Tampere University

Emma Huhtala

ANALYSIS OF COMPARTMENTAL MODELS FOR CORTICAL NEURONS

Faculty of Information
Technology and Communication
Sciences
Master of Science Thesis
April 2020

ABSTRACT

Emma Huhtala: Analysis of compartmental models for cortical neurons
Master of Science Thesis
Tampere University
Master's Degree Programme in Electrical Engineering
April 2020

Pyramidal cells are the primary excitatory neurons in the cerebral cortex of the mammalian brain. They have complex morphological structures which have been shown to influence the computational capabilities and higher cognitive functions of the brain. In addition to complex structure, pyramidal cells also have diverse biophysical properties. The biophysical properties are conventionally described using conductance-based models, where the neuronal cell membrane can be represented as an equivalent electrical circuit. Here, the lipid bilayer is described as a capacitor and the ion channels as conductances. To account for both the complex morphology and the intricate biophysical properties of pyramidal cells, a state-of-the-art compartmental modeling framework is adopted. In this framework, the morphology is compartmentalized and each segment is represented using a conductance-based approach.

Many different compartmental models of cortical pyramidal cells exist but thus far a systematic review of these models has not been conducted. The aim of this thesis is to review publications with compartmental models of cortical pyramidal cells and to evaluate the models quantitatively. The emphasis of this work lies on peer-reviewed models implemented in the NEURON simulation environment as well as models available in public repositories. NEURON is a simulation tool widely used in the field of computational neuroscience. The literature review comprehensively lists the publications with compartmental models of cortical pyramidal cells and analyzes the implemented models and their properties. The review provides guidelines on choosing a suitable compartmental model for a cortical pyramidal cell.

Another aim of the thesis is a simulation-based analysis of a selected model using two new and promising tools. At the same time, the properties of these tools are evaluated. First, the role of the morphological complexity of the model is assessed by using the Neuron_Reduce Python toolbox. Second, the uncertainty of model parameters is examined with the Uncertainpy Python toolbox. The conductance-based models often depend on uncertain parameters which cause inherent variability in model dynamics. The uncertainty arises from various sources, such as different measurement methods, measurement error or biological variability. The model used in the simulations was a layer 2/3 pyramidal cell model from the rat somatosensory cortex.

With moderate reduction of the model (preserving 10-25% of the original model compartments), the Neuron_Reduce toolbox simplifies the cell while replicating the behavior of the original model. However, a dramatic reduction (3% of the original model compartments) led to quantitative differences in the model dynamics. The results obtained with Uncertainpy revealed that some model parameters have a greater effect on the sensitivity of the model indicating that these parameters should be carefully considered. The simulations carried out in this work prove that both of the Python packages used here are useful tools in analyzing models in neuroscience.

Keywords: conductance-based neuron model, compartmental model, pyramidal cell, cerebral cortex, simulation tool

TIIVISTELMÄ

Emma Huhtala: Kortikaalisten hermosolujen kompartmenttimallien analyysi
Diplomityö
Tampereen yliopisto
Sähkötekniikan diplomi-insinöörin tutkinto-ohjelma
Huhtikuu 2020

Pyramidaalisolut ovat nisäkkäiden isoaivokuoren primaarisia eksitatorisia hermosoluja. Niiden rakenteellisen morfologian on osoitettu vaikuttavan korkeampiin kognitiivisiin aivotoimintoihin. Monimutkaisen rakenteen lisäksi pyramidaalisoluilla on myös monipuolisia biofysikaalisia ominaisuuksia. Biofysikaalisia ominaisuuksia voidaan mallintaa konduktanssiin perustuvilla malleilla, joissa hermosolujen solukalvon lipidikaksoiskerros esitetään kondensaattorina ja ionikanavat konduktansseina. Yksityiskohtaisempaa mallinnustapaa tarvitaan, kun otetaan huomioon sekä monimutkainen morfologia että monipuoliset biofysikaaliset ominaisuudet. Tässä työssä tutkitaan kompartmenttimallinnuspohjaista lähestymistapaa, vakiintunutta matemaattista menetelmää, jossa huomioidaan sekä solujen monimutkainen rakenne että monipuoliset biofysikaaliset ominaisuudet.

Aivokuoren pyramidaalisolujen erilaisia kompartmenttimalleja on useita, mutta toistaiseksi näitä malleja ei olla tarkasteltu systemaattisesti. Tässä työssä tarkastellaan 50 julkaisua, joissa on käytetty aivokuoren pyramidaalisolujen kompartmenttimalleja sekä evaluoidaan kyseiset mallit. Työssä keskitytään NEURON-simulaatioympäristössä toteutettuihin vertaisarvioituihin sekä julkisissa arkistoissa saatavilla oleviin malleihin. NEURON-simulaattori on simulointityökalu, jota käytetään laajalti laskennallisen neurotieteen alalla. Kirjallisuuskatsauksessa tarkastellaan kattavasti aivokuoren pyramidaalisolujen kompartmenttimalleja sisältävät julkaisut ja analysoidaan käytetyt mallit ja niiden ominaisuudet. Katsaus tarjoaa suosituksia sopivan aivokuoren pyramidaalisolun kompartmenttimallin valintaan.

Työn toinen tavoite on simulaatioon perustuva analyysi, jossa yhtä valittua mallia simuloidaan käyttämällä kahta viimeaikaista Python-työkalua. Ensin mallin morfologisen monimutkaisuuden merkitystä arvioidaan käyttämällä Neuron_Reduce Python -kirjastoa. Toiseksi mallin parametrien epävarmuutta tutkitaan käyttämällä Uncertainpy -nimistä Python-kirjastoa. Mallien konduktanssit riippuvat usein epävarmoista parametreista, mikä tarkoittaa, että kyseisillä parametreillä on luontainen vaihtelevuus. Epävarmuus voi johtua useista eri lähteistä, kuten esimerkiksi erilaisista mittausmenetelmistä, mittausvirheestä tai biologisesta vaihtelusta. Lisäksi työn tavoitteena on arvioida molemmat simulaatioissa käytetyt työkalut. Simulaatioissa käytetään rotan somatosensorisen aivokuoren kerroksen 2/3 pyramidaalisolumallia.

Maltillinen yksinkertaistaminen (10-25% alkuperäisistä kompartmenteista) Neuron_Reduce-työkalulla yksinkertaisti mallia siten, että kyseinen yksinkertaistettu malli käyttäytyi kuten alkuperäinen malli. Dramaattinen yksinkertaistaminen (3% alkuperäisistä kompartmenteista) johti kuitenkin kvantitatiivisiin eroihin mallin käyttäytymisessä. Uncertainpy-työkalulla saadut tulokset paljastivat, että joillakin malliparametreilla on suurempi vaikutus mallin herkkyyteen, mikä osoittaa, että näitä parametreja tulisi harkita huolellisesti. Työssä suoritetut simulaatiot osoittavat, että molemmat tässä käytetyt Python-kirjastot ovat hyödyllisiä työkaluja neurotieteen mallien analysointiin.

Avainsanat: konduktanssiin perustuva hermosolumalli, kompartmenttimalli, pyramidaalisolu, isoaivokuori, simulaatiotyökalu

PREFACE

This M.Sc. thesis was done in the Computational Neuroscience Research Group at Tampere University. This research has received partial funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 785907 (Human Brain Project SGA2). The work was also supported by Academy of Finland (decision Nos 297893 and 318879).

I wish to express my gratitude to my supervisor Adjunct Professor Marja-Leena Linne, Ph.D., for providing me with the interesting topic and introducing me to the world of computational neuroscience. I would like to thank my supervisor Senior Researcher Jugoslava Aćimović, Ph.D., for all the guidance and advice she has provided me throughout the thesis process. I would also like to express special thanks to my supervisor Mikko Lehtimäki, M.Sc., for helping me with the simulations and answering to my endless questions during the process. I also express my warmest thanks to all the other members of the CNS group for the supporting work atmosphere and especially Ippa for proof-reading the thesis.

Lastly, I would like to thank my family for all the support I've had during the M.Sc. work.

Tampere, 29th April 2020

Emma Huhtala

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AP	Action potential
BAP	Backpropagating action potential
EPSP	Excitatory postsynaptic potential
GABA	Gamma-aminobutyric acid
GUI	Graphical user interface
HH	Hodgkin-Huxley
IPSP	Inhibitory postsynaptic potential
ISI	Inter-spike interval
L*n*	Layer *n*
NA	Not available
PC	Pyramidal cell
PN	Pyramidal neuron
a	Radius
C_m	Membrane capacitance
g_x	*X* conductance
h	Gating variable
m	Gating variable
n	Gating variable
S_i	First order Sobol index
V_m	Membrane voltage
λ	Length constant
τ	Time constant

1. INTRODUCTION

What happens in our brain when we see, hear, think or learn something? Particularly, what happens in the cerebral cortex, in the brain area where these higher cognitive brain functions occur? We know that the cerebral cortex is inhabited by pyramidal cells, which suggests that these cells play important roles in higher cognitive brain functions. This has attracted interest among neuroscientists for over a hundred years. Our knowledge of pyramidal cells has increased considerably over the years, but some aspects are not fully understood. To help us comprehend these aspects, computational neuroscience was introduced.

Computational neuroscience uses principles from physics, chemistry, mathematics and computer science to analyze, model and understand the behavior of cells and their subcellular mechanisms, networks and systems in the brain. This thesis focuses on computational methods and tools used to study single neurons, and particularly the compartmental framework used to model cortical pyramidal cells. In compartmental modeling, a neuron is divided into smaller segments, compartments, that are represented as connected electric circuits [1]. Each compartment is presented as a group of coupled equations that describes the electrical properties at a certain point of the neuron.

Pyramidal cells, or pyramidal neurons, are the most populous excitatory cell type in the mammalian cortex and receive both excitatory and inhibitory inputs [2]. The sizes and shapes of pyramidal cells vary between the different brain areas and layers of the cortex [2]. Understanding the structure and function of pyramidal cells comprehensively will enhance our understanding of sensory information processing in the cerebral cortex and ultimately, the behavior of mammalian species.

The aims of this thesis can be divided into two parts. The literature review part reviews publications with compartmental models of cortical pyramidal cells and categorize the models. The emphasis is on resources that are available in public databases. The second part is a simulation-based analysis in which the replicability of a model is analyzed. The model is also studied with two recently published Python-based toolboxes. Uncertainpy [3] is used for sensitivity analysis of the uncertain parameters of a model and Neuron_reduce [4] for simplifying the morphological complexity of the model.

This thesis is structured as follows: after the introduction, the theoretical framework is presented. The theoretical framework is divided into neurobiology, bioelectricity and modeling sections. It is followed by a review of the existing compartmental models. After that, materials and methods used in the simulations are presented in Chapter 4. The results are presented in Chapter 5 and they are discussed in Chapter 6. The results and findings are concluded in Chapter 7.

2. THEORETICAL FRAMEWORK

This chapter introduces the scientific framework of the thesis. First, biological background is presented, in which the structure and function of the cerebral cortex is introduced, as well as the structure and function of cortical pyramidal cells. Biological background is followed by electrical background, in which the electrophysiological properties of neurons are introduced. The scientific framework ends with presentating the main focus of this thesis; computational and compartmental modeling.

2.1 Neurobiology

In the following subsections the cerebral cortex and pyramidal cells are introduced. In the first subsection, layers and brain areas of the cerebral cortex are presented as well as cell types and connections in the cerebral cortex. This section is followed by an introduction of pyramidal cells, and their bioelectric signaling and excitability.

2.1.1 The cerebral cortex

The cerebral cortex is the outer layer of the cerebrum. It is a convoluted and layered sheet of grey matter mainly composed of cell bodies. Human cerebral cortex is 2-3 mm thick but the surface area is several hundred square centimeters because of the convolution [5]. It has approximately 16 billion neurons and 60 billion glial cells [6]. In comparison, the whole brain contains approximately 86 billion neurons and 85 billion non-neuronal cells [6]. The cerebral cortex has four main lobes: the frontal, parietal, temporal and occipital lobes. These areas are important in complex brain functions, such as memory, perception, language and emotions.

The cerebral cortex can be divided into six layers. Each layer has its own characteristic cells. The shapes, sizes and densities of the neurons and glial cells vary between each cortical layer. The cells are categorized into different layers based on the location of the cell soma [5]. Even though some dendrites of a neuron might reach out to the other layers, the cell body determines the layer. Each layer is connected differently to other cortical areas as well as subcortical areas, such as to the thalamus. The cortical layers are numbered from superficial (layer 1) to deep (layer 6). Both numbers and Roman numerals are used.

Layer I (L1 or LI) is the molecular layer [5]. The molecular layer contains hardly any somas, only a few inhibitory neurons. Layer I consists of mainly glial cells and incoming axons and apical dendrites from other layers. Layer II (L2 or LII) is the external granular layer, named after its small granule cells. It also contains small pyramidal neurons and many stellate neurons [5]. The third layer, also known as the external pyramidal layer, is composed of medium-sized pyramidal neurons. Layer IV is called the internal granular layer, consisting mostly of granule and stellate cells [5]. The fifth layer is the internal

pyramidal layer, which contains large pyramidal cells [5]. Pyramidal cells are discussed in more detail in the next chapter. The deepest layer, layer VI, is called the multiform layer [5]. It contains small polymorphic and fusiform neurons, as well as a few large pyramidal cells.

Layers I-III are also called supragranular layers. The nerve cells in the supragranular layers form primarily intracortical connections [7]. The intracortical connections can be either associational, meaning that they are connected with other areas of the same hemisphere, or they can be commissural, meaning that they are connected to the opposite hemisphere, mainly through the corpus callosum [7]. In addition to being connected intracortically to the supragranular layers, layer IV has also thalamocortical connections [7]. The intragranular layers, including layers V and VI, connect mainly to the subcortical areas [7]. Layer V connects primarily to the basal ganglia, the brain stem and the spinal cord where as layer VI to the thalamus [7]. As mentioned before, they also have afferent projections from the other layers of the cerebral cortex.

In the beginning of the 1900s, German anatomist Korbinian Brodmann divided the cerebral cortex into 52 cytoarchitecturally distinct regions [8, p. 12]. Although the partitioning was based on the cellular organization of the cortex, it also corresponds well to the functionality of the cortex [7]. In a more generic way, cerebral cortex can be divided into three functionally different areas, which are motor, sensory and associative [8, p. 11]. These areas are illustrated in Figure 1.

Motor areas control voluntary movements. They are located in the precentral gyrus in both hemispheres [8, p. 11]. In Figure 1, motor regions are illustrated with pink color. Motor areas are divided into the primary motor cortex, the premotor cortex and the supplementary motor areas. Motor areas in the right half of the cortex are responsible for the movement of the left side of the body and areas in the left side are responsible for the right side of the body [8, p. 11].

Sensory areas receive and interpret sensory information, such as sight, touch, hearing, taste and smell. Most of the sensory information passes to the cortex via the thalamus. The brain areas that receive these inputs from the thalamus are called the primary sensory areas [7]. They include the primary visual cortex, the primary auditory cortex and the primary somatosensory cortex. The primary visual cortex is located in the posterior tip of the occipital lobe [7], which is illustrated in Figure 1 with green color. It is responsible for processing visual information. The primary auditory cortices interpret sound and are responsible for the ability to hear. In Figure 1, auditory cortex is represented with yellow. They are located in the transverse gyri of the temporal lobe [8, p. 10]. The primary somatosensory cortex receives and process information that is sensed with touch. The primary somatosensory cortex is located behind the primary motor cortex, in the postcentral gyrus [8, p. 10]. Like the motor areas, the sensory areas also receive and process information from the opposite side of the body. For example, the auditory cortex in the left hemisphere receives information from the right ear.

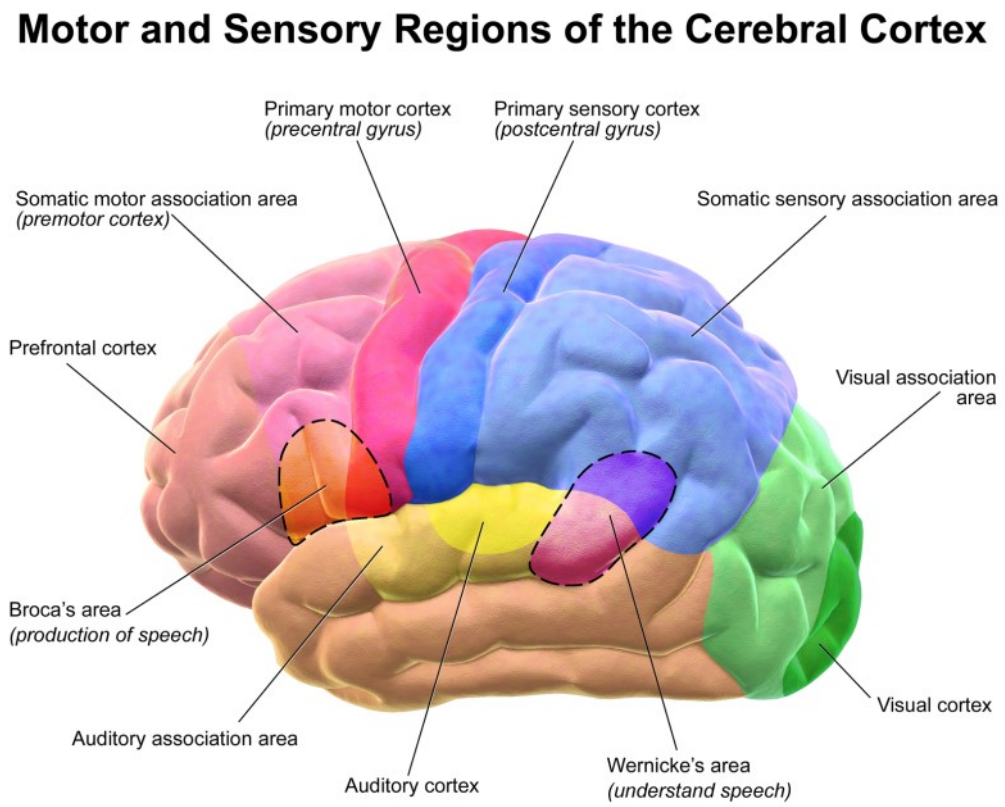

Figure 1: The cerebral cortex with functionally distinct areas, taken from [9].

Areas that do not belong to sensory or motor areas are called associative areas. Associative areas integrate information from other areas of the cortex and can relate the information to past experiences [8, p.10]. Associative areas are responsible for higher cognitive functions, such as thinking, memory and language. They are found in all four lobes [8, p.10]. In the frontal lobe, associative areas are especially responsible for thinking and behavior. The areas in the parietal, temporal and occipital lobes associate sensory information to the information stored in memory. The associative regions are illustrated in Figure 1 with lighter colors.

2.1.2 Pyramidal cells

Pyramidal cells (PCs) or pyramidal neurons (PNs) are the most common neurons in the cerebral cortex [2]. Many cognitive processes take place in the cerebral cortex and therefore pyramidal neurons play an important role in fundamental brain functions. Pyramidal neurons receive both excitatory and inhibitory inputs [2]. They are also found in mammalian hippocampus and amygdala [10]. Pyramidal cells were first discovered by the Spanish neuroscientist Santiago Ramón y Cajal in the late 1900th century [10].

Pyramidal cells are named after their pyramidal-shaped cell body, soma [10]. They have also other characteristic structural features, which are an apical dendrite, several basal dendrites and an axon [10]. The apical dendrite is a large dendrite, which arises from the apex of the soma. It bifurcates at various distances of the soma. The bifurcated branches may also bifurcate. At the head of the cell, the branches branch abundantly, forming an apical tuft. Basal dendrites emerge from the base of the soma. The further from the soma the basal dendrites are, the more they branch. The lone axon arises from the base of the soma and branches abundantly, forming many synaptic connections to other neurons. The large number of branches increase the ability to transmit and receive signals between other neurons [10]. Figure 2 illustrates the structure of a pyramidal cell.

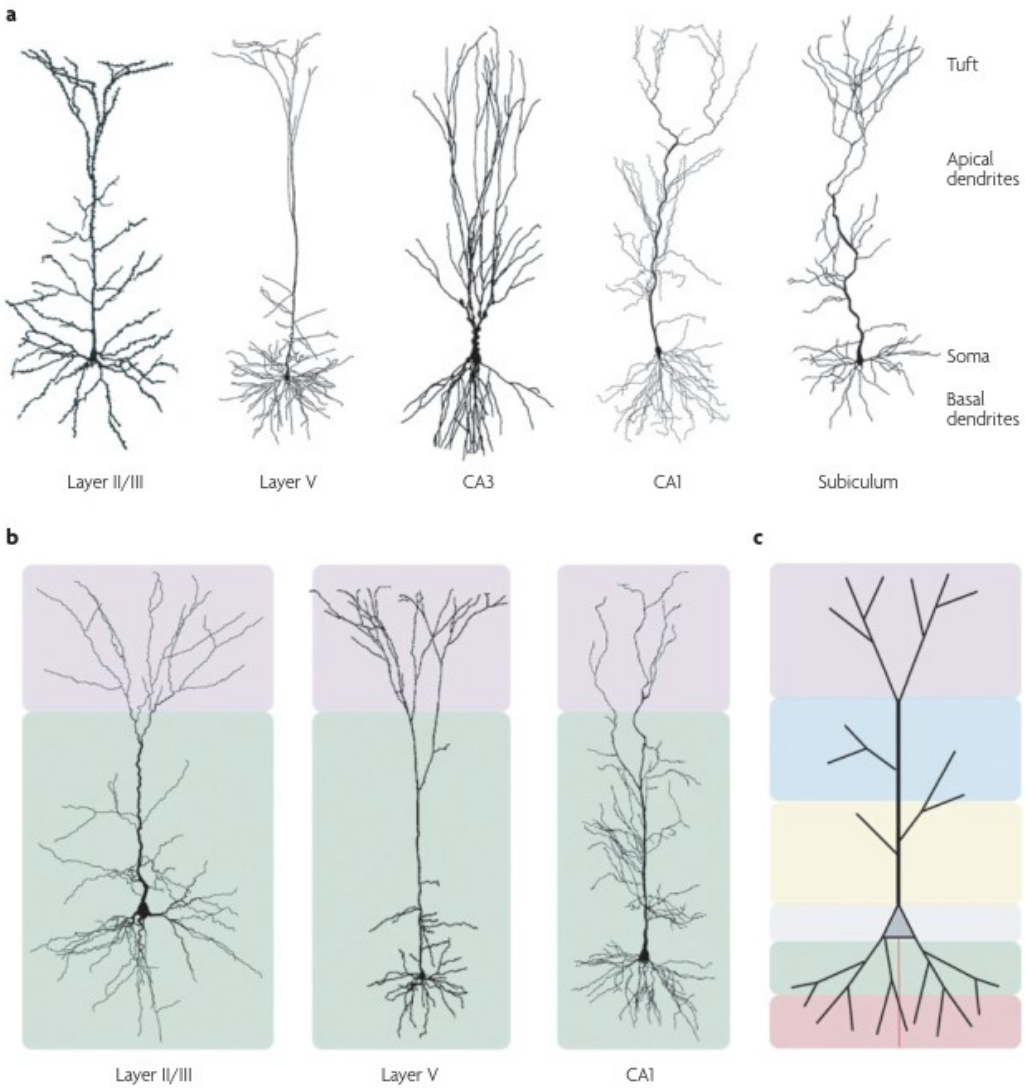

Figure 2: Structures of pyramidal cells. a-b) Layers II/III and V indicate areas in the cerebral cortex, CA1, CA3 and subiculum areas in the hippocampus. c) The principle structure of pyramidal cell. The complex morphology can be represented with multiple connected segments. The figure is from [10].

Layers II/III and V indicate areas in the cerebral cortex and CA1, CA3 and subiculum areas in the hippocampus. As seen in Figure 2, the structures of pyramidal cells vary based on the location of the neuron. For example, the apical dendrites of pyramidal neurons in the hippocampus branch closer to the soma compared to the apical dendrites of pyramidal neurons in L5 of the neocortex. Even the pyramidal neurons in different layers of the cerebral cortex have different structures. Panel C of Figure 2 illustrates the principle structure of a pyramidal cell, which shows also the pyramidal-shaped soma.

