Neuromechanical Modeling of Nematode *C. elegans* via Modular Integration and Deep Learning

Jimin Kim

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2025

Reading Committee:
Eli Shlizerman, Chair
Blake Hannaford
Sam Burden

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Neuromechanical Modeling of Nematode *C. elegans* via Modular Integration and Deep Learning

Jimin Kim

Chair of the Supervisory Committee:
Eli Shlizerman
Electrical and Computer Engineering

Neural circuits within the nervous system use coordinated activities to control behavior. The mediation of neural activities by individual neuron dynamics and their integration within the nervous system represents a fundamental question in neuroscience. Computational approaches that integrate modeling of the nervous system, muscles, and the body can assist in investigating functional pathways that guide neural activities and movement. Such approaches are referred to as neuromechanical models, as they incorporate models of the nervous system and biomechanics to achieve simultaneous simulation of neural activities and behavior.

Nematode *Caernorhabditis elegans* (*C. elegans*) is considered a viable framework for studying neuromechanics due to advances in the resolution of its nervous system connectomics, biomechanics, and electrophysiological recordings of neuronal activity. The availability of data allows for the construction of neuromechanical model candidates with varying scopes and modalities. In my PhD research, I proposed key methods for the identification, construction, and extension of neuromechanical models for *C. elegans*. In particular, I proposed the *modular integration* approach and its implementation, modWorm, for modeling and simulating neuromechanical model candidates. The modWorm software allows for the construction of a model as an integrated series of configurable and exchangeable modules, each describing specific biophysical processes.

Using modWorm, I proposed an initial candidate for the integrated neuromechanical model of *C. elegans*.

The model integrates the complete connectome and 7 biophysical modules, including intra- and extra-cellular neural dynamics, translation of neural dynamics to muscle dynamics, muscle dynamics to body postures, and proprioceptive feedback from the environment. The model recapitulates i) Known natural behavioral responses, such as forward and backward locomotion in response to associated neural stimuli or external forces, and ii) Transitional behaviors, such as avoidance and turns, through timed stimulus. We performed computational ablation studies on neurons to infer novel neural circuits involved in sensorimotor behaviors (e.g., touch response). Variations of the model's modules, such as more detailed intra- and extra-cellular dynamics, connectome mappings, and optimizations of associated parameters, can delineate possible mechanisms of locomotion and directions in which the model can be improved to fit experimental findings.

For an extension of modWorm modality, I developed mod-SenseWorm to incorporate environmental stimulus during the simulation of *C. elegans* behavior (e.g., chemotaxis). In particular, mod-SenseWorm incorporates the dynamic translation of external stimulus into neural stimulation to achieve a closed-loop simulation between neuromechanics and the surrounding environment. The translation algorithms employed by individual neurons can be configured by setting their stimulus encoding properties (e.g., tonic, phasic) and anatomical locations in the body (e.g., anterior, posterior). We applied mod-SenseWorm to study *C. elegans* O2 aerotaxis behavior and showed that the proposed model, in conjunction with the simulation of an O2 environment, recapitulates empirically observed avoidance behaviors associated with increased O2 levels. Furthermore, through the analysis of simulated neural activities, we show the use case of mod-SenseWorm to infer potential functional circuits associated with chemotactic responses.

Deep learning methods can assist in extending the scope of the proposed neuromechanical model by inferring the parameters of biologically detailed modules associated with empirical data. This led me to develop ElectroPhysiomeGAN (EP-GAN), a deep generative method for the estimation of biophysical neuron parameters associated with neuron models from recorded electrophysiological responses. Trained with simulation data, EP-GAN learns the translation from recorded neuron responses (e.g., membrane potential responses, steady-state currents) to biophysical model parameters associated with the detailed Hodgkin-Huxley (HH) model. Validation of EP-GAN by estimating HH-model parameters for 200 simulated non-spiking neurons, followed by 9 experimentally recorded neurons in *C. elegans*, showed EP-GAN's advantages in the accuracy of the estimated parameters and inference speed compared to existing estimation methods.

Control strategies can further extend the modality of the neuromechanical model by inferring supplemental mechanisms of neural circuits associated with behavior. In particular, I have introduced a possible employment of deep reinforcement learning (DeepRL) methods to develop control strategies for both neural stimulation (neuromodulatory control) and neural connection mapping (connectome control) that are applied on top of the proposed neuromechanical model to achieve aimed behaviors. The strategies learned by DeepRL can be used to identify dynamic neuromodulatory inputs between neurons (e.g., neuropeptidic currents) and perturbations of the connection wiring map for a local neural circuit, which result in empirically observed chemotactic behavior (e.g., attraction) in response to environmental stimuli. The results highlight the potential of utilizing DeepRL methods in conjunction with the neuromechanical model to infer potential neural interactions and circuitry that lead to specific behaviors.

ACKNOWLEDGEMENTS

I express my utmost gratitude to professor Eli Shlizerman for guiding me throughout my PhD years and cultivating the skills, knowledge, and wisdom necessary to grow as an independent researcher. Most of all, I thank professor Eli Shlizerman for helping me supplement my creativity and intuition with logical thinking, planning, and persistence, which were the aspects I lacked at the start of my PhD.

I thank both my former and current lab colleagues who have offered me advice and inspiration for my research and have provided me with valuable feedback and encouragement during important moments in my PhD years.

I thank my family members, who have always wholeheartedly supported my path with love, care, and resources. I wouldn't have been able to come this far without their help.

I thank professor Mari Ostendorf, professor Blake Hannaford, and professor Shih-Chieh Hsu, with whom I have worked throughout my teaching career, for providing me with valuable feedback and encouragement to cultivate my passion for teaching.

I thank and send my heartfelt love to all the students I have ever taught throughout my teaching career, who have always brightened my spirit, given me new inspiration, and helped me realize my core identity and mission of being of service to others.

Above all, I thank my father in heaven, who has always guided me with unconditional love, truth, and wisdom, and has given me the strength to be my authentic self to create with love and share with others.

DEDICATION

Dedicated to my Father in heaven, my Family, Friends, Teachers, Colleagues, and All of humanity.

Contents

List of Figures

List of Tables

List of Abbreviations

C. elegans *Caenorhabditis elegans*. iii

DDPG Deep Deterministic Policy Gradient. 17

DE Differential Evolution. 15

DeepRL Deep Reinforcement Learning. v, 5

DQN Deep Q-Network. 5, 86

EP-GAN ElectroPhysiomeGAN. iv, 63

FDB Feedback. 39

GA Genetic Algorithm. 69

GAN Generative Adversarial Network. 5, 64

GRU Gated Recurrent Units. 64

HH-model Hodgkin-Huxley Model. iv, 63

MDP Markov Decision Process. 86

O2 Oxygen. iv, 53

ODE Ordinary Differential Equations. 27, 67

RL Reinforcement Learning. 17

RMSE Root Mean-Squared Error. 71

RNN Recurrent Neural Network. 64

WGAN-GP Wasserstein GAN with gradient penalty. 64

Chapter 1

Introduction

The brain orchestrates behavior through the coordination of neural activity and the body. Neural activity is facilitated by neurons, a group of cells that use electrical and chemical processes to propagate neural signals. The nature of neural signals involves the encoding of external information (e.g., from the environment to neurons) or internal transmission (e.g., from neuron to neuron). The wiring circuitry between neurons, formed by synapses, facilitates signal propagation, leading to the formation of neuron populations that represent specialized functions, which range from encoding environmental stimulus into neural representations that generate rhythmic activity for muscles, such as locomotion (e.g., central pattern generators). Indeed, functional neural circuits have been universally found across different organisms, including small invertebrates such as locust, lamprey, Drosophila, and the nematode *C. elegans*, and larger vertebrates such as mice, primates, and humans [1; 2; 3; 4; 5; 6; 7; 8].

It is well known that the coordinated activity of neural circuits is an impetus for achieving animals' sensory and motor functions. Less understood, however, are their detailed mechanisms. In particular, at a microscopic level, the wiring principles of individual neurons (i.e., functional motifs) that facilitate the formation of a circuit with a specific function are not well described. On a macroscopic level, the principles behind the coordination and incorporation of neural circuits in the brain to orchestrate coherent motor functions (i.e., sensorimotor integration) are not yet resolved. While it appears to be a complex system to study, the fact that the system is composed of fundamental units (neurons) with known biophysical processes suggests that the principles governing its neural functions can be studied through computational approaches that emulate such architecture and processes.

A neuromechanical model, which aims to emulate the nervous system and the body through the simulation of neurons, is an example of such an approach. Using a neural wiring map as its foundation, such a model incorporates intra-cellular neuronal activities of individual neurons and inter-cellular neural interactions facilitated by neural connections. The model could incorporate additional layers of biomechanics to translate neural activities into movements. The ability to simultaneously simulate individual neurons and behavioral outcomes makes a neuromechanical model an attractive method for discovering neuron and neural circuit candidates that modulate behavior. Furthermore, by analyzing the simulated neural activities, neuromechanical modeling can assist in inferring the mechanisms of neural circuits and the computational principles through which the circuits, the nervous system, and the body interact [9].

The nematode organism *C. elegans* is a suitable candidate for neuromechanical modeling of the whole animal due to its well defined behavioral and anatomical aspects. *C. elegans* inherent locomotion patterns (forward/backward movements) and its response to environmental stimulus (e.g., turns) are well observed and characterized [10]. *C. elegans* neuronal wiring diagram, which maps the electrical and chemical neural connections between somatic neurons within its nervous system, is resolved and is constantly being updated across the organism's sexes and developmental stages [11; 12; 13; 14; 15; 16; 17; 18; 19; 20; 21]. *C. elegans* biomechanics, detailing the anatomical structure of muscles and the body, is well characterized [22; 23]. In addition, recent electrophysiological recordings of whole-cell responses provide biophysically relevant details of individual neuro-dynamical properties across different neuron types present in *C. elegans* [24; 25; 26; 27; 28]. These advances warrant the development of a fundamental neuromechanical model of *C. elegans* for investigating functional circuits and sensorimotor integration.

A fundamental neuromechanical model of *C. elegans* must integrate a series of models that encompass processes in the nervous system, muscles, and biomechanics. The integration of these models is necessary to incorporate numerous routes through which neural signals can flow during locomotion, such as synapses, gap junctions, neuromodulators, extra-synaptic signaling, muscles, body, and proprioception [29; 30; 31; 32; 33; 34; 35; 36; 37; 38; 39; 40; 41; 42; 43; 44; 45]. Identifying and integrating such models with varying scopes and modalities warrants the development of an effective modeling framework that allows for the simultaneous simulation and testing of individual models. In my work, I proposed modWorm: a modular integration

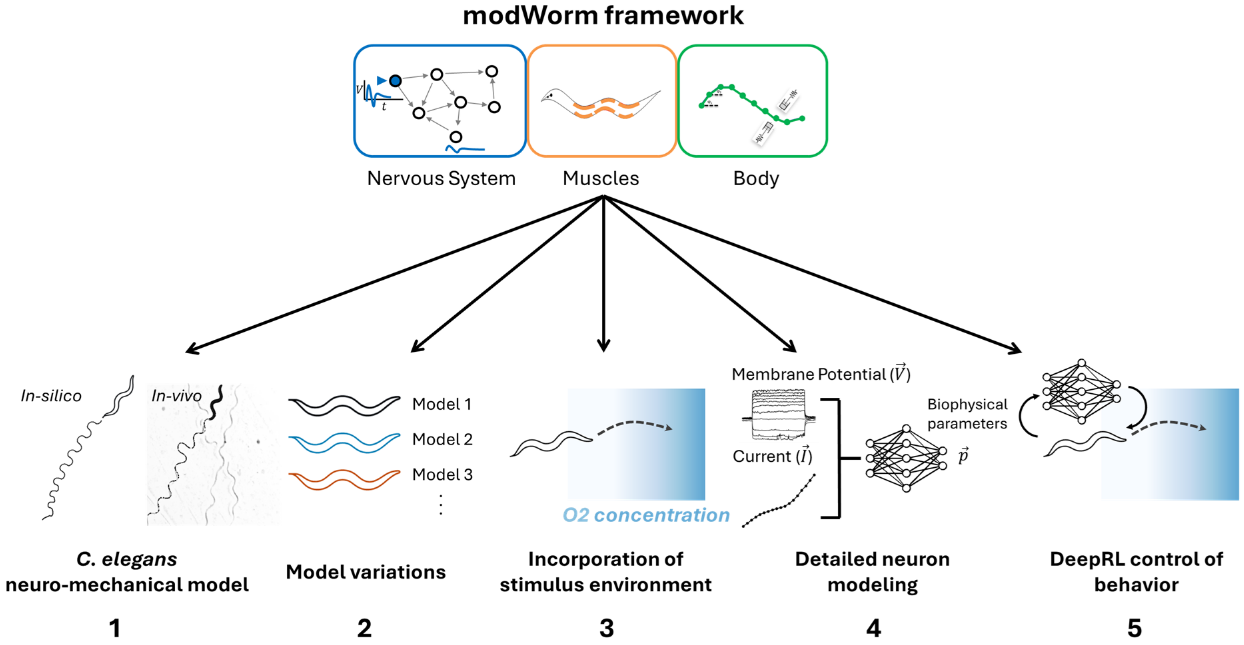


Figure 1.1 PhD research outline. In my PhD research, I have developed a fundamental framework for *C. elegans* simulation, modWorm. Using the framework I introduced (1) *C. elegans* neuromechanical model, (2) model variations to inform experimental analysis and model improvements, (3) incorporation of virtual environmental stimulus to emulate chemotaxis, (4) introduction of deep generative method to improve neuron modeling, and (5) implementation of DeepRL methods to infer neural mechanisms of behavior.

modeling and simulation framework for the development of neuromechanical models. The framework allows for the construction of a model as a sequence of configurable, exchangeable biophysical modules, each responsible for modeling distinct dynamic processes.

Using modWorm, we proposed an initial candidate for a fundamental neuromechanical model of *C. elegans*. The model adopts a modular architecture to encompass experimentally derived anatomical and electrophysiological data, as well as known biophysical processes of *C. elegans*. At the same time, the modules are designed to be generic, scalable, and unbiased by specific behaviors or experiments. Together, the proposed model is built upon the complete connectome and integrates a series of 7 modules: i) intra-cellular dynamics, ii) electrical and iii) chemical extra-cellular neural dynamics, iv) translation of neural activity to muscle calcium dynamics, v) muscle calcium dynamics to muscle forces, vi) muscle forces to body postures, and vii) proprioceptive feedback. We have integrated information from the literature known to us into a "baseline" model variant. We then tested and validated the baseline model through the generation of simulated neural/body dynamics driven by external forces and neural stimuli associated with locomotion, and analyzed them with respect to known experimental findings. To further validate the model, we extended the simulation studies to incorporate timed stimulus, to investigate behaviors induced during locomotion, such as avoidance and turn.

The modular structure of the model allows for multiple variations and extensions to be applied to its underlying modules. For example, we showed that the use of computational neuron ablations recapitulates locomotion behaviors associated with *in-vivo* ablation experiments of touch responses and performed further ablations to elucidate novel details for these experiments. We extended existing modules by incorporating empirically found ion channels of individual neurons and synapses, as well as more up-to-date full connectome mappings, to test the improved fit of locomotion to empirical findings. In addition, we considered variations driven by model optimization and theoretical assumptions, such as neural parameter fitting (e.g., synapse strengths) with respect to a given locomotion task and an alternative model for extra-cellular neural dynamics, to elucidate their effects on simulated behavior. Testing such variations highlights the possible mechanisms of the investigated locomotion, the directions in which the model can be improved, and how it can be incorporated into *in-vivo* investigations.

A natural extension of modWorm would be to simulate model responses to environmental stimulus (e.g., chemotaxis). Therefore, in the following work, I developed the mod-SenseWorm framework, an extension of modWorm with 2 new configurable modules: i) Virtual stimulus environment emulating the worm's surrounding environment, and ii) Spatial-temporal neural stimulation, which translates environmental stimulus into dynamic neural stimulation according to neurons' chemical receptor types and spatial locations. As an example, we showed how mod-SenseWorm can be used to study the oxygen aerotaxis behavior of *C. elegans* through the recapitulation of the avoidance responses observed *in-vivo* with respect to different O2 levels. The implementation of different sensory receptor types (e.g., tonic, phasic) suggests that the framework can assist in elucidating the relationships between stimulus encoding strategies and simulated behaviors. Furthermore, through selective stimulation of O2 sensory neurons, we observed that the framework can be used to identify potential functional pathways mediating the observed collective chemotactic behaviors, such as avoidance and attraction. As such, the framework mediates between individual neuromechanical mechanisms and their effects on collective spatial behavior.

Deep learning methods can assist in extending the scope of the neuromechanical model to improve its simulation accuracy. For example, deep learning based regression methods can predict more detailed biophysical parameters associated with experimental data. Therefore, in my work, I have developed the ElectroPhysiomeGAN (EP-GAN), a deep generative method that can generate biophysical parameters associated with neuron models. Built on top of the Generative Adversarial Network (GAN) architecture, EP-GAN takes recorded membrane potential and steady-state current responses as inputs and outputs parameters associated with detailed neuron models (e.g., Hodgkin-Huxley type model). EP-GAN is trained with simulated data to learn the translation from electrophysiological responses to HH-model parameters across different neuron types in *C. elegans*. Using EP-GAN, we showed that the predicted parameters can accurately capture ground-truth responses for both simulated and experimentally recorded *C. elegans* neurons. Through comparison with existing estimation methods, such as Differential Evolution (DE) and Genetic Algorithm (GA), we showed that EP-GAN is advantageous in both the accuracy of generated parameters and inference speed.

Complementing deep learning approaches, control strategies can further assist in extending the model's modality by inferring potential additional neural mechanisms associated with empirical behavior. I explored this direction by utilizing Deep Q-Network (DQN), a deep reinforcement learning (DeepRL) algorithm, to infer control strategies for i) Continuous neuromodulation inputs and ii) Static neural wiring configurations (i.e., connectome), applied on top of the neuromechanical model. We applied the methods to a sensory olfactory circuit in *C. elegans* nervous system associated with attraction chemotaxis and showed that the learned strategies can infer neuromodulatory currents between neurons (e.g., neuropeptidic currents) and connectome perturbations, which are associated with the desired behavioral outputs (e.g., avoidance and attraction). Furthermore, by expanding the action space of the control agent, we showed that the proposed methods can be applied to wider tasks, such as ground truth behavior recovery via repairing damaged neural circuits and full connectome inference of neural circuits from scratch. These results show the potential of the DeepRL methods as a viable control framework to infer the mechanisms of neural circuits.

Chapter 2

Background and Related work

This chapter includes the background knowledge and the review of related work pertinent to my PhD research. In particular, sections 2.1 to 2.5 discuss methods and related works for the neuromechanical modeling of the nematode *C. elegans*. Sections 2.6 and 2.7 focus on the applications of deep learning and control methods to assist in neuromechanical modeling and the inference of neural mechanisms.

2.1 Overview of *C. elegans* Body, Behavior, and Nervous system

Caenorhabditis elegans (*C. elegans*) is a nematode that primarily resides in a temperate soil environment. The body of *C. elegans* is approximately 1 mm in length, with a transparent outer exoskeleton and a set of internal organs that are found in larger animals, such as the pharynx, intestines, and reproductive organs [46]. *C. elegans* musculature consists of 95 muscle cells distributed from the anterior (head) to the posterior (tail) region, arranged in 4 quadrants. These quadrants each occupy the dorsal, ventral, left, and right portions of the body, allowing motion along 4 axes. While moving, *C. elegans* can only bend its body in dorsal/ventral directions, with the exception of the head, which can move in left/right directions to perform dispersal behavior (e.g., nictation) [47]. The typical sinusoidal movement of *C. elegans* is facilitated by a wave of dorsal/ventral muscle contractions, where wave propagation from anterior to posterior leads to forward movement, and from posterior to anterior leads to backward movement.

In addition to natural sinusoidal locomotion, *C. elegans* can perform coordinated responses to various mechanical, thermal, and chemical stimuli [48]. These responses are often accompanied by pirouette locomotion patterns, including gradual turning (weathervane response), sharp avoidance maneuvers (omega turn), and quick reversals, allowing the organism to efficiently navigate toward beneficial attractants (food sources) and avoid harmful repellents (toxins) in the environment. Coordinated modulations of locomotion are also critical to facilitating group survival strategies (social feeding), where worms aggregate around food sources to achieve more efficient consumption of food [48].

The nervous system of *C. elegans*, which controls its behavior, consists of 302 neurons, of which 279 are found within the somatic nervous system. The neurons in *C. elegans* are categorized into 3 classes - sensory, inter, and motor neurons, according to their anatomical locations and approximate functions. Neurons in *C. elegans* lack the Na^+ sodium ion channels, which are associated with spike generation. Thus, a majority of neurons exhibit graded intra-cellular membrane potentials. A few neurons in *C. elegans* are an exception and may exhibit calcium-regulated spiking activities [49; 24; 25]. *C. elegans* hosts a number of potassium and calcium ion channels to mediate graded intra-cellular neural activity [27; 28]. Experiments have so far identified approximately 25 potassium and calcium ion channels that are found in *C. elegans* and other organisms expressing homologous channels. Among these, 15 channels are quantitatively described and are further discussed in section 2.4.

On top of the intra-cellular activities of individual neurons, the neural connection wiring map (i.e. connectome) of *C. elegans*, which mediates the extra-cellular synaptic interactions between neurons, has been intensively studied and mapped. The connectome has been divided into two sub-connectomes that represent the wiring maps of i) gap junctions and ii) chemical synapses [14; 17]. Gap junctions mediate electrical interactions between neurons through direct physical contact mediated by connexon protein [50]. Chemical synapses, on the other hand, mediate chemical interactions between neurons through the release of neurotransmitters in the synaptic cleft. The chemical synapses are further divided into excitatory and inhibitory types according to the neurotransmitters expressed by the neurons. For example, neurons expressing Glutamate or Choline are typically considered excitatory, whereas neurons expressing GABA are typically considered inhibitory [51].

The availability of the connectome data allows for the quantification of the total number of neural connection parameters. This number is approximately 1100 parameters for gap junctions and 4400 parameters for synaptic connections (2200 connection weights + 2200 connection polarities) [17]. Combined together,

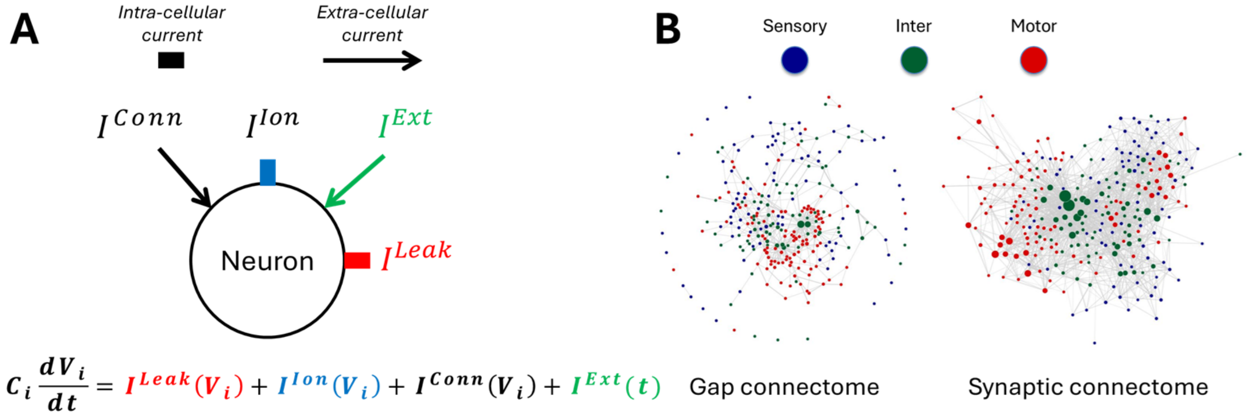


Figure 2.1 Visualization of a neuron and connectomes. **A**: Modeling of a neuron using single-compartment membrane model composed of intra-cellular and extra-cellular current sources. **B**: Graph visualization of gap and synaptic connectomes of *C. elegans* somatic nervous system (279 neurons).

the connectome of *C. elegans* somatic nervous system consists of approximately 5500 nonzero elements out of 233,523 possible parameters $(3 * (279 * 279))$.

2.2 Modeling Neural Circuits with Biophysical Neuron Model

Computational models of neural circuits aim to achieve a simulation of neural activities through the integration of connectomes and neuronal biophysical processes. The connectome defines the connectivity mappings of neurons across different types of synapses (e.g., electrical, chemical) within the neural circuit. Mathematically, a connectome can be described as a set of matrices (tensor), where each matrix describes a particular connection type (Gap, Glutamate, Choline, etc), and its elements represent the weight of the connection. For an arbitrary connectome tensor W, the connection weight w between neuron i and neuron j of connection type k can be defined as

$$w_{ij}^{k} = W_{ijk},\ i, j \in [1, N],\ k = \text{Type of connection}, \tag{2.1}$$

where N is the total number of neurons. The formulation can be applied to the established connectome data such as the somatic nervous system of *C. elegans*. For example, *C. elegans* connectome consists of $N = 279$ neurons and three types of channels $(k = 3)$, where $k = 1$: Gap, $k = 2$: Excitatory synaptic channels of

Glutamate and Choline synapses, and $k = 3$: Inhibitory synaptic channels of GABA synapses [14; 46].

Acting on top of the sub-connectomes are the neural activities (i.e., membrane potential dynamics) determined by intra- and inter-cellular processes. Assuming that the neurons are iso-potential (i.e., uniform membrane potential), these processes can be further divided into cell leakage and ionic currents (e.g., voltage-gated channels) for intra-cellular processes, synaptic (electrical, chemical) and extra-synaptic (e.g., neuropeptides, monoamine) currents for extra-cellular processes. The combined effects of these processes can be modeled as a system of Hodgkin-Huxley type (HH-model) single compartment membrane equations, which integrate the neuron's intra-cellular current terms $I^{Leak}(\vec{V})$ and $I^{Ion}(\vec{V})$, extra-cellular current term $I^{Conn}(\vec{V})$, and external current $I^{ext}(t)$ as follows

$$\begin{aligned}
&C\frac{dV_i}{dt} = I^{Leak}(V_i) + I^{Ion}(V_i) + I^{Conn}(\vec{V}) + I^{ext}(t), \qquad (2.2)\\
&\text{intra-cellular: } I^{Leak}(V_i) = -g_l(V_i - E_i^L),\ \ I^{Ion}(V_i) = -\sum_m g_m(V_i)(V_i - E_i^I),\\
&\text{extra-cellular: } I^{Conn}(\vec{V}) = -\sum_j g_{ij}^{Gap} w_{ij}^{Gap}(V_i - V_j) - \sum_j g_{ij}^{Syn}(\vec{V}) w_{ij}^{Syn}(V_i - E_j^S),\\
&\text{Ion activation: } g_m(V_i) = \overline{g}_m P_m^I(V_i),\ \ \frac{dP_m^I}{dt} \approx \alpha_m^I(V_i)(1 - P_m^I) - \beta_m^I(V_i)P_m^I,\\
&\text{Synaptic activation: } g_{ij}^{Syn}(\vec{V}) = \overline{g}_{ij}^{Syn} P_j^S(\vec{V}),\ \ \frac{dP^S}{dt} \approx \alpha^S(\vec{V})(1 - P^S) - \beta^S(\vec{V})P^S.
\end{aligned}$$

Here C is the membrane capacitance per unit area. g_l and E_i^L are the leakage conductance per unit area and leak reversal potential, respectively. g_m and E_m^I are the membrane potential dependent conductance per unit area and ion reversal potential for m-th ion channel, respectively. g_{ij}^{Gap} is the total conductivity of the gap junctions between neurons i and j scaled by the gap connection weight w_{ij}^{Gap}. g_{ij}^{Syn} is the membrane potential dependent conductance of the synapses from neuron j to i scaled by the synaptic connection weight w_{ij}^{Syn} [52]. Dynamic conductance $g_m(V_i)$ and $g_{ij}^{Syn}(\vec{V})$ are mediated by ion activation $P_m^I(V_i)$ and synaptic activation $P_j^s(\vec{V})$ functions multiplied by maximum m-th ion conductance $\overline{g}_m$ and maximum total conductivity of synapses $\overline{g}_{ij}^{Syn}$ between neuron i and j respectively. The dynamics of $P_m^I(V_i)$ and $P^s(\vec{V})$ are each governed by the differential equation involving channel opening rate functions $\alpha_m^I(V_i)$ and $\alpha^S(\vec{V})$ and channel closing rate functions $\beta_m^I(V_i)$ and $\beta^S(\vec{V})$ respectively, where they model the channel activation mediated by the membrane potentials. Combined together, Eqs (2.2) approximate the interactions between intra- and extra-cellular current sources and describe how membrane potential dynamics are propagated in the nervous system.

In Chapter 4 *Neuromechanical model of C. elegans*, the above equations are adapted for the simulation of *C. elegans* somatic nervous system.

2.3 Existing Works of Nervous System and Biomechanics Modeling

Models of neural dynamics and biomechanics adapting single-compartment membrane type models have been introduced for several model organisms, including *C. elegans*, the adult and larval Drosophila [53; 54; 55; 56; 57; 58], hydra [59], lamprey [60; 61], leech [62; 63] and rodents [64], whose anatomical, electrophysiological, biomechanical, and behavioral data make them suitable candidates for neuromechanical modeling [13; 14; 17; 19; 65; 49; 23].

For *C. elegans*, the availability of the connectome warrants searching for incorporated circuits using computational and experimental techniques. Indeed, groups of sensory-, inter- and motor-neurons have been associated with various types of locomotion including natural crawling motions [66; 67; 68], chemosensation [69; 70; 71; 72; 73; 74; 75], thermosesation [76; 77; 78], and mechanosensation [79; 80; 81; 82; 37]. However, it is still not fully resolved how these sensorimotor mechanisms are incorporated at the network level in *C. elegans* and what types of neural interactions lead to locomotion behaviors. Here, I survey existing works of neuromechanical modeling of *C. elegans*, which aim to address such challenges categorized into several broad approaches to highlight their contributions as well as limitations.

Several *C. elegans* models integrating *partial connectome data* (e.g., sub-neural circuits) with biomechanics models have been introduced. Such models include methods which describe ventral motor neurons as symmetric binary units controlling the body of *C. elegans* segmented into discrete rods and stretch receptors [83]. The model demonstrated gaits generating forward locomotion but also locomotion instabilities when neuron dynamic properties and arrangements are slightly changed, e.g., when binary motor units are replaced by time dependent neurons. Several studies also showed that pattern generators or connectome based locomotory sub-circuits (e.g., head motor neurons and ventral nerve cord) combined with body models can produce forward locomotion and basic navigation behaviors such as Klinotaxis [69; 84; 85; 86; 87; 88; 89; 90; 91]. While these analyses showed that oscillators and sub-neural circuits can produce *C. elegans* body postures typical for forward locomotion and subsequent sensory navigation, their relation to the full nervous system and how these patterns are generated remains unresolved. Furthermore, a unifying relation to other locomotion behaviors, such as backward locomotion, turns, and pirouettes, remains unclear.

Other lines of *connectome based* models introduced a dynamical model for the full somatic nervous system of *C. elegans* [92; 93; 94; 51]. These studies showed that the full nervous system can generate dynamic rhythms even when a few mechano-sensory neurons received a constant stimulus. These rhythms, however, could not be directly associated with behaviors since additional processes of biomechanics and proprioception were not included. Inspired by the human brain project, the OpenWorm collaborative project was established in 2011 as a crowd-sourcing platform aiming to develop generic bottom-up simulations of neuronal models, body and fluid simulations to lead to a full-scale *C. elegans* model [16; 92; 95]. While there has been progress in the development of generic tools for modeling *C. elegans* and other organisms, such as Geppetto, c302 (multiscale modeling) and Sibernetic (hydrodynamic simulation) [95; 96], the integration of these tools into a unified framework has not yet been achieved. Moreover, the incorporation of feedback between the nervous system and body, such as proprioception, remains unresolved.

Recent works include methods inspired by Zador et al [97] adopting machine learning techniques to enhance the model and improve its simulation accuracy. These methods target parameters such as neuron polarities, connection weights/strengths, and muscle-body parameters to be optimized using machine learning algorithms with respect to the established empirical data. Such methods have been applied to the nervous systems of both *C. elegans* and Drosophila to infer particular neural functions [98; 99; 100; 101]. Similar approaches have also been developed for the neuromechanical modeling of *C. elegans* to reproduce stereotypical behaviors such as forward locomotion [102; 103]. These models, however, generally include a number of compromises on model details, such as using the partial connectome, less accurate first order approximation for neural dynamics, and the absence of proprioception or body-environment interaction, to accommodate large scale optimization algorithms (e.g., Back-propagation through time) [104]. Furthermore, their ability to simulate locomotion behaviors in addition to forward locomotion, such as backward, turns and pirouettes, etc, are not guaranteed.

