i<net_in && j<net_in){
				r_net=&r_netprobInIn;
				Jpre=IPSP/(Vsyn[i]-
ELeak)*55.84311504;
			}
			else if (i<net_in){
				r_net=&r_netprobInEx;
				Jpre=IPSP/(Vsyn[i]-
ELeak)*55.84311504;;
			}
			else if (i>net_in && j<net_in){
				r_net=&r_netprobExIn;
				Jpre=EPSP/(Vsyn[i]-
ELeak)*55.84311504;;
			}
			else if (i>net_in){
				r_net=&r_netprobExEx;
				Jpre=EPSP/(Vsyn[i]-
ELeak)*55.84311504;;
			}
			if (r_net->operator()(r_seed)==1 && j!=i){
				J[i][j]=Jpre;
			}
			else{
				J[i][j]=0;
			}
		}

	}

	//~Endogenous cells
	double applied_current=63.3;
	uniform_real<double> r_real4(0,applied_current);

	//~Initialize parameter
	for (int k=0;k<N;k++){
		V_temp.push_back(-60);
```

```
                VV.push_back(-60);
                nn_fast.push_back(0.01);
                nn_slow.push_back(0.01);
                nn.push_back(0.01);
                s1.push_back(0.01);
                s2.push_back(0.01);
                spike.push_back(0);
                sum_spikes.push_back(0);
                sum_spikes2.push_back(0);
                IDC.push_back(r_real4(r_seed));
        }

        ofstream out_file2("spike");
        t=0;

        while(t<tend){

                #pragma omp parallel for
                for (int k=0;k<N;k++){

                        membrane membrane1;
                        potassium potassium1;
                        fastahp fastahp1;
                        slowahp slowahp1;
                        synapse synapse1;
                        synapses synapse2;

                        double m_inf=0.5*(1+tanh((VV[k]-V1)/V2));
                        double tau_R=1/cosh((VV[k]-V3)/(2*V4));
                        double n_inf=0.5*(1+tanh((VV[k]-V3)/V4));

                        potassium1. tau_R=tau_R;
                        potassium1. n_inf=n_inf;
                        potassium1. factor=factor;
                        potassium1.     r=runge_kutta_4th(potassium1,
nn[k], 0, dt);

                        fastahp1. v=VV[k];
                        fastahp1.       r_fast=runge_kutta_4th(fastahp1,
nn_fast[k], 0, dt);

                        slowahp1. v=VV[k];
                        slowahp1.   r_slow=runge_kutta_4th(slowahp1,
nn_slow[k], 0, dt);

                        synapse1. sp=sum_spikes[k];
                        synapse1. sy1=runge_kutta_4th(synapse1, s1[k],
0, dt);
                        s1[k]=synapse1. sy1;
```

```
                synapse2. sp2=sum_spikes2[k];
                synapse2. sy2=runge_kutta_4th(synapse2, s2[k],
0, dt);
                s2[k]=synapse2. sy2;

                membrane1. C=C;
                membrane1. GNa=GNa;
                membrane1. m_inf=m_inf;
                membrane1. ENa=ENa;
                membrane1. Gahpslow=1.8*slowahp1. r_slow;
                membrane1. GK=GK;
                membrane1. w=potassium1. r;
                membrane1. EK=EK;
                membrane1. GLeak=GLeak;
                membrane1. ELeak=ELeak;
                membrane1. IDC=IDC[k];
                membrane1. Is1=s1[k];
                membrane1. Is2=s2[k];
                membrane1.      q=runge_kutta_4th(membrane1,
VV[k], 0, dt);

                V_temp[k]=VV[k];
                VV[k]=membrane1. q;
                nn_fast[k]=fastahp1. r_fast;
                nn[k]=potassium1.r;
                nn_slow[k]=slowahp1.r_slow;
        }

        int ex_spike=0, inb_spike=0;

        for (int k=0; k<N; k++){
                spike[k]=0;
                sum_spikes[k]=0;
                sum_spikes2[k]=0;
        }

        for (int k=0; k<N;k++){

                if (VV[k]>Vspk && V_temp[k]<Vspk){
                        spike[k]=1;
                                for (int j=0;j<N;j++){
                                if (J[k][j]!=0){
                                        sum_spikes[j]+=J[k][j];

        sum_spikes2[j]+=J[k][j]*Vsyn[k];
                                }

                                if (k<net_in)
                                        inb_spike++;
                                else
```

```
                                        ex_spike++;
                                    }
                                }
                            }
                            out_file2<<ex_spike<<","<<inb_spike<<endl;
                            t=t+dt;
                        }
                        timing.toc();
                        cout << setw(10) << timing.diff()<< "s" << endl;
                        return 0;
}
```