The dendrites of pyramidal neurons are covered with thousands of spines. Spines are tiny membranous protrusions that consist of a spine head and a spine neck [11]. The structure of a spine is illustrated in Figure 3. Part A of Figure 3 illustrates pieces of dendrites with spines on their surfaces. As seen in part E of Figure 3, sizes and shapes vary between spines. The spine neck is a thin structure that connects the spine head to the dendrite. The spine head is usually a fuller structure but can be indistinguishable from the spine neck. In pyramidal cells, spines form the postsynaptic site of most excitatory (and some inhibitory) synapses and increase the receptive surface of the cell. A larger receptive area increases the ability to process information. However, the precise functions of spines are yet to be discovered [10].

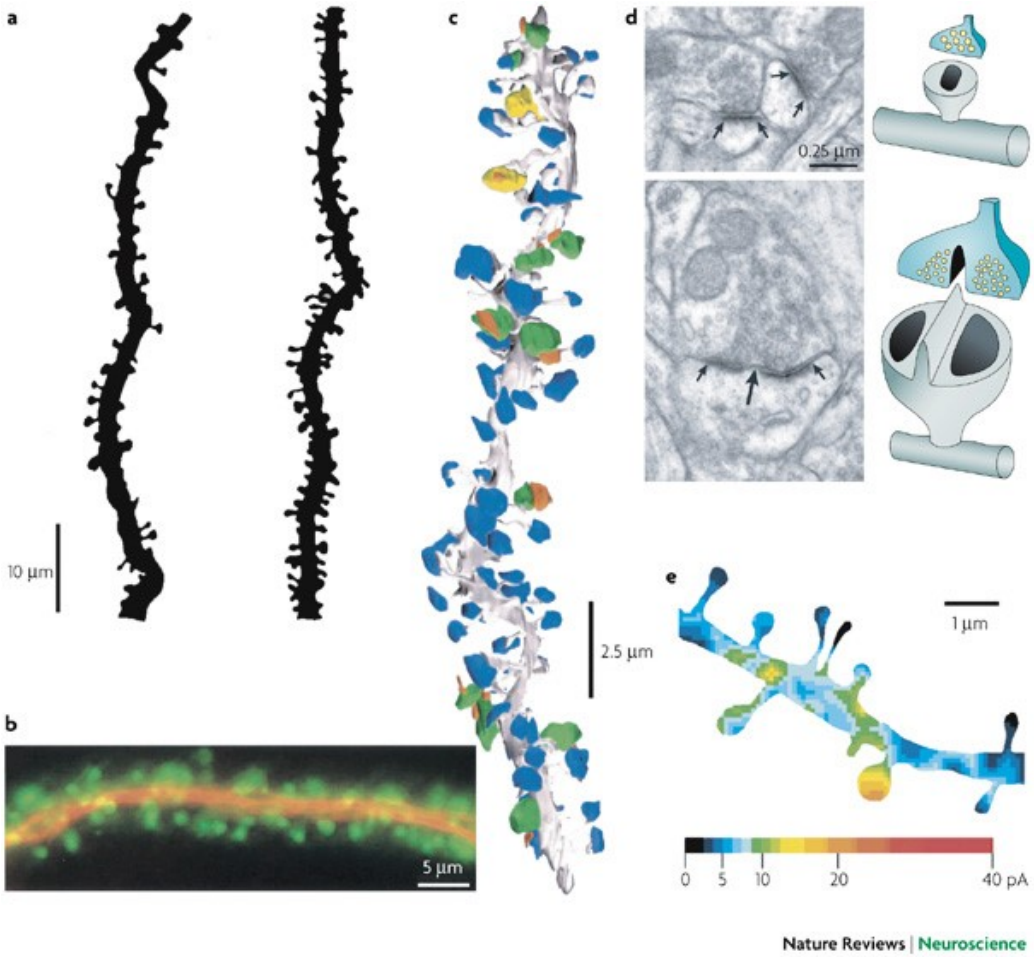

Figure 3: Structures of dendritic spines. a-c) Segments of dendrites with spines. d) Spines as postsynaptic sites. The figure is from [10].

2.1.3 Pyramidal cell excitability

Information is transferred between neurons electrically and chemically. As in other cells, the cell membrane of pyramidal cells consists of a lipid bilayer that is semipermeable to certain ions. The lipid bilayer is an insulator that separates two conducting solutions, the cytoplasm inside the cell and the extracellular fluid outside the cell. In the cell membrane, there are ion channels that allow ions to pass through the cell membrane. At the resting state the outside of the cell membrane has a positive charge and inside a negative charge due to the different distribution of the charged ions. This difference between the charges causes an electrical potential across the cell membrane called a membrane potential, denoted by V_m. It is defined as

$$V_m = V_{in} - V_{out}, \tag{1}$$

where V_{in} is the potential inside of the cell and V_{out} the potential outside of the cell. At rest the membrane voltage in neurons is usually between -60 mV and -70 mV. [8, p. 126]

The excitability of pyramidal cells depends on both passive and active electrical properties. Neurons have three different passive properties that are considered important in electrical signaling: the membrane resistance, the membrane capacitance and the intracellular axial resistance, [8, p. 140] which are described in more detail in Chapter 2.2.2. When a positive charge is injected into the cell the membrane voltage becomes less negative in a process called depolarization. To a certain range of depolarization, the neuron acts as a resistor [8, p.141]. When the membrane voltage reaches a threshold voltage, a point where the neuron will fire an action potential, the neuron no longer acts as a resistor [8, p. 141]. This can be explained with active electrical properties that arise from voltage-gated ion channels.

As mentioned above, an action potential is generated when a stimulus causes a sufficient increase from the resting membrane voltage. This depolarization causes the voltage-gated sodium (Na^+) ion channels to open. Then Na^+ ions are allowed to flow inward the cell membrane, which causes an increase in the membrane voltage. This causes more and more sodium channels along the cell membrane to open, which spreads the action potential along the cell membrane. When the membrane potential has become high enough, the sodium channels inactivate stopping the influx of sodium ions. The depolarization phase is followed by a repolarization phase, in which the voltage-gated potassium (K^+) ion channels open and potassium ions flow outside. It causes the membrane voltage to decrease below the resting potential meaning that the cell membrane hyperpolarizes temporarily. To re-establish the resting state, the potassium channels close and the sodium-potassium pump returns the Na^+ and K^+ ions to the initial conditions. [8, pp. 150-151]

An action potential is transferred to the next neuron via a junction called synapse [8, p.175]. In synapses, the input signal is called synaptic potential and output is usually an action potential. The process, in which a neuron determines the output from the input, is

called synaptic integration [12]. At a chemical synapse the transmitting cell (presynaptic cell) releases chemical substances called neurotransmitters to the junction [8, p. 182]. The neurotransmitters bind to the receptors that are located in the cell membrane of the postsynaptic cell [8, p. 182]. The response in the postsynaptic cell can be either excitatory or inhibitory. If the response is inhibitory, the signal received in the postsynaptic cell is called an inhibitory postsynaptic potential (IPSP) [8, p. 209]. In this case, the cell membrane becomes hyperpolarized and therefore is less likely to propagate an action potential [13]. If the response is excitatory, the cell membrane in the postsynaptic cell becomes depolarized and this increases the probability to generate an action potential [13]. This change in the membrane voltage is called an excitatory postsynaptic potential (EPSP) [13]. The neurotransmitters in the excitatory synapses are mostly glutamate and in the inhibitory synapses gamma-aminobutyric acid (GABA) [2].

Usually, the synaptic potentials generated by a single presynaptic neuron are not strong enough to excite the postsynaptic cell to initiate an action potential [8, p. 222]. Generally, either consecutive inputs from a single presynaptic neuron or simultaneous inputs from many presynaptic neurons are needed to trigger an action potential in the postsynaptic cell. The process, in which the repeated synaptic potentials from the same presynaptic neuron are added together in the postsynaptic cell, is called temporal summation [8, p. 222]. It is affected by the time constant, a parameter that is used to determine the time course of the synaptic potential [8, p. 222]. Another process, in which simultaneous inputs from multiple presynaptic neurons are added together, is called spatial summation [8, p. 223]. A parameter that describes how the voltage decays as it spreads passively is called the length constant [8, p. 223]. The definitions of time and length constants are presented in Section 2.2.2.

In addition to synaptic integration, dendrites of pyramidal cells have also other excitatory responses. One of them is backpropagating action potential (BAP). BAP is an action potential that is initiated in the axon and then propagates backwards into the dendrites [14]. The efficacy of back-propagating action potential can vary between the dendrites in the same pyramidal cell [14]. The reason for this is that the dendrites have different distribution and densities of the ion channels that are responsible for the propagation of action potential.

Action potentials, or spikes, can also be initiated in the dendrites and they are then called dendritic spikes [10]. These spikes are propagated through Na^+ and/or Ca^{2+} channels [14]. The local voltage threshold for generating these spikes in the dendrites is higher than the corresponding voltage threshold for action potential initiation in the axon meaning that the initiation of dendritic spikes needs relatively strong input [14]. The spike is propagated through opening of ion channels, in the same manner as in the axon.

2.2 Bioelectricity

In this chapter, the background on bioelectricity and computational framework for describing bioelectricity is presented. First, the Hodgkin-Huxley (HH) model for modeling

bioelectricity of neurons, is described. After that, neuronal cable theory, the framework for modelling signal propagation along axons and dendrites of neurons, is presented.

2.2.1 Hodgkin-Huxley model

Hodgkin-Huxley model, developed by Alan Lloyd Hodgkin and Andrew Fielding Huxley in 1952, is a mathematical model that describes how action potentials are initiated and propagated in neurons [15]. Hodgkin and Huxley performed experiments with the squid giant axon and found out that the membrane of the axon has three different types of currents. The currents were later understood and explained with time and voltage-dependent ion channels [16]. Two of the ion channels are voltage-gated and time-dependent meaning that their conductances depend on the time and voltage across the membrane. Part of the voltage-gated channels are selective to sodium ions and the other part to potassium ions. The third type of channels are known as leakage channels, in which the conductance is constant. They are responsible for the resting membrane potential.

The semipermeable cell membrane is considered as a small cylinder with cytoplasm inside and extracellular medium around it on the outside. The membrane works as an insulator between two conductors, forming a capacitor. An injected current $I(t)$ can either charge the capacitor (I_C) or flow through the channels (I_i, where i indicates the current type) in the cell membrane. The capacitive current I_C can be expressed as an ordinary differential equation $I_C = C\frac{dV_m}{dt}$, where C is the capacitance of the membrane. Then we can present the following equation

$$I(t) = C\frac{dV_m}{dt} + \sum_k I_i(t). \tag{2}$$

The channels are represented as resistors. The resistances for sodium and potassium channels are denoted by R_{Na} and R_K, respectively. The leak resistance is denoted by R_L. In many cases, it is more convenient to use conductance rather than resistance. According to Ohm's law, the leak current is $I_L = g_L(V_m - E_L)$, where V_m is the membrane voltage, E_L is the leak reversal potential and g_L is the leak conductance, which is an inverse of resistance, $g_L = \frac{1}{R_L}$. Similarly, the ionic currents can be expressed as $I_{Na} = g_{Na}(V_m - E_{Na})$ and $I_K = g_K(V_m - E_K)$.

Based on their measurements, Hodgkin and Huxley proposed a mathematical description, which represents how ions flow through the channels as a function of time and voltage. Hodgkin and Huxley introduced gating variables m, h and n, which describe the activation (*m*) and inactivation (*h*) properties of Na+ ionic currents and the activation (*n*) for the K+ current. The gating variables are defined with first order kinetic equations found in the literature [17]. Describing the ionic currents with their gating variables we get the following form

$$\sum_k I_k = g_{Na}m^3h(V_m - E_L) + g_K n^4(V_m - E_K) + g_L(V_m - E_L). \tag{3}$$

Inserting the sum of ionic currents (Eq. 3) into Equation 2, we get the complete form of Hodgkin and Huxley model

$$I(t) = C_m \frac{dV}{dt} + g_{Na} m^3 h (V_m - E_L) + g_K n^4 (V_m - E_K) + g_L (V_m - E_L). \quad (4)$$

Hodgkin and Huxley performed their experiments with the squid giant axon. However, if we want a model that represents more complex structures, such as mammalian cells, more conductances than sodium and potassium are required. The mammalian cells have more numerous and complex set of ion channels in their cell membrane (e.g. see [18] or [19]). These models, where the ion channels are represented as conductances and the lipid bilayer as a capacitor, are called conductance-based models [20]. They are the simplest biophysical representations of excitable cells. When the complexity of the morphology would like to be incorporated, compartmental models are used. More on these models can be found in Section 2.3.2.

2.2.2 Neuronal cable theory

Neuronal cable theory is a mathematical model, which describes passive propagation of charge through an insulated cable surrounded by a conductor. The theory was first developed by the neuroscientist Wilfrid Rall in 1959 [21]. His purpose was to develop a model that would describe the electrical properties of morphologically and physically realistic dendritic trees. In 1964, Rall complemented the cable theory with compartmental modelling approach [22].

Rall made some key assumptions when developing the cable theory. Firstly, he assumed that the membrane is passive and uniform. Secondly, the intracellular fluid has ohmic resistivity and the cross section is constant. He also assumed that the extracellular medium has zero resistivity. This means that the potential outside the membrane is equivalent to ground. Lastly, he assumed that all the inputs are currents. Taking into account the assumptions made by Rall, the cable equation can be derived. The following description is made based on the derivation by Idan Segev [1].

As described before, the membrane acts as a capacitor. An injected current can flow either longitudinally (along the x-axis) or through the membrane. Because of the resistance of the cytoplasm, the voltage decreases as the longitudinal current I_i encounters it. The cytoplasmic resistivity r_i is defined as a resistance per unit length along the x-axis, with the unit of Ω/cm. According to Ohm's law, we get the following form

$$\frac{1}{r_i} \frac{\partial V}{\partial x} = -I_i, \quad (5)$$

where the negative sign is a consequence of the current flowing down the potential gradient.

The membrane current can either pass the membrane, represented as a membrane resistance r_m, or charge the membrane capacitance per unit length c_m. The units for membrane resistance and membrane capacitance are Ω·cm and F/cm, respectively. The change per unit length of the longitudinal current, expressed as $\frac{\partial I_i}{\partial x}$, is equal to the density of the membrane current i_m. The total membrane current consists of the resistive current, expressed as $i_r = \frac{V}{r_m}$ and the capacitive current, expressed as $i_c = c_m \frac{\partial V}{\partial t}$. Combining these, we get

$$\frac{\partial I_i}{\partial x} = -i_m = -(\frac{V}{r_m} + c_m \frac{\partial V}{\partial t}). \quad (6)$$

Combining equations 5 and 6, we get a second-order partial differential equation, known as the cable equation

$$\frac{1}{r_i}\frac{\partial^2 V}{\partial x^2} = c_m \frac{\partial V}{\partial t} + \frac{V}{r_m}. \quad (7)$$

In the Eq. 7, the variables r_i, r_m and c_m are considered for a unit length of the cylinder. If we want to use the actual variables, which are not geometry specific, we need new expressions for the resistances and capacitance. They are called specific resistance and capacitance, denoted by R_m, R_i and C_m. The relation between the membrane resistance and specific resistance is expressed as

$$r_m = \frac{R_m}{2\pi a}, \quad (8)$$

where a is the radius of the cross-section of cylinder. The relation between capacitances is

$$c_m = C_m 2\pi a. \quad (9)$$

The relation between the intracellular resistance per unit length and the actual resistance of the cytoplasm R_i is expressed as

$$r_i = \frac{R_i}{\pi a^2}. \quad (10)$$

The values of neuronal membrane resistance and capacitance are usually assumed to be 10,000 Ωcm^2 and 1 μF/cm^2, respectively [23]. However, a recent study suggested that the value for capacitance in human neocortical neurons is only half of the commonly used value [24].The value used for intracellular resistance is usually 100 Ωcm. However, these values vary largely between different cell types and even in the same cell type in different species.

Sometimes it is useful to use a length constant λ and a time constant τ. They are defined as follows

$$\lambda = \sqrt{r_m / r_i} = \sqrt{(\frac{a}{2}) \frac{R_m}{R_i}} \tag{11}$$

and

$$\tau = r_m c_m = R_m C_m. \tag{12}$$

Multiplying Equation 7 with r_m and substituting the expressions of the length and time constant, we get:

$$\lambda^2 \frac{\partial^2 V}{\partial x^2} - \tau \frac{\partial V}{\partial t} - V = 0. \tag{13}$$

If we need dimensionless units, we can use $\chi = x/\lambda$ and $T = t/\tau$. Then, Equation 13 becomes

$$\frac{\partial^2 V}{\partial \chi^2} - \frac{\partial V}{\partial T} - V = 0. \tag{14}$$

These forms of the cable equation can be used only with passive dendrites. However, Rall noticed that dendrites did not function as passive cables. Therefore, he implemented a new approach, which took account of the active properties of the dendrites. The new approach divides the dendrites into small segments, compartments, which are represented as a group of differential equations. This approach is presented in the next section.

2.3 Modeling in neuroscience

In the following chapter a short introduction to computational modeling in neuroscience is presented. It is followed by a description of compartmental modeling of neurons, a framework that is used to model the complex structures of pyramidal cells.

2.3.1 Computational modeling

Computational modeling allows us to better understand how complex systems, such as biological neuronal networks, function. It represents the theory in the form of a set of mathematical equations. Computational modeling, combined with experimental techniques, is an excellent aid for neuroscientists to increase their understanding of the phenomena, test theories and predict the outcomes. Theories expressed by precise mathematical language are less ambiguous than verbally described ones. Mathematical models describe the theory clearly and logically allowing the *in silico* experiments to be repeated and thus verified. In many cases, computational modeling is cheaper, easier and faster than biological experiments. Thus, simulations can be used to reduce the amount

of unnecessary biological experiments. For example, a larger number of hypotheses can be tested in silico, and then the most promising ones can be further explored via biological experiments. In biological research, ethical issues need to be considered whereas with computational modeling that is not an issue. [17]

Constructing a computational model of a complex system is not straightforward. The challenges arise from various perspectives, e.g. the type of the model to use and how detailed it should be [17]. For example, modelling ion channels requires a different type of model than modelling synapses between nerve cells. Additionally, although the models are built at the same level of analysis, they may describe the phenomenon with various levels of detail [17]. The models might include parameters that cannot be measured experimentally. In that case, the modeler needs to decide, how to deal with the unknown parameters. At best, the parameters are good estimates for the real value, but sometimes they are educated guesses [17]. A model is never completely accurate and understanding the limitations of a model is important.

To analyze the effects of uncertain parameters of a computational model, sensitivity analysis can be used. Sensitivity analysis studies how uncertain parameters in a mathematical model contribute to the output of the model [3]. It improves one's confidence in the model and in the selected parameters, especially in complex models with many uncertain parameters, such as in compartmental models of pyramidal cells.

2.3.2 Compartmental modeling

Multi-compartmental modeling, or often simply compartmental modeling, is a useful tool to model complex structures, such as dendritic trees of pyramidal cells. In compartmental models, dendrites or axons are divided into smaller segments, compartments, which are represented as connected electric circuits [17]. The circuits are connected to each other with axial resistivities, illustrated in Figure 4. A larger number of compartments improves accuracy of the model but at the same time increases the complexity and the required computational time and power.

Cable theory assumes neurons to only have passive electrical properties but this assumption is overturned in compartmental modeling. In compartmental models, the continuum description of a neuron is replaced with a discrete description of a neuron by a group of differential equations. Each compartment is presented with a group of equation thus, one model can include thousands of equations. Generally, numerical methods are used to solve the set of equations. For example, a neuron can be divided into compartments of length l each. The flow of ions from the jth compartment through active channels can be presented as an additional current I_j. With these variables, we can express Equation 7 with the following form

$$\frac{l^2}{R_i}\frac{\partial^2 V_j}{\partial x^2} = C_m \frac{\partial V_j}{\partial t} + \frac{V_j}{R_m} + I_j, \tag{15}$$

where V_j is the voltage in the jth compartment, and the resistances and capacitance are the actual values defined with the Equations 8-10. Writing a Taylor's series expansion for small values of l, we can write Equation 15 with the following form

$$\frac{V_{j+1}-2V_j+V_{j-1}}{R_i} = C_m \frac{dV_j}{dt} + \frac{V_j}{R_m} + I_j, \tag{16}$$

where $j+1$ and $j-1$ are the adjoining compartments. Equation 15 can be extended to include more compartments. For N compartments, we get N coupled equations like Equation 16. The equations are solved simultaneously for each time step.

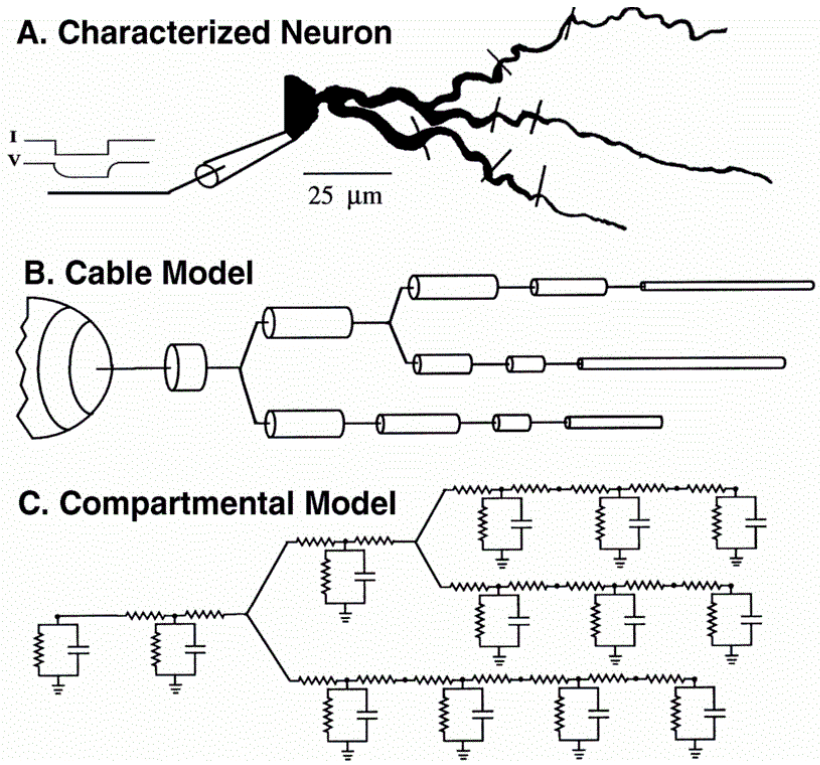

Figure 4: Compartmental model of a neuron, reproduced from [1].

Neurons, and especially pyramidal cells, have many other types of active channels in addition to sodium and potassium channels of the basic HH model. Different channels have an effect on the function of the neuron, including the shape of action potentials and the integration of synaptic inputs [17]. Although the densities of the ion channels have a critical impact on the function of the neuron, this information is sometimes not known. That means that they are often free parameters, and in a compartmental model with many compartments, this causes considerably large number of unknown parameters [17]. To reduce the number of free parameters, the morphology can be simplified or assumptions of the distribution of the ion channels can be made [17]. For example, the easiest way to do this is to place a single maximum conductance in all the compartments in the dendritic tree [17].

3. A REVIEW OF COMPARTMENTAL MODELS

Firstly, the publications covering the existing compartmental models of cortical pyramidal cells were listed in an Excel table (Appendix A.1). Over 50 studies that have been carried out between the years 1995-2020 were reviewed. Then the studies that used other simulation environments than NEURON were discarded as well as studies that had no model available in public databases. The thesis project work required a model that was implemented in NEURON because NEURON interfaces with the computational tools that are studied later. It is also one of the widest used simulation environments and it has a large user base. It is also actively developed and supported. The properties of the NEURON simulator are further discussed in Section 4.2.1.

For the 50 studies, 40 had the model available in public repositories. 36 of those were implemented with NEURON simulation software. The remaining 36 studies and their models were taken into closer consideration. We were interested in what kind of experimentally obtained data the models were validated on, such as whether the models are based on human or rodent data. We were also interested in which brain area and layer of the cortex the cells are from. Another aspect we considered was the morphological reconstructions of the models. For example, we analyzed how realistic the morphologies are and how many compartments the cells were divided into. Lastly, we analyzed what biophysical properties the models have and how biophysically realistic they are. The comparison criteria are collected in the following list.