In Chapter 4, I introduce an extension to existing approaches in the form of a generic neuromechanical model of *C. elegans* that aims to address these limitations by achieving simultaneous simulations and observations of the system in its entirety. The model adopts a modular architecture and integrates established biophysical processes and their approximate parameter values in *C. elegans*, including the complete somatic nervous system, muscles, body, and their interactions with the environment. I show that such a model incorporates and qualitatively reproduces known forward and backward locomotion and transitional behaviors, such as turns and avoidance.

2.4 Modeling *C. elegans* Neurons with Different Levels of Biophysical Detail

Incorporation of neuron intra-cellular dynamics that are descriptive of neurons' dynamical properties (e.g., electrophysiological dynamics) can improve the simulation accuracy of nervous system models. Here, I expand on section 2.2 to review the most up-to-date detailed Hodgkin-Huxley model of *C. elegans* neurons introduced in [28], which includes modeling of empirically found ion channels in *C. elegans* and other organisms expressing homologous channels.

The general equation describing the membrane potential dynamics of a single-compartment neuron is

$$C_m \frac{dV}{dt} = -I_{ion} + I_{Ext} \tag{2.3}$$

Where I_{ion} and I_{Ext} represent the ionic and external currents applied to a neuron respectively. The HH-model of *C. elegans* neurons has a total of 16 ionic current terms comprised of 11 voltage/calcium-gated potassium currents (I_{K^+}), 3 voltage-gated calcium currents ($I_{Ca^{2+}}$), leakage currents (I_{Leak}), and sodium leakage currents (I_{Na^+}) (Table 2.1). Consequently, the ionic current term I_{ion} of the considered HH-model can be written as

$$I_{ion} = I_{K^+} + I_{Ca^{2+}} + I_{Leak} + I_{Na^+} \tag{2.4}$$

Each ith ionic current can be modeled according to the Hodgkin-Huxley type formulation as follows:

$$I^i_{ion} = g_i \cdot m^p_i \cdot h^p_i \cdot (V - E_{rev}) \tag{2.5}$$

Where g_i is the maximum conductance and E_{rev} is the reversal potential of the channel. m^p_i and h^p_i each

Ion channel	**Ion Selectivity**	**# of parameters (Without activation parameters)**	**# of parameters (With activation parameters)**
SHL1	K^+	4	22
SHK1	K^+	3	14
EGL2	K^+	2	8
IRK1/3	K^+	2	10
UNC103	K^+	3	15
KQT1	K^+	3	17
EXP2	K^+	3	15
SLO1	K^+	2	2
SLO1-CaV	K^+	3	3
SLO2	K^+	2	2
SLO2-CaV	K^+	3	3
EGL19	Ca^{2+}	3	23
UNC2	Ca^{2+}	3	18
CCA1	Ca^{2+}	3	15
Leak	Leak	1	1
NCA	Na^+	1	1

Table 2.1 List of ion channels included in the detailed HH-model of *C. elegans* neurons. Ion selectivity for each channel and the number of parameters (not including reversal potentials) for models with and without parameters associated with activation variables. The model is adopted from [28].

represent the activation and inactivation variables of the channel with the following equations:

$$\frac{dm_i}{dt} = \frac{m_{i,\infty} - m_i}{\tau_{i,m}} \tag{2.6}$$

$$\frac{dh_i}{dt} = \frac{h_{i,\infty} - h_i}{\tau_{i,h}} \tag{2.7}$$

Where $m_{i,\infty}$ and $h_{i,\infty}$ each represent the steady-state activation and inactivation values, and $\tau_{i,m}$ and $\tau_{i,h}$ represent the activation and inactivation time constants. Note that each ionic channel may have a different mathematical description and parameters for variables $m_{i,\infty}$, $h_{i,\infty}$, $\tau_{i,m}$, and $\tau_{i,h}$ according to their dynamics and dependencies. While the majority of these variables are dependent on membrane potential, some channels are dependent on intracellular calcium (e.g., SLO1/2) or independent (e.g., $\tau_{i,h}$ of SHK1).

Leakage and sodium leakage currents are modeled with the following equations:

$$I_{Leak} = g_{Leak}(V - E_{Leak}) \tag{2.8}$$

$$I_{Na^{+}} = g_{NCA}(V - E_{NCA}) \tag{2.9}$$

where g_{Leak} and g_{NCA} each represent the conductance values for the leakage and sodium leakage currents. Notably, neuron dynamics described by Eqs (2.3 - 2.9) can capture dynamical properties observed in experimental neurons (e.g., through current or voltage-clamp experiments) more accurately than the simplified linear neurons employed by previous modeling works [93; 51; 105].

For the exact mathematical equations describing each individual channel, please refer to the supplementary materials of [28].

2.5 Existing Works of Inferring HH-model from Recorded Neuron Responses

Modeling neurons using the Hodgkin-Huxley type equations (HH-model) such as the one introduced in section 2.4 (adaptation of [28]) requires fitting a large number of parameters associated with ion channels found in the system (Table 2.1). For a typical single neuron, these parameters could be tuned via local optimizations of individual ion channel parameters estimated separately to fit their respective *in-vivo* channel recordings, such as activation/inactivation curves [106; 107; 108; 27; 24; 25]. Such a method requires multiple experiments to collect individual channel data, and when such experiments are infeasible, the parameters are often estimated through hand-tuning.

Alternatively, less data intensive direction for estimating such parameters has been the simultaneous estimation of all parameters of individual neurons from only the electrophysiological responses of cells, such as membrane potential responses and steady-state current profiles. A differential Evolution (DE) method was introduced to simultaneously estimate the parameters of a 3-channel or hybrid HH-model given a whole-cell membrane potential responses recording and steady-state current profiles [109; 110; 111]. From a statistical standpoint, Markov-Chain-Monte-Carlo method was introduced to obtain the posterior distribution of channel parameters for HH-models featuring 3 and 8 ion channels, given the simulated membrane potential responses data [112].

From an analytic standpoint, an iterative gradient descent based method was introduced to directly manipulate the HH-model to infer 3 conductance parameters and 3 exponents of activation functions given the measurements of membrane potential responses [113]. Recent advances in machine learning gave rise to deep learning based methods, which infer steady-state activation functions and posterior distributions of 3-channel HH-model parameters inferred by an artificial neural network model given the membrane potential responses data [114; 115].

These methods suggest that simultaneous parameter estimation from macroscopic cell data is indeed possible through a variety of techniques. For most of the above methods, the algorithms require an independent (from scratch) optimization process for fitting each individual neuron, making them difficult to scale up for tasks involving a large number of neurons. In Chapter 7, I introduce EP-GAN, an estimation method that adapts a deep generative model to address these aspects for the class of non-spiking neurons found in *C. elegans*.

2.6 Applications of Deep Learning for Neuromechanical Modeling

Recent advances in machine learning methods further assist data-driven approaches for developing neuromechanical models and vice versa [97]. In particular, artificial neural network models (ANNs) trained with deep learning algorithms (back-propagation and gradient descent) are commonly regarded as generic computing frameworks that can act as universal function approximators in both theory and empirical trials [116].

ANN's ability to approximate an arbitrary function makes it a powerful tool to assist in modeling the neuromechanics for the inference of unknown model parameters associated with empirical data. Indeed, methods adopting deep learning techniques have recently been introduced to enhance model simulation accuracy. These methods target parameters such as the directionality of neurons (excitatory/inhibitory), connection weights/strengths, and muscle-body parameters to be optimized using deep learning algorithms with respect to the established empirical data. As shown in section 2.3, such methods have been applied to the nervous systems of both *C. elegans* and *Drosophila* to infer particular neural functions [98; 99; 100; 101] and to assist in neuromechanical modeling of *C. elegans* to reproduce stereotypical behaviors such as forward locomotion [102; 103].

While these works show promising results aligning the model with empirical data, studies such as Kim et al [105] showed potential inaccuracies that could occur with the fitted parameters when simulating behaviors other than those used for optimization. Furthermore, as mentioned in section 2.3, these models generally include a number of compromises on model details to accommodate gradient-based optimization algorithms.

These limitations, alongside the need for additional generalization requirements during optimization, remain challenges to be resolved for future adaptations of deep learning for neuromechanical modeling.

In Chapter 4 (*Neuromechanical Model of C. elegans*) and chapter 5 (*Model Extensions to Improve Simulation Accuracy*), I propose to apply model optimizations as modular extensions on top of the initial generic neuromechanical model. Such an approach of implementing an initial base model using the modular framework, followed by variations and experimental validations, can assist in making informed decisions in which the model incorporates and reflects future details and *in-vivo* findings into an expanding model of the organism [9; 117; 118; 119].

2.7 Neural Control Strategies to Inform Mechanisms of Neural Circuit

Neural control strategies informed by simulated or experimental neural data can be used to infer potential neural mechanisms associated with particular behaviors. In terms of neural circuits, control strategies can be delivered through the control of i) External neural stimulation (i.e., neuro-modulation control), or ii) Internal circuitry of the nervous system (i.e. connectome control) to achieve a target behavior. Here, I review existing methods for achieving neuromodulation and connectome control.

Neuromodulation control of neural circuits takes neural activity as input and outputs external stimulation to modulate neural activity. Model based approaches often utilize a simplified model of the target neural circuit to establish control strategies that leverage the model's analytic features. For example, non-linear control methods for the coordination of local circuit spike trains used a simplified model of neurons to infer external stimuli to adjust oscillations related to animal locomotion patterns [120; 121]. Such approaches were also shown to be effective for the inference of stimuli to trigger neurons' natural rhythms [122] and in adding feedback to regulate oscillations in large networks [123; 124]. For control problems where the construction of a model is not feasible, model free neural control methods have been proposed. In particular, deep reinforcement learning methods, such as Deep Deterministic Policy Gradient (DDPG) were developed to control the stimulation of multiple neurons in oscillatory spike models and showed equivalent or preferable performance to the model-based methods [125]. Another example of applying RL in neural control is deep brain stimulation to suppress neural activity associated with epileptic seizures [126; 127; 128]. In these applications, ensemble learning and batch mode Q-learning were used to achieve adaptive closed loop neuro-modulation to eliminate seizures. The work showed that reinforcement learning methods can achieve successful neuromodulation and also demonstrated the sensitivity of these methods to various definitions of both state and reward functions.

Several methods that achieve the control of neural circuits via connectome inference (i.e., connectome control) have been proposed. For example, it was shown that the phase reduction method can be used to optimize the neural connection coupling matrix between a pair of limit-cycle oscillators to synchronize their activity [129]. In addition, the connectivity within the antenna lobe in insects was inferred by imposing model constraints [130]. Several experimental approaches were proposed to achieve connectome control through the utilization of neuroimaging, optogenetics, and bioengineering techniques. These techniques include genetically editing the connectome to add a gap junction synapse between neurons or synthesizing neural circuits with arbitrary structures [50; 131; 132; 133; 134; 135]. Notably, while connectome control corresponds to the discrete modulation type (e.g., adding/removing connections), it would require nonlinear dynamic control methods rather than static methods due to the nonlinear relationship between the connectome and neural processes [32; 136; 51; 137].

In Chapter 8, I introduce DeepRL methods to establish control strategies for both neuromodulation and connectome control of the neural circuit of *C. elegans*, which are then applied on top of the proposed generic neuromechanical model. In particular, I implement control agents employing a small discrete action space, associated with the incremental change in neuromodulatory input and connectome, to infer supplemental neural mechanisms (e.g., neuropeptidic currents) and connectome perturbations associated with empirically observed chemotaxis behavior.

Chapter 3

A Need for Modular Neuromechanical Modeling

In this chapter, I describe the need for a modular approach for neuromechanical modeling, followed by the introduction of modWorm, a modular framework for modeling and simulating neuromechanics. modWorm is used to construct the neuromechanical model of *C. elegans*, the central work presented in the thesis. The chapter is a partial adaptation of the paper [JK1].

3.1 Navigating the Scope and Modalities of Neuromechanical Model

Modeling the neuromechanics requires the integration of multiple biophysical models across different modalities, including the nervous system, the muscles, and the body. Each biophysical model can have multiple variations according to its scope. For example, neuron's intra-cellular dynamics can be modeled using i) linear dynamics (e.g., cell leakage channel), ii) simplified non-linear dynamics incorporating ion channel currents (e.g., polynomials), iii) detailed conductance-based models, such as Hodgkin-Huxley type equations (Figure. 3.1). Such variations can be extended to static parameters associated with biophysical processes. For example, connection weights of synapses may be represented as i) binary number (i.e., 0: no connection, 1: connection), ii) number of synapses between two neurons, iii) physical synapse properties (e.g., width of synapse cross-section).

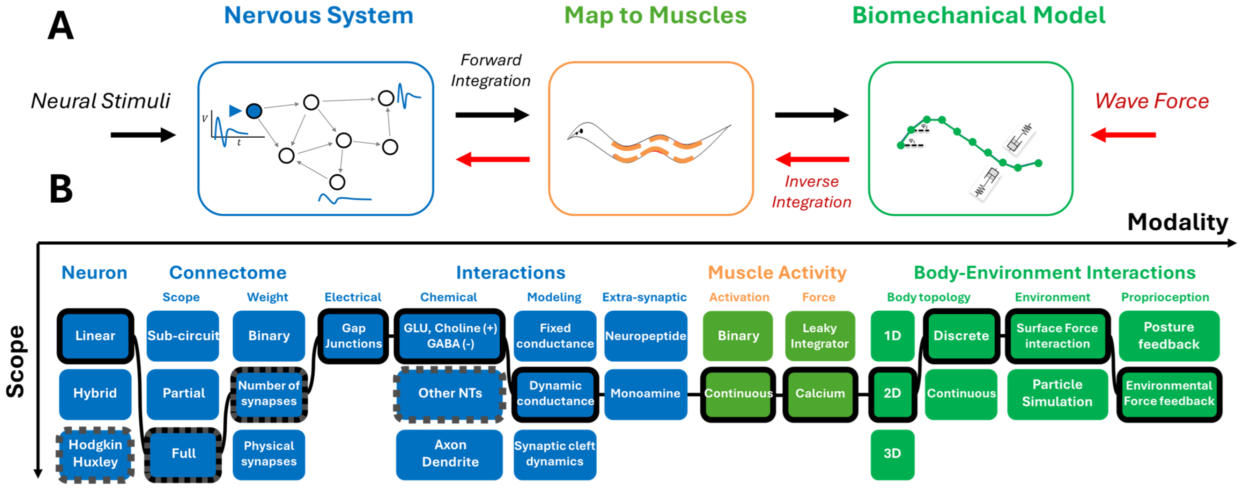


Figure 3.1 Constructing *C. elegans* neuromechanical model using modular integration approach. **A**: From left to right, Modeling the nervous systems as a dynamical system encompassing the full somatic connectome including graded ion-channel and connectivity neural dynamics, mapping neural dynamics to dynamic muscle impulses and forces, mapping muscle forces to a biomechanical model that incorporates body responses and interaction with the environment. Neural stimuli are integrated forward to resolve body movements (black arrow). External forces are propagated in an inverse direction to resolve corresponding neural dynamics (red arrow). **B**: From left to right: Schematics of each model aspect as flowchart for the nervous system (blue), map to muscles (orange) and biomechanical model (green) laid out in scope (y-axis) and modality (x-axis) space. Black highlighted boxes connected with solid lines are model aspects chosen by the proposed model and boxes with dark gray dotted edges represent the variations considered in the paper.

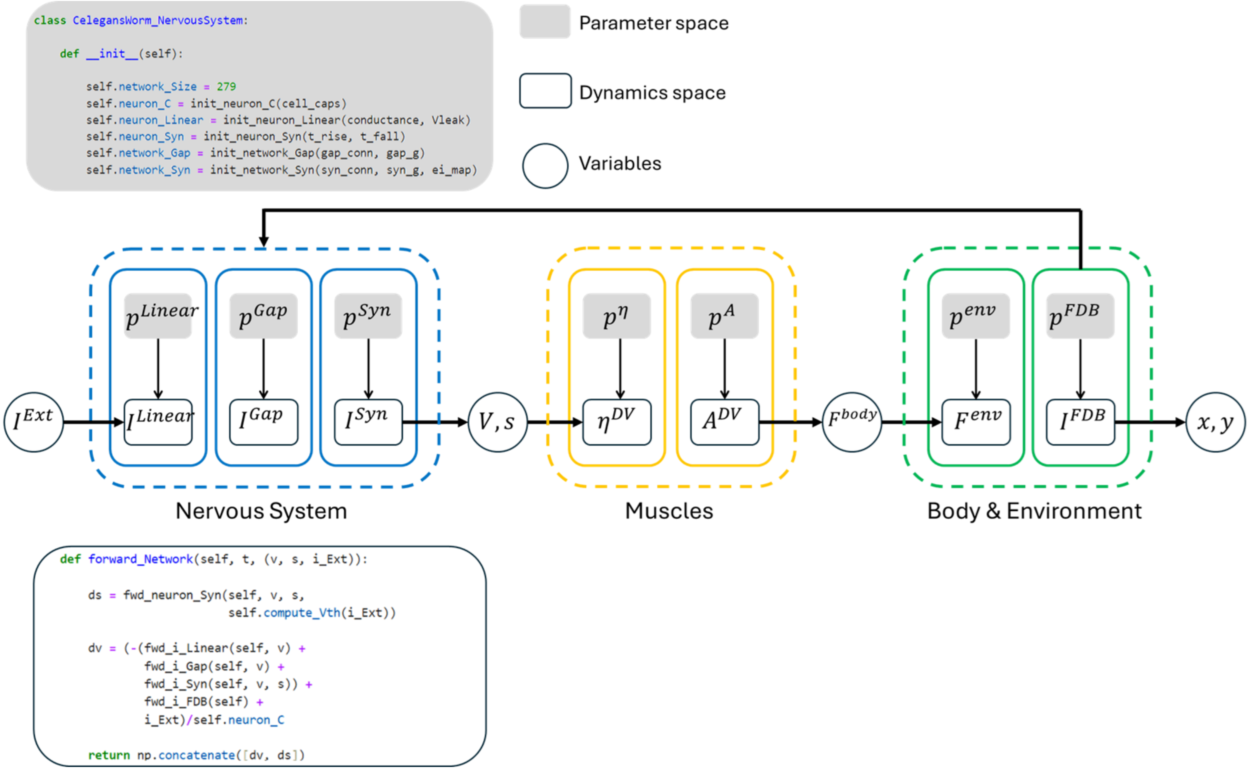
An intuitive way to visualize such variations across different biophysical models and parameters is to plot them in 2D space spanned by axes of scope and modality (Figure. 3.1). In this framework, the models are distributed according to their scope and modality attributes, providing a visual representation that maps their possible variations and interactions. A neuromechanical model can be described as a line connecting biophysical models (i.e., the path of integration), which constitute its architecture. Such a method of constructing a neuromechanical model may be referred to as *modular integration*, as it describes the model as an integrated series of smaller *modules*, each describing specific biophysical processes across different modalities. Such an approach can assist in identifying a neuromechanical model by providing simultaneous representations of both internal (interactions between modules) and external (distribution of scopes and modalities) aspects of the model.

Finding a combination of modules that constitutes the neuromechanical model entails surveying a large pool of possible model variations across different scopes and modalities. This is especially true for the modeling of *C. elegans* neuromechanics, where there are more than 50,000 model variations with 13 different model aspects encompassing the nervous system, muscles, and biomechanics, each with varying degrees of scope (Figure 3.1). Imposing constraints on module selection, such as computational feasibility and data availability, can make the survey size more manageable. Nevertheless, navigating possible module combination candidates that satisfy the biological and computational criteria necessitates a modular modeling and simulation framework that allows for the integration and testing of individual modules. In the section below, I describe modWorm, a software foundation that facilitates such a modular approach in neuromechanical modeling.

3.2 Software Framework for Modular Neuromechanical Modeling

modWorm is a modeling framework developed in Python and Julia programming languages. The framework is integrated with computing libraries such as NumPy, SciPy, DifferentialEquations.jl, and Jupyter Notebook for high-performance simulations and an intuitive interface [138; 139; 140; 141; 142; 143]. modWorm defines three model classes according to their scopes and scales: i) Module, ii) System, iii) Model. These three classes are inclusive of one another, such that Module $\subset$ System $\subset$ Model, where a Module describes an individual biophysical process within a system (e.g., individual ion channel current, synaptic current), System describes a specific system within the organism (e.g., nervous system, muscles, body), and Model describes the integrated system in its entirety (e.g., neuromechanics) (Fig. 3.2). Such a modular framework has also been adopted by existing modeling tools for the nervous system, such as NeuroML and machine learning libraries such as PyTorch [144; 145; 146].

All three model classes have two main components: parameters and dynamics. These components are fully customizable and describe the input-output relationship of the model. Acting as the smallest model class, Modules serve as building blocks for larger model classes. Each Module's parameters and dynamics are either pre-defined by the framework (e.g., *C. elegans* neuron leakage channel, gap/synaptic currents) or created by the user. All pre-defined Modules' parameter spaces are also fully configurable. A System is constructed by simply calling and combining the existing Modules to build its parameter and dynamics spaces (Fig. 3.2). Different Systems can be further combined into a Model where users can configure how inputs and outputs of Systems interact during simulations. The Model and its simulated dynamics can be



```
class CelegansWorm_NervousSystem:

    def __init__(self):

        self.network_Size = 279
        self.neuron_C = init_neuron_C(cell_caps)
        self.neuron_Linear = init_neuron_Linear(conductance, Vleak)
        self.neuron_Syn = init_neuron_Syn(t_rise, t_fall)
        self.network_Gap = init_network_Gap(gap_conn, gap_g)
        self.network_Syn = init_network_Syn(syn_conn, syn_g, ei_map)
```

```
def forward_Network(self, t, (v, s, i_Ext)):

    ds = fwd_neuron_Syn(self, v, s,
                        self.compute_Vth(i_Ext))

    dv = (-(fwd_i_Linear(self, v) +
            fwd_i_Gap(self, v) +
            fwd_i_Syn(self, v, s)) +
            fwd_i_FDB(self) +
            i_Ext)/self.neuron_C

    return np.concatenate([dv, ds])
```

Figure 3.2 ***C. elegans*** **neuromechanical model implemented in modWorm**. The model is comprised of individual modules (solid line boxes) each with parameter and dynamics equation space. Modules are combined to form systems (dotted line boxes) which are then integrated to form a model. For the nervous system, Python code snippets outlining the example definitions of parameter and dynamics functions are shown. From left to right, neural stimuli I^{Ext} is fed to the nervous system consisting of 3 modules (linear leak channel, gap, synaptic current) which outputs membrane potential V and synaptic activity variable s. V and s are then used as inputs to muscles system to calculate muscle calcium activities η^{DV} across dorsal-ventral muscles which are then translated to muscle activations A^{DV} and outputs muscle forces F^{body}. F^{body} is then fed to Body and Environment system to calculate corresponding environmental force F^{env} and proprioceptive feedback current I^{FDB} and output positional coordinates (x,y) for each body segment. I^{FDB} is used as a feedback input to the nervous system.

further analyzed and visualized (e.g., neural activity raster plots, animated body dynamics) using the plotting and animation tools provided by the framework. modWorm also allows saving the constructed model as a Python class object for future reusability.

On top of its modeling capability, modWorm incorporates an efficient simulation engine for the integrated model. The simulation engine is powered by Julia, a high performance programming language designed for numerical analysis and scientific computing [142]. In particular, modWorm uses the DifferentialEquations.jl library within the SciML ecosystem to achieve efficient simulation of Differential Equations [147]. The library implements a wide variety of high-performance ODE solvers for different types of systems (e.g., stiff and non-stiff). The library also supports advanced computing features, such as ensemble simulations that can utilize the parallel processing capabilities of both the CPU and GPU.

The modularity employed by modWorm offers additional benefits for model optimization tasks, such as parameter fitting. In particular, every parameter is coupled only with its associated biophysical module instead of the entire model. This allows users to extract or target a particular set of parameters for optimization without the need to deconstruct the entire model. Such a property makes modWorm a suitable platform to be used in conjunction with machine learning methods, such as deep learning algorithms, to further inform computational studies and improve model accuracy [97]. For example, modWorm could be used to generate a large amount of simulation data necessary to train machine learning methods that aim to interact with the biophysical neural or body model [117; 148]. Leveraging the modular structure, it would also be possible for modWorm to treat biophysical modules as learnable parameters (e.g., connectome weights, neuron directionalities) and optimize them with respect to particular neural and body tasks using gradient-based training methods (e.g., back-propagation through time) [104]. Indeed, such approaches have already been applied to *C. elegans* and other organisms to achieve model fitting or inference of neural functions with respect to experimental data [99; 102; 103; 100]. modWorm employs standard Python methods to construct its models (e.g., Python class) and data (e.g., NumPy arrays). The framework is thus expected to integrate well with existing machine learning libraries (e.g., PyTorch) which adapt similar model and data structures [146].

In Chapter 4, I use modWorm to construct the initial candidate for the foundational neuromechanical model of *C. elegans*, and leverage its modular structure to apply different variations to its modules (Chapter 5) to investigate their effects on simulated dynamics compared to experimental findings. The proposed model incorporates 7 modules, which include i) Linear intra-cellular dynamics, ii) Extra-cellular gap and iii) Synaptic dynamics, iv) Translation of neural activities to muscle dynamics, v) Muscle dynamics to muscle forces, vi) Muscle forces to body posture, and vii) Proprioceptive feedback from environmental forces (Figure 3.2).

Chapter 4

Neuromechanical Model of *C. elegans*

In this chapter, I adopt my publications [JK1, JK6] to develop the neuromechanical model for *C. elegans*. I first introduce the mathematical framework for modeling *C. elegans* nervous system, followed by the Neural Interactome, a framework for the visualization of simulated neural activity [JK6]. I incorporate biomechanics to translate simulated neural activity into muscle forces, muscle forces to body postures, and proprioceptive feedback in the form of inverse integration to emulate the effects of external forces on nervous system activity. We validate the neuromechanical model through the recapitulation of empirically observed natural locomotion, touch responses, associated neuronal ablation studies, and transitional behaviors (turn, avoidance).

4.1 *C. elegans* Nervous System Model

C. elegans neurons lack membrane potential-gated sodium (Na^+) channels to generate action potential spikes and thus typically exhibit graded membrane potential responses [49]. While many neurons of *C. elegans* exhibit other ion channels, such as potassium (K) and calcium (Ca^{2+}), only a few channels are needed to mediate synaptic vesicle release, implying that the dynamics are mainly driven by gap and synaptic interactions. Experiments also identify that many neurons in *C. elegans* are iso-potential with a relatively uniform distribution of membrane potential throughout the membrane. These characteristics allow neuron dynamics to be approximated as single-compartment membrane models as introduced in Chapter 2, section 2 as follows: [93; 51]:

$$
\begin{aligned}
&C\frac{dV_i}{dt} = I^{Leaky}(V_i) + I^{Conn}(\vec{V}) + I^{ext}(t) \qquad (4.1)\\
&\text{individual: } I^{Leaky}(V_i) = -g_l(V - E_i^L)\\
&\text{interaction: } I^{Conn} = -\sum_j g_{ij}^{Gap} w_{ij}^{Gap}(V_i - V_j) - \sum_j g_{ij}^{Syn}(\vec{V}) w_{ij}^{Syn^{\pm}}(V_i - E_j^{\pm})\\
&\text{Synaptic activation: } g_{ij}^{Syn}(\vec{V}) = \overline{g}_{ij}^{Syn} P_j^S(\vec{V}),\ \ \frac{dP^S}{dt} = \frac{a_r(1-P^s)}{1+\exp(-\beta(V_i - V_{th}))} - a_d P^s
\end{aligned}
$$

where C, g_l, E_i^L, $I^{ext}(t)$, g_{ij}^{Gap}, w_{ij}^{Gap}, $g_{ij}^{Syn}(\vec{V})$ and $\overline{g}_{ij}^{Syn}$ follow the identical definitions as defined for Eqs (2.2). $w_{ij}^{Syn^{\pm}}$ consists of $w_{ij}^{Syn^{+}}$ and $w_{ij}^{Syn^{-}}$ which are the connection weights of excitatory and inhibitory synaptic channels from neuron j to i respectively. Similarly, $E_j^{\pm}$ is divided into E_j^{+} and E_j^{-} which represent the reversal potentials for the excitatory or inhibitory pre-synaptic neurons, respectively. $g_{ij}^{Syn}(\vec{V})$ is mediated by the synaptic activity variable $P_j^S(\vec{V})$ where a_r and a_d correspond to the rise and decay time constants for synaptic activity, β is the width of the sigmoid function, and V_{th} which is the equilibrium membrane potential.

Using modWorm, Eqs (4.1) are implemented by combining 3 modules: (i) graded potentials, (ii) neural gap-junction interactions (iii) neural dynamic synaptic interactions including glutamatergic, cholinergic and GABAergic receptors. As an initial approximation, glutamatergic and cholinergic transmitter activated ion channels are assumed to be excitatory, and GABAergic receptors are assumed to be inhibitory. These settings are configurable for changes (as we show in the section *Model Variations to Improve Simulation Accuracy*). Notably, the synaptic dynamics are modeled as Hodgkin-Huxley model type ion channels, where the transmitter release is controlled using a dynamic gating variable that is simulated along with the membrane potential. The model incorporates local synaptic parameters determined by the connectome (i.e., the number of synapses) and global biophysical conductance coefficients per type of connection.

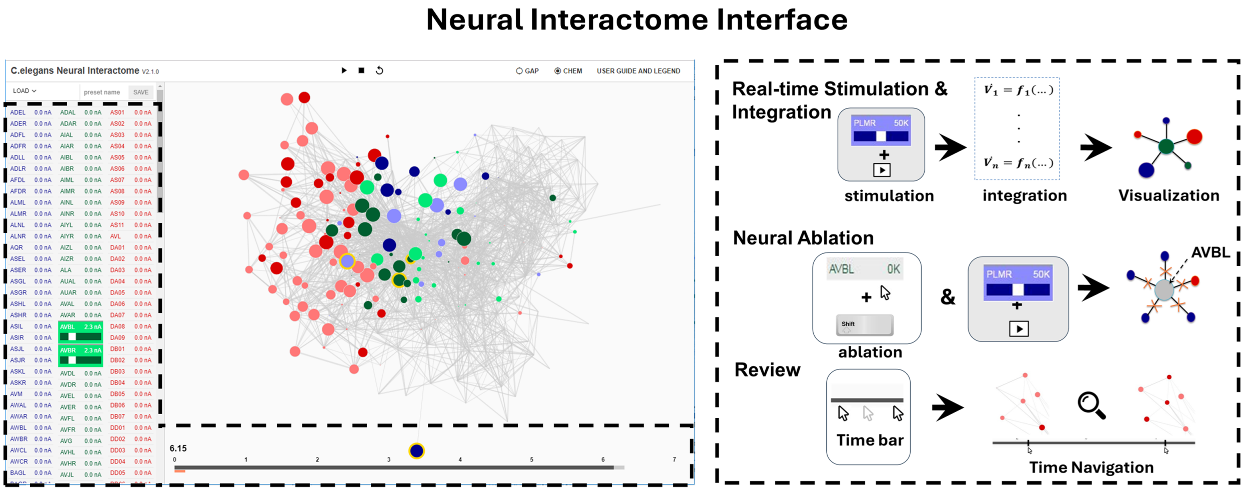
The neuronal physiological parameters in Eqs (4.1) are set to approximate values set by the literature and are applied to all neurons in the system. In particular, we assume that each individual gap junction and synapse has approximately the conductance of $100pS$ [14]. For each neuron, the membrane conductance g_l is taken as $10pS$ and membrane capacitance is set to $C = 1pF$ [14]. Leakage potentials are all taken as $E^L = -35mV$ [65]. Reversal potentials $E_j^{\pm}$ are set to 0 mV for excitatory synapses and $-48mV$ for inhibitory synapses [65]. For the synaptic variable, we choose $a_r = \frac{1}{1.5}$, $a_d = \frac{5}{1.5}$ and define the width of the sigmoid by $\beta = 0.125 mV^{-1}$. V_{th} is computed by imposing that the synaptic activation $\phi = 1/2$ at the equilibrium potential where $\phi = 1/(1 + exp(-\beta(V_i - V_{th})))$ which is then followed by solving for V by setting the left side of Eqs (4.1) to 0 [65].

When Eqs (4.1) are applied to the neurons of the somatic nervous system of *C. elegans*, they describe the membrane potential dynamics of 279 neurons given an arbitrary external current $I^{ext}(t)$. The system is an ODE system of 558th order (solving for both V and P^s) with more than 5500 parameters. The system also exhibits nonlinear stiff behavior due to the sigmoid non-linearity modeling the synaptic activation function. The integrated dynamics equations are solved using adaptive integration steps with an implicit backward differentiation formula (within SciPy or DifferentialEquations.jl numerical integration packages) for numerical stability. Such a numerical scheme allows for efficient simulation of the nervous system close to real-time, i.e. simulating the nervous system for a duration of T seconds will take about T seconds in the real world, or less to compute.