The comparison criteria of the models

- Implementation software
- Model availability
- Experimental data
- Brain area and layer of the cortex
- The number of compartments
- Biophysical properties

3.1 Experimental data

The experimental data is represented in Table 1. The experimental data in most studies were obtained from rodent brain slices, mostly rat slices and a couple of mouse slices. Four studies used human slices. However, from the four studies that used human brain slices three were done by the same group and they are all based on the same study [24,25,26]. Two *in vivo* studies are also included [27,28]. Some studies [23, 29-34] had no experimental data, but their models were based on other already validated models.

Table 1. *Publications and their validation data*

Author & Year	Validation data	Brain area & Layer
Aberra et al. 2018 [35]	rat slices	somatosensory, 2,3,5,6
Almog & Korngreen, 2014 [36]	rat slices	somatosensory, 5
Arkhipov et al. 2018 [37]	mouse slices	visual, 4
Bahl et al. 2012 [29]	no	5
Behabadi & Mel, 2014 [30]	no	5
Behabadi et al. 2012 [38]	rat slices	5
de Sousa et al. 2015 [31]	no	not specified
Deitcher et al. 2017 [26]	human and mouse slices	temporal, 2, 3
Doron et al. 2017 [32]	no	5
Eyal et al. 2016 [24]	human and mouse slices	temporal, 2,3
Eyal et al. 2018 [25]	human slices	temporal, 2,3
Galloni et al. 2019 [39]	mouse slices	visual, 5
Gidon et al. 2020 [40]	human slices	temporal, 2,3
Gouwens et al. 2018 [41]	mouse slices	visual, not specified
Hay et al. 2011 [18]	rat slices	5
Hu et al. 2009 [42]	rat slices, rat in vivo	prefrontal, 5
Kampa & Stuart, 2006 [43]	rat slices	somatosensory, 5
Kole et al. 2008 [44]	rat slices	somatosensory, 5
Kole et al. 2006 [45]	rat slices	somatosensory, 5
Larkum et al. 2009 [46]	rat slices	5
Li et al. 2013 [33]	no	5
Mainen et al. 1995 [47]	rat slices	5
Markram et al. 2015 [48]	rat slices	somatosensory, all
Mäki-Marttunen et al. 2017 [49]	no	5
Nevian et al. 2007 [50]	rat slices	5
Papoutsi et al. 2017 [51]	rat slices	prefrontal, 5
Poleg-Polsky, 2015 [23]	no	2,3,5
Polsky et al. 2009 [52]	rat slices	somatosensory, 5
Schaefer et al. 2003 [53]	rat slices	somatosensory, 5
Shai et al. 2015 [54]	mouse slices	visual, 5
Shu et al. 2007 [28]	ferret in vivo and slices	prefrontal, somatosensory, 5
Sidiropoulou & Poirazi. 2012 [19]	rat slices	prefrontal, 5
Smith et al. 2013 [27]	mice in vivo	visual, 2,3
Stuart & Spruston, 1998 [55]	rat slices	5
Traub et al. 2003 [56]	rat slices	temporal, 2,3
van Elburg & van Ooyen, 2010 [34]	no	visual, 5

Most of the experimentally validated cells are from layer 5, which was expected as that is the layer where the largest pyramidal cells are found. Some cells are from layers 2/3, which contain small and medium-sized pyramidal cells. Two studies [35, 48] included a pyramidal cell from layer 6, which was quite unexpected as pyramidal cells are sparse in that layer. A third of the publications did not specify the brain area and cell location. The brain area that were indicated as the source of the cells included the somatosensory,

prefrontal, temporal and visual cortices with the occurrences of 25%, 11%, 14% and 17%, respectively. In 36% of the studies the brain area was not specified. The layer where the modeled cells were from was 5 in 75% of the studies, layer 2/3 in 22% of the studies and layer 6 in 6% of the studies. In 6% of the studies the layer was unspecified. The percentage is over 100% because some studies had cells from more than one brain area and layer of the cortex.

3.2 Morphological reconstruction

The modelled cells have both simple and complex morphological reconstructions. The majority of the models have biologically realistic morphologies. Figure 5 illustrates both reduced and realistic morphologies of layer 5 pyramidal cells. The reduced morphology is reproduced with the script from [29] and the realistic, detailed morphology is reproduced with the script from [18]. The reduced morphology is very simple, a triangle-shaped soma, a basal dendrite, an apical dendrite, a dendritic tuft and an axon. The detailed, realistic morphology consists of a soma, several branching apical dendrites, abundant basal dendrites and an axon. The reduced morphology allows faster simulation but decreases the biological realism of the model. The morphology depends on the layer, but even cells in the same layer have different morphologies [26].

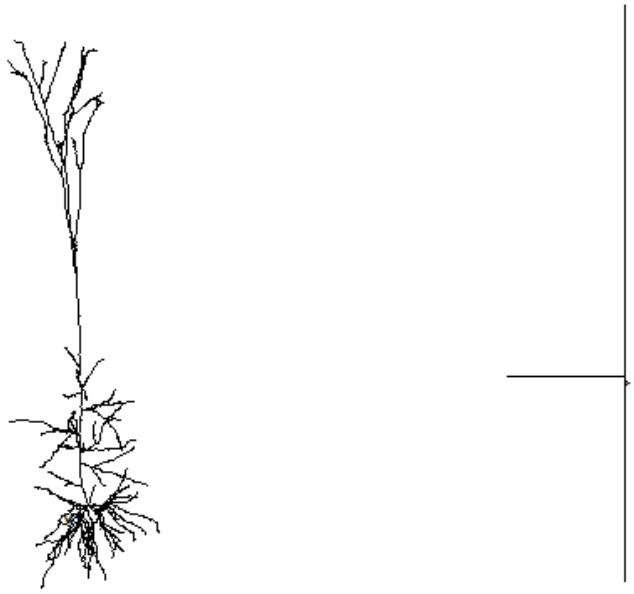

Figure 5: The realistic and reduced morphology of layer 5 pyramidal cell.

The number of compartments vary from two to a few thousands of compartments. The number of compartments in the existing models are listed in Table 2. Some publications did not mention the number of compartments in their model. Missing information is abbreviated in the table with NA, not available. Some publications mentioned only the amount of the dendritic compartments.

Table 2. *The number of compartments in the models*

Author & Year	Compartments
Aberra et al. 2018	NA
Almog & Korngreen, 2014	663
Arkhipov et al. 2018	91-264
Bahl et al. 2012	20
Behabadi & Mel, 2014	NA
Behabadi et al. 2012	2, 443
de Sousa et al. 2015	255 (dendritic compartments)
Deitcher et al. 2017	NA
Doron et al. 2017	NA
Eyal et al. 2016	738
Eyal et al. 2018	NA
Galloni et al. 2019	20, 200
Gidon et al. 2020	184
Gouwens et al. 2018	NA
Hay et al. 2011	average 200
Hu et al. 2009	836
Kampa & Stuart, 2006	1093
Kole et al. 2008	1903
Kole et al. 2006	1541
Larkum et al. 2009	2079
Li et al. 2013	176
Mainen et al. 1995	718
Markram et al. 2015	average 260
Mäki-Marttunen et al. 2017	4
Nevian et al. 2007	1331
Papoutsi et al. 2017	56
Poleg-Polsky, 2015	45-91 (L2/3), 135-200 (L5)
Polsky et al. 2009	45-251 (dendritic compartments)
Schaefer et al. 2003	NA
Shai et al. 2015	642
Shu et al. 2007	2723
Sidiropoulou & Poirazi. 2012	45
Smith et al. 2013	2953
Stuart & Spruston, 1998	610, 1099, 1112
Traub et al. 2003	88
van Elburg & van Ooyen, 2010	NA

To make some comparison on the morphologies, we divided the models into different categories based on the number of compartments. The first category includes models that have under 50 compartments and 19% of the models belong to that category. Models that have compartments from 50 to 100 belong to category 2. 14% of the models fall into that category. In the third category with compartments from 100 to 1000 includes 44% of the models. The rest of the models with over 1000 compartments belong to category 4 with the percentage of 22%. The number of compartments was not indicated for 22% of the models.

Almost 70% of the models have over 100 compartments, which indicates that the biological realism of the models is high. However, a large number of compartments does not necessarily mean that the model is biologically realistic. For example, L5 pyramidal cells are larger than pyramidal cells in L2/3 so they need more compartments.

3.3 Biophysiological properties

The first compartmental model of cortical pyramidal cell was introduced in 1995 and it included only the basic Hodgkin-Huxley type of channels, fast sodium and non-inactivating potassium channels [47]. Other models from the late 90s and early 2000s have fewer ion channels than the recent ones. This phenomenon is illustrated in Figure 6. Recent studies include ion channels also in the dendrites unlike the first models that had ion channels only in the soma and axon. The included ion channels are listed in Table 3. The ion channels are expressed in the same way they are mentioned in the publications. The abbreviations are described in Appendix A.2.

Table 3. *The ion channels, currents and receptors included in the models*

Author & Year	Ion channels, currents and receptors
Aberra et al. 2018	Nat, Nap, Kt, Kp, Im, SK, Ca_HVA, Ca_LVA, IH, Kd, Ka
Almog & Korngreen, 2014	Na, Ks, Kf, Ih, Ca_HVA, Ca_MVA, Kca, SK,
Arkhipov et al. 2018	CA_HVA, Ca_LVA, IH, Im, Kp, Kv, Kd, Kt, Nap, Nat, Kca, Nav
Bahl et al. 2012	HCN, Nat, Kfast, Kslow, Nap, Km, Cas, Kca, CP
Behabadi & Mel, 2014	Na, K, NMDA, AMPA
Behabadi et al. 2012	Na, K, NMDA, AMPA
de Sousa et al. 2015	Nat, Kv, Km, Kca, Ca_HVA, leak
Deitcher et al. 2017	Na, K
Doron et al. 2017	Nat, Nap, Kp, Kt, Km, SK, Ca_HVA, Ca_LVA, NMDA, AMPA
Eyal et al. 2016	Na, K
Eyal et al. 2018	Nat, Nap, Kp, Kt, Km, SK, Ca_HVA, Ca_LVA
Galloni et al. 2019	HCN, Nat, Nap, Kfast, Kslow, Km, Cas, Kca, CP
Gidon et al. 2020	Na, Kdr, Ca_HVA, Kca, Ca-pump, NMDA, AMPA, GABA
Gouwens et al. 2018	CA_HVA, Ca_LVA, IH, Im, Kp, Kv, Kd, Kt, Nap, Nat, Kca, Nav
Hay et al. 2011	Nat, Nap, Kp, Kt, Km, SK, Ca_HVA, Ca_LVA
Hu et al. 2009	Nat, Kv, Km, Ca_hva, Kca, leak
Kampa & Stuart, 2006	Na, Kv, Ka, Ca_hva, Ca_lva,

Kole et al. 2008	Na, Kv, Kv1, Km, Ih,
Kole et al. 2006	Na, Kv, KaP, Kca, Ca, Cat, Km
Larkum et al. 2009	Na, K, Kdr, Ka, Km, Kca, Cal, IH, NMDA, AMPA
Li et al. 2013	Na, Kv, Kca, Km, Ih, NMDA, AMPA
Mainen et al. 1995	Na, K
Markram et al. 2015	Kd, Kv, Ks, Kt, Kp, Nat, Nap, Ca_hva, Ca_lva, M, IH, Kca, Ca_dyn
Mäki-Marttunen et al. 2017	Nat, Nap, Ih, Km, Kp, Kt, Kv, Ca_hva, Ca_lva, SK, leak
Nevian et al. 2007	Na, K
Papoutsi et al. 2017	Nat, Nap, Kdr, Ks, Ka, H, Kca, CaL, CaR, CaN, CaT, NMDA, AMPA
Poleg-Polsky, 2015	Nat, Kdr, NMDA, AMPA
Polsky et al. 2009	Nat, Kf, Km, NMDA, AMPA
Schaefer et al. 2003	Nat, Ka, K, Km, Kca, Ca
Shai et al. 2015	Nat, Nap, Kt, Kp, Ca_HVA, Ca_LVA, SK, IH, NMDA; AMPA
Shu et al. 2007	Nat, Kv, Km, Kca, Im, Ih, Kd, Ca_hva, Ca pump, AHP
Sidiropoulou & Poirazi. 2012	Nat, Nap, Kdr, Ka, Kd, fAHP, sAHP, IH, Cat, Can, Cal, Car, NMDA, AMPA
Smith et al. 2013	Na_hva, Ca_HVA, Ca_LVA, Kv, Km, Kca,
Stuart & Spruston, 1998	Na, K, Ih
Traub et al. 2003	Nap, Nat, Kdr, Ka, K2, Kca, Km, KAHP, Cat, Cah,
van Elburg & van Ooyen, 2010	Nat, Km, Kv, Kca, Ca_HVA,

The curve in Figure 6 shows the trend through the years. Not only the number of modeled channels has increased but also the number of studies has increased through the years.

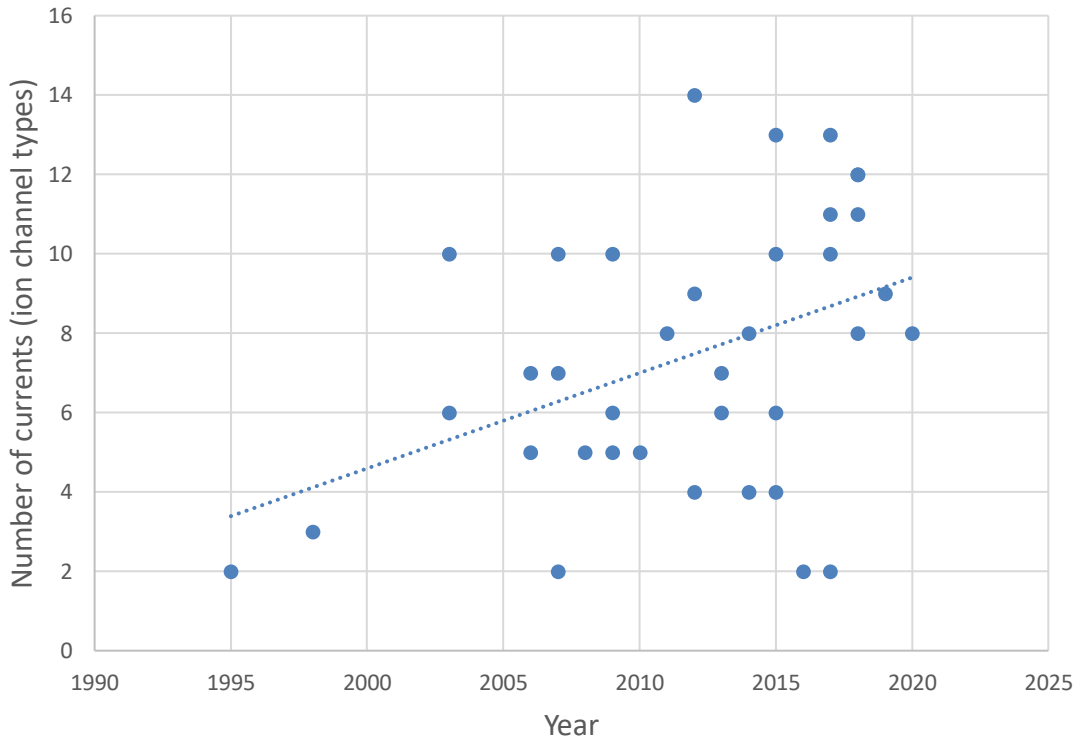

Figure 6: The number of currents (ion channel types) in the studies between the years 1995-2020.

As this review shows, the repertoire of compartmental models of cortical pyramidal cells is diverse. The models are used for modeling different kind of neuronal behaviors. The models represent the biological realism of cortical pyramidal cells in different level of detail as well as the bioelectrical properties of the cells. The diversity and extent of the models show that cortical pyramidal cells have captured the interest of neuroscientists and compartmental modeling is a useful tool for studying them.

4. MATERIALS AND METHODS

In this chapter, materials and methods used in the simulation-based analysis are described. The Materials section includes an explanation of how the model used in the simulations was selected. After that, a detailed description of the properties of the model is presented. The Methods section shows how the simulations were conducted. The simulation environment is presented as well as the two Python toolboxes that were tested.

4.1 Materials

In Chapter 3, we described and evaluated 36 models from the literature selected according to criteria given in the beginning of Chapter 3. Next we selected a model that we further analyzed using computational simulations with NEURON and Python. The model was selected according to following criteria.

As seen in Table 1, most of the studies modeled layer 5 pyramidal cells. Because they were studied most widely, we decided to choose a model from the layer that had not been studied as much. Thus, we chose a layer 2/3 model. We were particularly interested in a model that presented a cell from the somatosensory cortex. The model should also be morphologically and biophysically realistic hence we selected a model from [48].

The model we chose to simulate was from the publication Reconstruction and Simulation of Neocortical Microcircuitry [48]. Their study included a reconstruction of the microcircuitry across all layers of juvenile rat somatosensory cortex. The complete model includes over 31 000 neurons with 37 million synapses. We chose one of the layer 2/3 pyramidal cell models for our simulations. The morphology of the cell is presented in Figure 7. The experimental data, models and tools used for the reconstruction can be found in The Neocortical Microcircuit Collaboration Portal at https://bbp.epfl.ch/nmc-portal.

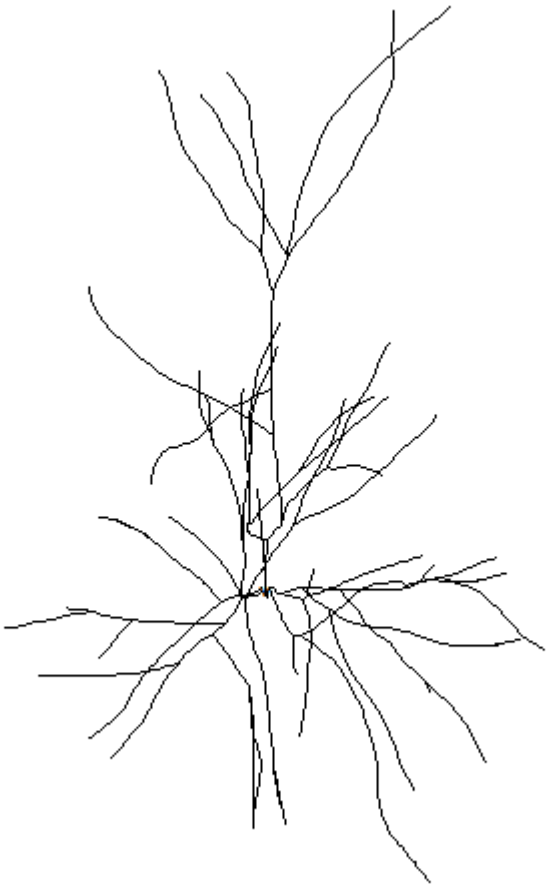

Figure 7: Morphology of the layer 2/3 pyramidal cell from rat somatosensory cortex.

The model included active electrical properties also in the dendrites. The model has different ion channels in different sections of the cell. These mechanisms are listed in Table 4.

Table 4. *Channel mechanisms*

Sections	Mechanisms
basal	Ih
apical	Im, NaTs2_t, SKv3_1, Ih
axonal	Ca_HVA, SKv3_1, SK_E2, Ca_LVAst, Nap_Et2, CaDynamics_E2, K_Pst, K_Tst, NaTa_t
somatic	NaTs2_t, SKv3_1, SK_E2, Ca_LVAst, Ih, Ca_HVA, CaDynamics_E2

We were interested in how many synapses layer 2/3 pyramidal cells form with cells from different layers of the cortex. The number of synapses and the types of synapses with different presynaptic cells are listed in Table 5. All the information can be found in The Neocortical Microcircuit Collaboration Portal.

Table 5. *Synapses between different presynaptic cells and layer 2/3 pyramidal cell*

	Presynaptic cell	**Total number of synapses**	**Synapses per cell**	**Number of synapses per connection**	**Type of the synapse**
Layer 2/3	Large basket cell	382 670	65	17	Inhibitory
	Small basket cell	199 790	34	23	Inhibitory
	Nest basket cell	298 310	51	17	Inhibitory
	Pyramidal cell	1 562 100	266	2.8	Excitatory
Layer 4	Large basket cell	55 596	9	14	Inhibitory
	Small basket cell	19 590	3	15	Inhibitory
	Nest basket cell	36 301	6	14	Inhibitory
	Pyramidal cell	256 610	44	2.4	Excitatory
	Star pyramidal cell	53 367	9	2.3	Excitatory
Layer 5	Large basket cell	16 363	3	8.2	Inhibitory
	Thick-tufted pyramidal cell with a late bifurcating apical tuft	100 530	17	2.3	Excitatory
	Thick-tufted pyramidal cell with an early bifurcating apical tuft	56 880	10	2.3	Excitatory
	Untufted pyramidal cell	13 694	2	2.2	Excitatory
	Slender-tufted pyramidal cell	12 417	2	2.2	Excitatory
Layer 6	Pyramidal cell with inverted apical-like dendrites	22 771	4	2.2	Excitatory

As seen in Table 5, layer 2/3 pyramidal cells receive most of the synapses with other cells in layer 2/3. They form most synapses with other layer 2/3 pyramidal cells. These cells also have more synaptic connections with other excitatory cells than with inhibitory cells.

The synapses between layer 2/3 pyramidal cells and different postsynaptic cells are listed in Table 6. The number of synapses is significantly higher with other pyramidal cells as postsynaptic cells compared to L2/3 pyramidal cells being postsynaptic cells (see Table 5). On the contrary, the number of synapses with inhibitory cells as postsynaptic cells are lower compared to L2/3 pyramidal cells being the postsynaptic cells.

Table 6. *Synapses between layer 2/3 pyramidal cells and different postsynaptic cells*

	Postsynaptic cell	Total number of synapses	Number of synapses per connection	Type of the postsynaptic cell
Layer 2/3	Large basket cell	123 320	8.1	Inhibitory
	Small basket cell	38 077	3.1	Inhibitory
	Nest basket cell	76 106	3.8	Inhibitory
	Pyramidal cell	1 562 100	2.8	Excitatory
Layer 4	Large basket cell	37 250	6.9	Inhibitory
	Small basket cell	15 493	7	Inhibitory
	Nest basket cell	32 694	6.7	Inhibitory
	Pyramidal cell	1 205 100	2.6	Excitatory
	Star pyramidal cell	392 300	2.7	Excitatory
Layer 5	Large basket cell	49 572	6.5	Inhibitory
	Thick-tufted pyramidal cell with a late bifurcating apical tuft	1 766 300	3.8	Excitatory
	Thick-tufted pyramidal cell with an early bifurcating apical tuft	1 680 600	3.8	Excitatory
	Untufted pyramidal cell	165 190	2.8	Excitatory
	Slender-tufted pyramidal cell	190 290	2.9	Excitatory
Layer 6	Pyramidal cell with inverted apical-like dendrites	89 135	2.3	Excitatory
	Tufted pyramidal cell with dendritic tuft terminating in L4	230 860	2.4	Excitatory
	Tufted pyramidal cell with dendritic tuft terminating in L1	254 060	2.4	Excitatory

4.2 Methods

In the following sections, methods used in the simulation part are described. First, the simulation software NEURON is introduced. After that, the two Python toolboxes are described.

4.2.1 Simulation environment: NEURON

The NEURON environment is a simulation software designed for modelling neurons and networks of neurons [57]. It was first developed by Michael Hines, John W. Moore and Ted Carnevale. NEURON is especially suitable for building models that are based on experimental data, particularly if the cells have complex morphologies and biophysical properties. NEURON can be installed free of charge at https://neuron.yale.edu and it runs on most popular operating systems, such as MSWin, UNIX, Linux and OS X.

NEURON allows the user to decide, which programming language to use. Most of the programming has been done with an interpreted programming language called hoc [57]. More recently, Python was added as an optional interpreter. Python and hoc are compatible hence all the functionalities can be accessed with both interpreters. However, using NEURON does not need any programming skills as the simulation environment includes a graphical user interface (GUI).

The simulations were started by studying the behavior of the model with NEURON simulation software. The NEURON GUI for the rodent layer 2/3 pyramidal cell model is shown in Figure 8. The panel A controls the simulation. It also includes a selection of the step current that is injected to the soma. The options are 120% higher than threshold (Step current 1), 130% higher than threshold (Step current 2), 140% higher than threshold (Step current 3) or no step current. Panel B shows the morphology of the L2/3 pyramidal cell. It also shows locations of the synapses when a presynaptic cell type has been selected from the panel C. The frequency of the synaptic input is defined with panel D. Panel E illustrates the somatic potential (as mV) as a function of time (ms).

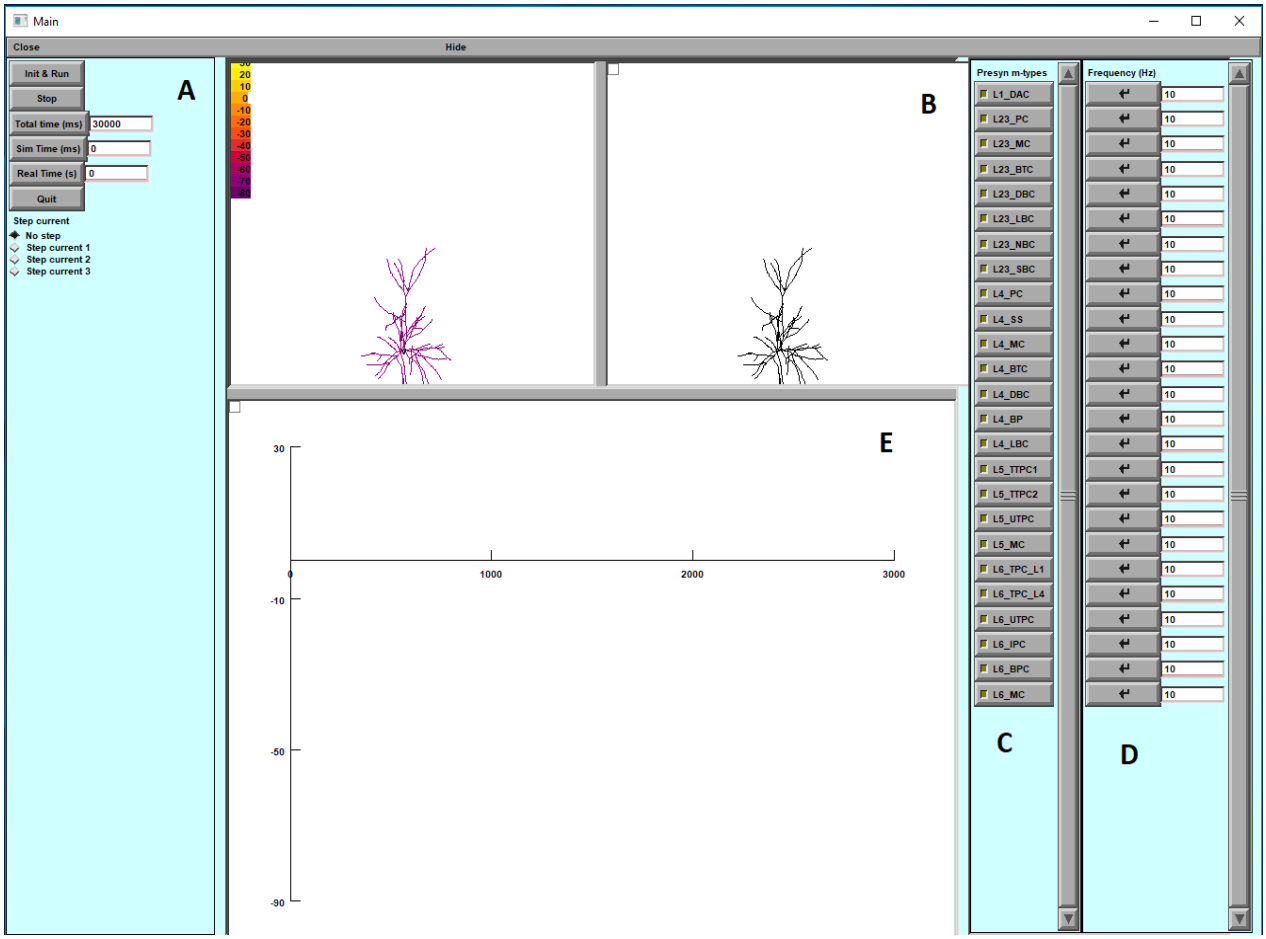

Figure 8: NEURON GUI for the rodent layer 2/3 pyramidal cell model. A) Controlling the simulations, choosing the external stimulus. B) The morphology of the cell. C) Selecting the presynaptic cell type. D) Defining the frequency of the synaptic input. E) Somatic potential (mV) as a function of time (ms).

We started the analysis of the excitability of the model with modeling the effect of excitatory synaptic inputs. First we studied which combinations of inputs can induce action potentials without an induced step current. We tested different combinations of the excitatory synapses with other pyramidal cells. We also simulated conditions in which the inputs from inhibitory synapses can prevent the cell from spiking when excited with excitatory synaptic inputs.

4.2.2 *Neuron_Reduce* toolbox

Simulating complex and detailed compartmental models with thousands of synapses is computationally expensive and time-consuming, which notably constrains the utility of the model. Therefore we were interested in Neuron_Reduce toolbox, which is a Python module for reducing the morphological complexity and computational time of nonlinear neuron models [4]. Neuron_Reduce significantly decreases the simulation time while replicating the voltage dynamics of the original model and dendritic computations [4]. The toolbox is publicly available at https://github.com/orena1/neuron_reduce.

In the second part of the simulations, we analyzed the reduced complexity of the model. The code used for reduction is presented in Appendix A.3. First, the code loads the original cell, which is named *complex_cell*. The complex cell and its mechanisms are defined with NEURONs hoc-files. After that, synapses are added, and the complex cell can be simulated. Variable *h.tstop,* given in milliseconds, is used to define the simulation time. Then the model is simulated for the given time and the voltage of the soma is recorded. After the recording, the simulation time for the complex cell is printed. Next, Neuron_Reduce is used to simplify the cell. The number of segments in the reduced cell can be defined with the variable *total_segments_manual.* The default value -1 will do the segmentation automatically. The reduced cell is simulated in the same manner as the complex cell and the simulation time for the reduced cell is printed. Finally, the soma voltages of both complex and reduced cell are plotted in a figure.

First we simulated the minimum number of compartments, which was eight with the model we used. The simulation time was 30 seconds. We then calculated the mean interspike intervals (ISIs) (time between two spikes) and the mean AP heights. For the calculations, we used the Electrophys Feature Extract Library (eFEL), which is a Python library for extracting features from neuron recordings. Documentation can be found in [58]. The same simulation and calculations were made also for reduced cells with 30, 50 and 78 compartments. The simulations were run on a virtual 64-bit Linux machine, with the host computer having 7 1.90 GHz processors and 32 GB memory. We then increased the simulation time to 10 minutes and repeated the simulations. The 10-minute simulations were run on a computing cluster.

4.2.3 *Uncertainpy* toolbox

In neuroscience, models often include many uncertain parameters, meaning that the exact value of the parameter is not always known. This may be due to the measurement errors or to the inherent variability of the parameter. To study the effect of the uncertain parameters, we used Uncertainpy, a Python toolbox for sensitivity analysis and uncertainty quantification [3]. Uncertainpy is particularly tailored to perform sensitivity analysis and uncertainty quantification for models in neuroscience [3]. It can use either polynomial chaos expansions or quasi-Monte Carlo methods [3]. Uncertainpy is publicly available at https://github.com/simetenn/uncertainpy.

For sensitivity analysis, Uncertainpy uses Sobol sensitivity indices. Sobol indices define how much of the variance in the model output each uncertain parameter or combination of parameters is responsible for. While Uncertainpy can calculate both the first order and total Sobol indices, the results presented in this thesis use only the first order Sobol indices that describe how each parameter individually affects the variance in the model output. The first order Sobol indices result in values from 0 to 1. Basically, a low (close to 0) first order Sobol index means that a change in the parameter does not affect the model output considerably. Similarly, a high (close to 1) Sobol index means that a change in the parameter causes a large variation in the model output.

The parameters we chose to study with Uncertainpy were the maximal conductances for the different ion channel types present in the soma, NaTs2, SKv3, SK_E2, Ca_LVA, Ca_HVA channels, as well as the maximal conductance for the Ih current. NaTs2 is an abbreviation for transient sodium channels which are responsible for the depolarization phase of action potentials [6, p. 156]. SKv3 is a type of Kv3 voltage-gated potassium channels that enables the fast repolarization phase of action potentials without compromising the initiation or height of the action potential [59]. Small-conductance calcium-activated potassium channels (SK_E2) cause hyperpolarization and therefore lower the spike firing frequency [60]. Ca_LVA is an abbreviation for low-voltage activated calcium channels that are activated with small voltage changes and are important for repetitive firing of action potentials [61]. Ca_HVA are high-voltage activated calcium channels that open in large depolarizations and can shape regenerative action potentials [62]. H-current, or hyperpolarization-activated current (Ih), is a cation current that affects on the firing frequency of action potential [63].

We run the sensitivity analysis for the original model and for the reduced model with 30 compartments from the previous simulations. Uncertainpy has a class called SpikingFeatures, which includes built-in features relevant for single neuron models that receive an external stimulus. The features we were interested in were spike rate, average AP overshoot and average AP width. Spike rate represents the action potential firing rate, average AP overshoot the average peak voltage of action potential and average AP width

the average action potential width taken at the midpoint between the onset and peak of the action potential [3].

The script used for uncertainty quantification and sensitivity analysis is presented in Appendix A.4. The first part of the code uses the Neuron_Reduce tool to simplify the complexity of the cell. Then a function needed in the uncertainty quantification is defined. After running the simulations, the parameters and distributions for the parameters are defined. We allowed each parameter to vary within the interval [0.5g, 1.5g], where g is the original value of the parameter. The parameter took values from this interval according to the uniform distribution. Lastly, the features are defined, and the uncertainty quantification is executed.

5. RESULTS

In this chapter, the results of the simulations are presented. In Section 5.1, results of studying the excitability of the model with NEURON simulator are shown. The analysis of the reduced complexity by using the *Neuron_Reduce* toolbox is presented in Section 5.2. Section 5.3 shows the results of the sensitivity analysis of the uncertain parameters.

5.1 Analysis of the excitability

First we simulated which combinations of excitatory inputs can induce an action potential in the postsynaptic cell. We started with synaptic inputs from a L2/3 pyramidal cells. The frequency of the synaptic inputs was 10 Hz. Locations of the synapses are illustrated with red dots in Figure 9.

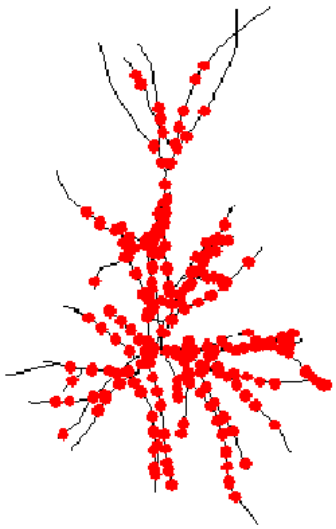

Figure 9: Layer 2/3 pyramidal cell's synapses with other layer 2/3 pyramidal cells.

Figure 10 illustrates the somatic potential as a function of time with synaptic inputs from L2/3 pyramidal cells. As seen in Figure 10, the synaptic inputs are capable of generating action potentials in the postsynaptic cell. Layer 2/3 pyramidal cells form more excitatory synapses with the L2/3 pyramidal cell than any other considered cell type so the probability for the action potential generation is the greatest in these synapses (see Table 5). There is also some synaptic activity between the action potentials.

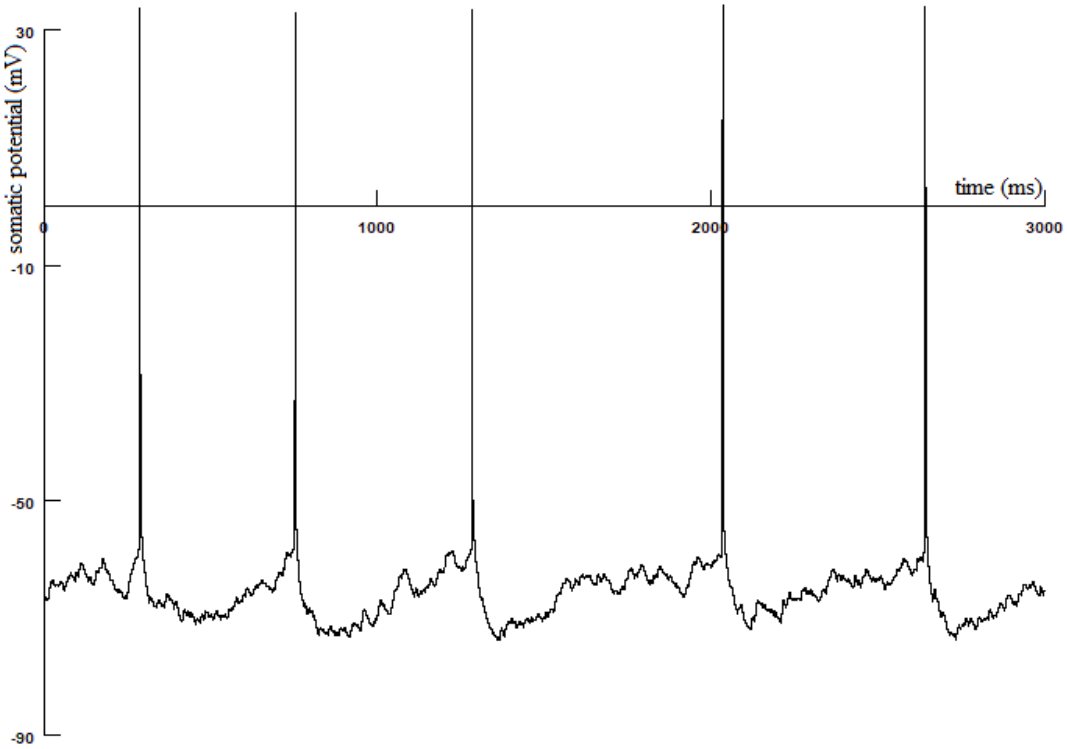

Figure 10: Somatic potential of layer 2/3 pyramidal cell excited with synaptic inputs from other layer 2/3 pyramidal cells.

Figure 11 illustrates the locations of synapses when layer 4 pyramidal cell is the presynaptic cell type. As seen in Figure 11, the number of connections is smaller between the cells than in the previous case. The changes in somatic potential are illustrated in Figure 12. Some synaptic activity can be seen but not enough to induce an action potential in the postsynaptic cell.

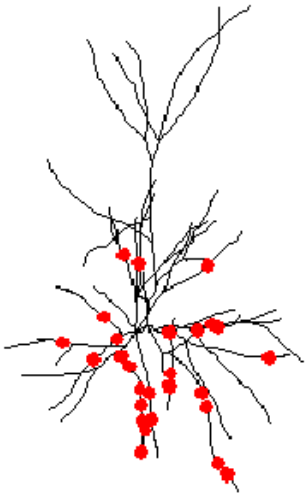

Figure 11: Layer 2/3 pyramidal cell's excitatory synapses with pyramidal cells from layer 4.

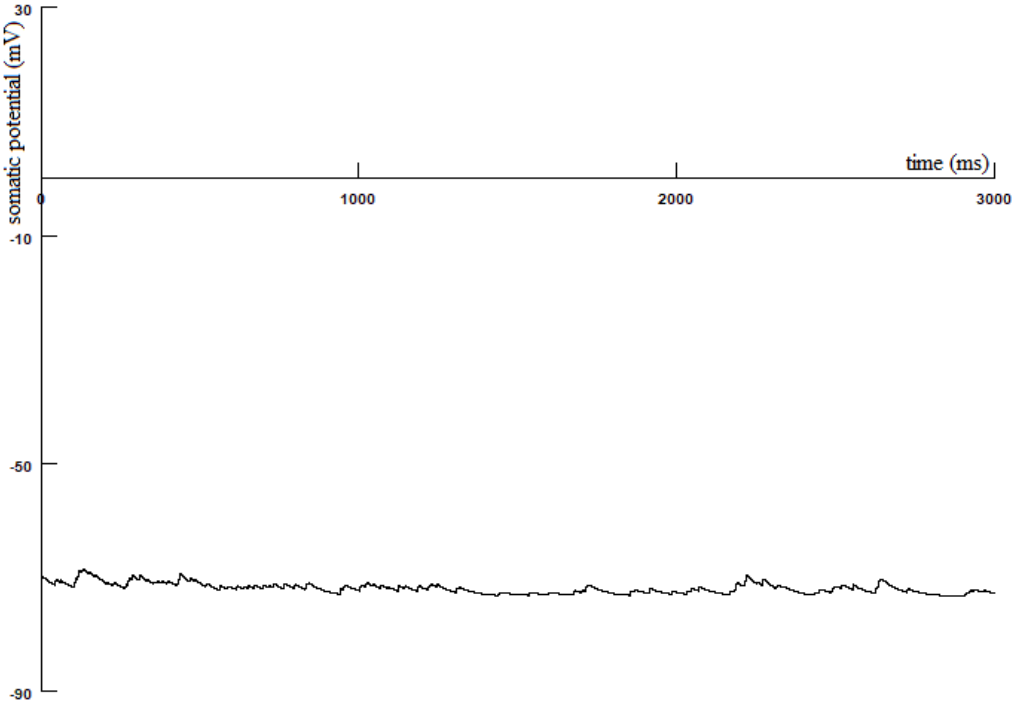

Figure 12: Somatic potential of layer 2/3 pyramidal cell excited with synaptic inputs from layer 4 pyramidal cells.

The locations of the synapses with all the other excitatory cells except layer 2/3 pyramidal cells are illustrated in Figure 13. It shows that the number of synapses here is substantially less compared to the number of synapses with L2/3 pyramidal cells. This indicates that the synaptic inputs may not be strong enough to induce action potential. The result is presented in Figure 14.

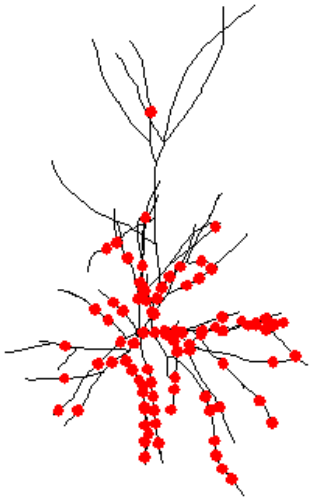

Figure 13: Layer 2/3 pyramidal cell's excitatory synapses with pyramidal cells from layer 4, 5 and 6.

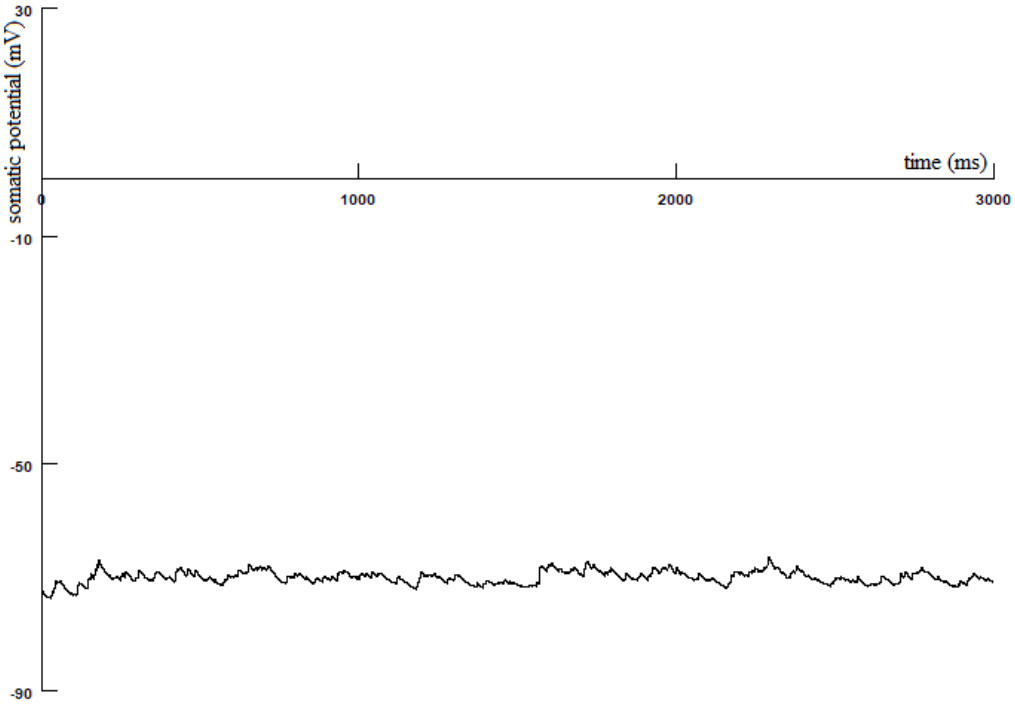

Figure 14: Somatic potential of layer 2/3 pyramidal cell when excited with synaptic inputs from layer 4, 5 and 6 excitatory cell types.

As seen in Figure 14, the sum of the inputs is not capable of generating action potential. However, the fluctuation is larger than in Figure 12.

Next we simulated the effect of inhibitory synapses. We chose layer 2/3 pyramidal cells that were able to induce action potentials and inhibitory synapses with all the inhibitory cell types. In Figure 15 the excitatory synapses are represented in red dots and inhibitory synapses with orange dots. The number of excitatory synapses is the same as in Figure 9. Figure 16 illustrates when the layer 2/3 pyramidal cell receives excitatory synaptic inputs from layer 2/3 pyramidal cells and inhibitory inputs (presented with orange dots) from all the inhibitory types of cells. As seen in the figure, the summation of inhibitory inputs prevents the cell from spiking. However, the synaptic activity is still larger than for example with only excitatory inputs received from L4 PC.

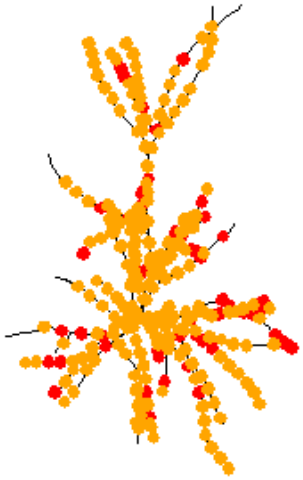

Figure 15: L2/3 pyramidal cell with excitatory synapses (red dots) with other layer 2/3 pyramidal cells and inhibitory synapses (orange dots) with all the inhibitory cell types.

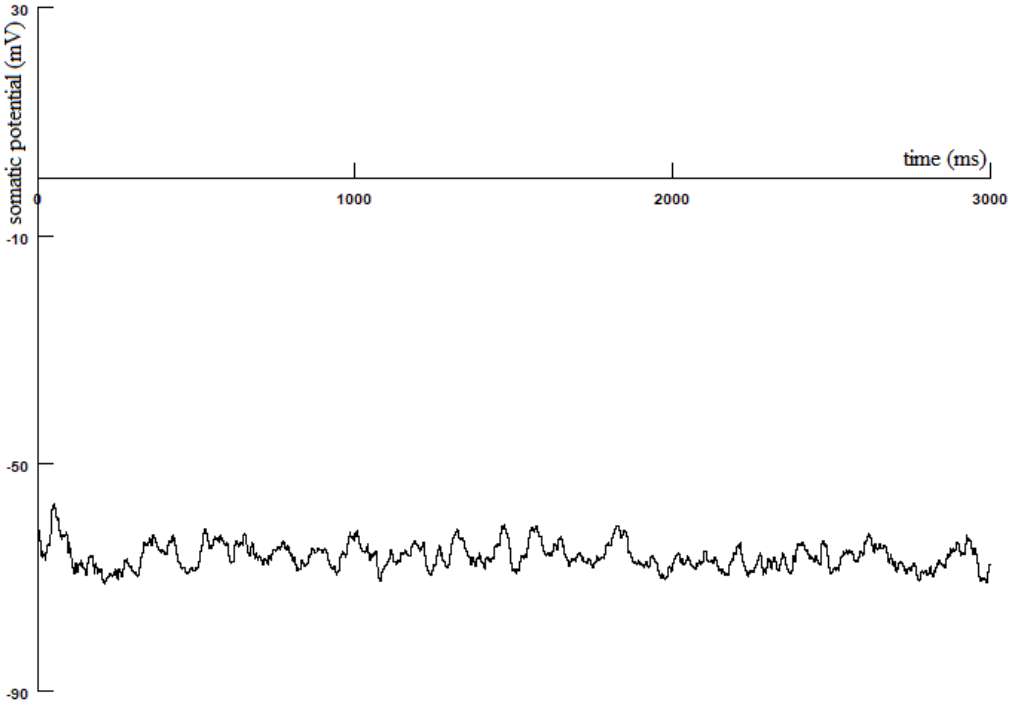

Figure 16: Somatic potential of layer 2/3 pyramidal cell when receiving excitatory synapses from layer 2/3 pyramidal cells and inhibitory synapses from all the inhibitory cell types.