Effectively, the nervous system model allows for computational clamping of neurons by external current injection. It was previously observed that the injection of constant current into model's sensory neurons, e.g., the posterior PLM mechanosensory neurons, evokes oscillatory neural responses in a subset of motor neurons, producing low dimensional attractor-like dynamics and transient dynamics with longer timescales than the intrinsic neural dynamics [94; 136]. Since our initial goal is to implement a base model that is generic and not specific to behavior or experimental biases, we do not fit the neural parameters of individual neuron channels and synapses. These values, however, are easily configurable through modWorm. In the section *Model Variations to Improve Simulation Accuracy* and Fig. 5.1, we perform plausible modulations to such parameters to examine the effect of additional experimental details or hypotheses [17; 24; 149; 150; 25].

4.2 Visual Framework for Investigating Neural Dynamics

The dynamic nervous system model of *C. elegans* has the potential to reveal neural pathways and functionalities [33; 151; 136]. These advantages, however, come with the cost of surveying and analyzing large volumes of high-dimensional neural data, which are challenging for brute force methods. Such a challenge warrants a framework that allows for the efficient investigation of the model simulations.



ADEL	0.0 nA	ADAL	0.0 nA	AS01	0.0 nA
ADER	0.0 nA	ADAR	0.0 nA	AS02	0.0 nA
ADFL	0.0 nA	AIAL	0.0 nA	AS03	0.0 nA
ADFR	0.0 nA	AIAR	0.0 nA	AS04	0.0 nA
ADLL	0.0 nA	AIBL	0.0 nA	AS05	0.0 nA
ADLR	0.0 nA	AIBR	0.0 nA	AS06	0.0 nA
AFDL	0.0 nA	AIML	0.0 nA	AS07	0.0 nA
AFDR	0.0 nA	AIMR	0.0 nA	AS08	0.0 nA
ALML	0.0 nA	AINL	0.0 nA	AS09	0.0 nA
ALMR	0.0 nA	AINR	0.0 nA	AS10	0.0 nA
ALNL	0.0 nA	AIYL	0.0 nA	AS11	0.0 nA
ALNR	0.0 nA	AIYR	0.0 nA	AVL	0.0 nA
AQR	0.0 nA	AIZL	0.0 nA	DA01	0.0 nA
ASEL	0.0 nA	AIZR	0.0 nA	DA02	0.0 nA
ASER	0.0 nA	ALA	0.0 nA	DA03	0.0 nA
ASGL	0.0 nA	AUAL	0.0 nA	DA04	0.0 nA
ASGR	0.0 nA	AUAR	0.0 nA	DA05	0.0 nA
ASHL	0.0 nA	AVAL	0.0 nA	DA06	0.0 nA
ASHR	0.0 nA	AVAR	0.0 nA	DA07	0.0 nA
ASIL	0.0 nA	AVBL	2.3 nA	DA08	0.0 nA
ASIR	0.0 nA			DA09	0.0 nA
ASJL	0.0 nA	AVBR	2.3 nA	DB01	0.0 nA
ASJR	0.0 nA			DB02	0.0 nA
ASKL	0.0 nA	AVDL	0.0 nA	DB03	0.0 nA
ASKR	0.0 nA	AVDR	0.0 nA	DB04	0.0 nA
AVM	0.0 nA	AVEL	0.0 nA	DB05	0.0 nA
AWAL	0.0 nA	AVER	0.0 nA	DB06	0.0 nA
AWAR	0.0 nA	AVFL	0.0 nA	DB07	0.0 nA
AWBL	0.0 nA	AVFR	0.0 nA	DD01	0.0 nA
AWBR	0.0 nA	AVG	0.0 nA	DD02	0.0 nA
AWCL	0.0 nA	AVHL	0.0 nA	DD03	0.0 nA
AWCR	0.0 nA	AVHR	0.0 nA	DD04	0.0 nA
BAGL	0.0 nA	AVJL	0.0 nA	DD05	0.0 nA

Figure 4.1 Neural Interactome interface. Left: Snapshot of the framework interface. Neuron classes are colored as follows: sensory (blue), inter (green), motor (red). Right: Main functionalities of the framework

Here, we develop Neural Interactome: a visual interactive framework that combines visualization with the interactions of nervous system simulations. Using the nervous system model as its base, the framework visualizes nervous system activity as dynamic graphs for an intuitive representation of neural dynamics. In addition, the framework allows real-time perturbations of the nervous system during the simulation, such as changing the external stimuli I^{Ext} and modifying the nervous system structure (e.g. neural ablation). These features allow for visual characterizations of neural patterns associated with particular external stimuli or perturbations and assist in the investigation of the nervous system functionalities.

For such a framework, an intuitive front-end interface that accommodates both visualization and interactions with the nervous system is needed. In Figure 4.1, we show the interface of Neural Interactome when applied to the nervous system of *C. elegans*. The interface is divided into a control panel on the left and a visualization panel on the right. The control panel lists all neurons in the nervous system categorized into three classes by columns. Each neuron class (sensory, inter, motor) is assigned a unique identification color for visual distinction. When clicking the name of the neuron, the panel prompts a scrollable stimuli bar, which can be used to set or change the constant stimuli (in nA unit) for a neuron before or during mid-simulation. The panel is also used to perturb the connectome structure in the form of neural ablation. By clicking on the name of the neuron while holding the 'shift' key, the neuron is grayed out on the panel. The process effectively removes the neuron from the network by deleting both its gap and synaptic connections. Repeating the process for the ablated neuron re-inserts the neuron into the network by restoring all of its connections in the connectome adjacency matrices. Similar to adjusting external stimuli, both ablation and re-insertion can be carried out either before or during mid-simulation.

The right panel consists of the network graph, which visualizes the nervous system simulations of *C. elegans*, and an interactive time bar that keeps track of the simulation time. Each node in the network graph represents an individual neuron, and the edge represents the gap/synaptic connection between neurons. To display the membrane potential dynamics in real-time, we set the radii of each node to change according to the neuron's membrane potential. Specifically, given the membrane potential V of ith neuron at time t, the radii r is determined by:

$$\begin{aligned} |R_i| &= \frac{R_{max}|V_i|^2}{\rho + |V_i|^2} \\ sign(R_i) &= sign(V_i) \end{aligned} \tag{4.2}$$

where ρ is the slope factor. The transformation takes the form of membrane potential translation to SR calcium-like activation. Such construction is advantageous for the nervous system visualization for several reasons. The SR calcium activation is often used to describe biophysical signals, such as the activation of muscle activity [152; 153; 154]. In addition, several recording techniques for neural dynamics are capable of measuring and monitoring SR calcium activity, which can be directly compared with the visualization. The interactive time bar under the network graph serves a similar function to those used for controlling video playback on websites (e.g., YouTube). It has a dark gray bar that keeps track of the simulated time throughout the simulation. Clicking on a specific point on the bar reverts the network graph visualization to the corresponding time point. Such a function allows for reviewing the simulation during a particular time interval.

In the Figure 4.2, we present the framework in action by investigating the locomotion scenarios of *C. elegans*. Specifically, we simulate neural patterns associated with forward locomotion by injecting constant stimuli into sensory neurons PLM and interneurons AVB, and backward locomotion by stimulating sensory neurons ALM and interneurons AVA, AVD, AVE [79]. In both scenarios, the Neural Interactome captures unique visual characterizations of the resulting neural patterns. By looking at the network graph, the neural

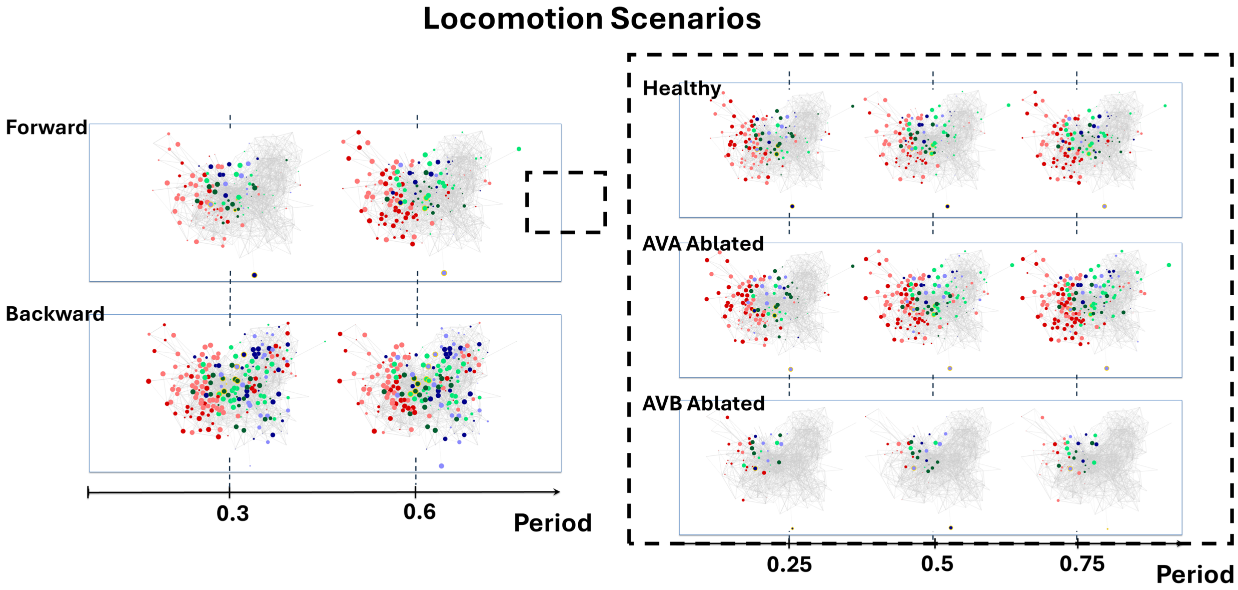


Figure 4.2 Simulation of locomotion scenarios. Simulation of locomotion scenarios (constant stimuli into PLM+AVB) visualized by network graph. The time axis is normalized by the oscillation period of the dynamics (Left), simulation of neural ablations for the forward locomotion scenario (Right)

pathways associated with the stimuli can be easily identified. Stimulation of PLM and AVB neurons activates VB and DB motor-neurons (ventral and dorsal type B) that are known to mediate forward locomotion [155]. Stimulation of ALM, AVA, AVD, and AVE neurons, on the other hand, activates VA and DA motor-neurons (ventral and dorsal type A) that are experimentally found to be associated with backward locomotion [79; 156]. In addition, the model identifies a list of neurons associated with the pathways that are not reported in the literature, such as LUA neurons in the forward locomotion scenarios.

We further test the framework's capability to perturb the connectome by investigating the ablation scenarios for both forward and backward locomotion. In particular, we investigate the effects of ablating AVB or AVA inter-neurons to the neural dynamics associated with the forward locomotion scenario. AVB neurons are identified as part of the sub-circuit experimentally shown to be associated with forward locomotion [79]. Removing AVB neurons has been shown to affect the neural dynamics associated with forward locomotion. On the other hand, AVA neurons are part of the sub-circuit associated with backward locomotion, and their ablation has been suggested to have little effect on the forward locomotion dynamics [79]. In Figure 4.2, we show that the model simulations corroborate such observations, as AVA ablation appears to preserve most of the visual characteristics observed in a healthy forward locomotion scenario, whereas AVB ablation leads to a significant deviation from it. The results from locomotion and ablation scenarios indicate that the Neural Interactome can assist in studying characteristic neural patterns associated with experimental observations.

4.3 Incorporating Biomechanics

While the dynamic model of the nervous system provides information on how stimuli are translated into neural patterns, it does not inform how such patterns are translated into physical behaviors. Resolving such a link requires incorporating three additional modules into the nervous system model in modWorm. These are (i) translation of nervous system activity to muscle calcium dynamics, (ii) calcium dynamics to muscle forces, (iii) muscle forces to body postures according to environment-body interactions (Fig. 3.2). Incorporation of these modules thus closes the loop between the nervous system and the body.

As a first step, we model the translation of neural activity to muscle forces. The process has two steps: (i) translating the nervous system activity into weighted muscle inputs, (ii) translating the muscle inputs into SR calcium activation, which produces the force. The translation of nervous system activity into muscle inputs utilizes the experimentally determined linear mapping between the neurons' membrane potential (279 neurons) and the discrete segments of muscles (96 segments) distributed throughout the body of *C. elegans* [16; 46]. Specifically, given the nervous system activity $\vec{V}(t)$, the weighted muscle inputs $\vec{u}(t)$ for the muscle segments at time t takes following linear transformation

$$\vec{u}(t) = M \cdot \vec{V}(t) \tag{4.3}$$

Where $u(t) \in R^{96\times1}$, $M \in R^{96\times279}$ and $\vec{V}(t) \in R^{279\times1}$. The translation of muscle inputs into calcium activation is modeled by a unidirectional linear cascade where the muscle inputs are converted into T-tubuli depolarisation response β and SR calcium release η using the following differential equations [157; 158].

$$\begin{aligned} \ddot{\beta} + c_1\dot{\beta} + c_2\beta &= c_3u(t) \\ \ddot{\eta} + c_4\dot{\eta} + c_5\eta &= c_6\beta(t) \end{aligned} \tag{4.4}$$

where $c_1 - c_6$ are prescribed CPG values. The muscle activation $A(t)$ is then obtained by

$$A(t) = \frac{a_0 + (\rho\eta)^2}{1 + (\rho\eta)^2} \tag{4.5}$$

Where ρ represents the muscle fatigue [159]. The final force due to muscle activation is then computed by multiplying $A(t)$ with the scaling factor F which represents the force per unit muscle cross section generated by unit activation [160; 152].

$$f_{D,V} = A_{D,V}F \tag{4.6}$$

where D, V subscripts indicate forces corresponding to Ventral and Dorsal directions, respectively. Eqs (4.6) describe the distribution of forces exerted by the muscles in dorsal/ventral direction throughout the *C. elegans* body. Consequently, $f_{D,V}$ allows for 2D representation of *C. elegans* postures. Such a representation is suitable for describing *C. elegans* locomotion in which the worm predominantly moves sideways, bending only in the dorsal-ventral plane [161].

Translation of the force distribution into body postures utilizes the visco-elastic rod model. The model has been proposed to study anguilliform swimming behavior exhibited by animals such as lampreys and eels [162; 163; 164]. The model describes the body as discrete segments of the elastic rod connected by joints and actuated by passive springs (emulating the gap junction between muscle segments), dashpots, and dynamic force generators. Muscle contraction is represented in the rod as a force applied to a segment. The surrounding fluid and its effect on the moving rod are also taken into account in the form of damping. Notably, the body discretization into segments is stable to neural stimulation and provides advantages over the gaited model, which uses synthetic muscle stimulation [83]. Moreover, modeling the body as two-dimensional rods and body-fluid interactions in the form of damping is an effective method for simulating *C. elegans* body in a 2D plane and its surrounding environment while maintaining computational efficiency. This is further described in section 5.4 *Effects of environment and proprioception on locomotion* where we show that the model can emulate locomotion body postures under environmental variations similar to *in-vivo*.

Mathematically, viscoelastic rod model is described entirely by the positions of each rod segment and their angles relative to the horizontal plane. The positions of the segments of the viscoelastic rod are described by the difference equations:

$$
\begin{aligned}
x_{i+1} &= \frac{h}{2}(cos(\phi_i) + cos(\phi_{i+1})) + x_i \\
y_{i+1} &= \frac{h}{2}(sin(\phi_i) + sin(\phi_{i+1})) + y_i
\end{aligned}
\tag{4.7}
$$

Where each x_i, y_i indicates the mid-point positions of body segments and ϕ indicates the angle of each rod relative to the horizontal plane. Due to the recursive structure of Eqs (4.7), once all ϕ are known, only the first pair of (x, y) is needed to compute the rest of the positions. The angles ϕ based on the vertical force (y-direction) g and horizontal force (x-direction) f are determined by partial differential equations:

$$
\begin{aligned}
&J\ddot{\phi} = M_i - M_{i-1} + \frac{h}{2}(g_i + g_{i-1})cos(\phi) - \frac{h}{2}(f_i + f_{i-1})sin(\phi) \\
&M_i = EI_i(\frac{(\phi_{i+1} - \phi_i)}{h} - k_i) + \delta_i(\frac{(\dot{\phi}_{i+1} - \dot{\phi}_i)}{h}) \\
&J_i = \rho h(I_i) + \frac{\pi}{12}r^2h^2 \\
&I = \frac{\pi D^4}{64}
\end{aligned}
\tag{4.8}
$$

where M_i represents the contact moment and J_i represents the moment of inertia. Contact moment M is a function of Young's modulus E which describes the elasticity of the rod, the preferred curvature k, the environmental damping coefficient δ, and the segment lengths h. The moment of inertia J is a function of the rod's material density ρ, rod's radius r, segment length h and the rod's diameter D. The preferred curvature k_i for each segment is determined by the muscle forces $f_{D,V}$ in Eqs (4.6) using the following equation

$$
k_i = \frac{4(f_{D_i} - f_{V_i})w}{8vw^2 - (f_{D_i} + f_{V_i})h^2}
\tag{4.9}
$$

where w is the cross-sectional radius of the rod and v is the stiffness parameter. In summary, the neural activity simulated by the model is translated into muscle forces according to Eqs (4.3) - (4.6). The forces are then used to compute the preferred posture of the worm's body at each time t according to its physical properties (Eqs (4.9)). The final posture is computed by adjusting the preferred posture with the forces from body-environment interactions (Eqs (4.7 - 4.8)).

In addition to integrating the nervous system and biomechanics in a feed-forward manner, we develop an inverse integration approach. Inverse integration computes the neural dynamics that would result from external forces applied to the body. This is based on the assumption that the environmental forces are proportional to the forces that the body exerts on the environment (i.e., Newton's third law). To account for the reaction time of the environment in producing such forces onto the body, we incorporate a time delay into the feedback. Mathematically, the process is approximated by multiplying both sides of Eqs (4.3) by the generalized inverse of M (i.e. psuedo-inverse) and solving for $\vec{V}_{Env}(t)$ given the $\vec{u}_{Env}(t - t_{delay})$ which represents delayed reactive external forces from the environment as follows:

$$\vec{V}_{Env}(t) = M^{+}\vec{u}_{Env}(t - t_{delay}) \tag{4.10}$$

where M^{+} is the pseudo-inverse of M. $\vec{V}_{Env}(t)$ is then integrated through the nervous system by setting $\vec{V} = \vec{V}_{NV} + \vec{V}_{Env}$ in Eqs (4.1) where $\vec{V}_{NV}$ corresponds to the neurons' membrane potentials generated by the nervous system. Note that Eqs (4.10) can also be written in the form of feedback current I_{FDB} when incorporated into Eqs (4.1). Since the reactive forces are proportional to those generated by the nervous system but with the opposite sign, $\vec{u}_{Env}(t)$ can be approximated as

$$\overrightarrow{u}_{Env}(t - t_{delay}) \approx \alpha \overrightarrow{u}^{*}(t) \tag{4.11}$$

where α is the scaling factor for the reaction force and $*$ subscript refers to the operation of switching the force (i.e. dorsal force becomes ventral force and vice versa).

4.4 Neural Dynamics induced by External Force and Neural Stimuli

We use the inverse integration approach to validate our model for three basic locomotion patterns: forward, backward, and turn movements. For each pattern, we apply a sinusoidal force wave traveling along the body with a variable frequency to infer the neural dynamics associated with it. These neural dynamics are then forward integrated by the nervous system to generate the body posture. We then simulate the integration and compare with the locomotion characteristics of freely moving animals. We use eigenworm characteristics to

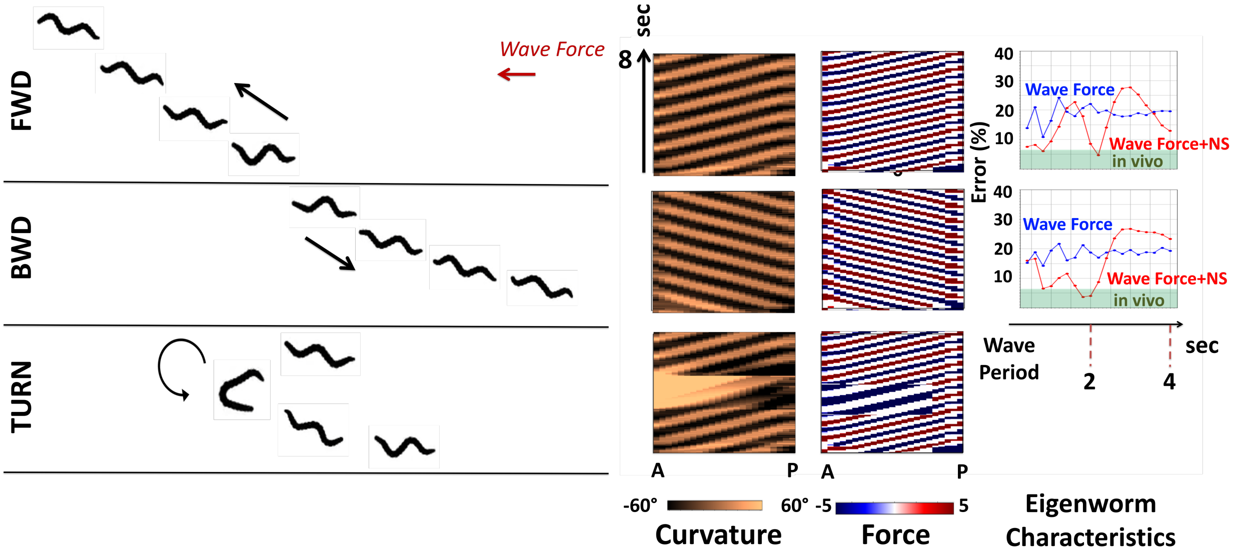


Figure 4.3 Typical locomotion patterns, body curvature and force dynamics generated by three types of external wave forces, corresponding to forward, backward, 180 degree turn movements From left to right: Simulated body snapshots driven by external wave force with minimal eigenworm posture error where each snapshot is sampled every 2 seconds for forward, backward and turn dynamics (Also see SM videos), associated body curvature and muscle force dynamics ordered from anterior to posterior direction for 8 seconds simulation duration, eigenworm posture errors (w.r.t normalized posture coefficients) in the function of varying external wave force periods. Posture errors resulting from direct external wave force on the body are labeled as blue curve with 'Wave Force'. External wave force inverse integrated to resolve neural dynamics which are integrated forward to simulate body dynamics are labeled as red curve with 'Wave Force+NS'. 'Wave Force' curve results with 20% mean error for both directions of the force and does not show preference to wave period. 'Wave Force+NS' appears to be selective to period and achieves minimal error for period 2s of 4.6$ (fwd) and 3.6$ (bwd) within the CI of in-vivo worms (green band; 6.7%, P=0.01) for both directions of movement.

compare the effect of inverse integration of external forces through the nervous system versus external forces acting directly on the body with no nervous system [10; 165; 166]. Postures generated by the integration of the nervous system result in a close match to the coefficients of freely moving worms, with a preference for particular periods (4.6% and 3.6% normalized coefficient errors within 6.7% *in-vivo* error interval obtained from freely moving animals). The optimal frequency of the force being selected is approximately 2s. These results indicate that the response of the nervous system shapes the external effect on the body in a nontrivial and nonlinear manner (Fig. 4.3).

Next, we investigate the neural dynamics associated with these locomotion patterns. In Fig. 4.4A, B, we show membrane potential traces for the full somatic nervous system, and a group of 58 motor neurons

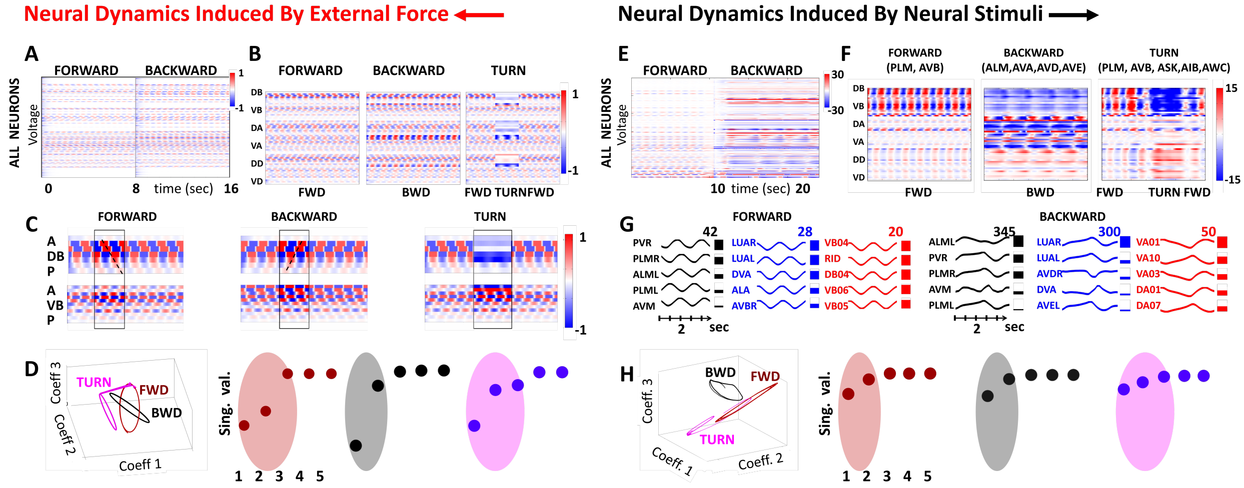


Figure 4.4 Neural responses of *C. elegans* somatic nervous system to external wave forces and neural constant stimuli. A: Color raster plot of membrane potential (difference from equilibrium) of 279 neurons inferred by inverse integration of wave forces to corresponding neural dynamics. Neural responses generated for 16 sec: 0-8 sec: spatial wave force generating forward movement; 8-9: transition; 9-16 sec: spatial wave force generating backward movement. **B**: Color raster plots of membrane potential of motor neurons for forward, backward and turn wave force profiles. **C**: Color raster plots of membrane potential of Ventral and Dorsal type B motor neurons for forward, backward and turn wave force profiles **D**: Evolution of temporal coefficients during forward, backward and turn neural responses (red, black, magenta). Temporal coefficients are associated with PC modes from SVD analysis of all three responses (i.e. projected to a common space of PC1-PC3). **E**: Color raster plot of membrane potential of optimal forward and backward current; 0-10s: forward (0.7nA into PLMR/L, 1.3nA into AVBR/L) 10-15s: transition; 15-25s: backward (2.8nA, 1nA, 0.5nA, 0.5nA into ALMR/L, AVAR/L, AVDR/L and AVER/L respectively). **F**: Color raster plots of membrane potential of motor neurons for forward, backward and turn stimulations (compare with Figure 3B). **G**: Top 5 neurons (which have largest elements in PC1 mode) from each group (sensory, inter and motor) of neurons for forward (left) and backward (right) stimuli. **H**: Evolution of temporal coefficients during forward, backward and turn neural responses (red, black, magenta); compare with Figure 3D.

part of the Ventral Nerve Cord (DB, VB, DA, VA, VD, DD) reported in terms of the difference from the rest membrane potential. We observe that the traces are qualitatively consistent with activity patterns identified in the literature [167; 168; 169; 170]. Most of the neurons are activated during locomotion, where (DB, VB) group is the most active group in forward locomotion and (DA, VA) group is the most active group in backward locomotion. When we focus on (DB, VB) neurons and order them by their physical location along the anterior to posterior axis, we find that within each period of oscillation, the activity propagates with a preferred spatial direction (Fig. 4.4C). During forward locomotion, membrane potential activity propagates from Anterior to Posterior (A→P) while for backward locomotion, the propagation is from Posterior to Anterior (P→A). These propagation directions are consistent with the direction of movement [171; 172].

Analysis of membrane potential responses using Singular Value Decomposition (SVD) elucidates their low dimensional characteristics. The SVD method decomposes the responses into spatial neuronal population modes (PC modes) and their temporal coefficients [93; 168; 173]. We first apply SVD to understand the representation of each individual movement to determine the number of spatial modes needed to represent each activity. The decomposition reveals that there are only a few (2-3) dominant spatial modes representing each movement, similar to empirical findings [165]. We thus use these modes to construct a unified low-dimensional basis of spatial neuronal modes to elucidate the discriminative signatures of forward and backward movement [169]. A viable candidate for such a basis is the set of the first three PC modes obtained from SVD of all motor neurons membrane potentials during forward, backward, and turn movements. Projection of forward and backward responses onto this basis (PC space) yields cyclic temporal trajectories that are well separated and appear to be orthogonal (Fig. 4.4D). When projecting the turn membrane potential dynamics onto this space, we observe that the trajectory departs from the forward cycle and approaches the region of the backward cycle, highlighting the shift in neural dynamics. Notably, the PC space and the coefficient trajectories are obtained from the raw membrane potential dynamics and not from the derivative of calcium dynamics, as described previously [169]. This allows us to identify the PC space as a low dimensional recognition space capable of determining the type and characteristics of movements the network performs from motor neural activity.

In addition to external body forces, we apply neural "clamping" to examine how these stimuli generate body movements. Our first aim is to explore movements created from a simple constant stimulus where most of the neurons do not receive any input. In later investigations, we extend the stimulus to be a dynamic timed stimulus (Fig. 4.7). To identify stimulus profiles that support coherent locomotion, we target the circuit associated with the touch response and seek stimulation of sensory- and inter-neurons that maximizes locomotion distance in either forward or backward directions [79; 167; 174]. From neurons in this circuit, our simulations identify a subset of both sensory- and inter-neurons related to behavioral responses: posterior-touch triggered forward locomotion (sensory: PLM (0.7nA), inter: AVB (1.3nA)), anterior touch triggered backward locomotion (sensory: ALM (2.8nA), inter: AVA (1nA), AVD (0.5nA), AVE (0.5nA)), and turn movement (sensory: +ASK (0.3nA), AWC (0.6nA), inter: +AIB (0.5nA)). The results show that stimulation of both sensory and interneurons promotes directed locomotion, as reported in experimental studies and control theory analyses [175; 176].

Membrane potential traces associated with neural clamping are consistent with membrane potential activity generated by the external force traveling waves. We observe similar active groups of motor neurons: (DB, VB) for forward, (DA, VA) for backward, a phase change in (DB, VB) in turn (Fig. 4.4E), and similar preferred spatial direction in each period of oscillation. SVD analysis of membrane potential traces indicates the occurrence of dimension reduction similar to the external force case. We select the top five neurons from each neural group (sensory, inter, motor), that received the highest weight in the first PC mode, and display their membrane potential over the 4 sec time (Fig. 4.4G). Notably, the selected neurons are those that are experimentally linked to forward and backward movements; for forward locomotion: PLM, PVR sensory neurons and VB motor neurons, and for backward: ALM, PLM sensory neurons and VA motor neurons. Furthermore, membrane potential response time patterns are characteristic to two different types of locomotion examined: for forward stimulus, these are clear sinusoidal membrane potential dynamics with period of 1.8sec in- and out-of-phase oscillations, and for backward stimulus, these are cusp like responses over a longer period of 3.4sec with two, in- and out phases as well. These oscillations are not present in the stimulus and are generated by the intrinsic neuronal network interactions. The projection of forward, backward locomotion, and turn responses onto 3 PC modes embedding yields well separated cyclic trajectories, as in the spatially traveling wave stimulation: the forward cycle is approximately orthogonal to the backward cycle, and turn membrane potential projected trajectory departs from the forward cycle to approach the region of the backward cycle (Fig. 4.4H).

4.5 Effects of Environment and Proprioception on Locomotion

Experiments indicate that the environment plays an important role in shaping coordinated movement [171; 177; 178]. We thereby explore environmental variations and their influence on the model with respect to parameters such as the viscosity of the surrounding fluid. In all variations, we inject a pulse of neural stimulus into PLM sensory neurons with proprioceptive feedback to simulate forward locomotion and examine the effective body movement. For fluid viscosity variation, we compare the postures generated by the model to *in-vivo* experiments in which the viscosity of the surrounding agar fluid was altered by dropping a water droplet (lower viscosity than agar) [179]. We mimicked such an experiment by simulating locomotion in two

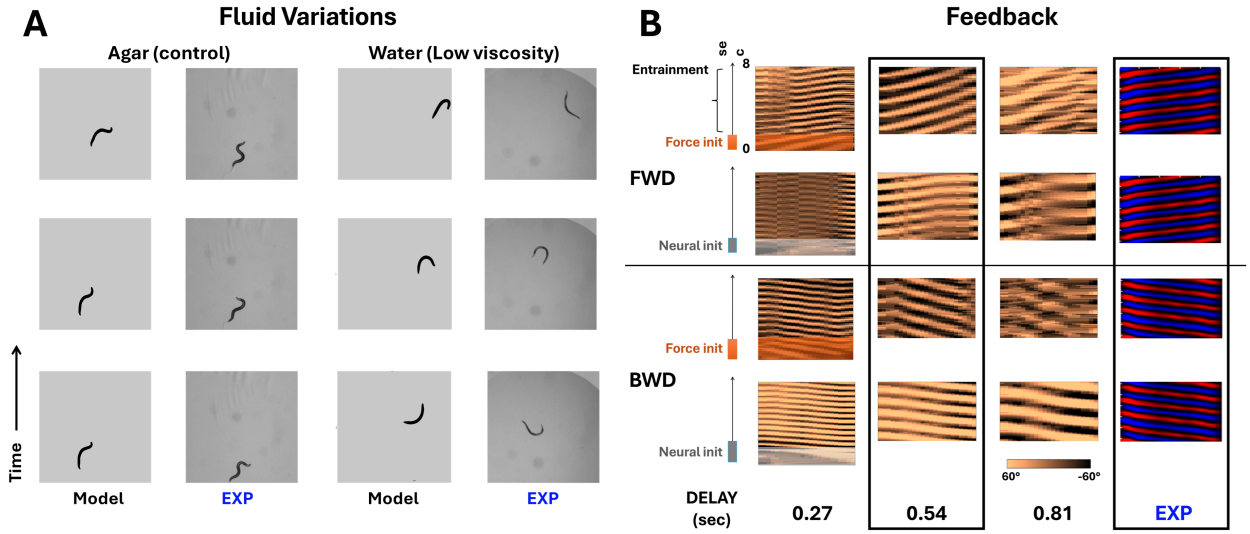


Figure 4.5 Appropriate fluid parameters and proprioceptive feedback and facilitate sustained locomotion. A: Surrounding fluid of model and experiment [179] are varied between agar (high viscosity, columns 1,2) and water (low viscosity, columns 3,4) respectively, to study their effects on locomotion. In the model, viscosity and fluid density are reduced from $10mPa/s$, $1g/cm^3$ to $1mPa/s$, $0.7g/cm^3$ to emulate water droplet in experiment. For each column, 3 locomotion snapshots are taken at 1 second apart. **B**: Feedback is initiated by a wave force (rows 1,3) or neural stimuli (rows 2,4). Columns 1-3 display various feedback time delays, modeling the environmental reaction time of producing external forces on the body, and the curvature profiles they produce and compared to column 4, experimentally recorded curvatures (adapted from [172]). Time delay of approximately 0.5 sec produces optimal forward and backward locomotion which is close to experimental locomotion (highlighted by dashed border).

different viscosity values corresponding to either agar (10mPa/s) or water (1mPa/s). We observe that for the control environment (agar), the model generates movement that features a sine-shaped posture qualitatively similar to that of the experiment, whereas for low viscosity (water) the same stimulation corresponds to strokes of C-shaped postures, atypical for *C. elegans* forward motion (Fig. 4.5A). When compared to experimental body curvatures in water, these postures have similar C-shaped postures as demonstrated in Fig. 4.5A. These results show that a particular neural activity can support various types of behavioral responses, which are determined not only by the precise choice of neural stimulation but also by the appropriate surrounding media for the movement.

In addition to the surrounding environments, experiments indicate that proprioception within the motor neurons circuit can facilitate locomotion and serve as an alternative to the stimulation of command interneurons [155; 177; 180; 181; 182; 183; 184]. To emulate proprioceptive feedback, we utilize an inverse integration method to incorporate additional inter-cellular dynamics to the model and its current term I_{FDB}

which accounts for external body forces by inverse integrating the force that acts on the body subject to a time delay. Such time-delayed feedback control has been suggested to play an important role in shaping coherent locomotion for anguilliform swimmers [185; 61; 88]. We then test feedback effects by initiating locomotion with external stimulation, either neural or external force stimulation. Once the feedback starts to entrain the movement, We gradually turn the stimulation off. We found that in both initiation procedures, feedback entrains the body into sustainable, coherent movements in forward and backward directions, such that the body moves solely due to feedback (Fig. 4.5B). Variation in feedback delay time influences coherency, and we find the delay of approximately 0.5 sec to be closest to experimentally measured patterns (Fig. 4.5B) [172].

4.6 Validation, Recapitulation, and Prediction of Touch Responses

The integration of the nervous system, biomechanics, and proprioceptive feedback constitutes the neuromechanical model of *C. elegans*. To validate the model, we link neural stimulation with proprioceptive feedback and examine locomotion responses studied in the literature for (i) gentle and (ii) harsh anterior/posterior touch. Neural and mechanical triggers for these behaviors have been identified, and we use them to validate the movements generated by the model [79; 82]. For all four touch responses, the model generates typical directional movement patterns: (i) forward; during posterior touch neural stimulation (PLM (Gentle) and PVD+PDE (Harsh)) and (ii) backward; during anterior touch stimulation (ALM+AVM (Gentle) and FLP+ADE+BDU+SDQR (Harsh)), as shown in Fig. 4.6A (Control (Wild)). To test how robust the recapitulation of the response is, we perform *in-silico* ablations of random pairs of neurons (Fig. 4.6A; Control (Rand)) as additional controls. Indeed, random ablations do not change, on average, the characteristic velocities of the four touch responses that we considered. Next, we consider targeted computational ablations as done in prior *in-vivo* experiments and compare velocities and directions of movements with respect to the descriptions published in these experiments [79; 82]; See Fig. 4.6A. Arrows indicate *in-vivo* reported data; colors of arrows indicate consistency (blue); or disparity (red).

Our ablation results are generally qualitatively consistent with previous *in-vivo* findings (19 scenarios out of 20). For the gentle anterior touch responses (left-top in Fig. 4.6A) ALM, AVM we observe that ablations of (AVA) or (AVD) are the most impactful. Ablation of (AVA) nearly stops movement, while ablation of

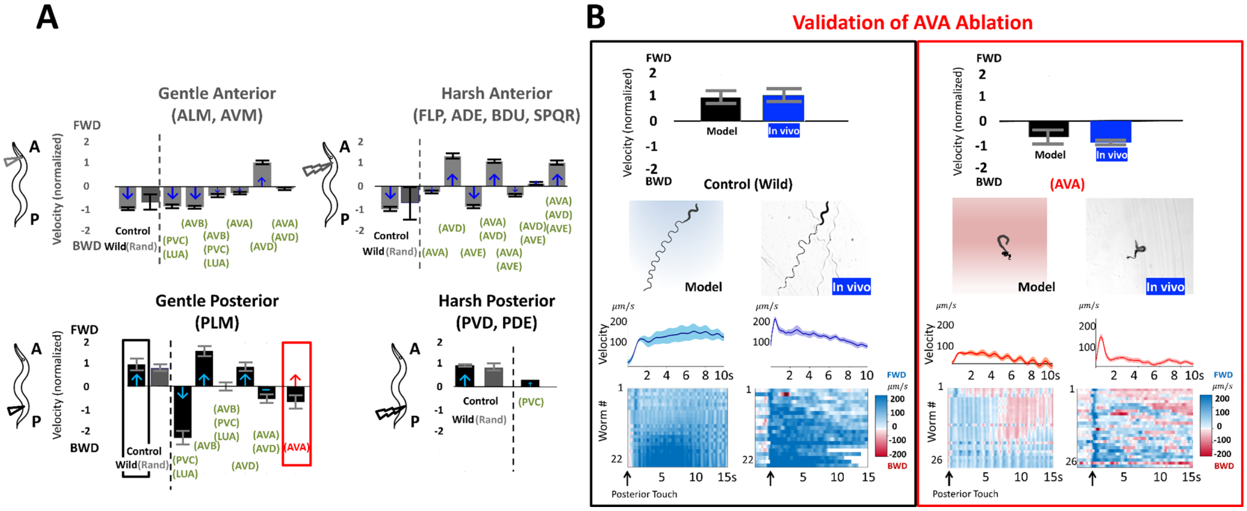


Figure 4.6 Validation, recapitulation, and prediction of locomotion behaviors for touch responses. **A**: Body velocities for neural stimulation associated with Top: Anterior Gentle (left) and Harsh (right) touch responses Bottom: Posterior Gentle (left) and Harsh (right) touch responses. Velocity amplitude is normalized according to the Control Wild (black label) locomotion and direction up/down are chosen as FWD and BWD direction respectively. Recapitulated velocities following ablation experiments in [79; 82](green label) are compared to the Control Wild velocities. Arrows indicate experimental *in-vivo* observations and color indicates match of model with published experimental observations (blue: match; red: mismatch) **B**: *in-vivo* validation of Gentle Posterior Touch model prediction for ablation of AVA. Model predictions are compared with *in-vivo* assays for control (left) and AVA ablated (right). Model and *in-vivo* responses at and after stimulus onset are compared with characteristics of (top to bottom) normalized velocities, response locomotion paths, instantaneous velocities after onset, individual instantaneous velocities (each row is a worm) color plot (blue; positive velocity (fwd); red: negative velocity (bwd)).

(AVD) reverses the direction to forward movement in some cases with respect to the control response of backward movement. Notably, effects of computational ablation are not simple nor additive, e.g., ablation of (AVD) alone is stronger than the ablation of (AVA) + (AVD); ablation of (AVB) + (PVC) + (LUA) causes movement to significantly slow down, while separate ablation of (AVB) or (PVC) + (LUA) permits typical backward movement. Similarly, for the stimulation (PVD, PDE (Harsh posterior scenario); right bottom in Fig. 4.6A) we observe that ablation of (PVC) does not lead to a reverse in direction as in *in-vivo* experiment. For (FLP, ADE, BDU, SDQR (Harsh anterior scenario); right-bottom in Fig. 4.6A) ablation of (AVD) or (AVA) + (AVD) is found to reverse the direction from backward to forward. This observation is similar to the observation described in *in-vivo* experiments [79; 82].

For (PLM stimulation (Gentle posterior scenario); left-bottom in Fig. 4.6A), computational ablations provide novel predictions: while multiple ablations are consistent with *in-vivo*, the model indicates that the ablation of either (AVA) + (AVD) or (AVA) significantly alters the response such that the movement is slower than control, and in about half of the simulated cases, invokes backward movement instead of the control forward movement. Since the ablation of (AVD) alone does not result in a significant change in the response, the model identifies AVA interneurons as having a vital role in forward movement, contradicting the classical classification of AVA as a backward command neuron and the reported results of [79; 82]. Such a finding on the AVA role in forward movement has been presaged by [186] and corroborated by recent experimental works [68].

Since such a response was not identified in the original experiment of gentle posterior touch response, we validate the prediction with a novel *in-vivo* experiment. We use optogenetic miniSOG method to ablate AVA neurons in ZM7198 mutants and compare their responses to gentle posterior touch with control wild type N2 animals (see Fig. 4.6B). Gentle posterior touch was performed mechanically with hair touching the posterior part of the body. Control simulated animals exhibited sustained forward movement, with an average instantaneous velocity of $142 \pm 22\mu m/s$ for a duration of at least 10s after the posterior touch onset. *in-vivo* control worms exhibited matching velocities, forward postures, and bearing with simulated control worms.

With AVA neurons ablated, *in-vivo* worms are unable to perform sustained forward movement. Behavioral assays indicate the average instantaneous velocity of $19 \pm 12\mu m/s$ after the posterior touch, which was in-line with the velocity of $13 \pm 22\mu m/s$ reported by computational AVA ablated. In addition, 61% of tested animals (16/26) performed spontaneous backward movement for a duration of at least 1s (as can be seen in Fig. 4.6B bottom; red color corresponds to backward). The result is in qualitative agreement with recent *in-vivo* findings by [68], and computational ablation of AVA (see side by side comparison in Fig. 4.6B).

4.7 Complex Behaviors and Timed Stimulus

The touch response case study leads us to explore extensions of neural stimulus complexity and to examine the effect of timed external stimuli impulses while the worm is performing locomotion. Here, we consider several case studies. The first case study is avoidance (Fig. 4.7A), which can be induced when forward locomotion is interrupted by ALM + AVM stimulus via anterior touch [187]. In such scenarios, *C. elegans* reacts to the stimulus by stopping and reversing. We examine this scenario by initiating forward locomotion, and after 6 seconds, applying ALM + AVM neural stimulation for 2 seconds. As we show in Fig. 4.7A, ALM

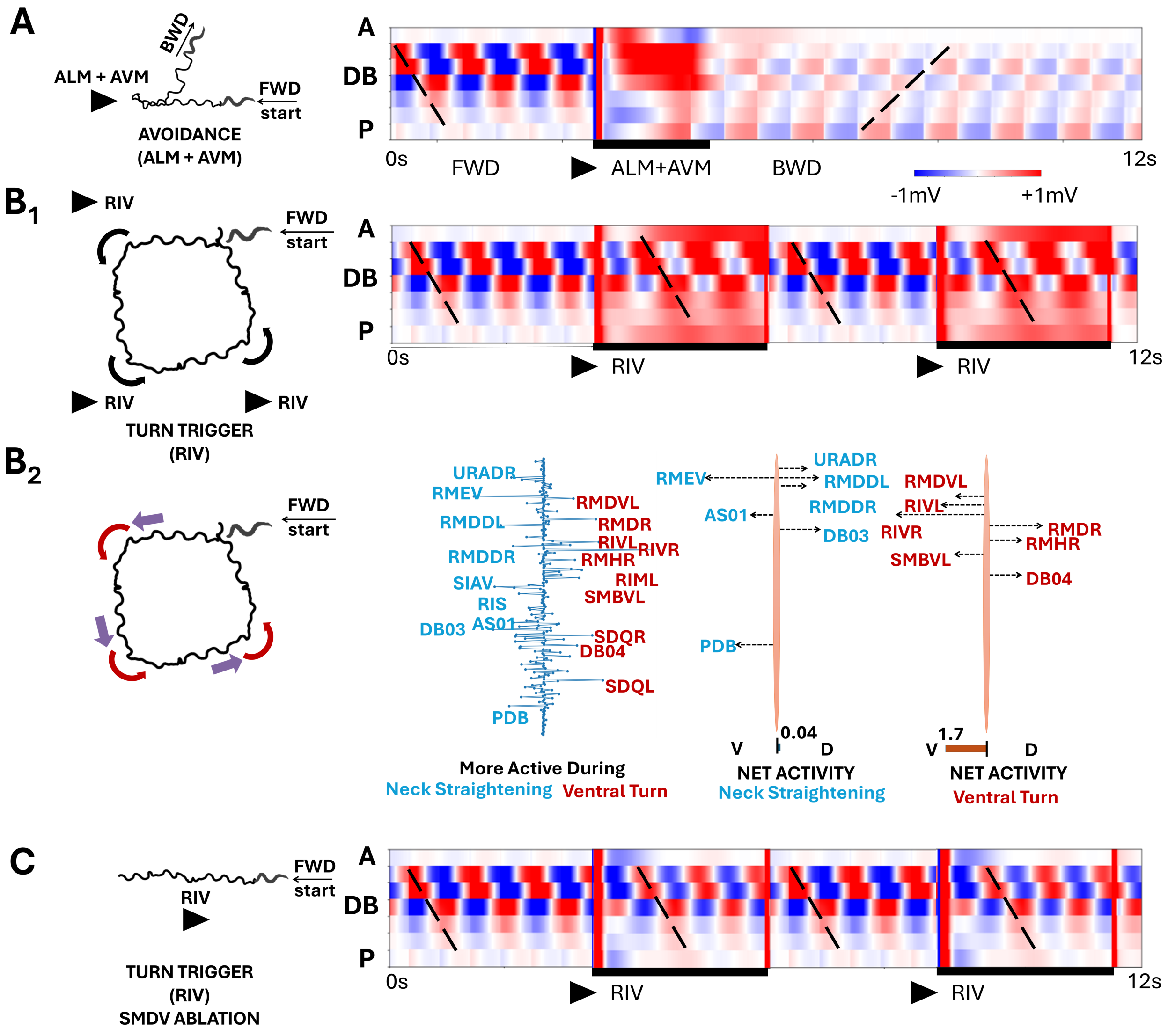


Figure 4.7 Neural impulses modify basal locomotion behavior. **A**: Avoidance behavior in which forward locomotion is interrupted by ALM+AVM impulse leading to backward locomotion (left: head trajectory, right: motor neurons (DB) activity). **B1**: RIV stimulus (2.7nA) is applied every 6 sec for duration of 3 sec during forward locomotion and rotates locomotion direction by 90°. **B2**: Dominant neurons and forces involved in rotation of locomotion direction (2nd column): (i) neck straightening-purple (ii) ventral turn-red. 3rd and 4th columns: dominant motor neurons and corresponding forces they generate in ventral and dorsal directions (3rd column - neck straightening, 4th column - ventral turn). **C**: The effect of SMDV ablation: typical RIV response (rotation by 90°) is disabled.

+ AVM neural stimulus indeed produces body dynamics similar to the avoidance behavior observed *in-vivo*, where the stimulus disturbs forward locomotion, followed by initiating backward locomotion. Inspection of the neural activity of motor neurons (DB neurons are A→P ordered in Fig. 4.7A) indicates that the stimulus induces a change in the directionality of the traveling wave of neural activity from A→P to P→A. The transition is marked as a high constant activity in the anterior motor neurons.

An additional case study that we consider is the sharp ventral turn that occurs during reorientation in the direction of locomotion (Fig. 4.7B). The RIV motor neurons synapse onto ventral neck muscles and have been implicated in the execution of the ventral turn [174; 37]. To examine the role of RIV, we stimulate RIV neurons every 6 seconds for a duration of 3 seconds, while the worm is performing forward locomotion. As we show in Fig. 4.7B1, each RIV stimulus causes a sharp ventral bend of the head, leading to a rotation of the forward locomotion course by approximately 90° while sustaining locomotion in the forward direction. Neural activity indicates that the turn corresponds to a bias added to the membrane potential activity of oscillating motor neurons. The rotation of the body is exhibited by two posture states: (i) neck straightening followed by (ii) ventral turn. These states are observed in experimental studies of the escape response as well [187]. We investigate these states by performing SVD on neural activity in each state and identify dominant neurons associated with the activity. We then compute the force magnitude resulting from the dominant motor neurons activity (Fig. 4.7B2). We find that during the neck straightening state, dorsal and ventral forces tend to be balanced and cancel each other out, while in the ventral turn state, there is a strong ventral force acting on muscle segments. Such analysis reveals neural participation at the cellular level in each state and how neural activity is superimposed to create a particular posture.

As in the touch response studies, systemic ablation can be used to further analyze the observed behaviors. Here, we utilize computational combinatorial ablation of neurons to seek which neurons are most correlated with this behavior. We select all pairs of neurons (R and L) from the group of neurons directly connected to RIV and separately ablate each pair. The analysis shows that the ablation of SMDV causes the most prominent change in dynamics, where it disables the turn behavior and causes the body to continue with forward movement, see Fig. 4.7C. Neural activity in the case of SMDV ablation is similar to neural activity during forward movement, suggesting that both SMDV and RIV neurons are required to facilitate a sharp turn.

4.8 Discussion

This chapter introduces a neuromechanical model of *C. elegans*. Built with the modWorm framework, the integrated model simulates the interaction between the nervous system, the muscles, and the body. The model includes a total of 7 sensory-neuro-mechano-environmental modules, which incorporate the connectome of the full somatic nervous system, its response to stimuli, and the effects of neural activity on body postures and proprioception. Applications of simple stimuli in the model show that the structures of the connectome and polarities set specific movement patterns enabled by neural dynamics and biomechanics. We showed that the transformations between the different constituent modules are in the form of dynamic mappings [32; 34]. This appears to be essential for determining whether neural activity can generate coherent body movements and cannot be resolved without incorporating the complete connectome.

We introduced methods for forward integration (stimulus to muscles) and inverse integration (muscles to neural activity) to find a correlation between neural stimuli and muscles, closing the loop between them through proprioceptive feedback. We tested the model by injecting constant currents into neurons in the touch response circuit. Through the Neural Interactome framework, we observed that stimulation of a few neurons (sensory and interneurons) in these circuits can generate coordinated neural patterns. Together with biomechanics, we showed that the simulated movements from neural dynamics are consistent with the directions and patterns in *in-vivo* experiments. We then examined the effect of proprioceptive feedback and showed that feedback with a time delay can entrain, smooth, and sustain locomotion initiated by neural or external force stimuli. We further tested the model against previous *in-vivo* touch response experiments, which used ablations to identify key neurons involved in the responses. We repeated these ablations and performed additional ablations to further elucidate the roles of neurons participating in these circuits.

The model's ability to generate robust directional locomotion allows for the identification of functional neural circuits and pathways associated with timed neural stimuli during locomotion. We showed examples of timed neural stimulus applications during locomotion, which give rise to intricate locomotion patterns and orchestrated behaviors. Specifically, we triggered behaviors such as avoidance and sharp turns through ALM + AVM and RIV neural impulses. We demonstrated that computational ablation surveys can identify neurons participating in the sensorimotor pathway of these behaviors, e.g. SMDV neuron in RIV impulse pathways.

Chapter 5

Model Extensions for Simulation Accuracy Enhancement

In-vivo validation from Chapter 4 indicates the potential of the model to inform and complement experimental studies. In-depth analysis of motion, however, shows that the characteristics of locomotion, such as eigenworm coefficients, are similar but do not precisely match with *in-vivo* coefficients (Fig.5.1 Left).

In this Chapter, I partially adapt studies from [JK1] and utilize modWorm to implement different variations and extensions into the neuromechanical model of *C. elegans* to achieve a refined fit of simulated behavior to experimental findings. In particular, I consider empirically driven variations to both intra- and extra-cellular dynamics, as well as data-driven and conceptually driven variations, such as model parameter optimizations with respect to empirical data and an alternative model description for the neural dynamics.

5.1 Empirically Driven Model Extensions

Novel experimental data show that there are additional higher-order properties that play roles in neural activity and behavior, such as spiking neurons, novel connectomics data, and extra-synaptic interactions, etc [17; 37; 24; 149; 25; 188]. The proposed model, through modWorm, supports the incorporation of these additional properties and potential future variations. We therefore, incorporate these model variations to explore the refined fits of eigenworm characteristics for the forward and backward movements in the touch

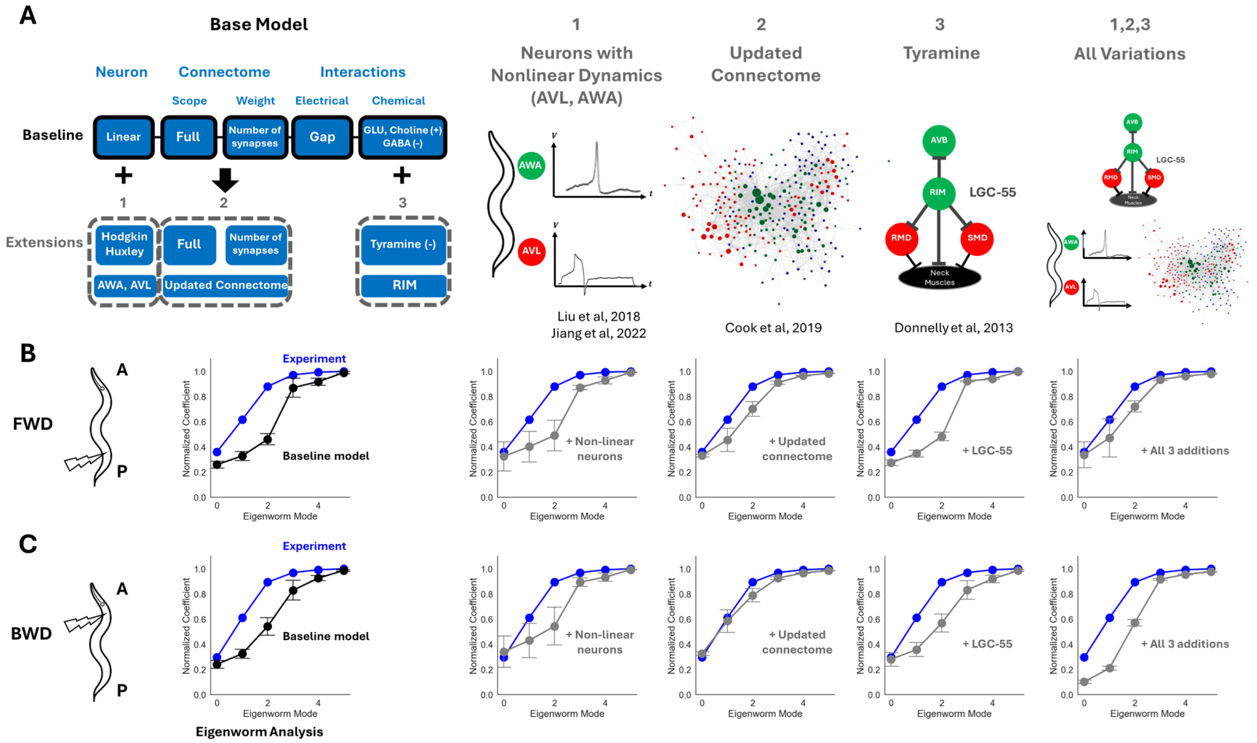


Figure 5.1 Model variations and their effects on eigenworm coefficients obtained from simulated FWD and BWD locomotion. **A**: Graphic illustrations of considered extensions to the model – from left to right: Neurons with "known" non-linear channels: AWA and AVL, updated connectome mappings, Tyramine gated chloride channels (LGC-55), and combination of all three additions. **B**: Comparison of cumulative eigenworm coefficients during FWD locomotion between experiment (blue) vs model (black) and each of the model extension (grey, N = 10, p = 0.05). FWD locomotion is simulated in the model by injecting a pulse of 3nA (±10% variations each trial) of current into PLML/R followed by entrainment of proprioceptive feedback. For updated connectome, the synaptic weights are randomly varied by ±10% each trial. **C**: Comparison of BWD locomotion between experiment vs model (black) and each of the model extension (grey, N = 10, p = 0.05). BWD locomotion is simulated in the model by injecting a pulse of 6.8nA and 3nA of current into ALML/R, AVM (±10% variations each trial) respectively with proprioceptive feedback. Synaptic weights are randomly varied by ±10% each trial for updated connectome addition.

response case study.

The variations we consider are (1) neurons with non-linear ion channels, (2) variations of the connectome, and (3) tyramine-gated chloride channels (LGC-55). Each of these variations was recently proposed in experiments to have a potential role in mediating locomotion [17; 37; 25; 188]. For variation (1) a group of (AWA, AVL) neurons was shown to exhibit all-or-none spiking action potentials [24; 25]. The spikes are mediated by nonlinear voltage-gated calcium and potassium channels and were initially observed in AWA sensory neurons. However, recent experiments found such channels facilitate action potentials in enteric motor neurons (AVL) as well. While both the AWA and AVL are not known to be directly associated with locomotion, we include these variations to demonstrate the incorporation of additional biophysical dynamics to the model and their possible roles in locomotion. For variation (2), we consider the variation of the connectome with an up-to-date electron microscopy reconstruction that is based on the analysis of both new and published electron micrographs [17]. The updated connectomics include additional connections for both gap and synaptic connectomes across the entire *C. elegans* nervous system that were previously missing or inaccurate in the old hermaphrodite connectome data [14]. Lastly, for variation (3), LGC-55 is expressed by neurons that are post-synaptic to tyraminergic motor neurons. These neurons receive inhibitory signals from tyraminergic motor neurons and are known to inhibit head movements and forward locomotion during the escape response [37]. In modWorm, incorporating these variations into the model can be achieved by modifying the intra-cellular module to integrate new individual neuron channels (variation (1)), extra-cellular modules (gap and synaptic currents) with updated connectivity weight matrices (variation (2)), or an updated neuron polarity matrix (variation (3)).

Figure 5.1 and Table 5.1 describe the error comparison of forward and backward locomotion eigenworm coefficients (*in-vivo* vs model) obtained with model simulations versus the models with variations. Variation (1), selected neurons having non-linear channels appear to have a slightly higher error (14.8 ± 2.1) compared to that of the model (13.1 ± 0.8), where the increase in error is mostly due to AVL modeled by the HH-model with nonlinear channels. Variation (2), updating the connectome data to the dataset published in [17] on the other hand, significantly decreased the error between the variant model and *in-vivo* coefficients (5.6 ± 0.7). Especially, the backward movement *in-vivo* coefficients were within the confidence interval (P = 95%) of the model obtained coefficients, indicating a closer match. From the incorporation of LGC-55, variation (3), we observe that errors for both forward and backward locomotion do not change significantly from their base counterparts (12.8 ± 0.8). This is expected since LGC-55 was found to be primarily associated with head turning behavior during the pirouette maneuver, e.g., omega turn, rather than basal locomotion [37; 188]. Combining all three variations decreased the error for forward (8.8 ± 2.3) but slightly increased the error for backward locomotion (13.0 ± 0.5). These results suggest that the effects of variations are not additive and indicate the existence of additional processes that may contribute to shaping *in-vivo* locomotion postures and the importance of connectomes on body dynamics.

Variation Type	Movement direction (posterior, anterior)	Mean eigenworm error	FWD eigenworm error	BWD eigenworm error
Static nonlinear synapses	(FWD, FWD)	$13.0 \pm 0.2\%$	$11.7 \pm 0.2\%$	$14.3 \pm 0.5\%$
Optimized connectomes (w.r.t. FWD locomotion)	(FWD, FWD)	$8.3 \pm 0.6\%$	$1.9 \pm 0.2\%$	$14.6 \pm 1.2\%$
Non-linear channels (AWA, AVL)	(FWD, BWD)	$14.8 \pm 2.1\%$	$14.7 \pm 2.7\%$	$14.9 \pm 2.4\%$
model	(FWD, BWD)	$13.1 \pm 0.8\%$	$14.0 \pm 1.3\%$	$12.2 \pm 1.7\%$
LGC-55 (Tyramine)	(FWD, BWD)	$12.8 \pm 0.8\%$	$13.3 \pm 1.1\%$	$12.3 \pm 1.3\%$
All variations	(FWD, BWD)	$10.9 \pm 1.2\%$	$8.8 \pm 2.3\%$	$13.0 \pm 0.5\%$
Updated connectomes	**(FWD, BWD)**	$\mathbf{5.6 \pm 0.7\%}$	$\mathbf{6.4 \pm 1.1\%}$	$\mathbf{4.8 \pm 1.6\%}$

Table 5.1 For each model variation (row), movement directions with respect to posterior/anterior touch stimuli and normalized eigenworm coefficient error vs experiment are shown. Model variations are first sorted with respect to correct movement direction associated with posterior and anterior touch (FWD, BWD) which are then sorted with respect to mean eigenworm coefficient error in descending order.

5.2 Model Optimizations

Following the improvements of simulated behavior with variation (2), updated connectome data, we asked how further optimization of the connectome parameters with respect to a specific behavior could affect the overall simulated behavior errors. Such task-optimized neural parameters have been successful in predicting experimental neural activity and behaviors for different organisms, including Hydra, *C. elegans*, and fruit fly [59; 99; 103]. We used a Genetic Algorithm to optimize individual synapse strengths of the updated connectome data from [17] (both gap and synaptic) with respect to *in-vivo* forward locomotion eigenworm coefficients. The optimization reduced the normalized coefficient error from 5.8 to 1.9 for simulated forward locomotion (Figure 5.2). To validate whether these parameters are generalizable to other types of locomotion, we applied neural stimulus associated with backward locomotion, and evaluated the errors with respect to the associated *in-vivo* eigenworm coefficients. This resulted in a significant increase in the error from 4.8 to 14.6, changing the movement direction from backward to forward (Table 5.1, row 2). The results indicate that neural parameters (e.g., connectome mappings) optimized for a specific behavior are not necessarily generalizable to other behavior types or indicate biophysical parameters.

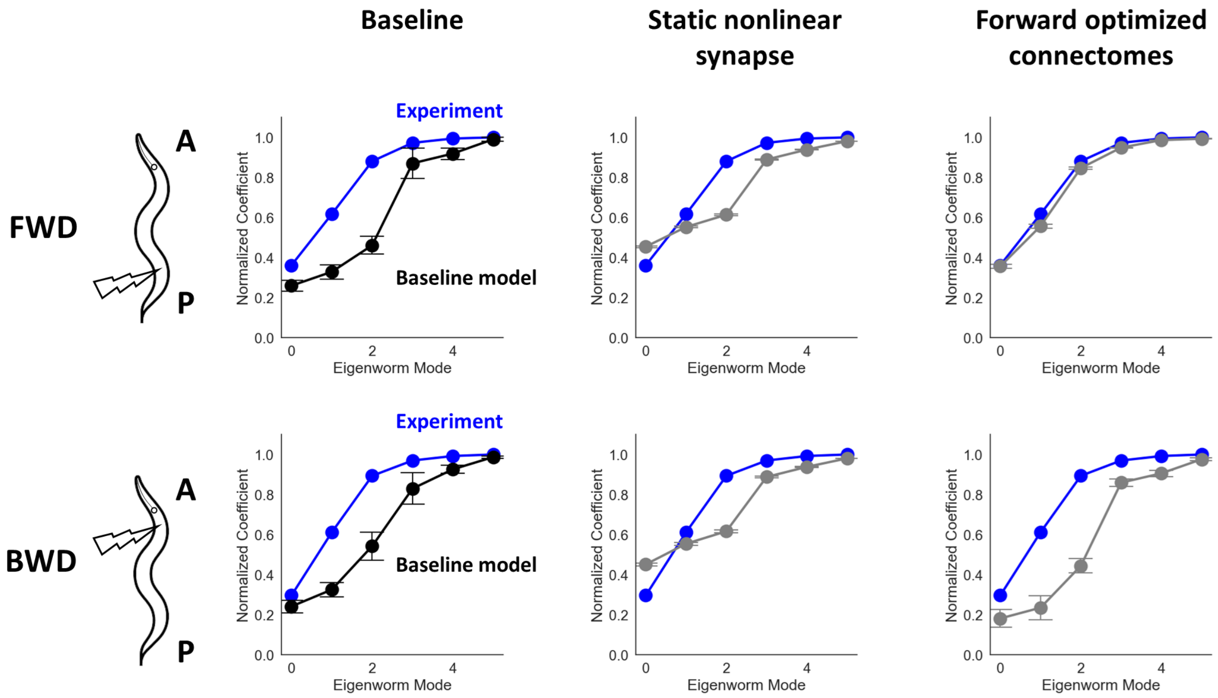


Figure 5.2 The cumulative eigenworm coefficients for forward and backward locomotion with respect to experimental coefficients (blue). From left to right, cumulative coefficients for the first 6 modes using baseline mode, baseline model using static nonlinear synapses of the form $s \sim \sigma(v)$ instead of Eqs (4.1) and using optimized connectomes with respect to forward locomotion coefficients.