Based on the simulations, the model seems to work as presented in the original publication. When the number of synapses exceeds a certain value, action potentials are initiated in the postsynaptic cell.

5.2 Analysis of the reduced complexity

Next we reduced the complexity of the model with the *Neuron_Reduce* toolbox. We simulated how the number of segments affects the model's behavior. The original model has 312 compartments and we reduced the number of segments to 8 (approximately 3% of the original model's compartments), 30 (approximately 10%), 50 (16%) and 78 (25%). Each of them was simulated for 30 seconds and 10 minutes of biological time. We also calculated the mean ISIs and mean AP heights.

Figure 17 illustrates the results for reducing the model to 8 segments. Upper panels show the membrane voltage of the original cell (blue trace) and reduced cell (orange trace). Lower panel shows a segment of the recordings with both traces superimposed. As can be seen from the figure, the heights of the action potentials of the reduced cell are too high compared to the original cell, which indicates that the model is reduced too much.

We calculated ISIs and AP heights for both cells. The mean ISI in 30 second simulation for the original cell was 369.76 milliseconds and for the reduced cell with 8 compartments 361.64 milliseconds. The difference between the ISIs is 2.2%, which does not seem to be significant. However, the mean AP height for the original cell was 84.49 millivolts and for the reduced cell 99.11 millivolts, which results in a large 17.3% difference.

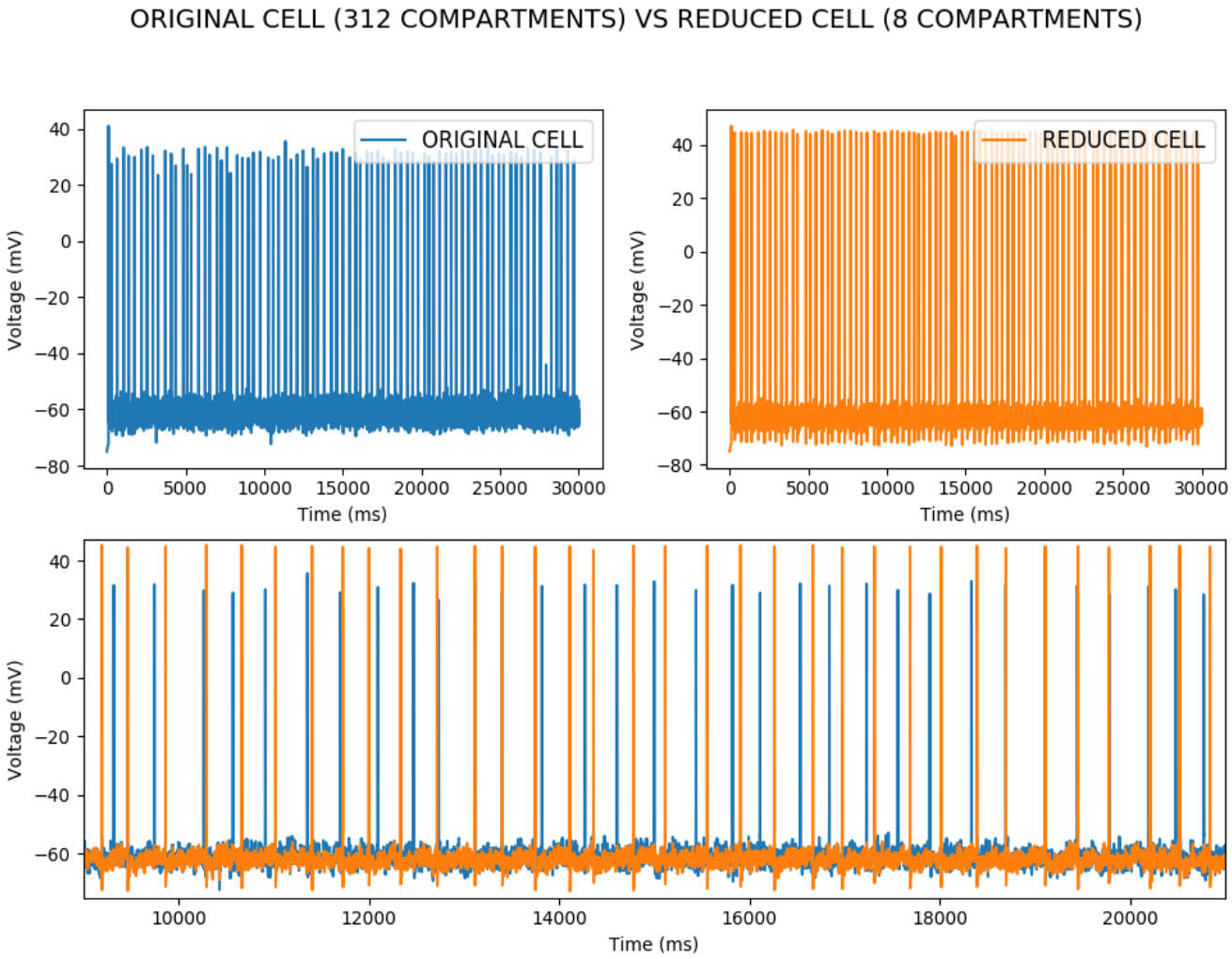

Figure 17: Voltage traces of the original cell with 312 compartments (blue trace) and reduced cell with 8 compartments (orange trace). The lower panel shows that the reduced cell was simplified excessively, and the reduced cell no longer behaves like the original cell.

When we increased the simulation time to 10 minutes, we got mean ISIs 368.29 milliseconds for the original cell and 342.96 milliseconds for the reduced cell. In this case, the difference was 6.9%. For the AP heights, we got 84.33 millivolts (original) and 97.48 millivolts (reduced), which still gives 16% difference. To better visualize the inter-spike intervals, they are illustrated in Figure 18. As can be seen from the histogram, it seems that the reduced model spikes more frequently and it has more regular activity than the original model.

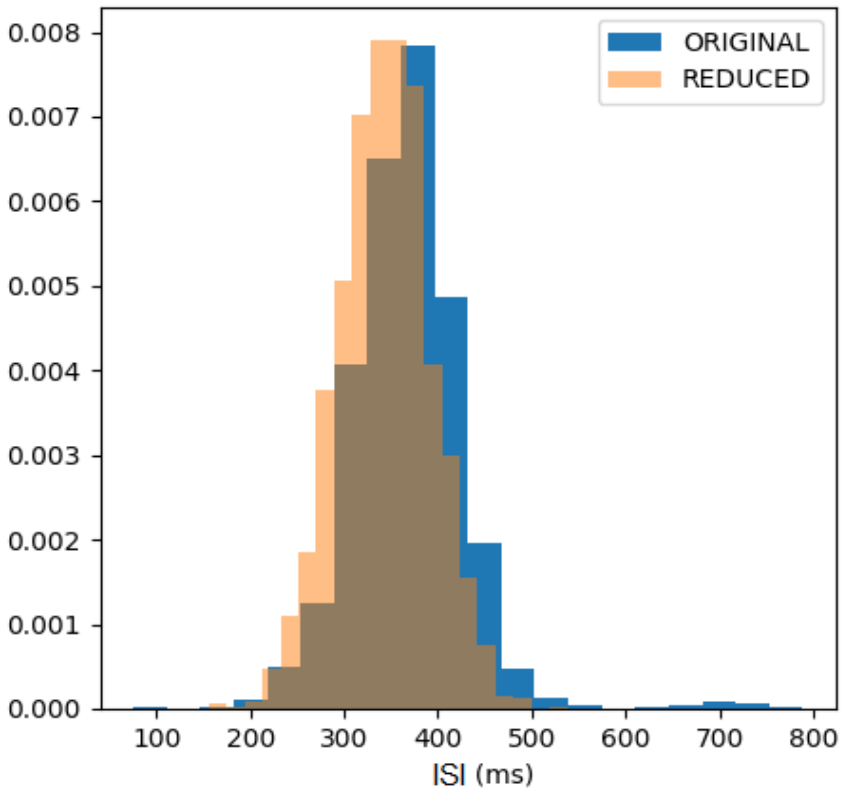

Figure 18: Histogram for inter-spike intervals for original cell (blue bins) and reduced cell with 8 compartments (orange bins). The histogram shows that the reduced cell spikes faster than the original cell.

Next we increased the number of compartments to 30 and repeated the simulations. For 30 second simulation time, we got 335.61 milliseconds mean ISI for the reduced model. The difference to the original model was 9.2%, which is even more than in the previous case. However, here the mean AP height was 84.82 millivolts, which resulted in only 0.4% difference, substantially less than in the previous case. The voltage traces for the 30-second simulation with 30 compartments are illustrated in Figure 19. As can be seen from the figure, the heights of action potentials are more similar to the original model than those obtained from the cell reduced to 8 compartments.

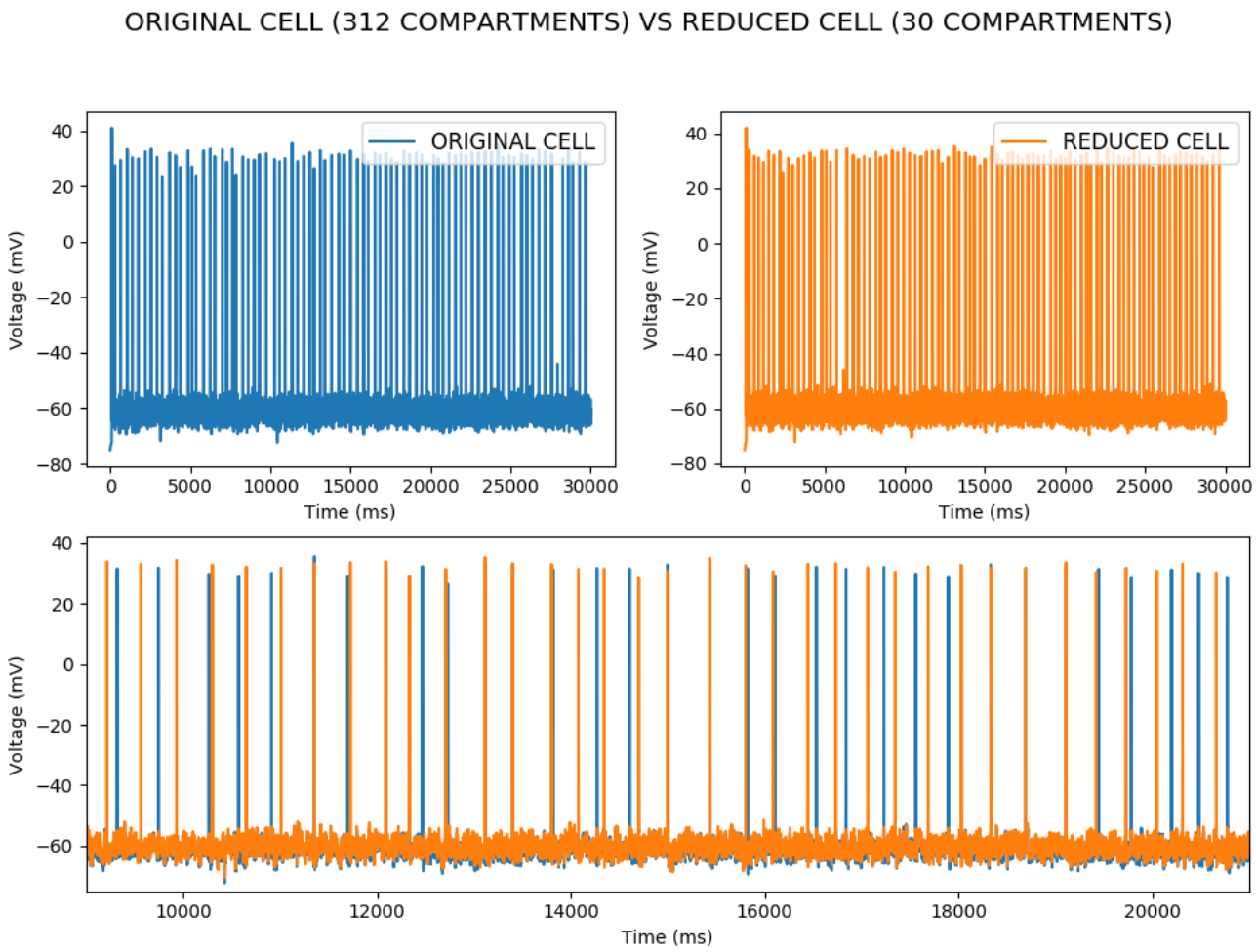

Figure 19: Voltage traces of the original cell with 312 compartments (blue trace) and reduced cell with 30 compartments (orange trace). The lower panel shows that the reduced model produces nearly the same results as the original cell.

With the longer 10-minute simulation time, the mean AP heights were 84.33 and 83.84 millivolts for original and reduced, respectively. The difference here is 0.6%. However, here the mean ISIs are even further, 368.29 milliseconds for the original and 328.08 milliseconds for the reduced, resulting in 10.9% difference. The ISIs are shown in Figure 20. The histogram shows that the 30-compartment reduced cell seems to fire even faster than the 8-compartment reduced cell.

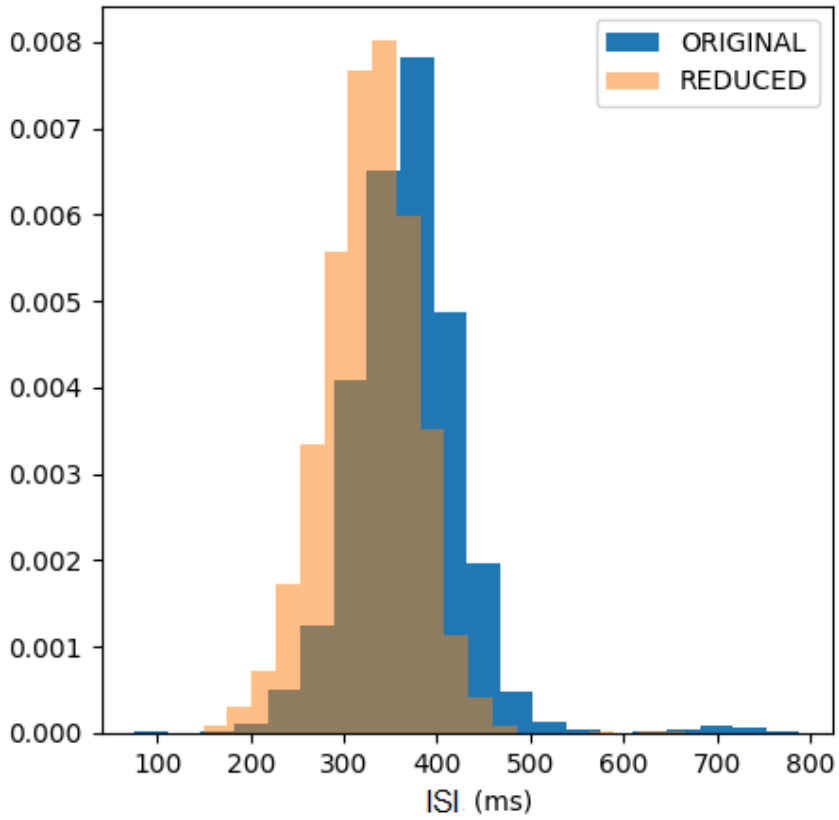

Figure 20: Histogram illustrating the inter-spike intervals, orange bins for the reduced cell with 30 compartments and blue bins for the original cell. The figure shows how the reduced cell spikes faster than the original cell.

Next we repeated the simulations with a cell reduced to 50 compartments. The 30-second simulation is shown in Figure 21. The lower panel shows that the voltage trace for 50-compartment cell is similar to the 30-compartment cell. Here the mean ISI was 351.63 milliseconds and the mean AP height 83.31 millivolts. These results show 4.9% and 1.4% differences compared to the original cell for the mean ISI and AP height, respectively.

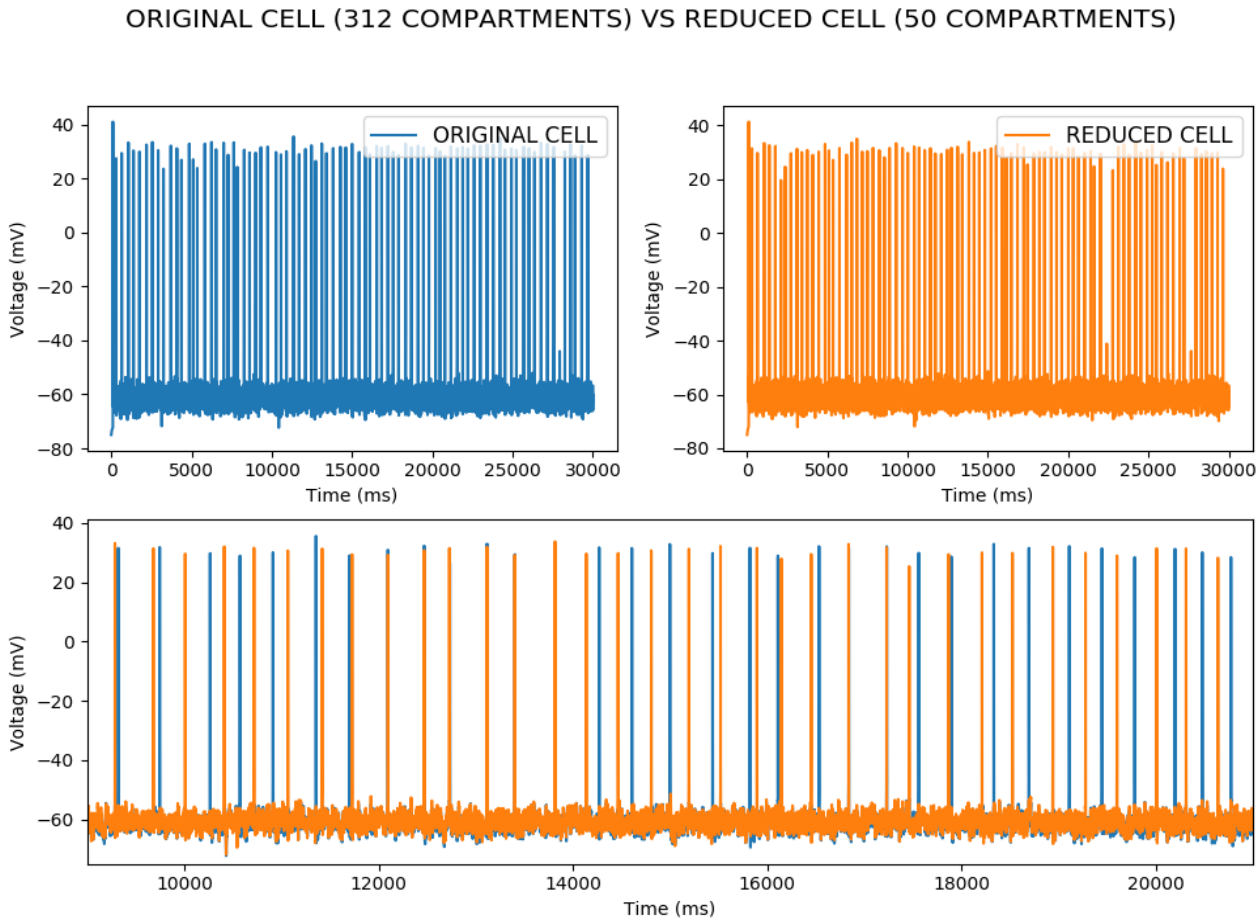

Figure 21: Voltage traces of the original cell with 312 compartments (blue trace) and reduced cell with 50 compartments (orange trace). The lower panel shows that the behavior of the reduced cell does not differ significantly from the original cell's behavior.

For the 10-minute simulation, the mean ISI for this reduced cell was 344.40 milliseconds and the mean AP height 82.65 millivolts. Compared to the original cell we got 6.5% and 2.0% differences for the mean ISI and AP height, respectively. The ISIs are shown in a histogram in Figure 22. It seems that the spiking is still faster than with the original cell.

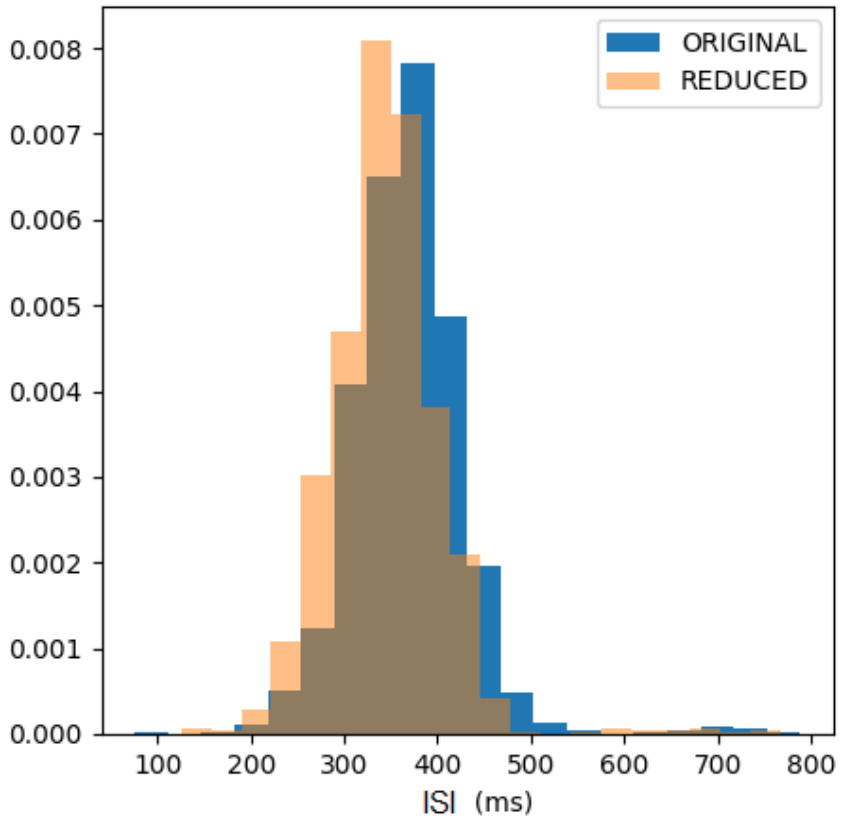

Figure 22: Histogram illustrating inter-spike intervals for reduced cell (orange bins) and original cell (blue bins). Reduced cell produces faster spiking than the original cell, but here the activity is not as regular as in the previous cases.

Lastly we increased the number of compartments to 78 and repeated the simulations. The results for the 30-second simulation were 340.30 milliseconds for the mean ISI and 82.98 millivolts for the mean AP height. Here the differences were 8.0% (ISIs) and 1.8% (AP heights). The 30-second simulation is illustrated in Figure 23. As can be seen from the figure, the behavior of the reduced cell is quite similar to the behavior of the original cell.