5.3 Alternative Neural Dynamics Model

In addition to model optimization with respect to neural parameters, we conduct variations on model dynamics equations to study their effects on simulated behaviors. We specifically modify the synaptic dynamics term I^{Conn} so that the transmitter release gating variable P^S is calculated with a static nonlinearity (e.g., sigmoid) dependent only on membrane potential (i.e., $P^S = 1/(1 + e^{-\beta V})$). Such a variation would be more comparable to the artificial neuron activation function and is a simpler approach than the original synapse model, where the gating variable P^S is governed by its own differential equations similar to the Huxley-Hodgkin channel [189]. The results indicate that while the simulated behavior under such variation has similar eigenworm coefficient errors as the base, it results in an abnormality in movement properties, where both posterior and anterior touch responses lead to forward locomotion, and the locomotion speed is significantly slower (Table 5.1, row 1). The results suggest that the detailed synaptic transmission model may play an important role in shaping the simulated locomotion behavior closer to *in-vivo*.

5.4 Discussion

In this chapter, I introduced systemic extensions and variations of the neuromechanical model to explore their contributions to locomotion. We incorporated model extensions based on experimental findings, such as neurons with individual non-linear channels (AWA, AVL), updates to connectome data, and synaptic channels driven by additional neurotransmitters (e.g., tyramine) that can assist in investigating which aspects contribute to achieving a refined fit in locomotion metrics between *in-vivo* and the model. The results indicate that the global network properties, such as connectomes, may play an important role in modulating locomotion body postures. In addition to empirical variations, I introduced data-driven and conceptually driven variations, such as task-optimized synaptic parameters and an alternative model for synaptic dynamics, to elucidate their effects on simulated behaviors. Such variations highlighted the model's sensitivity to the dynamical structure of underlying modules and associated parameters in achieving the accurate simulation of locomotion.

Beyond the considered variations, information for the connectome, neural dynamics, and processes such as monoamines signaling and neuropeptides activity modulation is rapidly becoming available [17; 24; 149; 150; 25; 188]. Moreover, modeling methodologies for biomechanics are starting to adopt three dimensions and continuous models [95; 190]. Incorporating such components via modWorm could further indicate behaviors and processes that are currently not included in the model and identify additional dynamics observed *in-vivo*, such as generative spontaneous behaviors.

Chapter 6

Connecting the Neuromechanical Model with the Environment

In this chapter, I introduce mod-SenseWorm, an extension of modWorm framework that incorporates environmental stimuli for the simulation of behavior. mod-SenseWorm translates environmental stimuli into dynamic neural stimulation to emulate the chemotaxis responses of *C. elegans*. In particular, I utilize mod-SenseWorm to study O2 aerotaxis responses of *C. elegans* and recapitulate empirically observed behaviors with respect to changes in O2 levels. By further analyzing the simulated neural activity, we show the framework's potential to identify functional pathways associated with chemotactic behavior.

6.1 Integrating Neuromechanics with Environmental Stimulus

mod-SenseWorm adds two optional, configurable modules in addition to modWorm modules described in previous chapters. These modules achieve closed-loop simulation between neuromechanics and environmental stimulus: i) Spatial environment module and ii) Spatial-temporal neural stimulation module. The spatial environment module defines a 2D spatial domain for simulation, i.e., chemical distribution surrounding the animal, in synthetic or natural environments. The module defines the environment as a two-dimensional scalar field, where each (x, y) coordinate is associated with an arbitrary stimulus function value $S(x, y)$. These values can be set as arbitrary numbers to represent specific units of measurement of the environmental

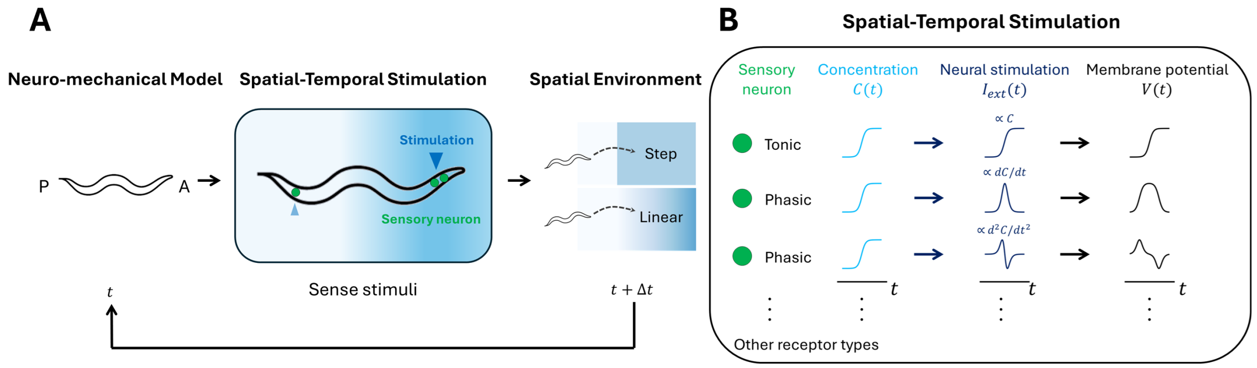


Figure 6.1 Closed-loop simulation of neuromechanics and spatial environment in mod-SenseWorm. **A**: At time t, stimuli are applied to sensory neurons according to the location of the worm's body followed by simulation of neuromechanics to update the worm's position in space. The new body position is then used to update the stimuli to repeat the process until the end of simulation. **B**: The stimulation module translates the scalar field value in stimulus space (e.g., concentration) to neural stimuli for each sensory neuron according to their receptor types (tonic vs phasic). The translated neural stimuli are then used to integrate nervous system model to generate membrane potential dynamics.

stimulus in space (e.g., concentration). The stimulus distribution can be any two-dimensional function with arbitrary shapes and sizes (e.g., step, linear, gaussian, etc). During simulations, the spatial coordinates of the worm's body are used to interpolate the stimulus values $S(x, y)$ that are associated with different parts of the body (Fig. 6.1A middle). These stimulus values are then converted to neural stimulation $I_{ext}(t)$ according to the anatomical locations of the neurons and their chemical receptor types as defined by the spatial-temporal neural stimulation module.

The spatial-temporal neural stimulation module links environmental stimulus with the model through the translation of $S(x, y)$ to neural stimulation $I_{ext}(t)$. The module has two configurable components: i) neurons to body mapping and ii) environmental to neural translation functions. Neurons to body mapping determines the spatial locations of neurons being stimulated relative to the worm's body; e.g., neurons that are close to the head (e.g., the nerve ring) can be mapped to the anterior body, whereas neurons close to the tail are mapped to the posterior body. Environmental to neural translation functions define the mathematical protocol f_i, in which the environmental stimulus value $S(x, y)$ at time t registered by neuron i is translated into neural stimulation $I_{ext}(t)$. Multiple translation functions can be associated with multiple neurons being stimulated (e.g., $f_1, f_2, ...$) depending on their receptor properties. For example, a neuron with a tonic receptor type can be modeled with a translation function of the form $I_{ext}(t) \sim S(t)$ whereas the phasic receptor type can be

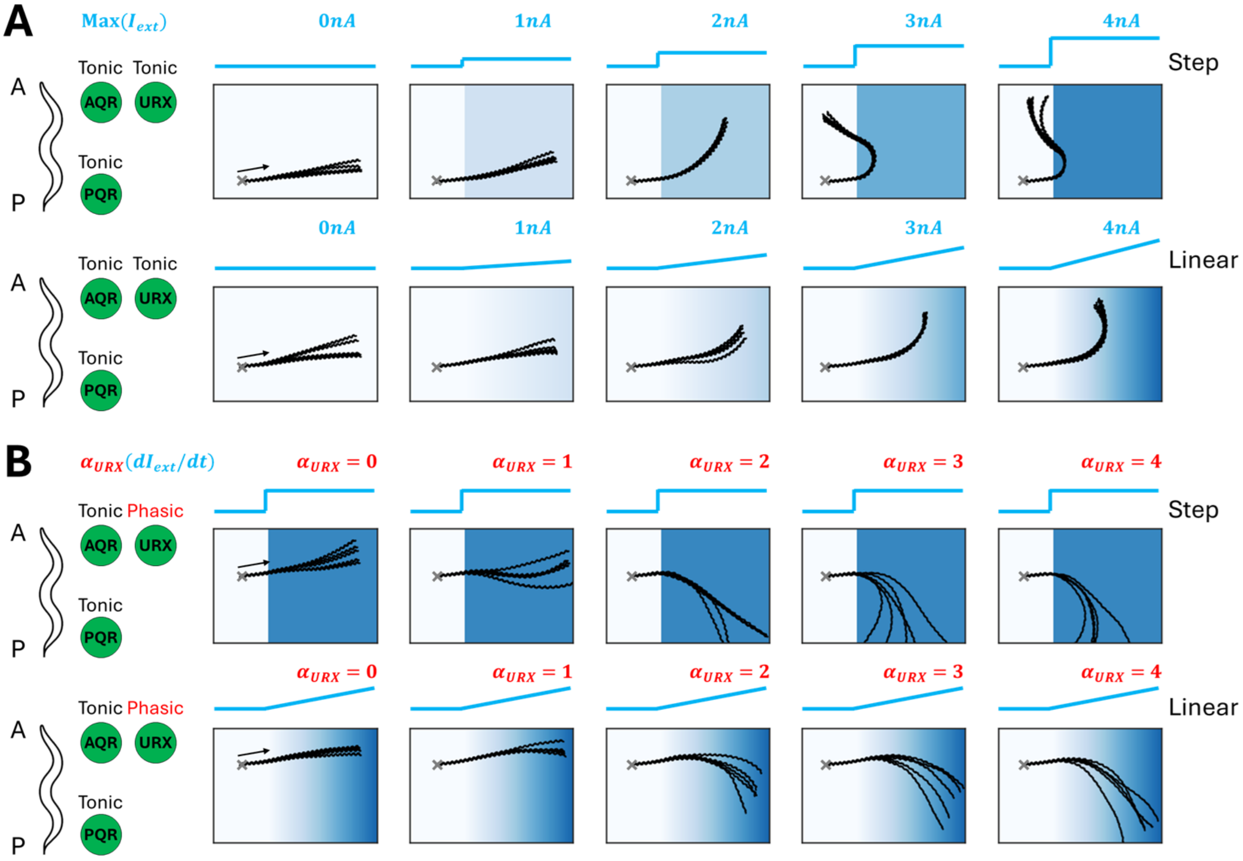


Figure 6.2 Simulated worm trajectories in response to fast and slow rising O2 environments. A: Simulated worm trajectories for AQR, PQR, URX exhibiting tonic receptors. For both fast and slow rising O2 environments, maximum O2 levels are varied 5-fold (0nA, 1nA, 2nA, 3nA, 4nA). **B**: Simulated worm trajectories when using for AQR, PQR exhibiting tonic receptor and URX exhibiting phasic receptor. For both fast and slow rising O2 environment, maximum O2 levels are fixed to 4nA whereas sensitivity coefficient α_{URX} is varied 5-fold (0, 1, 2, 3, 4). For all scenarios simulations are repeated 5 times (n = 5) with identical starting locations and simulation duration of 120 seconds (2 minutes).

modeled with derivatives of a scalar field of the form $I_{ext}(t) \sim \frac{d^n S(t)}{dt^n}$ (Fig. 6.1B).

6.2 Effects of Environment Variation on Simulated O2 Responses

We utilize the mod-SenseWorm framework to study O2 responses modulated by 3 sensory neurons: AQR, PQR and URX. Situated in both the anterior (AQR, URX) and posterior regions (PQR) of the body, these neurons are experimentally known as the main O2 sensors of *C. elegans* [48; 174; 191; 192]. To establish the baseline O2 response, we utilize the spatial environment module to incorporate environments representing both fast (step-like) and slow (linear-like) rising O2 levels. Note that for the fast-rising O2 environment, $dO2/dt \sim 0$ within the O2 region (i.e., S > 0) whereas for the slow-rising O2 environment, $dO2/dt > 0$. In the spatial-temporal neural stimulation module, the neurons are mapped to the body according to their anatomical locations. That is, AQR and URX are assigned to the anterior portion of the worm's body and PQR is assigned to the posterior portion. All 3 neurons are set to respond tonically to the O2 stimulus (i.e., $I_{ext}(t) \sim S(t)$) with an identical stimulus translation function as identified by the experiments [193]. To survey the range of behavior responses with respect to different O2 levels, we vary the maximum O2 levels 5-fold for both fast and slow increasing O2 environments where each level is represented as the maximum neural stimulation value $I_{ext}(t)$ associated with the max O2 level (Fig. 6.2).

Figure 6.2A shows the simulated body trajectories for both fast and slow rising O2 environments over 5-fold O2 levels (0nA – 4nA). For each O2 level, we repeat the simulations 5 times to ensure the reproducibility of the simulated responses. For the fast-rising O2 environments, we see that the simulated worms tend to increase their turning rate as O2 level increases (Fig. 6.2A, Top). The turn rate significantly increases for max O2 level > 3nA where all worms quickly escape the O2 region. Upon repeating the experiments with a slow rising O2 environment, we see similar trends of increased turn rate proportional to the max O2 level, but with a lesser turn angle than the fast-rising O2 scenarios (Fig. 6.3B, Top). For both fast and slow rising O2 environments, the speeds of the simulated worms are relatively similar across the O2 levels (Fig. 6.2A). These results highlight the avoidance response accompanied by turning as the primary simulated O2 response upon increased O2 levels with tonic AQR, PQR, and URX. Such an increased turning rate in response to elevated O2 levels has indeed been reported in *in-vivo* experiments [194; 191; 195].

6.3 Effects of Receptor Type Variation on Simulated O2 Responses

Having established the base O2 response, we utilize the spatial-temporal neural stimulation module to implement different receptor types for O2 sensors and study their effects on simulated behavior. Experiments have identified URX sensory neurons exhibiting both tonic and phasic properties in response to the rising O2 levels, whereas AQR and PQR primarily exhibit tonic receptors [193]. We therefore equip URX with a phasic receptor while keeping AQR, PQR as tonic O2 receptors in the stimulation module. Such a setup allows URX to respond to the derivative of environmental stimulus (dS/dt) instead of responding directly to

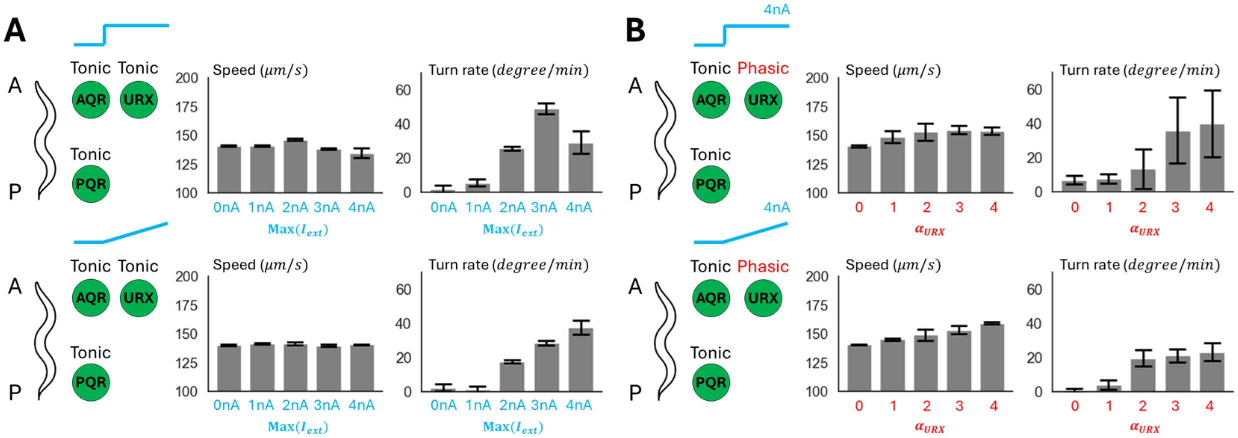


Figure 6.3 Speed and turn rate measured for simulated scenarios. Speed is measured in $\mu m/s$ whereas turn rate is measured in degrees/min. Each bar represents the mean speed/turn rate over n=5. Confidence interval of 95% confidence is used for all plots.

$S(t)$. To ensure that the observed behavior effects are solely due to the tonic → phasic receptor change in URX, we keep the max O2 level constant at 4nA for both fast and slow rising O2 environments while only varying the sensitivity coefficient α_{URX}, which is multiplied to dS/dt to calculate $I_{ext}^{URX}(t)$.

Figure 6.2B shows the simulated body trajectories for both fast and slow-rising O2 environments over 5-fold α_{URX} values (0 – 4). The results show a positive correlation between the sensitivity coefficient α_{URX} and the turn rate. Notably, we observe overall higher variance in movement trajectories and turn rates across the simulations (Fig. 6.3B). Such variance could be due to the noise caused by the worm's head oscillations during locomotion, leading to fluctuations in dS/dt and subsequently $I_{ext}(t)$. For a fast-rising O2 environment, most of the worms fully escape the O2 region with $\alpha_{URX} > 3$, indicating a positive association between phasic URX O2 sensitivity and turn behavior. For a slow-rising O2 environment, the turn rate continues to increase proportionally to α_{URX}, which plateaus after $\alpha_{URX} > 3$. Notably, for both the fast and slow-rising O2 environments, the worms turn in a ventral direction instead of a dorsal direction, as observed in tonic URX scenarios, indicating a potential link between turn bias and URX encoding strategy. We also notice an overall increase in movement speed proportional to α_{URX} (Fig. 6.3B). Such an increase in speed was also reported in experiments when the worms' environment shifted from low to high O2 levels [193; 191; 194; 196].

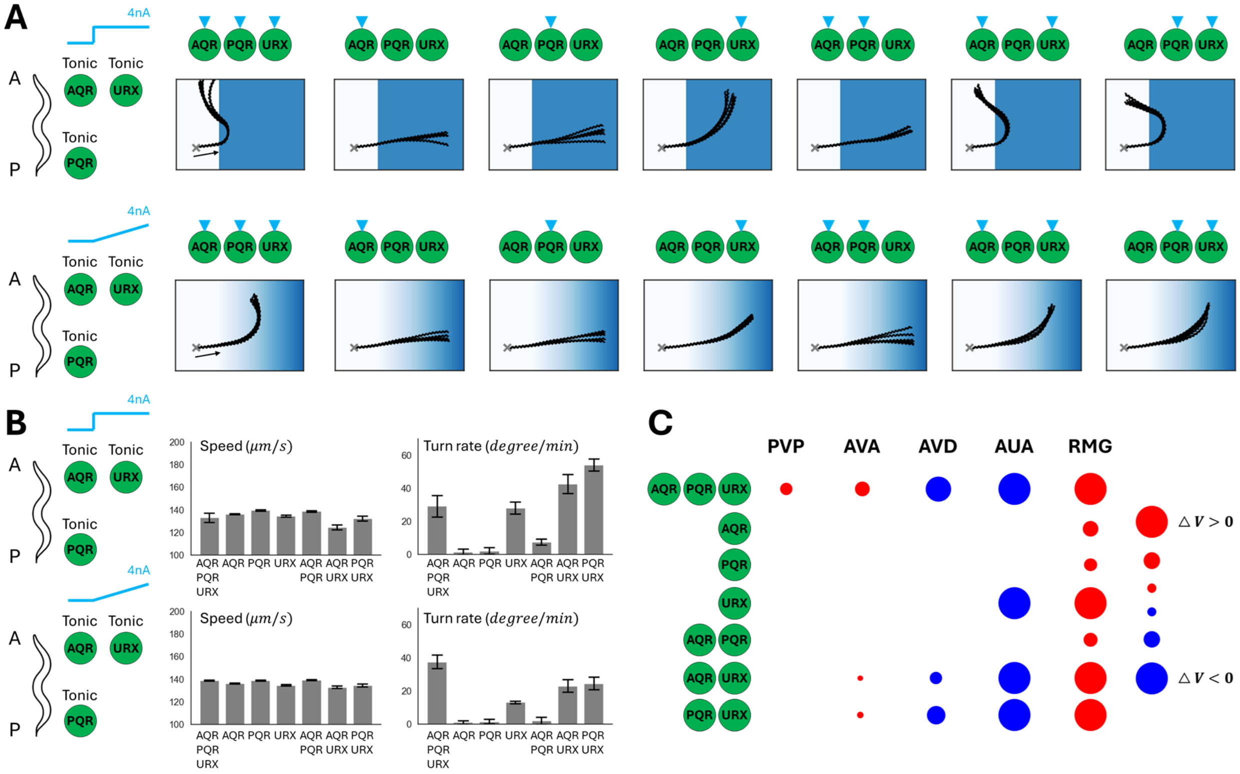


Figure 6.4 Simulated behavior and neural activities subject to selective stimulatios. **A**: Simulated worm trajectories for stimulation masking scenarios. Simulated body trajectories for all possible input masking combinations (n=7) for AQR, PQR and URX sensory neurons with respect to fast and slow rising O2 environments. AQR, PQR, URX exhibit tonic receptors for all scenarios. **B**: Speed and turn rate measured for each input masking scenario. **C**: Neural activities of most connected interneurons for each input masking scenario where the size of the circle represents the normalized magnitude of membrane potential and color represents the depolarization (red) and hyperpolarization (blue) with respect to resting potentials.

6.4 Selective Stimulation to Identify Neuron Contribution to Behavior

Experiments indicate that selective stimulation of sensory neurons can lead to different behavioral outcomes [193; 26]. Such experiments are indeed possible within the framework by simply turning off the O2 receptors for a selection of O2 sensors (i.e., stimulation masking). Neurons with their receptors turned off are unresponsive to environmental stimulus, allowing neural stimulation to be applied only to a subset of sensory neurons throughout the simulation. To study how different stimulation configurations affect O2 response, we consider all possible masking combinations (n = 6) of AQR, PQR, and URX while all 3 neurons are set to exhibit tonic O2 receptors. (Figure 6.4A).

Figure 6.4 shows the simulated O2 responses for all masking scenarios. From the results, we see that the individual stimulation of AQR or PQR and the co-stimulation of AQR + PQR do not invoke the turn response, but stimulating URX alone partially restores the turn (Fig. 6.4B). The results further suggest the role of URX in modulating the observed turn and avoidance. Interestingly, while URX alone can invoke a turn response, it results in a stronger turn angle when stimulated alongside AQR and PQR (Fig. 6.4B). Since stimulating AQR and PQR alone does not produce the turn, the results indicate potential neural pathways exclusive to the co-stimulation of AQR, PQR, and URX.

6.5 Survey of Downstream Neural Activities to Identify Sensorimotor Pathways

To identify such a pathway, we investigate simulated neural activity patterns of downstream interneurons of AQR, PQR, and URX for all masking scenarios. Such an analysis can be achieved within the framework by simply retrieving the simulated membrane potential activities of target interneurons for each simulation. In particular, we examine the membrane potentials of PVP, AVA, AVD, AUA, and RMG interneurons, which receive the most gap and synaptic connections from AQR, PQR, and URX [17].

Upon inspection of membrane potential responses while the worms are fully inside the O2 region (i.e., $S > 0$), we observe that the activities of RMG and AUA are associated with the individual stimulation of URX as reported in experiments [193; 197]. PVP, AVD, and AVA, on the other hand, respond only when URX is co-stimulated with AQR or PQR 6.4C), suggesting a possible link between PVP, AVD, AVA, and the modulation of turn rate by the co-stimulation of URX, AQR, and PQR neurons. Notably, PVP interneurons only became active when all 3 sensory neurons, AQR, PQR, and URX, were stimulated, exhibiting an AND-gate-like activation pattern 6.4C). Such a gate-like encoding strategy is also found in other interneurons (e.g., AIA), playing an important role in modulating the chemotaxis responses to chemicals (e.g., diacetyl) [26]. The results show the use case of the framework to perform a combinatorial survey to identify potential functional pathways associated with chemotaxis behavior.

6.6 Discussion

In this work, I introduced mod-SenseWorm: a framework to computationally study the chemotaxis behavior of C. elegans. mod-SenseWorm extends recently introduced modWorm with 2 additional modules: i) 2D spatial environment and ii) Spatial-temporal neural stimulation system. The extensions allow for closed-loop simulations between neuromechanics and environmental stimulus. To show the framework's potential to study C. elegans chemotaxis behavior, we apply mod-SenseWorm to investigate O2 aerotaxis response modulated by AQR, PQR, URX sensory neurons. Using the spatial environment module, we implement different O2 environment types, such as fast and slow rising O2 levels, and show that the framework can recapitulate avoidance responses for environments with high O2 levels as observed in *in-vivo*.

Complementing the spatial environment, the spatial-temporal neural stimulation module defines the neurons' encoding strategies governing the translation from environmental stimulus to neural stimulation, and the spatial locations of neurons relative to the body. We utilize this module to change the URX receptor type from tonic to phasic (i.e., responding to the derivative of environmental stimulus) to study its effects on simulated O2 responses. By surveying different URX sensitivity values to the derivative of environmental stimulus, we show that the baseline avoidance response observed with tonic URX persists with the addition of increased movement speeds.

Leveraging the spatial-temporal neural stimulation module also allows for the masking of neural stimulation for a selection of neurons. We use this capability to perform selective stimulation of AQR, PQR, and URX sensory neurons and elucidate individual sensory neuron contributions to the simulated avoidance response. The results show the importance of URX stimulation for facilitating the turn during avoidance, but they also indicate further strengthening of the turn rate with the co-stimulation of AQR or PQR. Through the inspection of the downstream interneurons for each stimulation scenario, we identify PVP, AVD, and AVA interneurons as potential neural pathways associated with co-stimulation that modulates turn reinforcement.

While this work focuses on the O2 aerotaxis response of *C. elegans*, mod-SenseWorm can be used to investigate other chemotaxis responses through configurations of spatial environments and spatial-temporal neural stimulation modules. Changing the environment types would only require redefining the 2D function of the spatial stimulus environment. Similarly, changing sensory neurons and their receptor properties associated with different environment stimuli can be achieved by modifying the neurons body mapping and environment to neural translation functions.

Using the modWorm as its base, mod-SenseWorm also allows for variations and extensions to its neuromechanical model. Updates to its nervous system, muscles, and biomechanics components can be achieved by simply modifying or exchanging their respective modules within the framework. Indeed, experiments identify additional dynamics such as extra-synaptic signaling (e.g., neuro-peptides, monoamines) and individual neuronal ion-channels that may play crucial roles in shaping chemotaxis behavior [88; 193; 149; 150; 25; 188]. Incorporating such additions into the model is expected to allow for more accurate simulations and systematic studies on the mechanisms of sensorimotor integration governing chemotaxis behavior.

Chapter 7

Deep Learning Method for Detailed Neuron Modeling

In this chapter, I partially adapt my previous work [JK2] to introduce ElectroPhysiomeGAN (EP-GAN), a deep generative method for estimating detailed neuron model parameters given the electrophysiological responses. EP-GAN can assist in extending the neuromechanical model by improving the simulation accuracy of intra-cellular neuron dynamics.

I first describe the architecture of EP-GAN by outlining its components: Encoder, Generator, and Discriminator, followed by their training process. We test EP-GAN by predicting parameters associated with Hodgkin-Huxley neuron models (HH-model) for both simulated and experimental neurons found in *C. elegans*. Furthermore, we perform ablation studies on the input data and EP-GAN components to test robustness against incomplete data and elucidate the contribution of individual model components in achieving accurate parameter predictions.

7.1 GAN Architecture for the Generation of Neuron Model Parameters

EP-GAN adopts the Generative Adversarial Network (GAN) architecture to translate the recorded neuronal responses into the neuron model parameters associated with Hodgkin-Huxley type model (HH-model). In this section, we lay out the motivation for choosing GAN as the base architecture, followed by a description of its components and the cost functions used for training them.

Deep Generative Model for Parameter Prediction. The method of estimating neuron model parameters from recorded neuron responses takes membrane potential responses and steady-state current profiles as input, and outputs a set of parameters associated with them in terms of simulating the inferred neuron model and comparing the simulated results with the inputs (Figure 7.1). EP-GAN adapts a deep generative model approach, specifically the Generative Adversarial Network (GAN), as its base architecture to implement such a method. The key advantage of GAN is its ability to generate artificial data that closely resembles real data. The generative nature of GAN is advantageous for addressing the one-to-many nature of our problem, where there are multiple parameter solutions for a given neuron recording. Indeed, several computational works attempting to solve the inverse HH-model noted the ill-posed nature of the parameter solutions [107; 113; 198; 199; 34; 200; 201; 202]. Our approach is therefore, to leverage GAN to learn a *domain of parameter sets* compatible with neuron recordings, instead of mapping directly onto a single solution. GAN consists of two separate networks, Generator and the Discriminator. The goal of the Generator is to generate outputs that are indistinguishable from real data, whereas the Discriminator's goal is to distinguish the generated outputs from the generator against the real data. Throughout training, the Generator and the Discriminator engage in a zero-sum game until they both converge to optimal states (i.e. Nash equilibrium) [203]. The particular architecture we use is Wasserstein GAN with gradient penalty (WGAN-GP), a variant of GAN architecture that offers more stable training and faster convergence [204].

Encoder Module. In addition to the Generator and Discriminator, we implement the Encoder module, which pre-processes the input membrane potential responses for the Generator and Discriminator (Figure 7.1 left). Specifically, the encoder serves two roles: i) compression of membrane potential responses traces along the stimulus space, thus reducing its dimension from 2-dimensional to 1-dimensional, and ii) translation of membrane potential responses traces into a latent space encoding a meaningful internal representation for the Discriminator and Generator. The Encoder module uses Gated Recurrent Units (GRU) architecture, a variant of Recurrent Neural Network (RNN) to perform this task [205]. Each input sequence to a GRU cell at step k corresponds to the entire membrane potential response of size 350 (i.e., 350 time points, representing $t = [4s, 11s]$) concatenated with the associated stimulus trace of equal size of 350. Since GRU is agnostic to the number of steps in input sequence, such an input structure allows EP-GAN to process a set of membrane

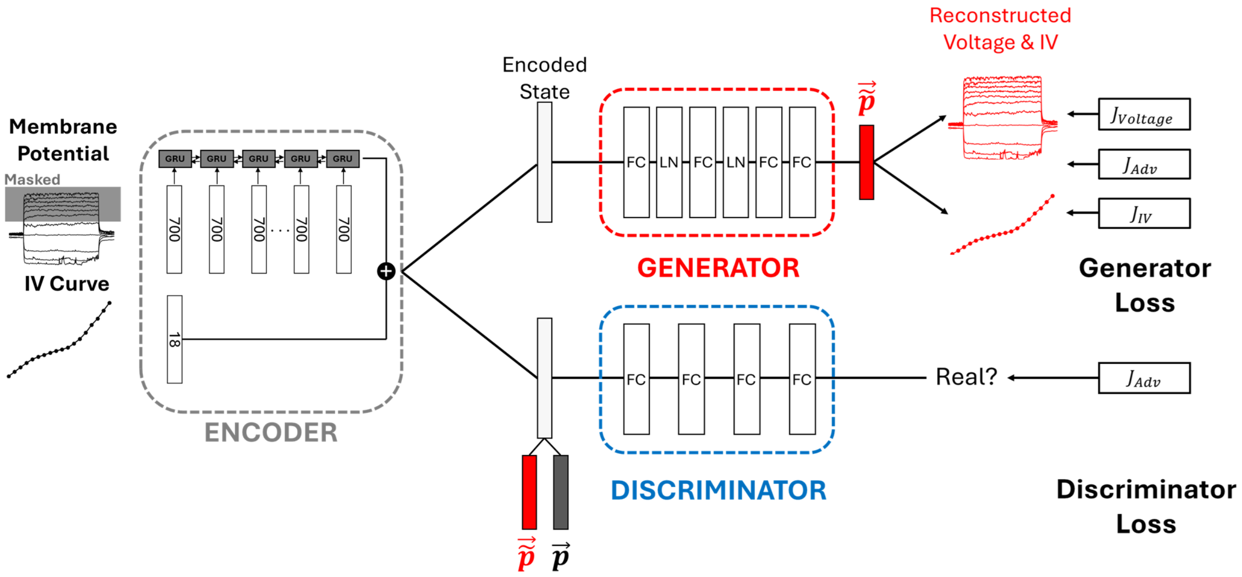


Figure 7.1 Architecture of EP-GAN. The architecture consists of an Encoder, Generator and Discriminator. Encoder compresses the membrane potential responses into a 1D vector (i.e., latent space) which is then concatenated with 1D steady-state current profile to be used as an input to both generator and discriminator. Generator translates the latent space vector into a vector of parameters $\vec{\tilde{p}}$ and the Discriminator outputs a scalar measuring the similarity between generated parameters ($\vec{\tilde{p}}$) and ground truths ($\vec{p}$). The Generator is trained with adversarial loss supplemented by reconstruction losses for both membrane potential responses and steady-state current profiles. The Discriminator is trained with discriminator adversarial loss only. Generator and Discriminator follow the architecture of Wasserstein GAN with gradient penalty (WGAN-GP) for more stable learning. During training, random masking is applied to input membrane potential responses which gradually decreases toward as the training continues.

potential traces with an arbitrary current-clamp protocol. The output of the Encoder is a latent space vector of size 1024, which encodes membrane potential responses information. The latent space vector is then concatenated with a 1D vector representing steady-state current profile, which is used as an input to both the Generator and Discriminator. During training, we randomly mask membrane potential traces to aid in better generalization in prediction [206; 207]. Specifically, we initially set the masking rate to 75% (i.e., 75% of membrane potential traces are randomly masked) and linearly decreases to 0% toward the end of training.