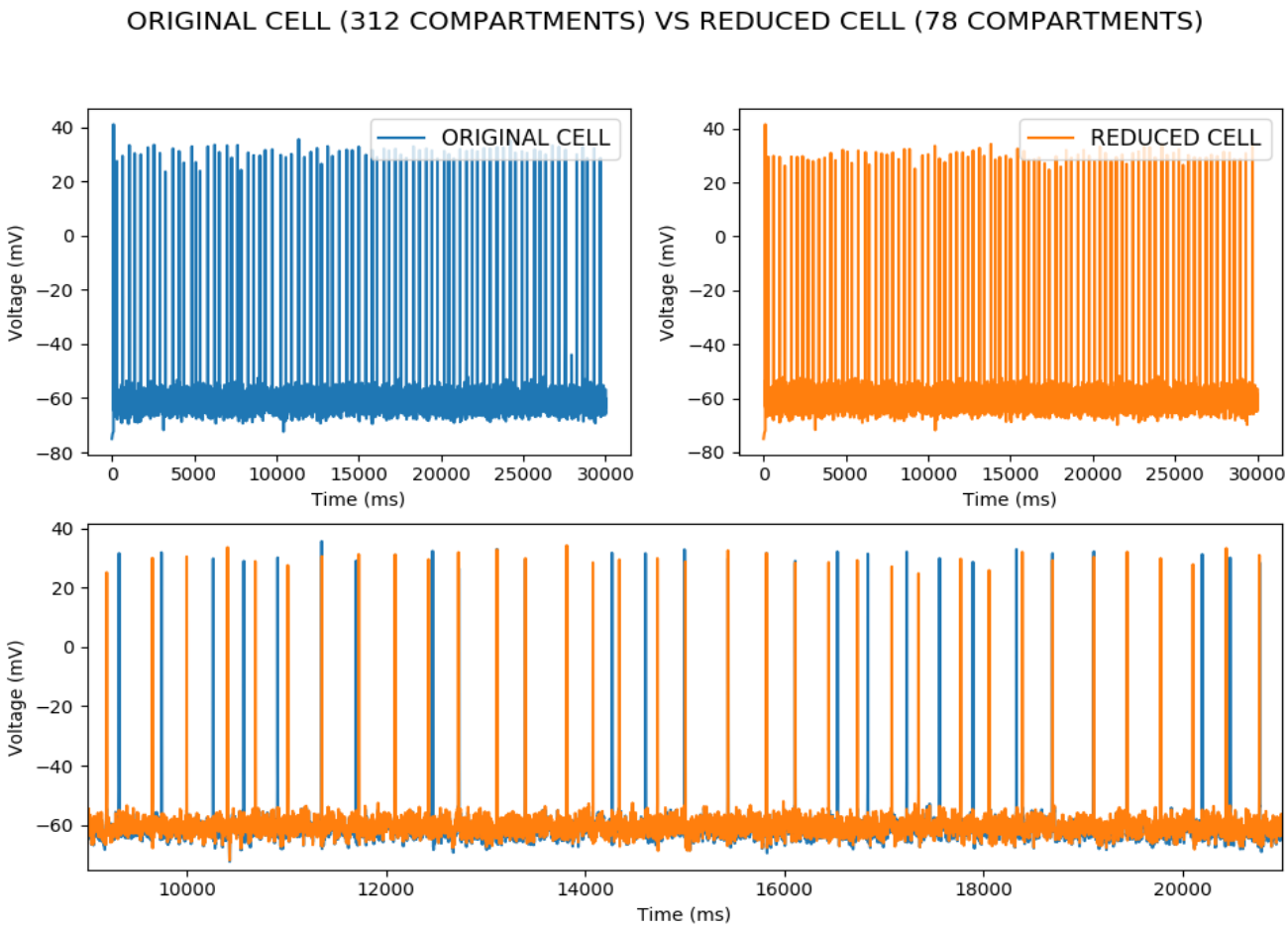

Figure 23: Voltage traces of the original cell with 312 compartments (blue trace) and reduced cell with 78 compartments (orange trace). The lower panel shows that the behavior of the reduced cell does not differ significantly from the original cell's behavior.

The results for the 10-minute simulation were 340.87 milliseconds for the mean ISI and 82.42 for the mean AP height, resulting 7.4% (ISIs) and 2.3% (AP heights) differences. The ISIs are illustrated in Figure 24. The histogram shows that the reduced cell seems to spike a little faster than the original cell, but the shapes of the bins are quite similar.

The results presented in this section are gathered in Table 7. As can be seen from the results, no big difference can be seen between reducing the cell to 30, 50 or 78 compartments. However, if the cell is reduced too much, it no longer produces similar results than the original cell.

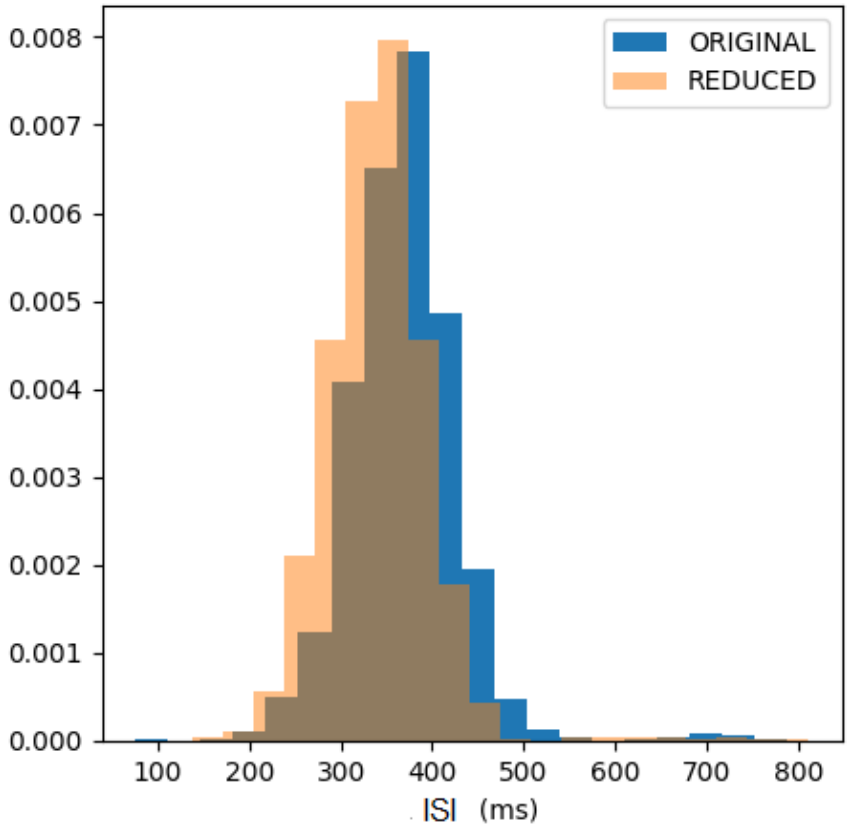

Figure 24: Histogram presenting the inter-spike intervals of a reduced cell with 78 compartments (orange bins) and original cell (blue bins). The reduces cell produces faster spiking but the shapes of the bins are similar.

Table 7. *Results for the analysis of reduced complexity*

	mean ISI (ms)	mean AP height (mV)	difference to the complex (%)	
Complex cell			**ISI**	**AP height**
30-second	369.76	84.49		
10-minute	368.29	84.33		
8 compartments				
30-second	361.64	99.11	2.2	17.3
10-minute	342.96	97.48	6.9	15.6
30 compartments				
30-second	335.61	84.82	9.2	0.4
10-minute	328.08	83.84	11	0.6
50 compartments				
30-second	351.63	83.31	4.9	1.4
10-minute	344.40	82.65	6.5	2.0
78 compartments				
30-second	340.30	82.98	8.0	1.8
10-minute	340.87	82.42	7.4	2.3

5.3 Sensitivity analysis of the parameters

Lastly, we analyzed the sensitivity of the different parameters of the model. The parameters chosen for sensitivity analysis were the different maximum conductances. We studied how the uncertainty of the parameters affect the different features of the model. Figure 25 illustrates the average of the first order Sobol indices for the average AP overshoot in the original model. As can be seen from the figure, the uncertainty of the AP overshoot depends primarily on the maximum sodium conductance (gNaTs2). The sodium conductance is known to be responsible for the rising phase of action potential, so the results are as expected. A slight contribution of potassium conductance (SKv3) can also be recognized.

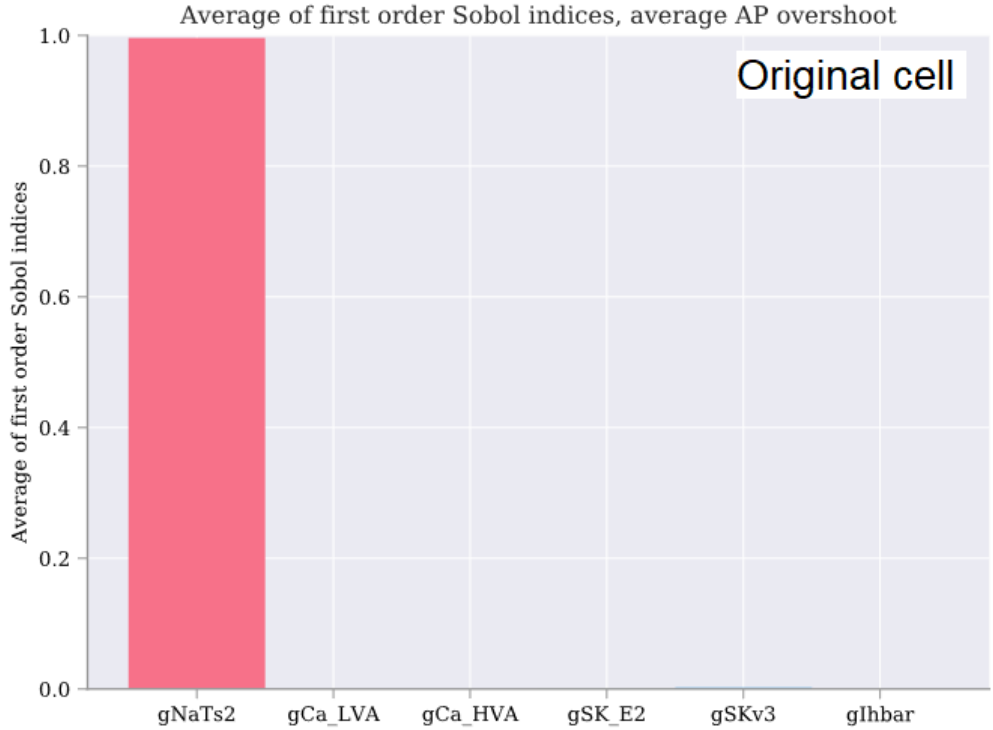

Figure 25: Average of the first order Sobol indices for the average AP overshoot in the original model. The uncertainty of the AP overshoot depends primarily on the sodium conductance (gNaTs2).

The same analysis was done also for the reduced cell with 30 compartments. The average of the first order Sobol indices for average AP overshoot in the reduced model is illustrated in Figure 26. Here the maximum sodium conductance is also the main parameter that causes uncertainty for the average AP overshoot.

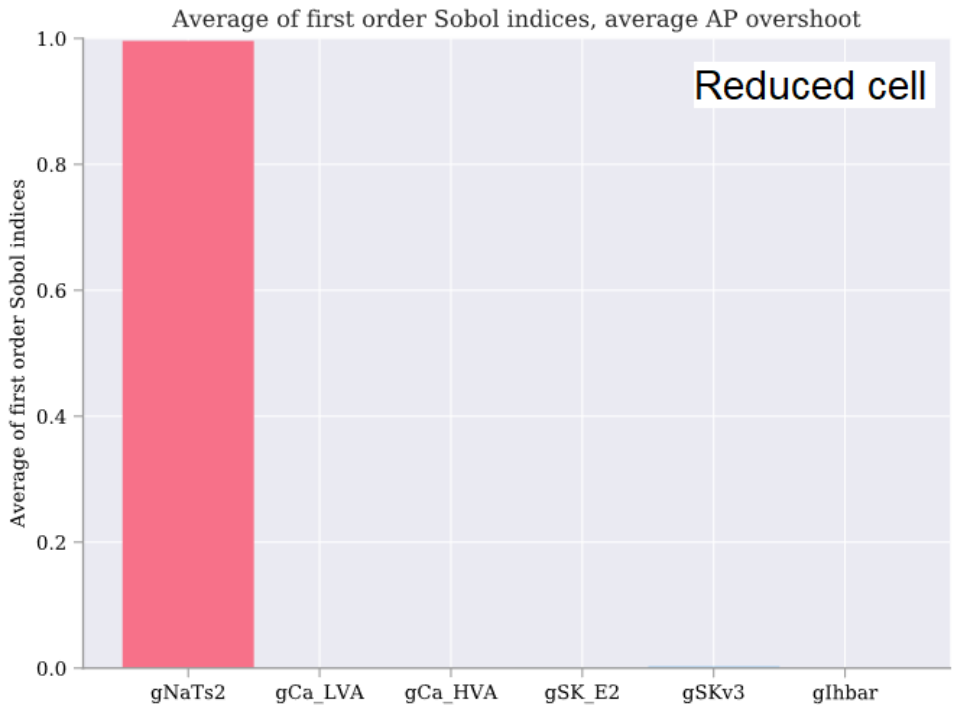

Figure 26: Average of first order Sobol indices for average AP overshoot in the reduced model. The uncertainty of the AP overshoot depends primarily on the sodium conductance.

Next feature that was studied was the average AP width. Figure 27 shows the average of the first order Sobol indices for the average AP width in the original model. Here the sodium conductance contributes almost 75% of the uncertainty of the model output, while the potassium conductance (gSKv3) contributes approximately 25% of the uncertainty. A minor contribution of high-voltage activated calcium (Ca_HVA) is also visible in Figure 27. The uncertainty for the average AP width in the reduced model is illustrated in Figure 28. Here 79% of the uncertainty arises from the sodium conductance and 21% from potassium conductance. In the reduced model, no contribution of the high-voltage activated calcium can be seen.

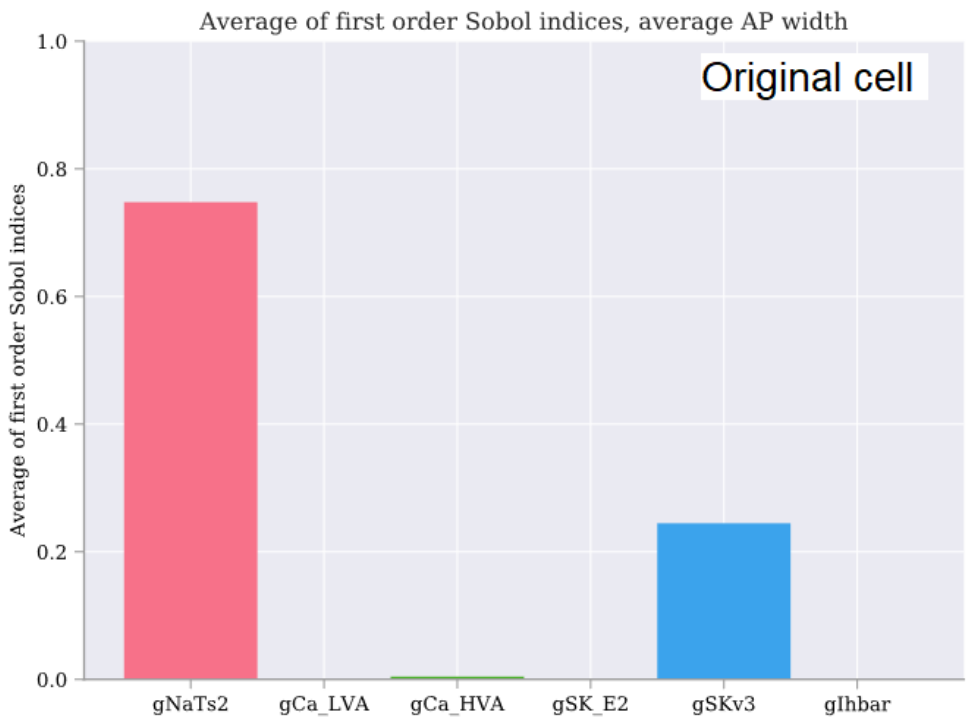

Figure 27: The average of the first order Sobol indices for average AP width in the original model. Here the uncertainty arises mostly from the sodium conductance, less from potassium conductance and a little from high-voltage activated calcium conductance.

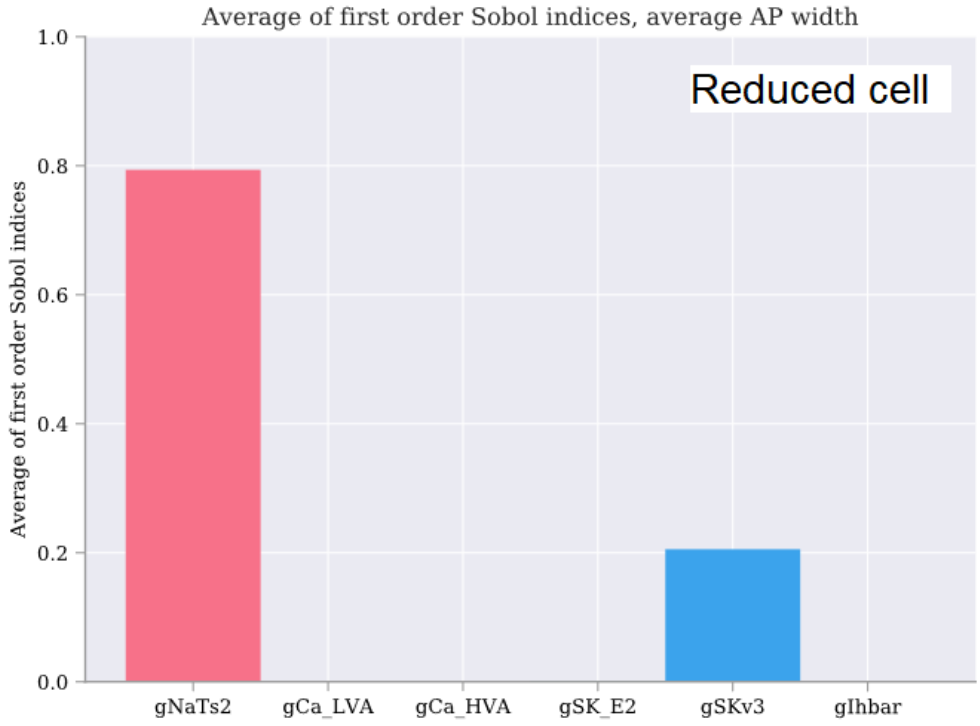

Figure 28: The average of the first order Sobol indices for average AP width in the reduced model. Here the uncertainty contributes mostly on the sodium conductance and less of potassium conductance.

The third feature that was studied was the spike rate. The average of the first order Sobol indices for spike rate in the original model is shown in Figure 29. Here the sodium conductance contributes approximately 22% of the uncertainty, then low-voltage activated calcium (Ca_LVA) approximately 6%, calcium activated potassium (SK_E2) and voltage-dependent potassium (SKv3) almost 4% and high-voltage activated calcium (Ca_HVA) and h-current (Ihbar) close to 1%. The corresponding analysis for reduced cell is illustrated in Figure 30. In this case, the sodium conductance contributes approximately 70% of the uncertainty. The contribution of sodium conductance is much higher in the reduced cell than in the original cell. Calcium activated potassium (SK_E2) and voltage-dependent potassium (SKv3) contribute 17% and 16% of the uncertainty, respectively. Neither high or low-voltage activated calcium affects the uncertainty for the spike rate nor the h-current.

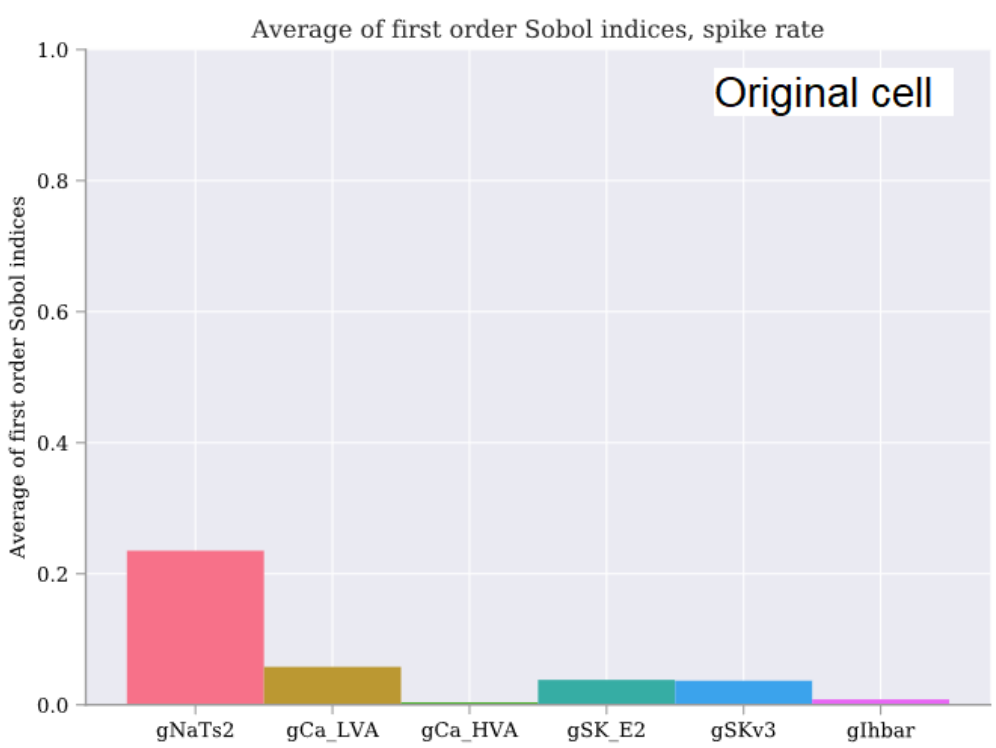

Figure 29: Average of first order Sobol indices for spike rate in the original model. Sodium conductance is the main source of uncertainty, but also all the other conductances have some effect on the output of the model.

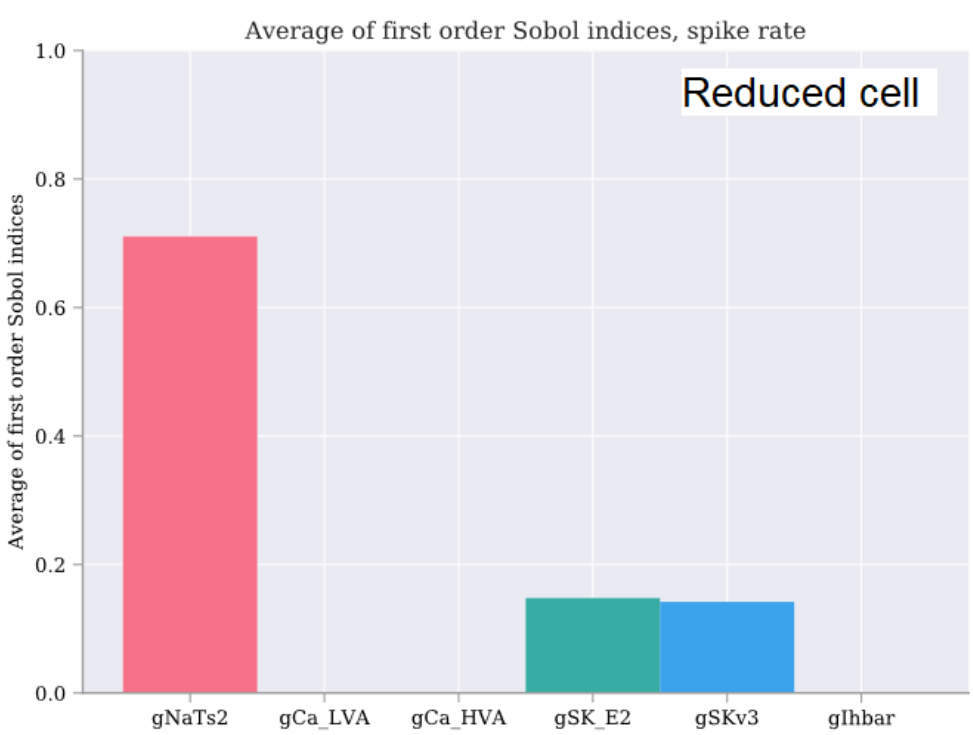

Figure 30: Average of first order Sobol indices for spike rate in the reduced cell. The uncertainty arises mostly from the sodium conductance and less from the calcium activated potassium and the voltage-dependent potassium conductances.

As can be seen from the figures, sodium conductance is the main source of uncertainty for all the considered features. Potassium, calcium activated potassium and low-voltage calcium conductances have some effect on the uncertainty and high-voltage activated calcium and h-current only a minor effect on the features studied here. It seems that the uncertainty for average AP overshoot and average AP width is very similar in the original and reduced cell. For spike rate, the uncertainty in the original model is distributed with more parameters than in the reduced model.

Uncertainpy also analyzes how the sensitivity of the uncertain parameters change as a function of time. It measures the sensitivity for each considered parameter from the model output, which in this case is the membrane voltage. For the original model, this is illustrated in Figure 31. It seems that during spiking the uncertainty arises from sodium conductance (gNaTs2) and just after spiking from potassium conductance (SKv3). The spikes can be seen approximately every 250 milliseconds starting at 50 milliseconds. The uncertainty between the spikes arises from high-voltage activated calcium (Ca_HVA) and calcium activated potassium conductances (SK_E2).

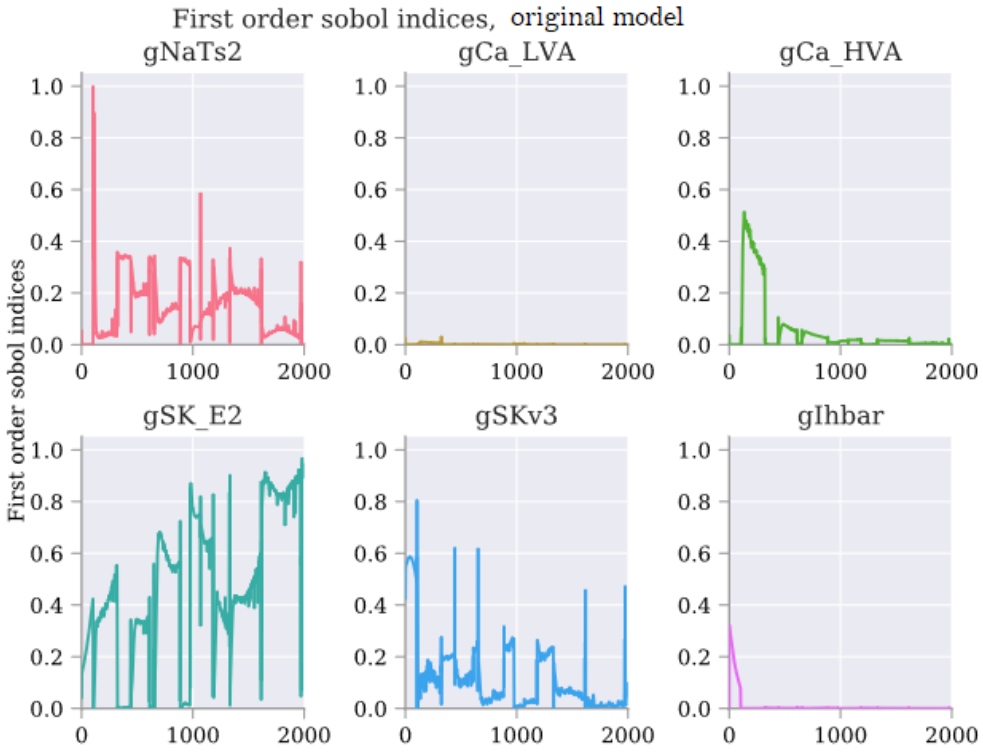

Figure 31: Sensitivity analysis of the parameters in the original model changing as a function of time.

The same analysis for the reduced model is illustrated in Figure 32. Here it can be seen even more clearly that during spiking the uncertainty arises from sodium conductance and just after spiking from potassium conductances. Between the spikes the uncertainty arises from the calcium activated potassium and high voltage activated calcium channels.

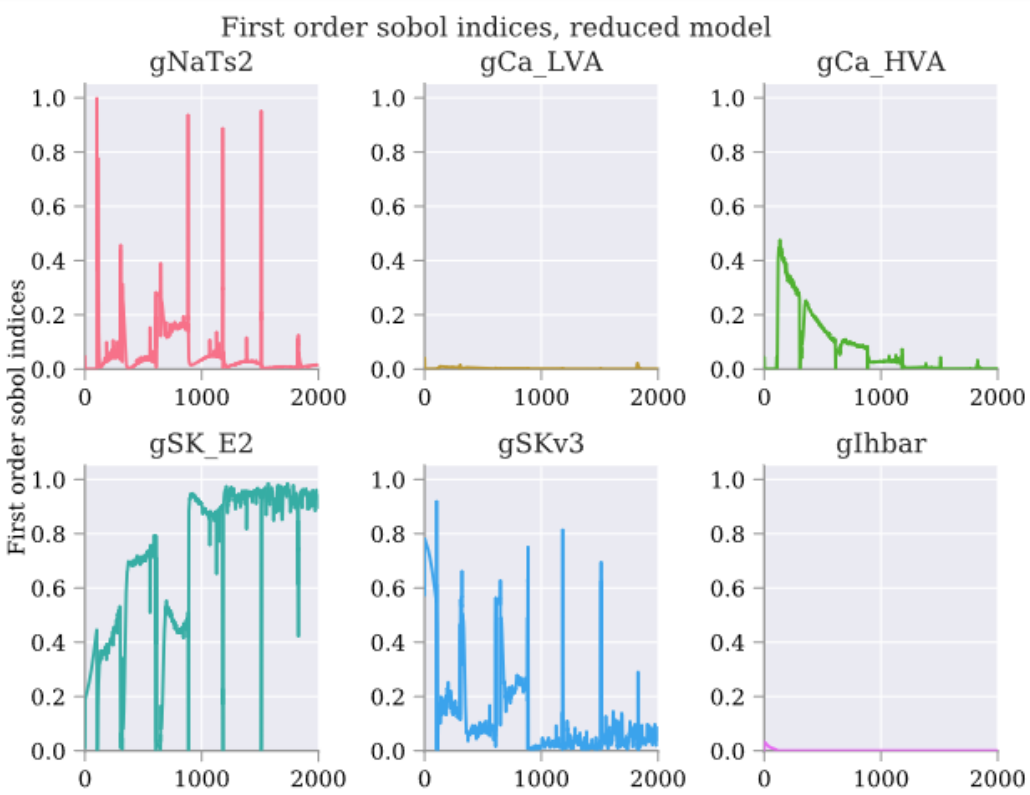

Figure 32: Sensitivity analysis of the parameters in the reduced model changing as a function of time.

Sensitivity analysis and uncertainty quantification are important when building models with large number of parameters that have an inherent variability. The results presented here show that Uncertainpy toolbox is a very useful tool for analyzing how the uncertainty of the parameters affect the output of the model. Uncertainpy is a very promising tool and will be very useful when used for analyzing one's own model.

6. DISCUSSION

The aim of this thesis was to review existing compartmental models of cortical pyramidal cells. The models were evaluated, and their features were analyzed. Another aim was to simulate the excitability of a model and test two recent and promising Python-based toolboxes, one for simplifying the complexity of a model, and the other for sensitivity analysis of the uncertain parameters in the chosen model. In this chapter, the results of the thesis are discussed.

6.1 A review of compartmental models

The review introduced 50 publications with cortical pyramidal cell models. The models were compared with the following criteria: 1) implementation software, 2) model availability in public repositories, 3) experimental data that the model were verified against, 4) the brain area and layer of the cortex that the experimental data were originated from, 5) the number of compartments the model had and 6) the biophysical properties that were modeled. 40 models were available in public databases and 36 models were implemented in NEURON simulator. Those 36 models were evaluated based on the described criteria.

Majority of the models were based on rodent experimental data, mostly from rats, but a couple from mice. Four studies used human experimental data. The studies that specified the region where the cells were form included cells from somatosensory, prefrontal, temporal and visual cortex. Majority of the models represented large pyramidal cells from layer 5, but also models from layer 2/3, 4 and 6 were included. The number of compartments in the models varied from 2 to almost 3000 and the amount of different ion channels was also diverse. However, the models are designed for different purposes and their features vary depending on what kind of phenomenon the model is intended to represent.

The review included only models that were presented in the publications. There are also numerous neuron models in different repositories, such as in ModelDB [64], Blue Brain Portal [65] and Allen Brain Atlas [66]. This study focused on models that were implemented with NEURON simulator but also other simulation environments exist, for example Nest [67], Brian [68] and GENESIS [69].

The review has some limitations. The data is gathered from the publications and some publications had limited information of their models. The manner how the information was presented varied a lot. Some publications provided the specification of the model in the paper, but in some cases the information had to be searched from the model source code files. Additionally, the style of the information was not standard. For example, comparing the ion channels was quite demanding because the publications used different abbreviations for the same channels (i.e. fast sodium channel and transient sodium channel). Furthermore, when testing the models, we noticed that some of the models did

not work properly or only part of the code was working. We also noticed that some models included only part of the code so we could not reproduce the results mentioned in the publications completely. Some models were only demo versions and did not produce entirely similar results than the actual model in the publication.

To conclude, the review shows that the repertoire of compartmental models of cortical pyramidal cells is diverse. The models are used for modeling different kinds of neuronal behaviors. The models represent the biological realism of cortical pyramidal cells as well as the bioelectrical properties of the cells in different levels of detail. The diversity and extent of the models show that cortical pyramidal cells have captured the interest of neuroscientists and compartmental modeling is a useful tool for studying them.

6.2 *Neuron_Reduce*

Neuron_Reduce is an open-source Python toolbox that is used for reducing the complexity of detailed conductance-based nonlinear neuron models [4]. It accelerates the simulation speed by 40-250 folds while replicating the original model's voltage dynamics and dendritic computations. In this section, the user experience with the tool is discussed as well as the results obtained using the tool.

Neuron_Reduce toolbox is a very recent tool, published in 2019, and has still limited documentation. Most of the instructions needed for use the tool can be found in the comments of the source code. However, a more complete documentation is not available at the moment. To use Neuron_Reduce, some programming skills are required to understand how the tool works. A detailed example code is provided so advanced programming skills are not needed.

Based on our user experience, some recommendations for advancing the tool could be given. For the tool to be used more widely, a description for the structure of the model could ease the use of the tool. Currently, the model needs to have a certain structure that is compatible with the tool. The NEURON model must have a name, which is then called in the code. The file format for morphology needs to be either .asc or .swc, other file formats are not supported. It is not either clear, what commands the NEURON template should have. In addition, for the future use, it could be beneficial if the reduced model could be saved and modified. In the current state, it is not clear how to execute such an operation.

Next, the results obtained with the Neuron_Reduce toolbox are discussed. We used the toolbox with the layer 2/3 pyramidal cell model from rat somatosensory cortex. We reduced the cell to 8, 30, 50 and 78 compartments and simulated them for 30 seconds and 10 minutes of biological time. The simulation times were substantially shorter in the reduced cases. We also noticed that the simulation times for the reduced models increased when the number of compartments increased. However, these simulation times cannot be compared as such, because the complex cell simulation times also varied. For example, for the 30 second biological time, the complex cell simulation time varied from 1500 seconds to over 1700 seconds. This may be due to the interface between Python and

NEURON simulator. For comparison, the simulation time for reduced cell with 30 compartments was between 35 and 40 seconds. In this case, the simulation speed was accelerated 40-45-fold.

With 30, 50 and 78 compartments, the reduced cells seemed to produce similar behavior than the original cell. It was surprising that all the moderately reduced cells produced similar behavior. It could be assumed that the reduced cell with 78 compartments would produce more similar results than the reduced cell with 30 compartments. However, when the cell was reduced dramatically, the reduced cell no longer behaved like the original cell. Hence, it seems that Neuron_Reduce works when the model is not over-reduced.

We also noticed that the ISIs for the reduced models are shorter than the ISI for the complex cell in every case. It also seemed that the reduced models produce more regular activity than the original. Some difference in the behavior was expected as the input was random synaptic events. Also, it is not completely clear how the model and its features affect the results. To get an overview, more models should be studied.

All things considered, Neuron_Reduce is a very promising tool for reducing the complexity of the model. It accelerates the simulation speed notably. With the tool, longer simulations and larger neuronal networks can be simulated. Neuron_Reduce has some limitations, but it is expected for a new tool that can still benefit from extensive testing.

6.3 *Uncertainpy*

Uncertainpy is a Python toolbox for sensitivity analysis and uncertainty quantification [3]. It is tailored towards neuronal models but can also be used for universal models. The analysis is based on either polynomial chaos expansions or quasi-Monte Carlo methods. In this section, the user experience with Uncertainpy is discussed as well as the results obtained with the tool.

Getting started with Uncertainpy is easy. The tool is provided with extensive documentation. It includes detailed description of the tool and a couple of example codes. All the functionalities are described exhaustively with given examples. Uncertainpy also utilizes the eFEL features [58], which automatically extract features from the data recorded from neurons.

Uncertainpy has some specific features that must be considered when using it. The toolbox makes some assumptions of NEURON simulator models. The tool has a class for NEURON models, but we were unable to utilize the class with the model used in the simulations. We also noticed that Uncertainpy uses Xvfb display server that is not compatible with Windows and therefore, we could not use the tool with Windows machines. Another issue that requires consideration is the memory usage. The analysis is computationally heavy and requires a lot of memory, time and power.
Next, the results obtained with Uncertainpy toolbox are discussed. Sensitivity analysis was performed for the original model and for the reduced model with 30 compartments.

The parameters chosen for the analysis were the maximum conductances in the soma. We studied how the uncertainty of the different parameters affect the spiking features of the model. The features included were average AP overshoot, average AP width and spike rate. We chose the distributions for the parameters to be 50% around the original value.

For the average peak voltage of the action potential, the transient sodium conductance is the primary source of uncertainty in both original and reduced model. Transient sodium conductance controls the depolarization phase of the action potential, so it was expected that a substantial amount of uncertainty arises from sodium conductance. For the average AP width, transient sodium conductance was the main source of uncertainty, but some uncertainty arises also from the potassium conductance. In the original model, high-voltage activated calcium had also a minor effect but otherwise the results were similar in both models. The largest difference between the original model and reduced model was when the uncertainty for the spike rate feature was analyzed. In the original model, all parameters affect the sensitivity of the feature. For the reduced model, sodium conductance had the biggest impact and then calcium activated potassium conductance and voltage dependent potassium conductance some. In this case, neither of the voltage activated calcium conductances nor h-current had effect on the sensitivity.

The sensitivity of the parameters was also analyzed as a function of time. The time-dependent analysis visualizes even better how the spikes are depended on sodium and potassium conductances. During the intervals between spikes, calcium activated potassium conductance and high-voltage activated calcium are responsible for the uncertainty. These can be seen even more clearly in the reduced model's analysis.

Sensitivity analysis and uncertainty quantification are important when building models with large number of parameters that have an inherent variability. Uncertainpy toolbox is a very useful tool for analyzing how the uncertainty of the parameter affect the output of the model. Uncertainpy will be very useful when used for analyzing one's own model. However, using the tool with complex neuron models needs a lot of memory and computational power, which must be taken into account when using the tool.

7. CONCLUSIONS

The purpose of this thesis was to review publications with compartmental models of cortical pyramidal cells and evaluate the models. Another goal was a simulation-based analysis consisting of three parts: 1) testing the excitability of the neuron models, 2) testing Neuron_Reduce Python toolbox for simplifying the complexity of a model and 3) testing the Uncertainpy Python toolbox for sensitivity analysis of the uncertain parameters of a model.

The literature review showed that more than 50 publications with compartmental models of cortical pyramidal cells were published between the years 1995-2020. The complexity of the morphology and the biophysical properties of the model varied based on what functions the model was intended to represent. Therefore, comparing the models was not straightforward. However, the review offers some guidelines on model selection and gives an overall view of the existing models.

In the simulation-based analysis, two Python toolboxes were tested. The model used in the simulations was a layer 2/3 pyramidal cell model from rat somatosensory cortex. The first tool that was tested is called Neuron_Reduce and it is used for reducing the complexity of neuron models. It accelerated the simulations substantially while replicating the behavior of the original model. With Neuron_Reduce toolbox, longer simulations and larger neuronal networks can be simulated. The tool was published recently so some limitations were expected. For the future, it could be beneficial if a comprehensive documentation of the tool could be provided.

Another Python module was also tested. Uncertainpy toolbox is used for sensitivity analysis and uncertainty quantification. It is tailored towards neuronal models but can also be used for universal models. In this work, the studied parameters were the different maximum conductances in the soma. The results obtained with Uncertainpy were quite as expected and the difference between the features in the original and reduced models could be nicely seen. The major issue with Uncertainpy is the memory capacity needed for the analysis, which will constrain the usability of the tool.

Computational models in neuroscience are often morphologically complex and have many parameters that have inherent variability. The toolboxes tested in this work will be useful to tackle those issues. Both tested toolboxes, Neuron_Reduce and Uncertainpy, have some specific features that may limit their use. But they also have a large potential to become very useful in building one’s model from scratch.

REFERENCES

[1] I. Segev, “Cable and Compartmental Models of Dendritic Trees,” in *The Book of GENESIS*, New York, NY: Springer New York, 1998, pp. 51–77.

[2] J. M. Bekkers, “Pyramidal neurons,” *Current Biology*, vol. 21, no. 24. Cell Press, 20-Dec-2011.

[3] S. Tennøe, G. Halnes, and G. T. Einevoll, “Uncertainpy: A Python Toolbox for Uncertainty Quantification and Sensitivity Analysis in Computational Neuroscience,” *Front. Neuroinform.*, vol. 12, p. 49, Aug. 2018.

[4] O. Amsalem *et al.*, “An efficient analytical reduction of detailed nonlinear neuron models,” *Nat. Commun.*, vol. 11, no. 1, pp. 1–13, 2020.

[5] S. Shipp, “Structure and function of the cerebral cortex.,” *Curr. Biol.*, vol. 17, no. 12, pp. R443-9, Jun. 2007.

[6] F. A. C. Azevedo *et al.*, “Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain,” *J. Comp. Neurol.*, vol. 513, no. 5, pp. 532–541, Apr. 2009.

[7] R. Swenson, “The Cerebral Cortex,” in *Review of Clinical and Functional Neuroscience*, Dartmouth, 2006.

[8] T. M. Kandel, E.; Schwartz, J.; Jessel, *Principles of Neural Science*, 5th ed. McGraw-Hill Companies, 2013, pp. 10-11, 126, 140-141, 150-151, 175, 182, 209, 222

[9] “Sensory Cortex | Visual, Auditory Cortex | Facts & Summary.” [Online]. Available: https://human-memory.net/sensory-cortex/. [Accessed: 09-Jan-2020].

[10] N. Spruston, “Pyramidal neurons: Dendritic structure and synaptic integration,” *Nature Reviews Neuroscience*, vol. 9, no. 3. pp. 206–221, Mar-2008.

[11] E. A. Nimchinsky, B. L. Sabatini, and K. Svoboda, “Structure and Function of Dendritic Spines,” *Annu. Rev. Physiol.*, vol. 64, no. 1, pp. 313–353, Mar. 2002.

[12] M. Hiroyoshi, “Synaptic Integration,” in *Encyclopedia of Neuroscience*, M. D. Binder, N. Hirokawa, and U. Windhorst, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 3952–3956.

[13] V. L. Harvey and A. H. Dickenson, “EPSPs and IPSPs,” in *Encyclopedia of Psychopharmacology*, I. P. Stolerman, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, p. 489.

[14] M. Hausser, N. Spruston, and G. J. Stuart, “Diversity and dynamics of dendritic signaling,” *Science*, vol. 290, no. 5492. pp. 739–744, 27-Oct-2000.

[15] A. L. Hodgkin and A. F. Huxley, “A quantitative description of membrane current and its application to conduction and excitation in nerve.,” *J. Physiol.*, vol. 117, no. 4, pp. 500–44, Aug. 1952.

[16] E. Neher and B. Sakmann, “Single-channel currents recorded from membrane of denervated frog muscle fibres,” *Nature*, vol. 260, no. 5554, pp. 799–802, 1976.

[17] D. Sterratt, B. Graham, A. Gillies, and D. Willshaw, *Principles of computational modelling in neuroscience*. Cambridge University Press, 2011.

[18] E. Hay, S. Hill, F. Schürmann, H. Markram, and I. Segev, “Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties,” *PLoS Comput. Biol.*, vol. 7, no. 7, Jul. 2011.

[19] K. Sidiropoulou and P. Poirazi, “Predictive features of persistent activity emergence in regular spiking and intrinsic bursting model neurons,” *PLoS Comput. Biol.*, vol. 8, no. 4, Apr. 2012.

[20] F. Skinner, “Conductance-based models,” *Scholarpedia*, vol. 1, no. 11, p. 1408, Nov. 2006.

[21] W. Rall, “Branching dendritic trees and motoneuron membrane resistivity,” *Exp. Neurol.*, vol. 1, no. 5, pp. 491–527, 1959.

[22] W. Rall, I. Segev, J. Rinzel, and G. M. Shepherd, *The theoretical foundation of dendritic function : selected papers of Wilfrid Rall with commentaries*. MIT Press, 1995.

[23] A. Poleg-Polsky, “Effects of neural morphology and input distribution on synaptic processing by global and focal NMDA-spikes,” *PLoS One*, vol. 10, no. 10, Oct. 2015.

[24] G. Eyal *et al.*, “Unique membrane properties and enhanced signal processing in human neocortical neurons,” *Elife*, vol. 5, no. OCTOBER2016, 2016.

[25] G. Eyal *et al.*, “Human cortical pyramidal neurons: From spines to spikes via models,” *Front. Cell. Neurosci.*, vol. 12, Jun. 2018.

[26] Y. Deitcher *et al.*, “Comprehensive Morpho-Electrotonic Analysis Shows 2 Distinct Classes of L2 and L3 Pyramidal Neurons in Human Temporal Cortex.,” *Cereb. Cortex*, vol. 27, no. 11, pp. 5398–5414, 2017.

[27] S. L. Smith, I. T. Smith, T. Branco, and M. Häusser, “Dendritic spikes enhance stimulus selectivity in cortical neurons in vivo,” *Nature*, vol. 503, no. 7474, pp. 115–120, Nov. 2013.

[28] Y. Shu, A. Duque, G. Yu, B. Haider, and D. A. McCormick, “Properties of action-potential initiation in neocortical pyramidal cells: Evidence from whole cell axon recordings,” *J. Neurophysiol.*, vol. 97, no. 1, pp. 746–760, Jan. 2007.

[29] A. Bahl, M. B. Stemmler, A. V. M. Herz, and A. Roth, “Automated optimization of a reduced layer 5 pyramidal cell model based on experimental data,” *J. Neurosci. Methods*, vol. 210, no. 1, pp. 22–34, Sep. 2012.

[30] B. F. Behabadi and B. W. Mel, “Mechanisms underlying subunit independence in pyramidal neuron dendrites,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 111, no. 1, pp. 498–503, 2014.

[31] G. de Sousa, R. Maex, R. Adams, N. Davey, and V. Steuber, “Dendritic morphology predicts pattern recognition performance in multi-compartmental

model neurons with and without active conductances," *J. Comput. Neurosci.*, vol. 38, no. 2, pp. 221–234, 2015.

[32] M. Doron, G. Chindemi, E. Muller, H. Markram, and I. Segev, "Timed Synaptic Inhibition Shapes NMDA Spikes, Influencing Local Dendritic Processing and Global I/O Properties of Cortical Neurons," *Cell Rep.*, vol. 21, no. 6, pp. 1550–1561, Nov. 2017.

[33] X. Li, K. Morita, H. P. C. Robinson, and M. Small, "Control of layer 5 pyramidal cell spiking by oscillatory inhibition in the distal apical dendrites: a computational modeling study," *J. Neurophysiol.*, vol. 109, no. 11, pp. 2739–2756, Jun. 2013.

[34] R. A. J. van Elburg and A. van Ooyen, "Impact of Dendritic Size and Dendritic Topology on Burst Firing in Pyramidal Cells," *PLoS Comput. Biol.*, vol. 6, no. 5, p. e1000781, May 2010.

[35] A. S. Aberra, A. V Peterchev, and W. M. Grill, "Biophysically realistic neuron models for simulation of cortical stimulation," *J. Neural Eng.*, vol. 15, no. 6, 2018.

[36] M. Almog and A. Korngreen, "A quantitative description of dendritic conductances and its application to dendritic excitation in layer 5 pyramidal neurons," *J. Neurosci.*, vol. 34, no. 1, pp. 182–196, Jan. 2014.

[37] A. Arkhipov *et al.*, "Visual physiology of the layer 4 cortical circuit in silico," *PLoS Comput. Biol.*, vol. 14, no. 11, Nov. 2018.

[38] B. F. Behabadi, A. Polsky, M. Jadi, J. Schiller, and B. W. Mel, "Location-dependent excitatory synaptic interactions in pyramidal neuron dendrites," *PLoS Comput. Biol.*, vol. 8, no. 7, Jul. 2012.

[39] A. R. Galloni, A. Laffere, and E. A. Rancz, "Apical length modulates dendritic excitability in L5 pyramidal neurons," *bioRxiv*, p. 754499, Sep. 2019.

[40] A. Gidon *et al.*, "Dendritic action potentials and computation in human layer 2/3 cortical neurons," *Science (80-.).*, vol. 367, no. 6473, pp. 83–87, Jan. 2020.

[41] N. W. Gouwens *et al.*, "Systematic generation of biophysically detailed models for diverse cortical neuron types," *Nat. Commun.*, vol. 9, no. 1, pp. 1–13, Dec. 2018.