Discriminator Module. The goal of the Discriminator is given the input membrane potential responses and current profiles, to distinguish generated parameters from real ground truth parameters. The Discriminator receives as input the latent space vector from the Encoder, concatenated with generated or ground truth parameter vector, and outputs a scalar representing the relative distance between the two parameter sets (Figure 7.1, Eqs (7.1). Such a quantity is called Wasserstein distance or Wasserstein loss and differs from the vanilla GAN Discriminator, which only outputs 0 or 1. Wasserstein loss is known to remedy several common issues that arise from vanilla GAN such as vanishing gradient and mode collapse, leading to more stable training [204]. To further improve the training of WGAN architecture, we supplement Wasserstein loss with a gradient penalty term, which ensures that the gradients of the Discriminator's output with respect to the input have unit norms [208]. This condition is called Lipschitz continuity and prevents Discriminator outputs from having large variations when there are only small variations in the inputs [209; 208]. Combined together, the Discriminator is trained with the following loss:

$$J_D = \mathbb{E}[D(\vec{\tilde{\boldsymbol{p}}})] - \mathbb{E}[D(\vec{\boldsymbol{p}})] + \lambda \mathbb{E}[(||\nabla_{\hat{\boldsymbol{p}}} D(\hat{\boldsymbol{p}})||_2 - 1)^2] \tag{7.1}$$

where $\mathbb{E}[D(\vec{\tilde{\boldsymbol{p}}})]$ and $\mathbb{E}[D(\vec{\boldsymbol{p}})]$ are the mean values of Discriminator outputs with respect to generated samples $\vec{\tilde{\boldsymbol{p}}}$ and real samples $\vec{\boldsymbol{p}}$, respectively, and $\lambda \mathbb{E}[(||\nabla_{\hat{\boldsymbol{p}}} D(\hat{\boldsymbol{p}})||_2 - 1)^2]$ is the gradient penalty term modulated by λ where $\hat{\boldsymbol{p}} = t\vec{\tilde{\boldsymbol{p}}} + (1-t)\vec{\boldsymbol{p}}$ is the interpolation between generated and real samples with $0 \leq t \leq 1$.

Generator Module. As the adversary network of the Discriminator, the goal of the Generator is to fool the Discriminator by generating parameters that are indistinguishable from the real parameters. The Generator receives as input the concatenated vector from the Encoder and outputs a parameter vector (Figure 7.1). The module consists of 4 fully connected layers, with layer normalization applied after the first two layers for improved model convergence [210]. Each parameter in the output vector is scaled between -1 and 1, which is then scaled back to the parameters' original scales. The module is trained using 3 loss terms: i) Generator adversarial loss, ii) Membrane potential responses reconstruction loss, and iii) Steady-state current reconstruction loss as follows:

$$J_G = -\mathbb{E}[D(\vec{\tilde{\boldsymbol{p}}})] + J_{\vec{\boldsymbol{V}}} + J_{\vec{\boldsymbol{IV}}} \tag{7.2}$$

$$J_{\vec{\boldsymbol{V}}} = \sum_{i=1}^{n} |\vec{\boldsymbol{V}}_{groundtruth} - \vec{\boldsymbol{V}}_{reconstructed}| \tag{7.3}$$

$$J_{\vec{\boldsymbol{IV}}} = \sum_{i=1}^{n} |\vec{\boldsymbol{IV}}_{groundtruth} - \vec{\boldsymbol{IV}}_{reconstructed}| \tag{7.4}$$

where $-\mathbb{E}[D(\vec{\tilde{\boldsymbol{p}}})]$ is the Generator adversarial loss, which is the reciprocal of the mean Discriminator outputs with respect to generated samples, and $J_{\vec{\boldsymbol{V}}}$ and $J_{\vec{\boldsymbol{IV}}}$ are L_1 regression losses for reconstructed membrane potential responses and current profiles, respectively. It's important to note that $J_{\vec{\boldsymbol{V}}}$ and $J_{\boldsymbol{I}\vec{\boldsymbol{V}}}$ are part of the Generator's computation graph and thus force the Generator to optimize them on top of the adversarial loss (Figure 7.1). The composite loss function of the Generator makes EP-GAN a "model-informed" GAN, as the HH-model itself becomes part of the training process. Such networks have been shown to be more data efficient during training, as they do not rely solely on training data to learn an effective strategy [211; 212]. The mathematical description of membrane potential responses and current reconstruction from generated parameter set $\vec{\tilde{\boldsymbol{p}}}$ is as follows:

$$\vec{\boldsymbol{V}}_{reconstructed}(t) = \nabla^{-1}\left(\frac{d\vec{\boldsymbol{V}}}{dt}(\vec{\boldsymbol{V}}, t, \vec{\tilde{\boldsymbol{p}}})\right),\ \boldsymbol{I}\vec{\boldsymbol{V}}_{reconstructed}(V) = IV(\vec{\boldsymbol{V}}, \vec{\tilde{\boldsymbol{p}}}) \tag{7.5}$$

Here $\frac{d\vec{\boldsymbol{V}}}{dt}(\vec{\boldsymbol{V}}, t, \vec{\tilde{\boldsymbol{p}}})$ is the right-hand-side function of the HH-model, which computes the membrane potential responses derivative at time t given the membrane potential responses $\vec{\boldsymbol{V}}$ and parameter set $\vec{\tilde{\boldsymbol{p}}}$, and $IV(\vec{\boldsymbol{V}}, \vec{\tilde{\boldsymbol{p}}})$ is the function that evaluates the neuron's current I given the membrane potential states $\vec{\boldsymbol{V}}$ and the parameter set $\vec{\tilde{\boldsymbol{p}}}$. Membrane potential responses are reconstructed by first evaluating their derivatives with respect to the ground truth membrane potential responses and generated parameters $\vec{\tilde{\boldsymbol{p}}}$ at regularly sampled time points. This is followed by the forward integration operation ∇^{-1} similar to Euler's method, to approximate $\vec{\boldsymbol{V}}$ at sampled time points given the initial condition $\vec{\boldsymbol{V}}_{init}$:

$$\vec{\boldsymbol{V}}_{t+1} = \vec{\boldsymbol{V}}_t + h\vec{\boldsymbol{V}}'(t),\ \vec{\boldsymbol{V}}_{t=0} = \vec{\boldsymbol{V}}_{init} \tag{7.6}$$

where h is the time interval between sampled derivatives. $\vec{\boldsymbol{V}}_{init}$ can be selected at any time point within the ground truth membrane potential state (e.g., pre-activation, mid-activation, post-activation) to reconstruct different membrane potential features. Notably, all computation steps consisting of the forward integration process are expected to be differentiable. This is necessary to incorporate the forward integration process as part of the generator network, which requires full differentiability and is trainable with the back-propagation algorithm [213]. Computationally, we achieve this by manually implementing the forward integration process with discrete array operations that support auto-differentiation and vectorization (e.g. PyTorch Tensors) instead of simulating the membrane potential with ODE solvers [146]. The variable h can also be adjusted to increase the accuracy of the membrane potential responses reconstruction in exchange for an increased computational cost. The current profile is reconstructed by directly evaluating $IV(\vec{\boldsymbol{V}}, \vec{\boldsymbol{p}})$ which uses the generated parameter set $\vec{\tilde{\boldsymbol{p}}}$ over the range of membrane potential responses values. We show in Table 7.4 that reconstruction losses are essential for the accuracy of predicted parameters.

7.2 Generating Training Data for EP-GAN

EP-GAN is trained entirely using simulated data to learn the translation from recorded neuron responses to associated neuron model parameters. For the successful training of EP-GAN, the simulated data must be of a sufficient number of samples, denoised, and diverse. To ensure these conditions are met with a simulated dataset, we employ a three-step process for generating training data (Figure 7.2).

Step 1. Randomly generate parameter sets associated with the target HH-model by sampling each parameter within a pre-defined distribution. This distribution is a skewed normal distribution for channel conductance parameters and uniform distributions for other parameters. The range is determined according to the biologically feasible ranges reported in the literature [214; 215; 24]. In particular, search ranges for channel parameters are set to baselines $\pm 50\%$, where the baseline parameters are the default parameters given by the HH-models [28].

Step 2. Simulate steady-state current traces for each sampled parameter set, followed by imposing bifurcation structure constraints on each of them. This is done by calculating the first derivative of the currents dI/dV with respect to membrane potential states and ensuring they are within the 98% confidence intervals of experimentally obtained dI/dV bounds. Specifically, we evaluate each parameter set with respect to the 3 neuron types found within *C. elegans* non-spiking neurons - Type 1: Transient outward rectifier, Type 2: Outward rectifier type, and Type 3: Bistable type. During data generation, approximately the same number of neurons are generated for each type to ensure balance across all neuron types. The step can be further extended with new neuron types by incorporating additional dI/dV bounds.

Step 3. Impose (minimum, maximum) constraints on the membrane potential response across the current-clamp protocol. These values are set to (-100mV, 150mV) respectively for (-15pA, 35pA) current-clamp range. Parameter sets that do not satisfy the steady-state currents (Step 2) and membrane potential responses constraints are then removed from the training set. These constraints serve two purposes: i) remove parameter sets that result in non-realistic membrane potential responses/steady-state current profiles from the training

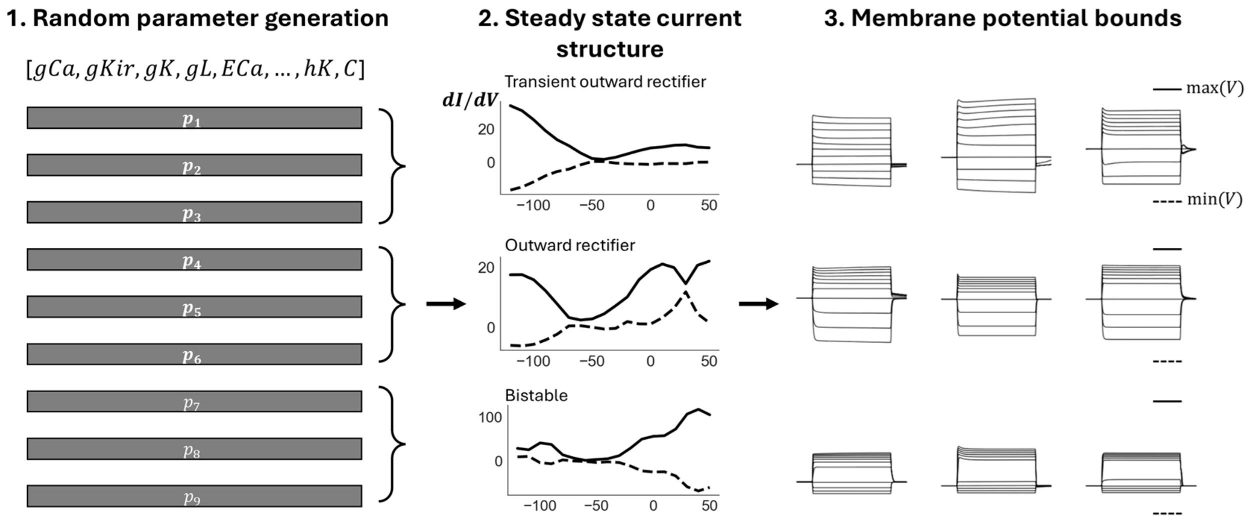


Figure 7.2 Training data generation. In Step 1, each parameter is initially sampled from biologically plausible ranges using both skewed gaussian (channel conductance) and uniform sampling. A parameter set consists of 175 parameters spanning 16 known ion channels in *C. elegans* and similar organisms. In Step 2 and 3, steady-state currents and membrane potential responses are evaluated for each set to ensure they satisfy the predefined constraints such as bifurcation structure represented by dI/dV bounds and minimum-maximum membrane potential across current-clamp protocols. Only parameter sets that meet both constraints are included in the training set.

set and ii) serve as an initial data augmentation process for the model. The constraints can also be extended or adjusted if deemed necessary for the improvement of the training process. Once constraints are applied, Gaussian noise is added to the membrane potential responses training data to mimic the measurement noise in the recorded experimental membrane potential responses data.

7.3 EP-GAN Testing Methodologies

We evaluate EP-GAN with respect to 4 existing evolutionary algorithms introduced for general parameter estimation: NSGA2, DEMO, GDE3, and NSDE. Specifically, NSGA2 is a variant of the Genetic Algorithm (GA) that uses a non-dominated sorting survival strategy and is a commonly used benchmark algorithm for multiobjective optimization problems, including HH model fitting [216; 217; 218]. DEMO, GDE3, and NSDE are variants of multi-objective differential evolution (DE) algorithms that combine DE mutation with pareto-based ranking and crowding distance sorting applied in NSGA2's survival strategy [219; 220; 221]. These methods have been proposed as more effective than direct DE for the estimation of HH-model parameters [111; 222; 223; 224]. In particular, DEMO has been successfully applied to estimate HH-model parameters

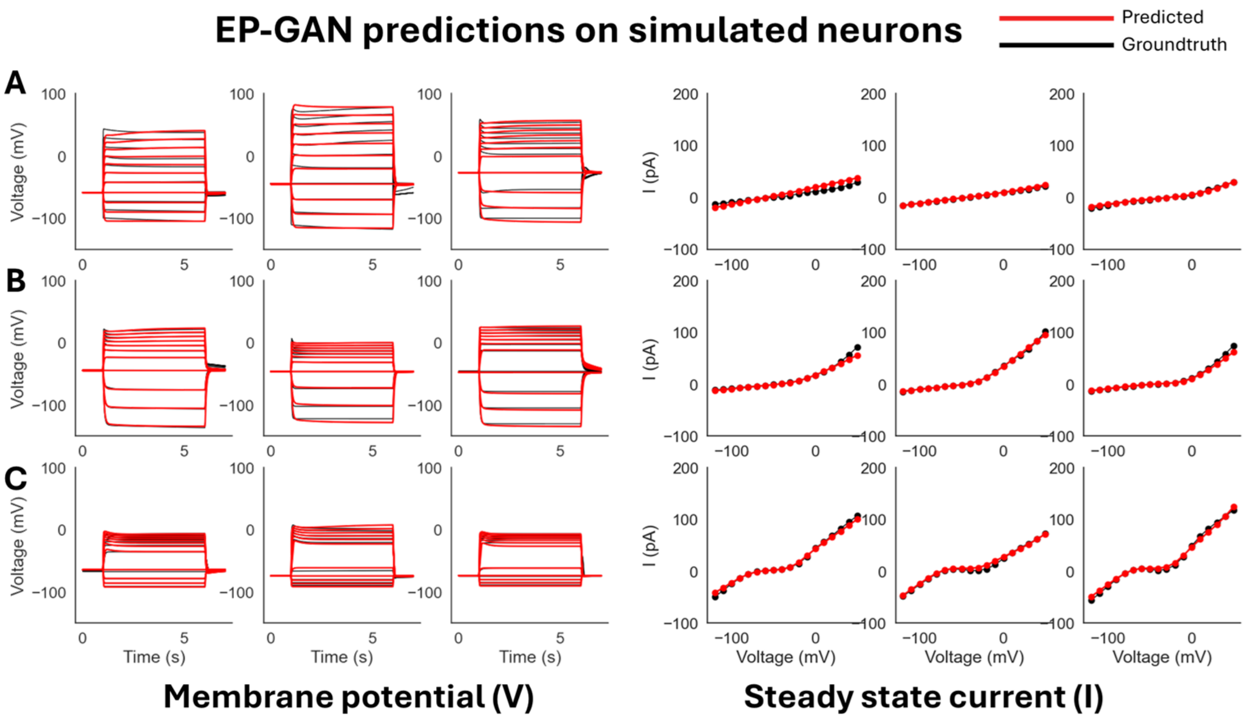


Figure 7.3 EP-GAN (32k) predictions on simulated neurons. **A**: EP-GAN predicted membrane potential traces and steady-state currents (Red) overlaid on top of groundtruth counterparts (black) for Transient outward rectifier neuron type. **B**: Outward rectifier neuron type. **C**: Bistable neuron type.

for non-spiking neurons in *C. elegans* [111]. All 4 methods support multi-objective optimization over a large parameter space, allowing them to have a similar setup as EP-GAN. All 4 methods were implemented in Python, where DEMO uses the algorithm proposed in [111] whereas NSGA2, GDE3 and NSDE were implemented using Pymoo package [225].

For the HH-model to be estimated, we use the formulation introduced in [27; 28]. The model features 16 ion channels that were found in *C. elegans* and other organisms expressing homologous channels, and it is considered the most detailed neuron model for the organism. The model has a total of **203** parameters, of which we identify 175 that have approximate ranges with lower and upper bounds that can be inferred from the literature [214; 215; 24]. We thus target these 175 parameters as trainable parameters for all methods.

Neuron	Duration	Current-clamp (min:step:max)	Stimulation period	Voltage-clamp (min:step:max)
Simulated	15s	-15pA:5pA:35pA	5s - 10s	-120mV:10mV:50mV
RIM	15s	-15pA:5pA:35pA	5s - 10s	-120mV:10mV:50mV
DVC	15s	-2pA:1pA:8pA	5s - 10s	-120mV:10mV:50mV
HSN	15s	-2pA:1pA:8pA	5s - 10s	-120mV:10mV:50mV
AIY	15s	-15pA:5pA:35pA	5s - 10s	-120mV:10mV:50mV
URX	15s	-4pA:2pA:16pA	5s - 10s	-120mV:10mV:50mV
RIS	15s	-4pA:2pA:16pA	5s - 10s	-120mV:10mV:50mV
AFD	15s	-15pA:5pA:35pA	5s - 10s	-120mV:10mV:50mV
AWB	15s	-4pA:2pA:16pA	5s - 10s	-120mV:10mV:50mV
AWC	15s	-4pA:2pA:16pA	5s - 10s	-120mV:10mV:50mV

Table 7.1 Simulation protocols for simulated and experimental neurons.

7.4 EP-GAN Predictions on Simulated Neurons

We first validate EP-GAN by training and testing using the simulated neurons. Each simulated neuron training sample consists of two inputs: i.) Simulated membrane potential traces concatenated with associated external stimuli traces and ii.) Steady-state currents across 18 membrane potential points. For each neuron, the output is the set of 175 HH-parameters to be inferred. Each membrane potential trace is simulated for 15 seconds for a given stimulus according to the current-clamp protocol, where the stimulation is applied for 5 seconds at [5, 10] seconds, and no stimulation is applied at $t = [0, 5)$ and $t = (10, 15]$. These time intervals are chosen to ensure sufficient stabilization periods before and after stimulation. For the membrane potential input, the responses during the $t = (4, 11)$ interval are used, consistent with the time interval used by the experimental recordings. Similarly, steady-state currents are computed across 18 membrane potential states according to the voltage-clamp protocol (see Table 7.1 for detailed current/membrane potential clamp protocols used for simulated neurons). The output HH-parameters are of the simulated neurons chosen randomly from lower and upper bounds, as previously described. For training EP-GAN, we simulate a total of 32,000 (32k) neurons, where EP-GAN achieves both good predictive performance and training time. Specifically, 32k is a training data size in which membrane potential errors from the test set are within the mean RMSE recording error ($\sim$ 4.8mV, averaged over pre-, mid-, post-activation periods) obtained from experimental neurons with multiple membrane potential recording data.

To initially test EP-GAN performance, we evaluated EP-GAN predicted parameters for 200 simulated neurons outside of the training set (test set). To emulate *C. elegans*' neuronal diversity, neurons in the test set are divided into 3 response types - i) Transient outward rectifier type, ii) Outward rectifier type, and iii) Bistable type - that are currently found in non-spiking neurons of *C. elegans* according to their steady-state responses (Figure 7.3) [111; 24]. For a given neuron being evaluated, EP-GAN predicted HH-parameters are obtained as follows: For each training epoch, EP-GAN generates a set of HH-parameters for the neuron, and at the end of the training, the parameter set that achieved the lowest Root Mean Square Error (RMSE) of membrane potential responses averaged across three time intervals - pre-activation (4, 5] seconds, mid-activation [5, 10] seconds, post-activation [10, 11) seconds - with respect to ground truths is reported (detailed descriptions of its calculation are provided in Figure 7.6A). In the case of multiple N-neurons being evaluated, the same procedure is followed, except EP-GAN generates N-parameter sets in parallel at each epoch. Such multi-inference is possible due to EP-GAN being a neural network, where the parallel processing of inputs can be done with minimal impact on inference speed. Using these procedures, EP-GAN predicted HH-parameters result in a mean membrane potential RMSE error of **2.37mV** for the test set (See Figure 7.3 for representative samples). These errors are within the mean recording RMSE error of 4.8mV obtained from experimental neurons, and their distributions were skewed uni-modal type, where the majority of the errors fall within 4mV.

7.5 EP-GAN Predictions on Experimental Neurons

We apply EP-GAN trained and tested on simulated data to predict HH-parameters for 9 experimentally recorded non-spiking neurons found in *C. elegans* - RIM, AFD, AIY, AWB, AWC, URX, RIS, DVC, HSN. Among these neurons, AWB, AWC, URX, RIS, DVC, HSN are novel recording data that were not previously modeled, whereas RIM, AIY and AFD neurons are publicly available, and their modeling descriptions were elaborated in previous works [111; 214; 226]. Similar to prediction scenario on simulated neurons, we categorize experimental neurons into different response types according to their steady-state current responses. In particular, we classify (RIM, DVN, HSN) as transient outward rectifier type, (AIY, URX, RIS) as Outward rectifier type, and (AFD, AWB, AWC) as Bistable type.

We compare the performance of EP-GAN with 4 existing parameter inference methods: NSGA2, DEMO, GDE3, and NSDE. Unlike EP-GAN which can optimize multiple neurons in parallel, these are evolutionary methods where the optimization is done **from scratch** for each neuron. We therefore evaluate their respective performances relative to EP-GAN by normalizing on the *total number of simulated neuron samples* during the entire optimization task. Specifically, for all methods, we set the maximum number of neuron samples used during optimization to equal sizes. e.g., if EP-GAN is trained with 32k neuron samples to predict 9 neurons, NSGA2, DEMO, GDE3, and NSDE are each allocated up to 3.5k samples during the search phase of HH-parameters for each neuron, thus adding up to a total of $\sim$32k samples for all 9 neurons. To test how the performance of each method scales with the size of samples during optimization, we evaluate each method with both 32k and 64k total neuron samples.

The parameter selection process for NSGA2, DEMO, GDE3 and NSDE is as follows: During the search phase for each neuron, the parameter set candidates (i.e., population) are recorded at each iteration. At the end of the search phase, steady-state current profile of each parameter set candidate is evaluated with respect to the experimentally known bifurcation structure (i.e., dI/dV) of the neuron being inferred (e.g., bistable type). Upon evaluation, only the parameter sets satisfying the dI/dV bound constraints ($\sim 98\%$ confidence interval) are kept. Such an initial selection process is similar to those employed by previous methods utilizing evolutionary algorithms and is partially adopted during EP-GAN training data generation, as outlined in section 8.2 [111]. The final parameter set is then chosen by selecting the one with the lowest RMSE membrane potential responses error, averaged across pre-, mid- and post-activation periods, identical to that of EP-GAN parameter selection process. For Multi-objective DE methods (DEMO, GDE3 and NSDE2), we follow the same configurations used in the literature to set their optimization scheme [111]. Specifically, we set the crossover parameter CR and scale factor F to 0.3 and 1.5 respectively. For all 4 methods, NP (population size) is set to 600 with a total of 6 and 12 iterations (i.e., $\sim 3.6k$ and $\sim 6.2k$ samples per neuron for 32k and 64k total neuron samples respectively). For all methods, loss functions identical to those used for EP-GAN training (mean absolute errors) are used to calculate the errors for membrane potential responses and steady-state currents for multi-objective optimization.

Small HH-model scenarios (47 parameters). We first test EP-GAN and existing methods with a "smaller" version of HH-model of 47 parameters, where the individual channel parameters ($n = 129$)

Method	Sample size	Median Error	RIM	DVC	HSN	AIY	URX	RIS	AFD	AWB	AWC
GDE3	32k	16.9mV	20.9	58.1	8.9	13.0	16.9	25.4	6.3	32.5	4.7
		5.9pA	5.8	5.8	19.8	4.8	2.4	5.9	19.1	7.2	11.9
	64k	6.0mV	11.5	24.1	13.7	6.0	5.1	7.1	4.1	5.6	3.6
		7.9pA	7.5	4.6	6.8	7.2	18.7	7.9	42.1	17.6	46.7
NSDE	32k	7.1mV	38.7	8.3	20.2	5.7	7.1	11.0	5.5	6.3	6.6
		13.6pA	3.1	21.5	9.7	13.6	14.2	5.7	24.7	9.6	14.5
	64k	5.5mV	15.4	8.7	20.2	13.5	5.2	5.5	4.9	4.9	4.0
		9.7pA	2.6	5.9	9.7	3.4	11.7	3.2	64.4	12.4	19.0
DEMO	32k	6.7mV	35.9	14.1	5.8	13.2	9.0	6.7	5.0	4.9	3.8
		11.5pA	6.5	13.8	14.6	5.2	5.4	9.7	23.8	11.5	18.7
	64k	4.8mV	12.3	10.5	5.8	10.2	4.8	4.7	**3.1**	4.4	2.9
		14.6pA	4.4	6.5	14.6	4.1	15.2	10.3	41.0	23.1	17.8
NSGA2	32k	7.5mV	12.4	15.4	2.6	9.8	6.1	6.4	7.5	9.4	6.6
		10pA	4.0	7.4	14.3	4.6	16.6	5.0	14.4	11.9	10.0
	64k	4.3mV	10.5	19.0	5.2	13.3	3.5	3.2	4.2	4.3	3.1
		8.4pA	8.4	1.8	5.2	2.3	21.2	6.5	26.6	12.6	49.4
EPGAN (Ours)	32k	2.5mV	3.4	2.4	1.6	2.5	3.2	1.7	4.9	2.5	**2.0**
		13.8pA	4.0	13.8	10.3	10.7	38.9	16.8	48.0	9.6	28.9
	64k	**2.4mV**	**3.4**	**2.4**	**1.6**	**2.4**	**2.9**	**1.4**	3.4	**2.5**	2.1
		13.1pA	3.2	13.9	2.5	9.8	16.5	13.1	49.7	11.6	27.9

Table 7.2 Small HH-model scenarios RMSE errors for predicted membrane potential responses and steady-state currents. Each method is tested with 32k or 64k total sample sizes where the top row shows membrane potential responses RMSE errors averaged across pre-activation, mid-activation, post-activation periods and the bottom row shows steady-state currents RMSE errors across 18 voltage values.

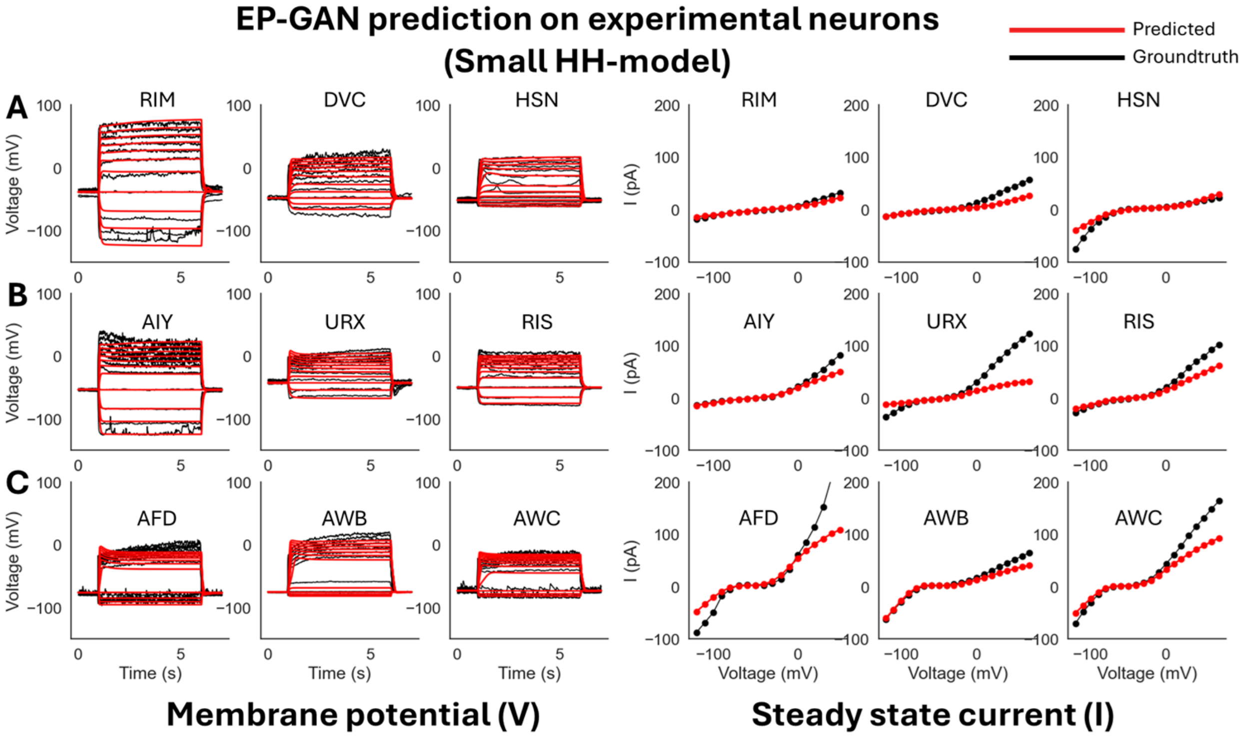


Figure 7.4 EP-GAN (32k) Prediction on experimental neurons (small HH-model). **A**: EP-GAN predicted membrane potential traces and steady-state currents (Red) overlaid on top of groundtruth counterparts (black) for Transient outward rectifier neuron type (RIM, DVC, HSN). **B**: Outward rectifier neuron type (AIY, URX, RIS). **C**: Bistable neuron type (AFD, AWB, AWC).

are frozen to the default values given by [28]. The considered parameters consist of 16 conductance parameters of each channel (g_{Ch}), 4 reversal potentials for calcium, potassium, sodium and leak channels (V_{Ca}, V_K, V_{Na}, V_L), 1 cell capacitance C, and 26 initial conditions for membrane potential V_0 and channel activation variables (m_0, h_0). Such a parameter set is commonly targeted when fitting HH-models for individual neurons [27; 28]. For all methods, we test both 32k and 64k total sample sizes for the inference of 9 experimental neurons. Figure 7.4 illustrates that EP-GAN can reconstruct membrane potential responses close to ground truth responses. Indeed, upon inspecting the RMSE error for membrane potential responses (averaged over pre-, mid- and post-activation), EP-GAN (32k) median error of $2.5mV$ over 9 neurons is $\sim 50\%$ lower than that of NSGA2 (64k) $4.3mV$ followed by DEMO (64k) $4.8mV$, NSDE (64k) $5.5mV$ and GDE3 (64k) $6.0mV$. Among all 9 neurons being inferred, EP-GAN (32k) showed the best accuracy for HSN with $1.6mV$ and the lowest accuracy for AFD with $4.9mV$. With EP-GAN (64k), the median error further decreased ($2.5mV \rightarrow 2.4mV$) where URX and RIS errors improved by $0.3mV$ and AFD error improved by

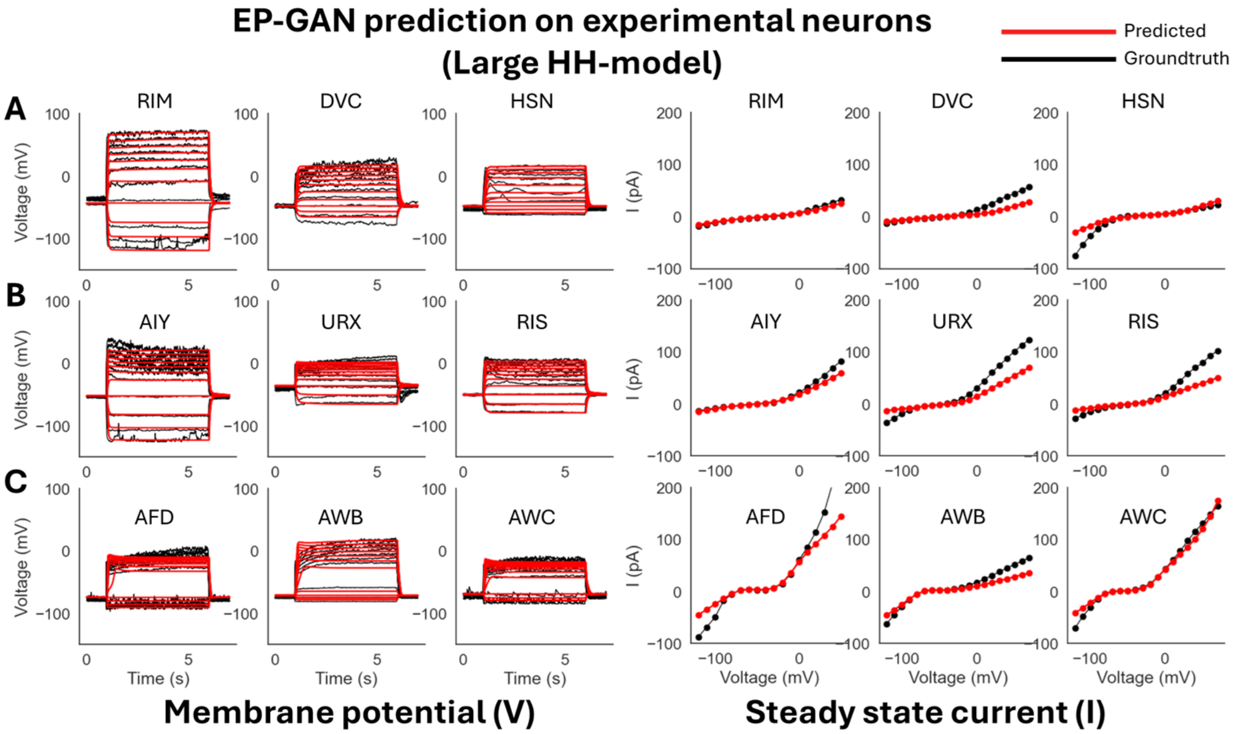


Figure 7.5 EP-GAN (32k) Prediction on experimental neurons (large HH-model). **A**: EP-GAN predicted membrane potential traces and steady-state currents (Red) overlaid on top of groundtruth counterparts (black) for Transient outward rectifier neuron type (RIM, DVC, HSN). **B**: Outward rectifier neuron type (AIY, URX, RIS). **C**: Bistable neuron type (AFD, AWB, AWC).

$1.5mV$ over their 32k counterparts. Interestingly, we note that the high accuracy of EP-GAN in predicting membrane potential is not necessarily complemented with steady-state currents (Figure 7.6B). EP-GAN's overall steady-state currents errors are generally higher than those of existing methods. A possible reason could be that the majority of these errors stem from lower and upper membrane potential ranges, where the recording variations are high and thus could conflict with membrane potential responses optimization. Such a competitive nature between membrane potential vs steady-state current optimizations has indeed been reported in previous works [111].

Large HH-model scenarios (175 parameters). We expand the domain of parameters being inferred by testing with respect to all 175 HH-model's trainable parameters including the 47 parameters from the previous scenario + 129 channel parameters. The inclusion of channel parameters allows methods to further fine tune the HH-model. The minimum and maximum values for channel parameters are set to $\pm 50\%$ from their default values. Such a task introduces further challenges as the methods need to estimate $\times 3$ more parameters

Method	Sample size	Median Error	RIM	DVC	HSN	AIY	URX	RIS	AFD	AWB	AWC
GDE3	32k	12.8mV	14.0	12.5	12.8	19.4	9.0	15.2	29.4	10.8	9.2
		9.6pA	3.2	23.5	12.8	10.2	6.3	6.0	7.9	18.1	9.6
	64k	10.5mV	14.0	11.0	10.5	11.7	12.4	9.0	5.0	9.5	4.7
		4.9pA	3.2	6.5	7.4	3.8	4.8	4.0	16.9	14.8	4.9
NSDE	32k	16.1mV	31.5	19.0	8.7	12.1	8.6	23.8	9.8	27.1	16.1
		7.2pA	7.8	8.0	9.6	6.8	5.1	4.2	18.2	7.2	6.2
	64k	8.6mV	33.9	8.6	12.4	5.9	8.6	8.0	16.6	12.5	4.6
		8.1pA	4.0	29.6	8.1	4.7	5.1	4.3	9.1	15.0	14.9
DEMO	32k	16.6mV	28.0	17.8	16.6	10.2	22.9	18.1	6.0	13.1	5.1
		11.9pA	7.6	11.9	21.4	7.2	11.1	6.8	18.2	11.9	30.9
	64k	10.5mV	10.5	29.5	11.0	10.2	6.8	18.4	6.0	13.1	4.4
		8.0pA	7.4	2.6	21.3	7.2	8.0	7.0	51.1	11.9	9.8
NSGA2	32k	13.4mV	13.4	16.1	29.2	11.3	8.6	13.5	8.2	11.2	13.6
		7.6pA	6.6	7.6	5.8	8.7	5.1	2.7	24.1	10.9	7.6
	64k	7.5mV	10.6	16.0	22.5	7.5	4.6	13.4	5.4	6.9	6.6
		4.9pA	4.9	3.2	3.7	1.2	9.5	3.1	24.7	13.7	6.9
EPGAN (Ours)	32k	2.7mV	3.2	**2.5**	3.0	2.7	**3.0**	1.8	4.5	2.6	**2.1**
		12.4pA	3.2	12.4	17.4	10.5	36.6	21.9	43.2	9.3	8.4
	64k	**2.6mV**	**3.2**	2.9	**2.5**	**2.6**	3.4	**1.8**	**4.4**	**2.6**	2.2
		8.6pA	3.6	8.6	3.3	4.9	37.9	9.2	31.8	5.1	10.2

Table 7.3 Large HH-model scenarios RMSE errors for predicted membrane potential responses and steady-state currents. Each method is tested with 32k or 64k total sample sizes where the top row shows membrane potential responses RMSE errors averaged across pre-activation, mid-activation, post-activation periods and the bottom row shows steady-state currents RMSE errors across 18 voltage values.

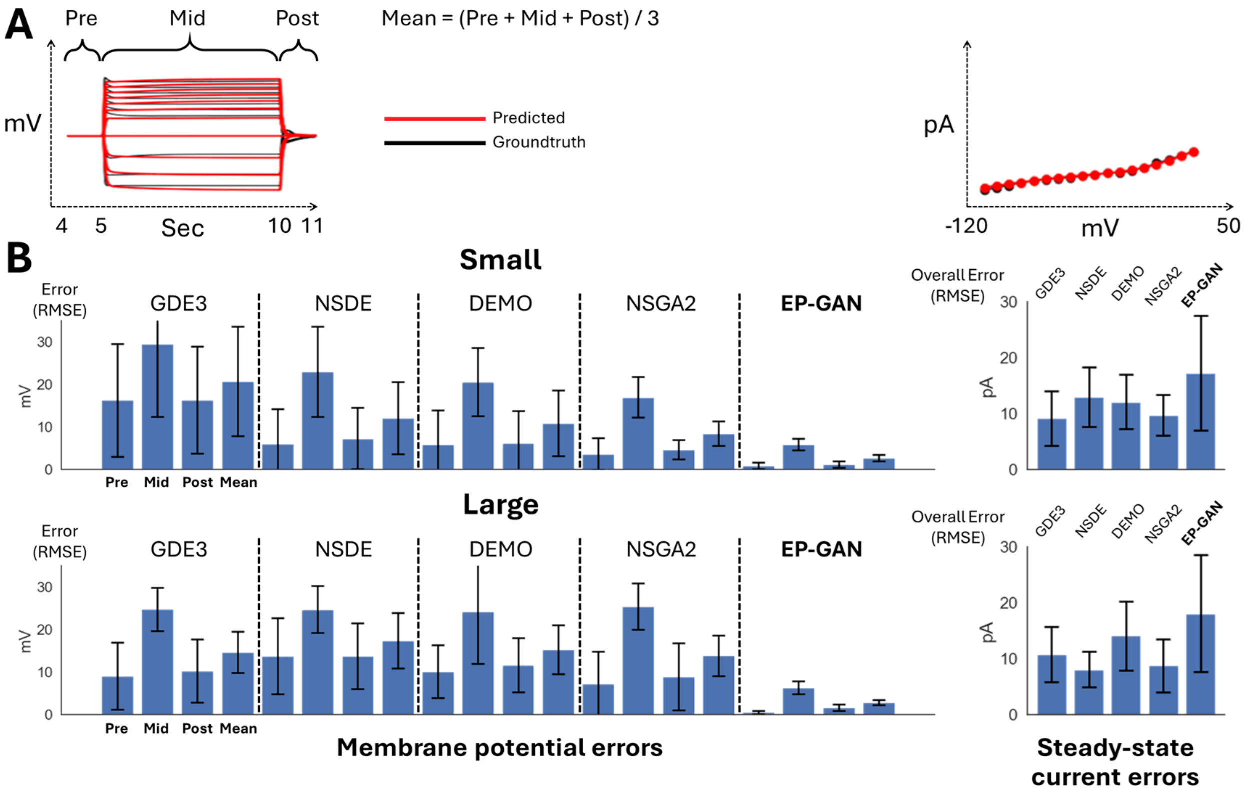


Figure 7.6 Bar plot showing the mean RMSE errors for membrane potential responses (pre-, mid-, post-activation periods, averaged error) and steady-state currents for 9 experimental neurons. A: Membrane potential responses (left) and steady-state currents (right) diagrams showing the time and membrane potential intervals of which the RMSE errors are computed. **B**: Bar plots showing RMSE errors for membrane potential responses and steady-state currents for small HH-model scenarios (Top) and Large HH-model prediction scenarios (Bottom). All methods use 32k sample size for both scenarios.

compared to the small HH-model scenario. From Figure 7.5, Figure 7.6B, we see that while EP-GAN (32k) median membrane potential error increases slightly from $2.5mV$ to $2.7mV$, its performance gaps over existing methods widen with $\sim 60\%$ lower error than NSGA2 (64k) $7.5mV$, followed by NSDE (64k) $8.6mV$ and DEMO (64k), GDE3 (64k) $10.5mV$. EP-GAN trained for large HH-model also slightly improved the overall steady-state current error ($13.8pA \rightarrow 12.4pA$) alongside membrane potential errors for RIM ($3.4mV \rightarrow 3.2mV$) and URX ($3.2mV \rightarrow 3.0mV$) indicating different selectivity for individual neurons for small vs large HH-model. Further increasing the sample size to 64k improved median errors of both membrane potential ($2.7mV \rightarrow 2.6mV$) and steady-states responses ($12.4pA \rightarrow 8.6pA$). Taken together, these results show EP-GAN predicting capabilities for HH-parameters with higher membrane potential responses accuracy and its ability to generalize to a larger parameter space with minimal performance loss.

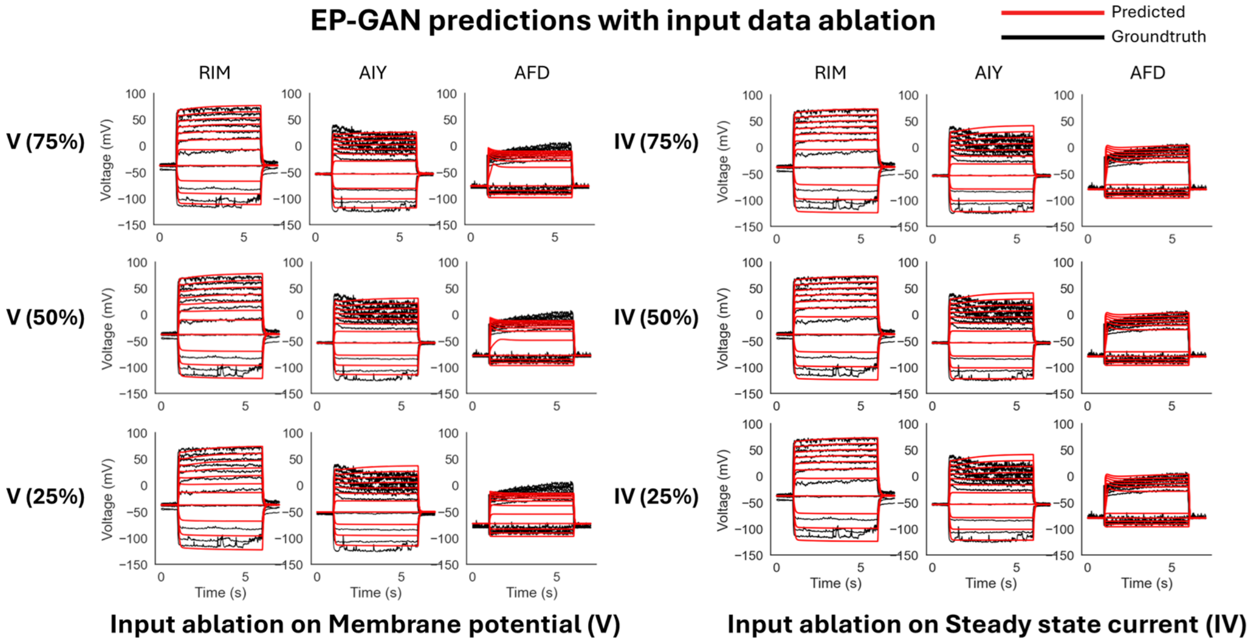


Figure 7.7 Input data ablation on EP-GAN (32k). **Left**: Reconstructed membrane potential responses for RIM, AIY and AFD when given with incomplete membrane potential responses data. Percentages in parenthesis represent the remaining portion of input membrane potential responses trajectories. **Right**: Reconstructed membrane potential responses for RIM, AIY, and AFD when given with incomplete steady-state current input.

7.6 Ablation Studies

EP-GAN architecture also allows its membrane potential inputs to have an arbitrary current-clamp protocol due to its RNN encoder component. To test the robustness of EP-GAN when incomplete input data is given, we provide the model with membrane potential responses and steady-state current inputs with missing data points. For each membrane potential responses and current profile, the data is reduced by 25%, 50%, and 75% each. For membrane potential responses data, the ablation is done in the stimulus space where a 50% reduction corresponds to removing the upper half of the membrane potential responses traces, each associated with a stimulus. For the steady-state current profile, we remove the first n-data points, which are instead extrapolated using linear interpolation with existing data points.

Our results show that EP-GAN largely preserves accuracy even when both membrane potential and steady-state current inputs are masked (Figure 7.7, Table 7.4). In particular, EP-GAN preserves the median membrane potential error ($3.3mV$) up to 50% of its input remaining, but becomes less accurate when up to 25% input remains ($3.3mV \rightarrow 5.4mV$). Surprisingly, AFD neuron membrane potential error improves when

Input Data Ablation	**Sample size**	**Median Error**	**RIM**	**AIY**	**AFD**
EP-GAN (25% membrane potential)	32k	5.4mV	3.8	5.4	8.9
		14.9pA	2.8	14.9	34.4
EP-GAN (75% steady-state)	32k	3.5mV	3.3	3.5	5.1
		15.6pA	3.7	15.6	68.9
EP-GAN (50% steady-state)	32k	3.5mV	3.3	3.5	5.1
		15.5pA	3.7	15.5	68.9
EP-GAN (25% steady-state)	32k	3.5mV	3.3	3.5	5.1
		15.6pA	3.7	15.6	68.9
EP-GAN (75% membrane potential)	32k	3.4mV	3.4	2.7	4.9
		11.3pA	3.6	11.3	44.0
EP-GAN (Full)	32k	3.3mV	3.3	2.7	5.2
		10.5pA	3.2	10.5	39.5
EP-GAN (50% membrane potential)	32k	3.3mV	3.3	2.9	4.5
		13.5pA	3.4	13.5	43.2
Loss Ablation	**Sample size**	**Median Error**	**RIM**	**AIY**	**AFD**
EP-GAN (Adv)	32k	14.4mV	14.4	6.1	24.5
		20.3pA	3.1	20.3	75.4
EP-GAN (Adv + Steady state)	32k	6.0mV	5.7	6.0	3.9
		19.1pA	2.8	19.1	23.1
EP-GAN (Adv + Steady state + Membrane potential)	32k	3.3mV	3.3	2.7	5.2
		10.5pA	3.2	10.5	39.5

Table 7.4 Ablation studies. **Top**: membrane potential responses and steady-state current errors achieved for EP-GAN (32k) when provided with incomplete input data. **Bottom**: membrane potential responses and steady-state current errors achieved for EP-GAN upon using only adversarial loss (Adv) and using adversarial + current reconstruction loss (Adv + Steady state) and all three loss components (Adv + Steady state + Membrane potential)

only 50% of input data is considered ($5.2mV \rightarrow 4.5mV$). These results could be attributed to the random input masking employed during EP-GAN training, which allows EP-GAN to make robust predictions even when conditioned with varying degrees of masked inputs.

We also performed ablation studies on model architecture by removing each loss component of the Generator module, allowing us to evaluate the relative contribution of each loss to accuracy. From Table 7.4 bottom, we see that removing the membrane potential loss term (V) results in a loss of performance for RIM and AIY, but an increase in accuracy for AFD. The result is consistent with input data ablation scenarios, indicating AFD's higher dependence on steady-state responses for a good EP-GAN prediction. Upon removing the steady-state current reconstruction loss term (IV) in addition to the membrane potential reconstruction loss, we see a further reduction in overall performance. These results highlight the significance of the reconstruction losses in aligning the Generator to produce the desired outputs.

7.7 Parameter Inference Time

We also evaluate EP-GAN for its scalability by assessing its overall inference time and computational cost, and comparing these to existing methods. Indeed, for estimation tasks involving many neurons, it is essential that the method is scalable so that the predictions are done within a reasonable time. In particular, for EP-GAN, the total time T needed for modeling N neurons, including data generation and training time, can be written as

$$T(N) \sim T_{Data\,generation} + T_{Train} + T_{Inference}$$

, whereas for existing methods, the $T(N)$ follows the form

$$T(N) \sim N \cdot T_{Inference}$$

, which increases linearly as N increases. Since $T_{Inference}$ for EP-GAN is nearly instantaneous, it has a strong advantage in the parameter prediction task involving multiple neurons. As an example, given a hypothetical task of modeling all 279 somatic neurons in the *C. elegans* nervous system, it would take DEMO, GDE3, NSDE or NSGA2 more than 44 days (assuming 7.2k samples per neuron and our available computing environment), whereas for EP-GAN, the process would be done within a day under a similar training setup. For a larger number of neurons, the computational requirements of existing methods would grow linearly, while EP-GAN would require constant time to complete the inference. Such scalability largely benefits from the neural network being an inherently parallel architecture, allowing it to take multiple neuron profiles and output corresponding parameters in a single forward pass [227].

7.8 Discussion

In this chapter, I introduced a novel deep generative method and system called ElectroPhysiomeGAN (EP-GAN), for estimating Hodgkin-Huxley model (HH-model) parameters given the recordings of neurons with graded membrane potential (non-spiking). The proposed system encompasses the RNN Encoder layer to process the neural recordings information, such as membrane potential responses and steady-state current profiles, and the Generator layer to generate a large number of HH-model parameters ($N > 175$). The system can be trained entirely on simulation data informed by an arbitrary HH-model. When applied to neurons in *C. elegans*, EP-GAN generates the parameters of HH-model, which membrane potential responses are closer to ground truth responses than existing methods, such as Differential Evolution and Genetic Algorithms. The advantage of EP-GAN lies in the accuracy and inference speed achieved through fewer training samples than existing methods, and it is generic in that it does not depend on the number of neurons for which inference is to be performed [27; 28; 111]. In addition, the method largely preserves performance when provided with input data with partial information, such as missing membrane potential responses (up to 50% missing) or steady-state current traces (up to 75% missing).

While EP-GAN is a step forward toward the ElectroPhysiome model of *C. elegans*, its inability to support neurons with spiking membrane potential responses remains a limitation. The reason stems from the fact that neurons with spiking membrane potential responses are rare during the generation of training data of 16 ionic channels HH-model without the spike associated neuron channels. The relative sparsity on spiking responses makes their translation strategies to parameter space difficult to learn. A similar limitation is present with bi-stable membrane potential responses, e.g., AFD, AWB and AWC, although to a lesser extent. While the limitations for these profiles can be partially remedied through more training samples of their neuron types, their relative sparseness in the training data tends to result in lower quality for the predicted parameters. Indeed, previous studies of *C. elegans* nervous system found that the majority of neurons exhibit graded membrane potential response instead of spiking [27; 49]. Furthermore, the limitation could lie within the current architecture of EP-GAN, as it processes data directly without a component that discerns and processes spiking membrane potential responses. Improving the sampling strategy for training data alongside the enhancement of the network architecture could address these limitations in the future.

As discussed in section 7.1, it is worth noting that EP-GAN does not necessarily recover the ground truth parameters associated with the input membrane potential responses and steady-state current profiles. This is mainly due to the fact that there may exist multiple parameter regimes for the HH-model that support the given inputs [107; 113; 198; 199; 34; 200; 201; 202]. The parameters generated by a single forward pass of EP-GAN (i.e., a single flow of information from the input to the output) could thus be interpreted as a one-time sampling from such a regime, and a perturbation to the input may result in a different set of parameters. Such sensitivity to perturbation could be adjusted by supplementing the training samples or inputs with additional recording data (e.g., multiple recordings per neuron).

EP-GAN allows additional modifications to accommodate different configurations of the problem. For instance, an update to the HH-model would only require retraining the network without changes to its architecture. Indeed, neuronal genome of *C. elegans* indicates additional voltage-gated channels that could be further incorporated into the HH-models introduced in [27; 28] to improve their modeling accuracy of membrane potential dynamics [228]. Extending the inputs to include additional data, e.g., channel activation profiles, can also be done in a straightforward manner by concatenating them to the input vectors of the Encoder network. Extending EP-GAN prediction capabilities to new neuron types can also be achieved by incorporating additional constraints during the generation of training data.

Despite its primary focus on *C. elegans* neurons, EP-GAN and its future extensions could be viable for modeling a variety of neurons in other organisms. Indeed, there are increasing advances in resolving the connectomes of more complex organisms and techniques for recording large-scale neural activities [229; 230; 231; 232]. As neurons in these organisms can be described by a generic HH-model or a similar differential equation model, EP-GAN is expected to be applicable and contribute to the development of biologically detailed nervous system models of neurons in these organisms.

Chapter 8

DeepRL Control to Inform Additional Neural Mechanisms

In this chapter, I adapt my research published in [JK5] to describe deep reinforcement learning (DeepRL) methods for achieving control of neural systems. The control method is sought for modulating dynamic neural stimulation (neuromodulation control) and the structure of connectome mappings (connectome control) which achieve target dynamics (e.g., empirically observed behavior).

I target the olfactory circuit of *C. elegans* modulated by AWC sensory neurons to demonstrate the performance of neuromodulation and connectome control strategies. In particular, I show that strategies can infer potential neuropeptidic currents between neurons that are not included in the initial neuromechanical model and the perturbations of connectome mappings for a sensory circuit to achieve empirically observed behaviors (attraction and aversion) in response to environmental stimuli.

8.1 Reinforcement Learning and Deep Q-Learning

In this section, we lay down background knowledge related to reinforcement learning and its deep learning adaptation, Deep Q-Learning, which forms the basis for the proposed neuromodulation and connectome control methods.

Agents in Reinforcement Learning. The goal of the agent in reinforcement learning is to learn the best *action* given the *environment* to maximize the *scalar rewards*. This interaction of the agent and the environment is formalized as a Markov Decision Process (MDP), or as a tuple $[S, A, T, r, \gamma]$ governed by

$$T(s, a, s') = P[S_{t+1} = s' | S_t = s, A_t = a] \tag{8.1}$$

where S is the finite set of states, A is the finite set of actions, and T is the stochastic transition function from state s to s' given the action a. $r(s, a)$ is the scalar reward function obtained at time $t + 1$ given the state s and action a with a discount factor $\gamma \in [0, 1]$. For each state S_t at time t, the selection of the action is given by a policy $\pi(s, a)$, which maps each state to a particular action. The discounted reward is defined as $R_t = \sum_{k=t}^{T} \gamma^{k-t} r_k(s_k, a_k)$ and corresponds to the discounted sum of future rewards that are gained by the agent. The goal of the agent is to learn the optimal policy $\pi^*(s, a)$, which maximizes R_t for all t.

An effective way to learn such a policy is to formulate it as a function of the discounted reward to each of the state-action pairs, i.e., q-values,

$$q^{\pi}(s, a) = E_{\pi}[R_t | S_t = s, A_t = a] \tag{8.2}$$

This method is referred to as Q-learning. By setting $\pi = \pi^*$, one can then learn the optimal policy π^* by learning the q-values for every state-action pair. Once the q-values are learned, the optimal policy at a given state is to choose the action with the highest value with probability $(1 - \epsilon)$ where ϵ is the exploration factor. Such selection of the action is called a ϵ-greedy action.

Deep Q-Learning (DQN). Learning q-values becomes intractable for large and complex state spaces. DQN was introduced to address this issue by estimating q-values with deep neural networks [233]. In DQN, the agent is a multi-layered deep neural network that takes S_t as input and outputs q-values for possible actions. The DQN agent is trained as follows: For each step, the state S_t is fed into the input layer, and the agent outputs the action A_t by following the ϵ-greedy principle. The *experience* is then defined as the tuple $[S_t, A_t, R_t, S_{t+1}]$ where R_t is the reward corresponding to action A_t and S_{t+1} is the new state. The experience is then stored in experience relay memory, which holds a finite number of past experiences. The

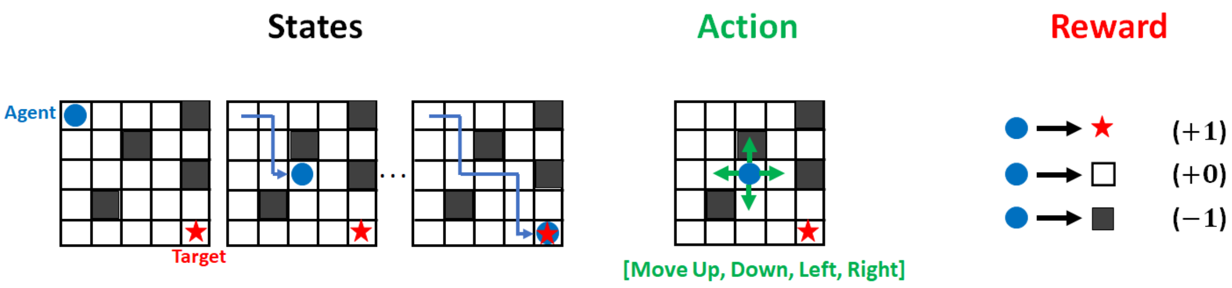


Figure 8.1 Gridworld navigation as an example of discrete action reinforcement learning. The goal of the agent (blue circle) is to reach to target (red star) while avoiding the obstacles (dark grey grids)

parameters of the DQN agent are then trained using gradient descent to minimize the loss

$$ls = (R_{t+1} + \gamma_{t+1} max_{a'} q_{\overline{\theta}}(S_{t+1}, a') - q_{\theta}(S_t, A_t))^2 \tag{8.3}$$

where t is randomly selected time point from the experience relay memory. The training process maintains two networks: the *evaluation network* with the parameters θ used to evaluate the q-values and to choose actions, and the *target network* with parameters $\overline{\theta}$ which is a periodic copy of the evaluation network to help the DQN agent achieve stable learning of q-values. While DQN was a major leap in reinforcement learning, it also had several limitations, leading to several extensions to address them. The most prominent extensions are *Double DQN, Prioritized experience relay*, and *Dueling DQN*. Each of these extensions brings improvements in resolving the overestimation bias of q-values, more effective sampling of past experiences, and better evaluation of q-values through decomposition into value and advantage. Notably, these extensions can be used together to synergistically achieve better performance.

Designing the Discrete Action Space in DQN. Notably, while the state space of DQN can be continuous, its action space is discrete. It follows directly from Eqs (8.2) that the size of the action space is proportional to the number of q-values (i.e. possible state-action pairs) to be learned. Specifically, q-values increase by N for each new action, where N is the total number of possible states in the environment. This makes having a large action space undesirable for DQN, as approximating q-values for all state-action pairs becomes infeasible.

It thus follows that keeping the action space small is advantageous for cultivating an effective learning environment for the DQN agent. An example problem following this principle is gridworld navigation.

Gridworld navigation is often considered a canonical example in Q-learning. The problem consists of spatial grids where the grid space represents a state and an action space of size 4, allowing for movements in [Up, Down, Left, Right]. At $t = 0$, the agent is placed at the starting grid point and needs to learn the optimal path toward the reward while avoiding obstacles (Figure 8.1). The small action space of gridworld navigation assists Q-learning in converging to an optimal path, given sufficient training episodes and a well designed reward function [234]. In sections below, we adapt a small action space principle inspired by gridworld navigation for the implementation of the neural control strategies.

8.2 Mapping Neural Control into Discrete Environment and Action Space

We formulate the neural control problems into discrete action reinforcement learning problems similar to Gridworld navigation. This requires i) State-space that well reflects the control environment and responds according to the actions, ii) Small discrete action spaces that do not compromise the control capability of the agent, and iii) Reward functions that lead to effective and efficient learning.

Satisfying these requirements is challenging for both neuromodulation and connectome control problems. My approach is to take the Gridworld navigation task as an example and design each action to result in *incremental change* to the state and neural metric being controlled. In particular, for neuromodulation control, we set the action space to be an incremental change to the amplitude of the control input, i.e., increase, decrease by a small amount or do nothing. For connectome control, the action space is set to be an incremental change to the connectome, i.e., each action is to determine either to add, subtract, or do nothing to the connection between a given pair of neurons. In the sections below, we describe these methods in detail by defining the state, action, and reward for both neural control problems.