[42] W. Hu, C. Tian, T. Li, M. Yang, H. Hou, and Y. Shu, "Distinct contributions of Nav1.6 and Nav1.2 in action potential initiation and backpropagation," *Nat. Neurosci.*, vol. 12, no. 8, pp. 996–1002, Aug. 2009.

[43] B. M. Kampa and G. J. Stuart, "Calcium spikes in basal dendrites of layer 5 pyramidal neurons during action potential bursts," *J. Neurosci.*, vol. 26, no. 28, pp. 7424–7432, Jul. 2006.

[44] M. H. P. Kole, S. U. Ilschner, B. M. Kampa, S. R. Williams, P. C. Ruben, and G. J. Stuart, "Action potential generation requires a high sodium channel density in the axon initial segment," *Nat. Neurosci.*, vol. 11, no. 2, pp. 178–186, Feb. 2008.

[45] M. H. P. Kole, S. Hallermann, and G. J. Stuart, "Single Ih channels in pyramidal neuron dendrites: Properties, distribution, and impact on action potential output," *J. Neurosci.*, vol. 26, no. 6, pp. 1677–1687, Feb. 2006.

[46] M. E. Larkum, T. Nevian, M. Sandier, A. Polsky, and J. Schiller, "Synaptic

integration in tuft dendrites of layer 5 pyramidal neurons: A new unifying principle," *Science (80-.).*, vol. 325, no. 5941, pp. 756–760, 2009.

[47] Z. F. Mainen, J. Joerges, J. R. Huguenard, and T. J. Sejnowski, "A model of spike initiation in neocortical pyramidal neurons," *Neuron*, vol. 15, no. 6, pp. 1427–1439, 1995.

[48] H. Markram *et al.*, "Reconstruction and Simulation of Neocortical Microcircuitry," *Cell*, vol. 163, no. 2, pp. 456–492, Oct. 2015.

[49] T. Mäki-Marttunen *et al.*, "A stepwise neuron model fitting procedure designed for recordings with high spatial resolution: Application to layer 5 pyramidal cells," *J. Neurosci. Methods*, vol. 293, pp. 264–283, Jan. 2018.

[50] T. Nevian, M. E. Larkum, A. Polsky, and J. Schiller, "Properties of basal dendrites of layer 5 pyramidal neurons: A direct patch-clamp recording study," *Nat. Neurosci.*, vol. 10, no. 2, pp. 206–214, Feb. 2007.

[51] A. Papoutsi, G. Kastellakis, and P. Poirazi, "Basal tree complexity shapes functional pathways in the prefrontal cortex," *J. Neurophysiol.*, vol. 118, no. 4, pp. 1970–1983, Oct. 2017.

[52] A. Polsky, B. Mel, and J. Schiller, "Encoding and decoding bursts by NMDA spikes in basal dendrites of layer 5 pyramidal neurons," *J. Neurosci.*, vol. 29, no. 38, pp. 11891–11903, Sep. 2009.

[53] A. T. Schaefer, M. E. Larkum, B. Sakmann, and A. Roth, "Coincidence detection in pyramidal neurons is tuned by their dendritic branching pattern," *J. Neurophysiol.*, vol. 89, no. 6, pp. 3143–3154, Jun. 2003.

[54] A. S. Shai, C. A. Anastassiou, M. E. Larkum, and C. Koch, "Physiology of Layer 5 Pyramidal Neurons in Mouse Primary Visual Cortex: Coincidence Detection through Bursting," *PLOS Comput. Biol.*, vol. 11, no. 3, p. e1004090, Mar. 2015.

[55] G. Stuart and N. Spruston, "Determinants of Voltage Attenuation in Neocortical Pyramidal Neuron Dendrites," 1998.

[56] R. D. Traub, E. H. Buhl, T. Gloveli, and M. A. Whittington, "Fast rhythmic bursting can be induced in layer 2/3 cortical neurons by enhancing persistent Na+ conductance or by blocking BK channels," *J. Neurophysiol.*, vol. 89, no. 2, pp. 909–921, Feb. 2003.

[57] N. T. Carnevale and M. L. Hines, *The NEURON Book*. Cambridge: Cambridge University Press, 2006.

[58] "Electrophys Feature Extraction Library — eFEL 3.0.80 documentation." [Online]. Available: https://efel.readthedocs.io/en/latest/index.html. [Accessed: 08-Apr-2020].

[59] B. Rudy and C. J. McBain, "Kv3 channels: Voltage-gated K+ channels designed for high-frequency repetitive firing," *Trends in Neurosciences*, vol. 24, no. 9. Elsevier, pp. 517–526, 01-Sep-2001.

[60] E. S. L. Faber and P. Sah, "Functions of SK channels in central neurons," in *Clinical and Experimental Pharmacology and Physiology*, 2007, vol. 34, no. 10, pp. 1077–1083.

[61] J. Engbers, “Low-Voltage-Activated Calcium Channels,” in *Encyclopedia of Computational Neuroscience*, Springer New York, 2014, pp. 1–5.

[62] S. Solinas, S. Masoli, and S. Subramaniyam, “High-Voltage-Activated Calcium Channels,” in *Encyclopedia of Computational Neuroscience*, Springer New York, 2014, pp. 1–7.

[63] A. Momin, H. Cadiou, A. Mason, and P. A. Mcnaughton, “Role of the hyperpolarization-activated current Ih in somatosensory neurons,” *J. Physiol.*, vol. 586, no. 24, pp. 5911–5929, Dec. 2008.

[64] “ModelDB: Home.” [Online]. Available: https://senselab.med.yale.edu/modeldb/. [Accessed: 17-Apr-2020].

[65] “Blue Brain Portal – Digitally reconstructing and simulating the brain.” [Online]. Available: https://portal.bluebrain.epfl.ch/. [Accessed: 16-Apr-2020].

[66] “Brain Map - brain-map.org.” [Online]. Available: https://portal.brain-map.org/. [Accessed: 16-Apr-2020].

[67] T. Fardet *et al.*, “NEST 2.20.0.” 2020. [Online]. Available: https://www.nest-simulator.org/. [Accessed: 16-Apr-2020].

[68] “The Brian Simulator | The Brian spiking neural network simulator.” [Online]. Available: https://briansimulator.org/. [Accessed: 16-Apr-2020].

[69] “The GENESIS Simulator.” [Online]. Available: http://genesis-sim.org/. [Accessed: 16-Apr-2020].

APPENDIX A:

A.1 Publications of compartmental models of cortical pyramidal cells in the years 1995-2020

Author	Year	Name of the publication	Model online/Model Database	Simulator
Aberra et al	2018	Biophysically realistic neuron models for simulation of cortical stimulation	yes, ModelDB	NEURON
Almog & Korngreen	2014	A Quantitative Description of Dendritic Conductances and Its Application to Dendritic Excitation in Layer 5 Pyramidal Neurons	yes, ModelDB	NEURON
Arkhipov et al	2018	Visual physiology of the layer 4 cortical circuit in silico	yes, AI	NEURON
Bahl et al	2012	Automated optimization of a reduced layer 5 pyramidal cell model based on experimental data	yes, ModelDB	NEURON
Beaulieu-Laroche et al	2018	Enhanced Dendritic Compartmentalization in Human Cortical Neurons	no	MATLAB
Behabadi & Mel	2014	Mechanisms underlying subunit independence in pyramidal neuron dendrites	yes, ModelDB	NEURON
Behabadi et al	2012	Location-Dependent Excitatory Synaptic Interactions in Pyramidal Neuron Dendrites	no, available on request	NEURON
Branco & Häusser	2011	Synaptic Integration Gradients in Single Cortical Pyramidal Cell Dendrites	no	NEURON
Caze et al	2017	Dendrites Enable a Robust Mechanism for Neuronal Stimulus Selectivity.	yes, ModelDB	NEURON
de Sousa et al	2015	Dendritic morphology predicts pattern recognition performance in multi-compartmental model neurons with and without active conductances	yes, link in the paper	NEURON
Deitcher et al	2017	2 Distinct Classes of L2 and L3 Pyramidal Neurons in Human Temporal Cortex	yes, ModelDB	NEURON
Doron et al	2017	Timed Synaptic Inhibition Shapes NMDA Spikes, Influencing Local Dendritic Processing and Global I/O Properties of Cortical Neurons.	yes, ModelDB	NEURON
Eyal et al	2018	Human Cortical Pyramidal Neurons: From Spines to Spikes via Models	yes, ModelDB	NEURON
Eyal et al	2016	Unique membrane properties and enhanced signal processing in human neocortical neurons	yes, ModelDB	NEURON

Felton et al	2018	Resonance Analysis as a Tool for Characterizing Functional Division of Layer 5 Pyramidal Neurons	used the model by Traub	GENESIS
Galloni et al	2019	Apical length modulates dendritic excitability in L5 pyramidal neurons	used the code by Hay et al	NEURON
Gidon	2020	Dendritic action potentials and computation in human layer 2/3 cortical neurons	yes, ModelDB	NEURON
Gouwens et al	2018	Systematic generation of biophysically detailed models for diverse cortical neuron types	yes, AI	NEURON
Hay et al	2011	Models of Neocortical Layer 5b Pyramidal Cells Capturing a Wide Range of Dendritic and Perisomatic Active Properties	yes, ModelDB	NEURON
Hu et al	2009	Distinct contributions of Nav1.6 and Nav1.2 in action potential initiation and backpropagation	yes, ModelDB	NEURON
Huys et al	2006	Efficient estimation of detailed single-neuron models.	yes, ModelDB	MATLAB
Kalmbach et al	2018	h-Channels Contribute to Divergent Intrinsic Membrane Properties of Supragranular Pyramidal Neurons in Human versus Mouse Cerebral Cortex	yes, AI	NEURON
Kampa & Stuart	2006	Calcium spikes in basal dendrites of layer 5 pyramidal neurons during action potential bursts	yes, ModelDB	NEURON
Keren et al	2009	Experimentally guided modelling of dendritic excitability in rat neocortical pyramidal neurones	used the model based on Mainen et al (1995)	NEURON
Kole et al	2008	Action potential generation requires a high sodium channel density in the axon initial segment	yes, ModelDB	NEURON
Kole et al	2006	Single Ih channels in pyramidal neuron dendrites: properties, distribution, and impact on action potential output	yes, ModelDB	NEURON
Larkum et al	2009	Synaptic integration in tuft dendrites of layer 5 pyramidal neurons: a new unifying principle	yes, ModelDB	NEURON
Li et al	2013	Control of layer 5 pyramidal cell spiking by oscillatory inhibition in the distal apical dendrites: a computational modeling study	yes, ModelDB	NEURON
Mainen et al	1995	A Model of Spike Initiation in Neocortical Pyramidal Neurons	yes, ModelDB	NEURON
Markram et al	2015	Reconstruction and Simulation of Neocortical Microcircuitry	yes, BBP	NEURON

Mohan et al	2015	Dendritic and Axonal Architecture of Individual Pyramidal Neurons across Layers of Adult Human Neocortex	no	NEURON
Mäki-Marttunen et al	2018	A stepwise neuron model fitting procedure designed for recordings with high spatial resolution: Application to layer 5 pyramidal cells	yes, ModelDB	NEURON
Nevian et al	2007	Properties of basal dendrites of layer 5 pyramidal neurons: a direct patch-clamp recording study	yes, ModelDB	NEURON
Papoutsi et al	2017	Basal tree complexity shapes functional pathways in the prefrontal cortex	yes, ModelDB	NEURON
Poleg-Polsky	2015	Effects of Neural Morphology and Input Distribution on Synaptic Processing by Global and Focal NMDA-Spikes	yes, ModelDB	NEURON
Polsky et al	2009	Encoding and decoding bursts by NMDA spikes in basal dendrites of layer 5 pyramidal neurons	yes, ModelDB	NEURON
Sarid et al	2007	Modeling a layer 4-to-layer 2/3 module of a single column in rat neocortex: Interweaving in vitro and in vivo experimental observations	no	NEURON
Schaefer et al	2003	Coincidence detection in pyramidal neurons is tuned by their dendritic branching pattern	yes, ModelDB	NEURON
Schiller et al	2000	NMDA spikes in basal dendrites of cortical pyramidal neurons	No	NEURON
Shai et al	2015	Physiology of Layer 5 Pyramidal Neurons in Mouse Primary Visual Cortex: Coincidence Detection through Bursting	yes, ModelDB	NEURON
Shu et al	2007	Properties of Action-Potential Initiation in Neocortical Pyramidal Cells: Evidence From Whole Cell Axon Recordings	no	NEURON
Sidiropoulou & Poirazi	2012	Predictive Features of Persistent Activity Emergence in Regular Spiking and Intrinsic Bursting Model Neurons	yes, ModelDB	NEURON
Smith et al	2013	Dendritic spikes enhance stimulus selectivity in cortical neurons in vivo	yes, ModelDB	NEURON
Stiefel et al	2009	The effects of cholinergic neuromodulation on neuronal phase-response curves of modeled cortical neurons	yes, ModelDB	NEURON

Stuart & Spruston	1998	Determinants of Voltage Attenuation in Neocortical Pyramidal Neuron Dendrites	yes, ModelDB	NEURON
Traub et al	2003	Fast rhythmic bursting can be induced in layer 2/3 cortical neurons by enhancing persistent Na+ conductance or by blocking BK channels	yes, ModelDB	NEURON
van Elburg & van Ooyer	2010	Impact of dendritic size and dendritic topology on burst firing in pyramidal cells	yes, ModelDB	NEURON
Weaver et al	2006	The role of action potential shape and parameter constraints in optimization of compartment models	yes, ModelDB	NEURON
Yi et al	2017	Dendritic properties control energy efficiency of action potentials in cortical pyramidal cells	yes, ModelDB	MATLAB
Yi et al	2017	Action potential initiation in a two-compartment model of pyramidal neuron mediated by dendritic Ca2+ spike	described in the paper	MATLAB

A.2 Abbreviations of ion channels, currents and receptors

Ca dyn	Calcium dynamics
Ca_HVA	High-voltage activated Ca channels
Ca_LVA	Low-voltage activated Ca channels
Cal	L-type voltage-activated calcium channels
Can	N-type voltage-activated calcium channels
Cap	P-type voltage-activated calcium channels
Cas	Slow Ca2+ channels
Cat	T-type voltage-activated calcium channels
CP	Calcium-pump
HCN	Hyperpolarization-activated cation channels
IH	H-current
IM	M-current
Ka	Transient K+ channels
KAHP	After hyperpolarization potassium current
KCa	Ca2+ dependent K+ channels
Kd	Delay current
Kdr	Delayed-rectifier K+ current
Kfast	Fast voltage-activated K+ channels
Km	Muscarinic K+ channels
Ks	Stochastic K+ channels
Kslow	Slow voltage-activated K+ channels
Kv	Voltage-gated K+ channels
Na	Fast Na+ channels
Nap	Persistent voltage-activated Na+ channels
Nat	Transient voltage-activated Na+ channels
SK	Small-conductance calcium-activated potassium channels

A.3 Code for reducing the complexity of neurons

```
from __future__ import division
from neuron import gui,h
import numpy as np
import neuron_reduce
import time
import matplotlib.pyplot as plt

#Loading original cell
h.load_file('biophysics.hoc')
h.load_file("import3d.hoc")
h.load_file('template.hoc')
complex_cell = h.cADpyr229_L23_PC_5ecbf9b163(0) #Name of the cell
h.celsius = 37
h.v_init = complex_cell.soma[0].e_pas

#Add synapses to the model
synapses_list, netstims_list, netcons_list, randoms_list = [], [], [],
[]

all_segments = [i for j in map(list,list(complex_cell.apical)) for i in
j] + [i for j in map(list,list(complex_cell.basal)) for i in j]
len_per_segment = np.array([seg.sec.L/seg.sec.nseg for seg in all_seg-
ments])
rnd = np.random.RandomState(10)
for i in range(10000):
  seg_for_synapse   =   rnd.choice(all_segments,        p=len_per_seg-
ment/sum(len_per_segment))
  synapses_list.append(h.Exp2Syn(seg_for_synapse))
  if rnd.uniform()<0.85:
     e_syn, tau1, tau2, spike_interval, syn_weight = 0, 0.3, 1.8,
     1000/2.5, 0.0016
  else:
     e_syn, tau1, tau2, spike_interval, syn_weight = -86, 1, 8,
     1000/15.0, 0.0008

  #Set synaptic variables
  synapses_list[i].e, synapses_list[i].tau1, synases_list[i].tau2 =
  e_syn, tau1, tau2

  #Set netstim variables
  netstims_list.append(h.NetStim())
  netstims_list[i].interval,
  netstims_list[i].number,
  netstims_list[i].start,
  netstims_list[i].noise = spike_interval, 9e9, 100, 1

   #Set random
   randoms_list.append(h.Random())
   randoms_list[i].Random123(i)
   randoms_list[i].negexp(1)
```

```
    netstims_list[i].noiseFromRandom(randoms_list[i])

    #Set netcon varibales
    netcons_list.append(h.NetCon(netstims_list[i], synapses_list[i] ))
    netcons_list[i].delay, netcons_list[i].weight[0] = 0, syn_weight

#Simulate the full neuron for 1 second
soma_v = h.Vector()
soma_v.record(complex_cell.soma[0](0.5)._ref_v)

time_v = h.Vector()
time_v.record(h._ref_t)

h.tstop = 1000 #Define the simulation time
st = time.time()
h.run()
print('complex cell simulation time {:.4f}'.format(time.time()-st))
complex_cell_v = list(soma_v)

# Apply Neuron_Reduce to simplify the cell
reduced_cell, synapses_list, netcons_list = neuron_reduce.subtree_re-
ductor(complex_cell,   synapses_list,   netcons_list,   reduction_fre-
quency=0, total_segments_manual=-1)

for r in randoms_list:r.seq(1) #Reset random

# Running the simulation again but now on the reduced cell
st = time.time()
h.run()
print('reduced cell simulation time {:.4f}'.format(time.time()-st))
reduced_celll_v = list(soma_v)

# Plotting the results
plt.figure()

plt.plot(time_v, complex_cell_v, label='complex cell')
plt.plot(time_v, reduced_celll_v,  label='reduced cell')
plt.legend()
plt.show()
```

A.4 Code for sensitivity analysis

```
from __future__ import division
from neuron import gui,h
import numpy as np
import neuron_reduce
import time
import matplotlib.pyplot as plt

import uncertainpy as un
import chaospy as cp


# Load the NEURON model that will be reduced
print("Loading files")
#Create a L5_PC model
h.load_file('biophysics.hoc')
h.load_file("import3d.hoc")
h.load_file('template.hoc')
print("Loading cell cADpyr229_L23_PC_5ecbf9b163")
complex_cell = h.cADpyr229_L23_PC_5ecbf9b163(0)
h.celsius = 37
h.v_init = complex_cell.soma[0].e_pas
save_step = 5
stimulus_start = 0
stimulus_end = 2000

# Add synapses to the model
synapses_list, netstims_list, netcons_list, randoms_list = [], [],
[], []
print("assembling synapses")
all_segments = [i for j in map(list,list(complex_cell.apical)) for i
in j] + [i for j in map(list,list(complex_cell.basal)) for i in j]
len_per_segment  =  np.array([seg.sec.L/seg.sec.nseg  for  seg  in
all_segments])
rnd = np.random.RandomState(10)
for i in range(10000):
    seg_for_synapse  =  rnd.choice(all_segments,      p=len_per_seg-
ment/sum(len_per_segment))
    synapses_list.append(h.Exp2Syn(seg_for_synapse))
    if rnd.uniform()<0.85:
        e_syn, tau1, tau2, spike_interval, syn_weight = 0, 0.3, 1.8,
1000/2.5, 0.0016
    else:
        e_syn, tau1, tau2, spike_interval, syn_weight = -86, 1,   8,
1000/15.0, 0.0008


    synapses_list[i].e, synapses_list[i].tau1, synapses_list[i].tau2
    = e_syn, tau1, tau2

    netstims_list.append(h.NetStim())
```

```
        netstims_list[i].interval,netstims_list[i].number,
    netstims_list[i].start,  netstims_list[i].noise  =  spike_interval,
    9e9, 100, 1

        randoms_list.append(h.Random())
        randoms_list[i].Random123(i)
        randoms_list[i].negexp(1)
        netstims_list[i].noiseFromRandom(randoms_list[i])

        netcons_list.append(h.NetCon(netstims_list[i],  synapses_list[i]
))
        netcons_list[i].delay, netcons_list[i].weight[0] = 0, syn_weight

    # Set defaults
    defaults = {}
    defaults['gSK_E2bar_SK_E2'] = complex_cell.soma[0].gSK_E2bar_SK_E2
    defaults['gIhbar_Ih'] = complex_cell.soma[0].gIhbar_Ih
    defaults['gNaTs2_tbar_NaTs2_t’] =
    complex_cell.soma[0].gNaTs2_tbar_NaTs2_t
    defaults['gSKv3_1bar_SKv3_1'] =
    complex_cell.soma[0].gSKv3_1bar_SKv3_1
    defaults['gCa_HVAbar_Ca_HVA'] =
    complex_cell.soma[0].gCa_HVAbar_Ca_HVA
    defaults['gCa_LVAstbar_Ca_LVAst'] =
    complex_cell.soma[0].gCa_LVAstbar_Ca_LVAst

    # Uncertainpy needs a function
    def run_pyramidal_model(gNaTs2, gCa_LVA, gCa_HVA, gSK_E2, gSKv3, gIh-
    bar):
        # reset random seed
        for r in randoms_list:
            r.seq(1)

        complex_cell.soma[0].gNaTs2_tbar_NaTs2_t = gNaTs2
        complex_cell.soma[0].gCa_LVAstbar_Ca_LVAst = gCa_LVA
        complex_cell.soma[0].gCa_HVAbar_Ca_HVA = gCa_HVA
        complex_cell.soma[0].gSK_E2bar_SK_E2 = gSK_E2
        complex_cell.soma[0].gSKv3_1bar_SKv3_1 = gSKv3
        complex_cell.soma[0].gIhbar_Ih = gIhbar

        # Simulate
        soma_v = h.Vector()
        soma_v.record(complex_cell.soma[0](0.5)._ref_v)
        time_v = h.Vector()
        time_v.record(h._ref_t)
        h.tstop = stimulus_end
        h.run()

        voltage = np.array(list(soma_v))
        info = {'stimulus_start': stimulus_start, 'stimulus_end': stimu-
lus_end}
        return   time_v.as_numpy()[::save_step],   voltage[::save_step],
info
```

```
print("simulating")
start = time.time()
cell_t, cell_v, _ = run_pyramidal_model(
    defaults['gNaTs2_tbar_NaTs2_t'],
    defaults['gCa_LVAstbar_Ca_LVAst'],
    defaults['gCa_HVAbar_Ca_HVA'],
    defaults['gSK_E2bar_SK_E2'],
    defaults['gSKv3_1bar_SKv3_1'],
    defaults['gIhbar_Ih'])
stop= time.time()-start
print("stop")
plt.plot(cell_t, cell_v)
plt.show()

low = 0.5
high = 1.5
parameters = {
    'gNaTs2': cp.Uniform(defaults['gSK_E2bar_SK_E2']*low,
                         defaults['gSK_E2bar_SK_E2']*high),
    'gCa_LVA': cp.Uniform(defaults['gCa_LVAstbar_Ca_LVAst']*low,
                          defaults['gCa_LVAstbar_Ca_LVAst']*high),
    'gCa_HVA': cp.Uniform(defaults['gCa_HVAbar_Ca_HVA']*low,
                          defaults['gCa_HVAbar_Ca_HVA']*high),
    'gSK_E2': cp.Uniform(defaults['gSK_E2bar_SK_E2']*low,
                         defaults['gSK_E2bar_SK_E2']*high),
    'gSKv3': cp.Uniform(defaults['gSKv3_1bar_SKv3_1']*low,
                        defaults['gSKv3_1bar_SKv3_1']*high),
    'gIhbar': cp.Uniform(defaults['gIhbar_Ih']*low,
                         defaults['gIhbar_Ih']*high)}

features = un.SpikingFeatures(features_to_run=['spike_rate',
                                               'average_AP_overshoot',
                                               'average_AHP_depth',
                                               'average_AP_width'])
UQ = un.UncertaintyQuantification(
     model=run_pyramidal_model,
     parameters=parameters,
     features=features
  )
data = UQ.quantify(seed=0)
```