8.3 State and Reward for Neuromodulation Control

The objective of continuous neuromodulation control is to infer time dependent control input into a set of neurons to achieve target behavior. Such behavior is often defined as a sequence of spatial positions $\boldsymbol{p}(t) = (x(t), y(t))$ or as a function of environmental factors experienced by the neural circuits, such as sensory input $I^s(t)$ for $t \in [t_0, T]$. Let us define the current position and the target position at time t as $\boldsymbol{p}(t)$

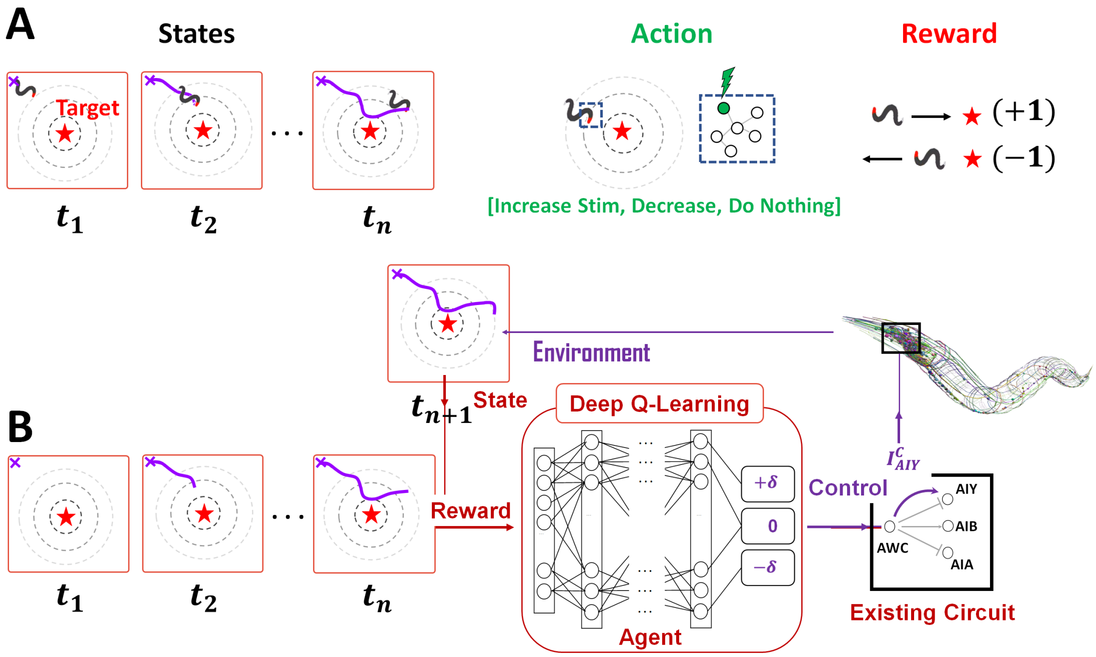


Figure 8.2 Continuous neuromodulation control for existing circuits with a Deep Q-Learning agent. **A** Formulating the continuous neuromodulation control as RL problem consisting of state, action and reward. **B**: Given the location of the organism, Deep Q-learning agent incrementally changes the stimuli to a neuron in existing circuit. The reward is computed based on the post-action distance to the target.

and $\boldsymbol{p}^*(t)$ respectively. We define the environment state and possible actions at time t as

$$S_t = [d_x(t), d_y(t), I^s(t), I^c(t)], \;\; A_t = [-\delta, 0, +\delta] \tag{8.4}$$

where $d_x(t), d_y(t)$ are x and y components of the difference vector $\Delta\boldsymbol{p}(t) = \boldsymbol{p}(t) - \boldsymbol{p}^*(t)$, $I^s(t)$ and $I^c(t)$ are environmental stimuli (e.g., sensory input) experienced by the neural circuit and neuromodulatory control input, respectively, at time t. The action space, i.e. the neural control output, A_t defines the incremental change to neuromodulatory control input $\Delta I^c(t)$ at time t where δ is some small scalar value. Given the definitions of S_t and A_t, we define the reward $R_t = r_t(s_t, a_t)$ as follows

$$r_t(s_t, a_t) = \begin{cases} -tanh(\alpha \Delta d(t+1)) & t < T \\ f(I^s(T)) & t = T \end{cases} \tag{8.5}$$

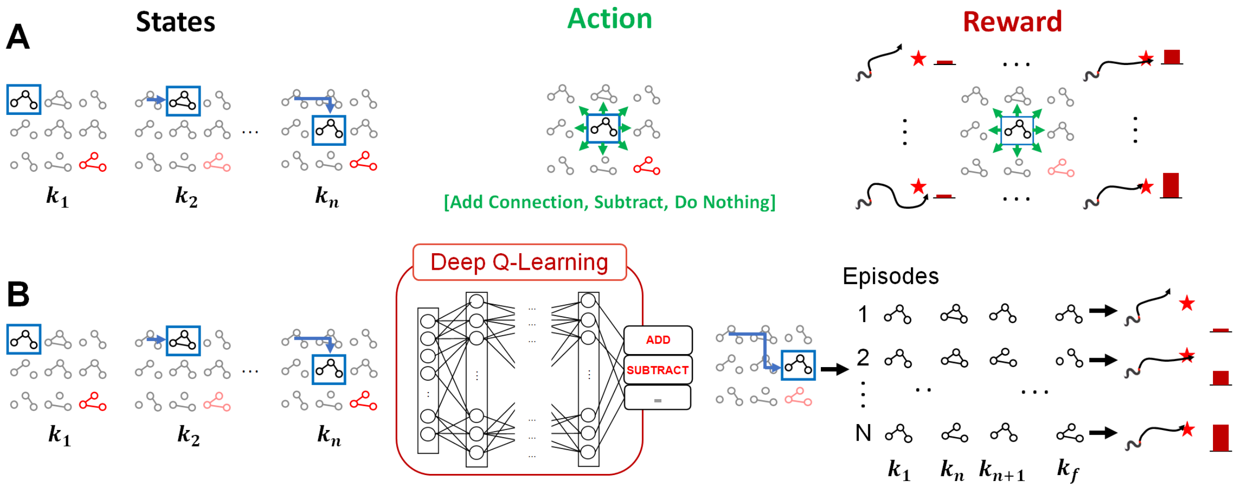


Figure 8.3 Connectome control implemented with Deep Q-Learning agent. **A** Formulating connectome control as the RL problem. **B**: Given the current connectome path, Deep Q-learning agent decides next step in path by adding, subtracting or preserving the connection between two neurons. For each path (i.e. episode), the model simulates the environment and evaluate a reward.

where $\Delta d(t+1) = |\Delta \boldsymbol{p}(t+1)| - |\Delta \boldsymbol{p}(t)|$, α is a scalar that adjusts the slope of the $tanh$ function, and f is an evaluation function that maps $I^s(T)$ to a scalar. The state S_t can be thought of as a grid coordinate in a grid-world at time t, where the DQN agent needs to pick which direction to move to get closer to the reward (Figure 8.2). In our formulation, these directions take the form of increasing, decreasing, or not changing the neuromodulatory control input, ensuring continuity while also keeping the action space small and discrete. The reward function (Eqs (8.5)) is designed such that the agent minimizes the error between the target and the controlled behavior throughout the episode while maximizing the reward at the end of the episode.

8.4 State and Reward for Connectome Control

An alternative way of achieving an aimed behavior is to modulate the architecture of the circuit. Unlike neuromodulatory control, which is time dependent, connectome modulation is static and once set, does not require the agent. Specifically, the goal of connectome control is to infer the connectome matrix G^*, which achieves the target behavior $\boldsymbol{p}^*(t)$ given the sensory input $I^s(t)$ for $t \in [t_0, T]$. Connectome modulation can be implemented in two forms. One can either start from an existing connectome and apply perturbations to the circuit by inserting or deleting connections, or start from a connectome with no connections and infer the full circuit architecture from scratch. Both of these cases take the form of $G^* = G^0 + \Delta G$, where G^0 is the initial connectome matrix and ΔG records changes that have been made to the initial connectome. We propose to use a DQN agent to infer ΔG using a small discrete action space.

We represent ΔG as a function of *state transition* steps similar to a grid-world environment, i.e. $\Delta G = \Delta G_{k_f}$ where k_f is the end step of each episode. We assign an enumeration of all possible neuron pairs within the circuit to serve as transition steps, such that the agent modifies a single connection at a time. The state and the action spaces are then defined as follows

$$S_k = [\Delta G_k, k, m_k, d_k],\ A_k = [-1, 0, 1] \tag{8.6}$$

where k is the step index, i.e. the neuron pair being modified, ΔG_k is ΔG at step k, m_k are the available number of insertions at step k and $d_k = k - m_k$ is the distance to *trapped* state at step k. The Trapped state is defined as at step k, $d_k < 0$, i.e. there are more remaining insertions than transition steps, leading to $m_{k_f} > 0$. m_k monitors whether the agent attempts to insert a new connection when there are no remaining insertions, whereas d_k monitors if there will be unused insertions by the end of the episode. The action space is a triplet selection between -1, 0 and 1 where -1 means deletion, 0 means no change, and 1 means insertion of new connection with some arbitrary weight δ. The reward function $R_k = r_k(s_k, a_k)$ is defined as follows:

$$r_k(s_k, a_k) = \begin{cases} +0.04 & m_{k+1} \geq 0 \text{ and } d_{k+1} \geq 0 \mid a_k \\ -0.75 & m_{k+1} \leq 0 \text{ or } d_{k+1} \leq 0 \mid a_k \\ f(\Delta G_{k_f}) & k = k_f \end{cases} \tag{8.7}$$

where f is an evaluation function that maps G_{k_f} to some scalar. A small positive reward is given for a *valid* action that does not attempt to insert when $m_k = 0$ AND its insertion does not lead to the "trapped" state. A negative reward is given for an *invalid* action that either attempts to insert when $m_k = 0$ OR leads to the "trapped" state. The formulation is analogous to an agent navigating a grid-world with random obstacles and a limited number of steps, where the grids are replaced with different realizations of the connectomes [235]. The reward function intends to guide the DQN agent to make modifications to the connectome at appropriate steps, leading to maximum reward at the end step while avoiding either inserting too many or too few connections.

8.5 RL Environment Modeling Setup

The DQN methodology imposes no constraints on its environment. For our methods, we use the computational neuromechanical model of *C. elegans* to set up the RL environment. The model can simulate full nervous system activity in response to external stimuli alongside body dynamics [51; 105]. In order to mimic the typical chemotaxis environment, we use mod-SenseWorm introduced in Chapter 6, which introduces environmental stimuli into olfactory neurons that are tonically translated into neural stimuli [236]. For each episode, we place a circular stimulus source (e.g., 2D gaussian) on the left side of the spatial environment and the worm on the right side. At $t = 0$, we initialize the worm with forward motion and simulate until t reaches the final time point. We define the behavior for each episode to be the trajectory of the worm head $s(t) = (x(t), y(t))$ parametrized by time t. Here $s(t)$ is the function of parameters being controlled, such as external control stimulation into a set of neurons or modulated connectomes.

8.6 Target Neural Circuit in *C. elegans* Nervous System

We apply neuromodulation and connectome control methods to AWC olfactory circuit (i.e. AWC sensory neurons and adjacent neighboring neurons) in *C. elegans* nervous system to control the associated chemotaxis behavior. First, we infer a continuous neuromodulatory control to emulate *neuropeptidic currents* from AWC sensory neurons to AIA and AIY interneurons to obtain the baseline attraction behavior in response to environmental stimuli associated with AWC. The experiment serves as a validation of literature findings that such currents between AWC, AIY, and AIA are important in achieving attraction behavior [44]. Next, we apply connectome control to the AWC circuit to *modulate the baseline chemotaxis behavior* by introducing new electrical synapses, i.e., gap junctions into the circuit. The results are validated against experimental findings that inserted synthetic gap junctions in particular locations of the AWC circuit and observed a change in chemotaxis behavior [50; 131]. To further test the generality and scalability of connectome control, we expand the method to *infer the full connectome* of the circuit that achieves the desired chemotaxis behavior.

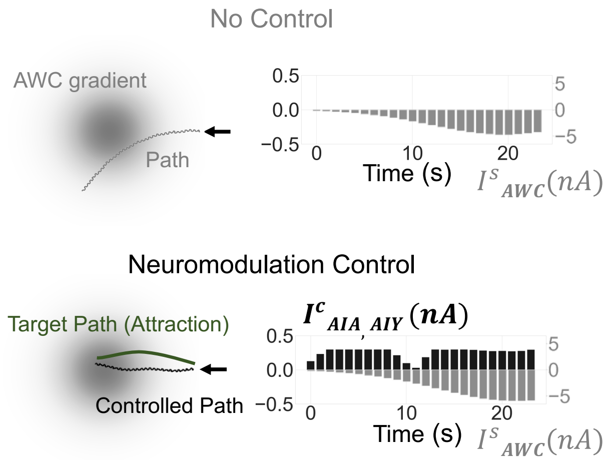


Figure 8.4 Neuromodulation control in *C. elegans* chemotactic circuit to assume a target path of attraction to AWC gradient stimulus. Top: In the absence of control, the path passively passes through the gradient stimulus. Bottom: Neuromodulation control into AIA and AIY neurons changes the course such that *C. elegans* is attracted to the center of AWC gradient stimulus.

8.7 Continuous Neuromodulatory Control Results

Figure 8.4 top shows *C. elegans* simulated behavior without neuromodulation control when navigating AWC stimuli gradient. In particular, we construct an environmental stimulus to have a negative distribution (i.e. $I^s < 0$) for all points throughout the space to mimic typical AWC activity (i.e., OFF neurons) during chemotaxis [236]. In the bottom part, neuromodulation control I^c is applied to AIA and AIY. Both control input I^C and the signal from AWC gradient I^s are plotted as a function of time to show the interaction between the two signals, where a larger amplitude of I^s represents the worm getting closer to the target at the center of the environmental stimulus source.

From movement paths, we observe that neuromodulation control steers the worm toward the stimulus source, whereas the absence of control leads to the worm escaping the stimulus region without interaction. The control input overall maintains the range of $\sim +0.3nA$, but temporarily dips around $t = 10s$ when the worm encounters an increase in stimulus, i.e., high $|dI^s/dt|$. Indeed, experiments have reported such adaptation of neuromodulatory currents in response to the negative derivative of the chemical stimulus during chemotaxis [236; 237; 238].

8.8 Connectome Control Results

Having established the baseline chemotaxis behavior for the environmental stimulus with neuromodulatory currents, we proceed to use this baseline, i.e., a closed loop system, for connectome control to introduce

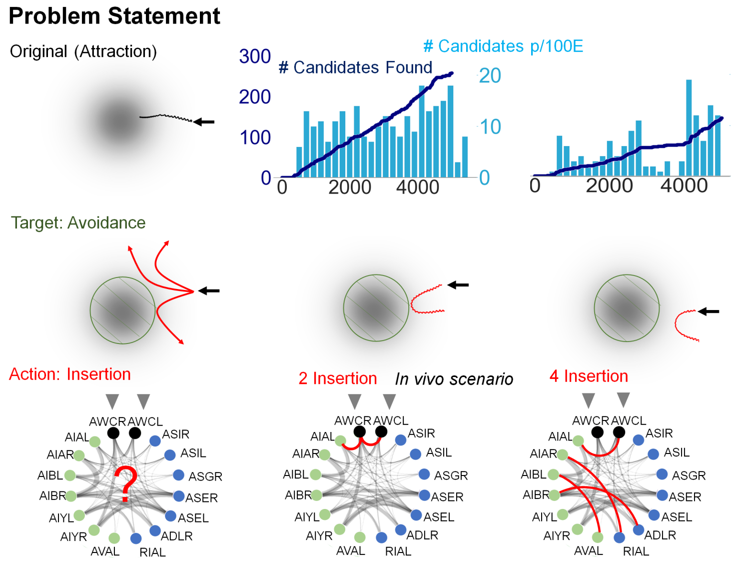


Figure 8.5 Gap junctions insertion to change baseline chemotactic behavior. Left: from top to bottom, problem statement showing original behavior before the insertion, target behavior to be achieved with insertion , and visualization of circuit subject to insertion. Middle and Right: from top to bottom, number of viable insertion configurations found as the function of training episodes, behavioral trajectory obtained from one of the candidates, visualization of the circuit corresponding to behavioral trajectory with inserted gap junctions marked in red

new connections into the circuit for novel behavioral output. We apply our method to three experiments to measure and test its capabilities.

Baseline behavior modulation In this scenario, we add gap junctions to the existing AWC circuit to modulate its baseline chemotaxis behavior. Specifically, RL agent is tasked with inserting gap junctions in appropriate locations in the circuit to change the behavior from *baseline attraction* to *repulsion*. The target is implicitly set such that for the given AWC gradient signal $I^s(t)$, the connectome that produces a worm movement path with a large curvature (i.e. steeper turn) receives a higher reward. In Figure 8.5, we show the training results for 2 and 4 insertion scenarios. Each column shows the learning progress in terms of the number of viable candidates (i.e. the insertions that result in behavior with scores higher than a certain threshold) found, the movement trajectory with one of such candidates, and the visualization of its circuit architecture. For the 2 insertion scenario, RL agent consistently finds viable configurations throughout training, which are reflected by the linear trend in the number of candidates found. One such configuration that results in behavior close to the target is connecting AWCL, AWCR, AWCR, and AIAL. Notably, we find that such a configuration is confirmed by experiment to indeed cause a behavior switch from attraction to repulsion in AWC modulated chemotaxis response [134].

Increasing the available insertions to 4 results in a more challenging search, as it increases both the number of possible configurations and the uncertainty in network activity by introducing more connections. This is indeed reflected in the lower number of candidates throughout the training compared to 2 insertion

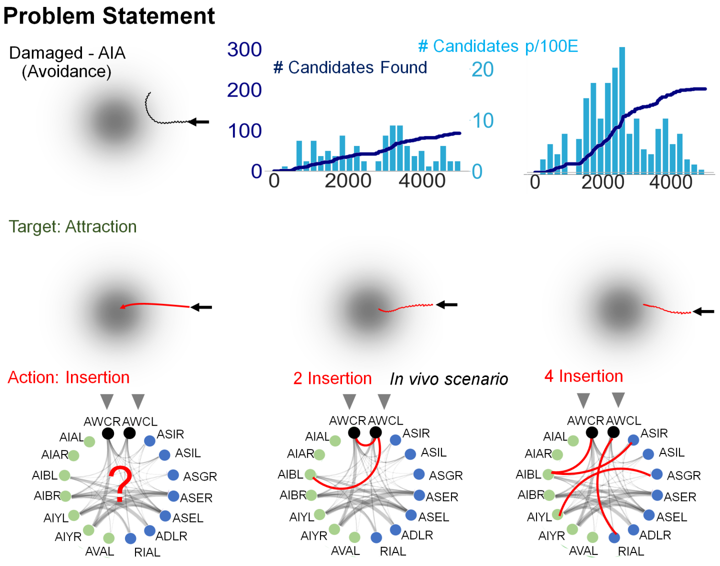


Figure 8.6 Gap junctions insertion to repair a damaged circuit (AIA ablated). Left: from top to bottom, problem statement showing chemotactic behavior with AIA ablated circuit, target behavior to be achieved with insertion, and visualization of circuit subject to insertion. Middle and Right: from top to bottom, number of viable insertion configurations found as the function of training episodes, behavioral trajectory obtained from one of the candidates, visualization of circuit corresponding to behavioral trajectory with inserted gap junctions marked red

scenario (Figure 8.5). Interestingly, we notice that many candidates still include an insertion between AWC and AIA, implying that the RL agent retains information on which insertions are valuable. We also notice a growing number of candidates that is unexplored in the literature. Such a candidate is shown in the third column of Figure 8.5. Apart from the insertion between *in-vivo* AIAL and AWCL, the other three connections are new, and their effects are unknown. This could imply that the RL agent learns how to ’engineer’ the circuit with multiple insertions, where the combined effects achieve a target behavior.

A damaged circuit repair. To further test the robustness of our method, we examine whether it can *repair* the damaged circuit and restore lost functionality by inserting new gap junctions. In particular, we ablate AIA inter-neurons from the AWC circuit, leading to an absence of attraction response [131]. The change in behavior is indeed accurately reproduced in the computational model of *C. elegans* (Figure 8.6). We then repeat the same procedure of inserting 2, 4 and 8 gap junctions into the damaged circuit to restore the baseline attraction behavior.

Figure 8.6 shows the training results for 2 and 4 insertions scenarios which lead to behavioral recovery. One of the candidates for the 2 insertions scenario connects AWCL with AIBL and AWCR. Indeed, such a configuration has been recently shown to restore the attraction behavior in AIA ablated circuits, as the connection bypasses the ablated AIA neurons and allows direct communication between AWC and AIB [131]. We notice that many of the candidates in 4 insertions scenario continue to connect AWC and AIB, further confirming the essential role of AWC-AIB coupling in behavioral recovery. Interestingly, we obtain a

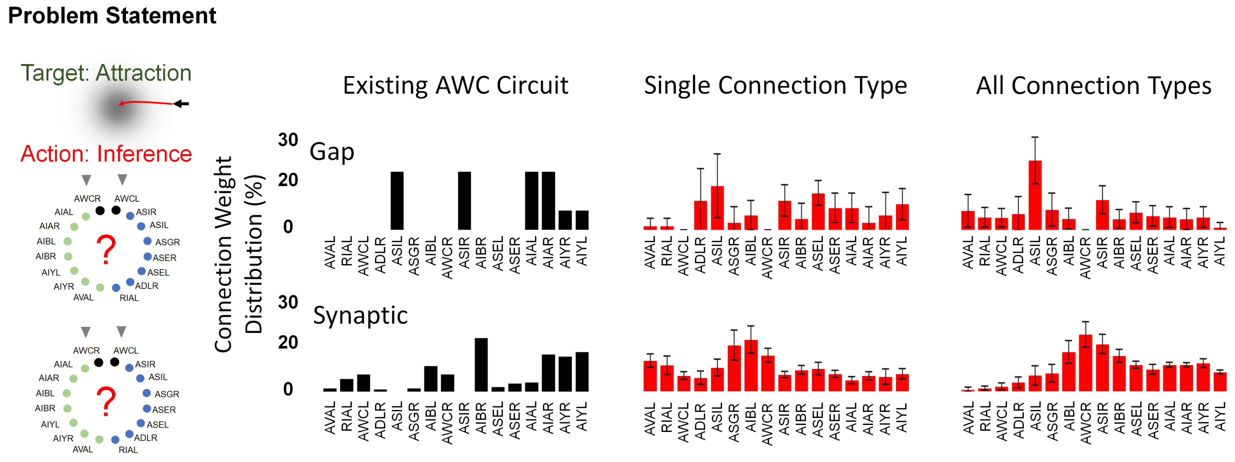


Figure 8.7 Inferring circuit architectures to achieve target chemotactic behavior. Each column shows the connection distribution throughout the neurons in existing AWC circuit (left), inferred each connection types when other type was kept static (middle) and inferred circuits of all connection types when types are initialized with empty circuits.

number of candidates that achieve a higher reward for 4 insertions than 2 insertions. One possible explanation is that since AIA is one of the most connected neurons in the circuit, it is easier for the RL agent to find candidates with 4 insertions than 2 to compensate for lost connections.

Full circuit architecture inference. The ability to insert new connections into an existing circuit can be extended to circuits without connections, i.e., building the circuit connectome from scratch. In such a case, the action space of the RL agent turns into a discrete set of numbers, where each action is *synaptic weight* to be added to a particular pair of neurons, allowing the agent to infer full circuit connectivity. To test such an extension, we designed three experiments with the same objective of inferring the circuit that can produce the baseline attraction behavior as in the existing AWC circuit with neuromodulation control. Specifically, experiment 1 was tasked with inferring both gap and synaptic connectomes from scratch, whereas experiments 2 and 3 were each tasked with inferring a single connection type: synaptic or gap connectome, while keeping the other type of connectome intact. In order to compare the distribution of inferred connections with that of the existing circuit, we set the number of insertions for each type of connectome to be the same as those in the existing connectomes (i.e., the number of synapses).

Figure 8.7 shows connection distributions between existing connectomes and inferred circuits that successfully produce baseline attraction behavior (averaged over 10 best candidates). We notice that the connection distributions of inferred connectomes are generally different from those of existing connectomes.

Notably, inferring all connection types results in a distribution that is more similar to the existing circuit than a single connection type. This implies the existence of several circuit realizations to facilitate attraction. However, inferred circuits also highlight some of the hub neurons found in existing circuits, such as ASI neurons in the gap connectome. It still remains an open question why the inferred connectomes do not necessarily converge to those of the existing circuits. One possible explanation is that inferred connectomes are optimized for a *single* behavior type, whereas the existing circuit is for *multiple* behaviors. There is a large body of evidence that shows the AWC circuit participates in sensory functions other than just olfactory [239; 240; 241]. Therefore, while many circuit realizations may facilitate a single behavior type, as the method suggests, they might not produce sensory functions in other areas.

In summary, for both continuous neuromodulatory control and connectome control, our methods adapting the small action space, such as that of grid world navigation, were able to successfully control the circuit of interest and infer the full circuit architecture from scratch to achieve the target behavior.

Chapter 9

Conclusion

In this thesis, I propose methods for developing a neuromechanical model of the nematode *Caenorhabditis elegans*. Such a model integrates the models of the nervous system, muscles, and body, and has the potential to reveal the mechanisms of sensorimotor integration that govern the translation of neural activity to behavior. In particular, I propose and implement the *modular integration* approach to describe the neuromechanical model as an integrated series of biophysical modules. Modular integration can assist in the identification of neuromechanical model candidates by representing their architectures across varying biophysical scopes and modalities. I developed the software implementation of modular integration, *modWorm*, for the construction of a neuromechanical model of *C. elegans* (Chapter 3). modWorm allows for the construction of a model as a series of configurable and exchangeable biophysical modules. Combined with its simulation capability, the framework is a generic modeling platform for neuromechanics studies to integrate different biophysical processes that constitute behavior.

In Chapter 4, I describe how modWorm is used to introduce the fundamental neuromechanical model candidate of *C. elegans*. The model integrates a series of 7 modules, which encompass the intra- and extra-cellular neural dynamics, muscle dynamics, biomechanics, and proprioception. We validated the model with respect to known typical locomotion scenarios, such as forward and backward locomotion, and transitional behaviors during locomotion, such as avoidance and turn, by applying associated neural/external force stimulation. Furthermore, I performed computational neural ablations to recapitulate locomotion behaviors associated with *in-vivo* ablation scenarios and validate model predicted neural pathways associated with locomotion through novel *in-vivo* experiments.

Leveraging the modular structure of the model, Chapter 5 introduces model variations and extensions motivated by both experiments and theoretical concepts. Empirically driven variations, such as alternative intra-cellular dynamics, connectome mappings, and synaptic channel dynamics, highlight their individual contributions to achieving a refined fit of locomotion compared to the experiments. Data-driven and conceptually driven variations, such as model parameter tunings and alternative equations for extra-cellular dynamics, show model sensitivities with respect to the dynamics of individual components and associated parameters in achieving simulated behavior close to the experiments.

In Chapter 6, I introduce mod-SenseWorm, an extension of modWorm, to incorporate the stimulus environment in which the neuromechanical model can interact during the simulation. The framework allows for the configuration of neurons' translation strategies from environmental stimuli to neural stimulation to achieve simulations of more complex locomotion scenarios, such as chemotaxis. In particular, we studied the O2 aerotaxis responses of *C. elegans* and showed that the model can recapitulate the avoidance responses associated with increasing O2 level as observed *in-vivo* experiments. Furthermore, through the inspection of underlying neural activities, we showed that mod-SenseWorm can be used to infer potential functional pathways associated with simulated responses.

In Chapter 7, I introduce ElectroPhysiomeGAN (EP-GAN) to assist in the extension of *C. elegans* neuromechainical model through the inference of biophysical parameters associated with a detailed neuron model. Trained entirely on simulated data, EP-GAN takes neuron electrophysiological recordings as input and generates a large number of parameters associated with neuron models (e.g., Hodgkin-Huxley type model). We tested EP-GAN by predicting HH-model parameters (N > 175) for both simulated and experimental neurons found in *C. elegans* nervous system. The results showed EP-GAN's advantage in both the accuracy of the generated parameters and inference speed compared to existing methods such as Differential Evolution and Genetic Algorithms.

In Chapter 8, I develop DeepRL based control strategies to further assist in model extension by inferring supplemental neural mechanisms (e.g., neuropeptidic currents, connectome perturbations) associated with empirically observed behavior. In particular, we developed control methods for dynamic neural stimulation (neuromodulation control) and neural connection mappings (connectome control), which are applied on top of the neuromechanical model. Using the Deep Q-Network as the RL agent, we showed that the control strategies identified by the proposed methods can be used to infer extra-synaptic interactions between neurons (e.g., neuropeptidic currents) and connectome configurations of a local neural circuit associated with target behaviors (e.g., attraction and avoidance) in response to environmental stimuli. Furthermore, by extending the action space of the RL agent, we showed that the method can be applied to a wider set of problems, such as repairing damaged neural circuits to restore behavior and inferring the full connectome of a neural circuit associated with a target behavior.

Taken together, the thesis presents a series of novel contributions to assist in the development of neuromechanical modeling of *C. elegans*. Leveraging neural data, biophysical models, and deep learning methods, my research extends the understanding of currently known neuromechanical functions and potentially leads to the unraveling of novel *C. elegans* brain circuits responsible for locomotion and behavior. Furthermore, as neural data for other organisms become increasingly available, I expect my work to pave the way for future model development endeavors for such organisms and ultimately accelerate our understanding of how the brain orchestrates behavior.

Chapter 10

Future Directions

In this section, I describe three potential future extensions for my PhD research. I categorize these directions into three broad approaches: i) Identification and validation of functional neural circuits in *C. elegans*, ii) Development of AI methods to further assist with the modeling and investigation of biological neural systems, and iii) Neuromechanical modeling of other organisms beyond *C. elegans*.

10.1 Identification and Validation of Functional Neural Circuits in *C. elegans*

modWorm and the proposed neuromechanical model of *C. elegans* provide a powerful platform for studying network functionality. Using the model to extract existing functional circuits distributed throughout the network, followed by identifying their computational roles in behavior, stands out as a promising research direction. Here, we show examples of how they could be achieved.

Performing simulations of the network in response to different combinations of stimulus (similar to the studies in Chapter 6.4, 6.5), followed by analyzing individual neuron activities, can reveal the relationships between the neurons and their associated transfer functions. In extensions of such an approach, the information flow from the well characterized olfactory sensory neurons to the downstream interneurons could be analyzed to investigate the network's initial encoding of environmental stimuli.

The studies could reveal novel functional circuit motifs between neurons, as recently identified by several experiments [26] and potential circuits that combine such motifs to implement computations such as NAND and XOR gates. Associating such circuits with behavior through computational ablation studies would assist in understanding how discrete functional units collectively perform to fulfill network-wide functions.

10.2 AI Methods to Further Bridge *in-silico* and *in-vivo* Neural Systems

Deep learning based methods that I have described here, such as EP-GAN, neuromodulation, and connectome control, show the potential of AI methods to enhance and interact with computational models to further improve their simulation accuracy. Developing extensions of these methods to further resolve the gap between computational and *in-vivo* neuromechanical systems can be pursued by leveraging both simulated and empirical data.

Simulated data, when generated from a sufficiently detailed model, can be used to train AI models to learn the translation from dynamics (e.g., neural or body) to biophysical model parameters that generate them. As demonstrated with EP-GAN, such models can be used to infer unknown model parameters associated with empirical data [JK2]. Future versions of EP-GAN that are capable of generating neuron model parameters for additional neuron types (e.g., spiking neurons) and the development of AI methods for the inference of extra-cellular biophysical parameters, such as neural connection weights/directionality (e.g., excitatory/inhibitory) from recorded population neural activity, would fall into this category. Another direction for using the simulated data is to train physics-informed AI models (e.g., neural operators), which, given the initial conditions, directly map the future dynamics of the neural systems at arbitrary time points. Such methods have been applied in areas such as weather forecasting to efficiently solve partial differential equations (e.g., Navier-Stokes equations) while maintaining high-resolution predictions [242; 243]. Since neuromechanical models are also governed by partial differential equations, it is expected that similar approaches can be sought to achieve fast simulations of neural and body dynamics with significantly lower computational costs.

An alternative approach for model parameter inference is to treat them as trainable parameters and optimize them directly with respect to empirical data using a gradient descent method. Such an approach would require fewer computational resources, as it does not require generating a large amount of simulated data for training. Indeed, several works have adopted such training methods in the form of deep mechanistic networks to infer the neural parameters (e.g., synaptic weights) and functions of organisms such as *C. elegans* and *Drosophila* visual system [100; 102; 103; 99]. These optimizations, as discussed in Chapter 5 *Model Extensions for Simulation Accuracy Enhancement*, need to be performed alongside the generalization requirements to mitigate potential simulation inaccuracies due to overfitting. Additional optimization methods (e.g., multi-objective cost functions) in conjunction with gradient based training algorithms can thus be investigated to mitigate such discrepancies. Furthermore, these approaches often make compromises on model details to allow for the training of a large set of parameters. My future research will thus aim to develop optimization methods that focus on addressing the challenges of model generalization while retaining detailed dynamical properties that are essential for the accurate simulation of a nervous system.

10.3 Neuromechanical Modeling Beyond *C. elegans*

Recent advances in imaging, optogenetics, and electrophysiological methods allow for the resolution of connectome mappings and electrophysiological neuron responses in organisms other than *C. elegans*. In particular, connectomes of both larvae and adult drosophila (fruit-fly) have been fully resolved alongside their biomechanics [53; 56; 55; 57; 58]. Since the modWorm framework is a generic modeling tool not limited to *C. elegans*, it is expected that similar procedures and implementation steps that leverage the modular modeling approach used for constructing *C. elegans* neuromechanical model can be followed to construct the model of drosophila and other organisms' neuromechanics.

Similar to *C. elegans* locomotion studies discussed in Chapter 4, initial simulations of drosophila neuromechanics can focus on recapitulating basic flight, followed by more advanced movements such as flight maneuvers. Non-locomotive behaviors, such as grooming, antenna movements, and object detection by visual systems, can also be studied to complement existing simulation work and discover novel mechanisms supported by different model variations [100]. Having larger sensory and behavioral regimes, the modeling of *Drosophila* is expected to further assist in understanding the computational principles that connect neural functions and behavior.

Jimin Kim's Publications

[JK1] Jimin Kim, Jeremy T Florman, Julia A Santos, Mark J Alkema, and Eli Shlizerman. Modular integration of neural connectomics, dynamics and biomechanics for identification of behavioral sensorimotor pathways in caenorhabditis elegans. *arXiv preprint arXiv:2504.18073*, 2025.

[JK2] Jimin Kim, Minxian Peng, Shuqi Chen, Qiang Liu, and Eli Shlizerman. Generation of biophysical neuron model parameters from recorded electrophysiological responses. *eLife*, 13, 2025.

[JK3] Jimin Kim and Eli Shlizerman. C. elegans nervous system model reveals a xor-gate mechanism mediating oxygen sensation. *Cosyne Abstracts*, 2021.

[JK4] Jimin Kim and Eli Shlizerman. Deep reinforcement learning of extra-synaptic control in c. elegans chemosensory response. *Cosyne Abstracts*, 2020.

[JK5] Jimin Kim and Eli Shlizerman. Deep reinforcement learning for neural control. *arXiv preprint arXiv:2006.07352*, 2020.

[JK6] Jimin Kim, William Leahy, and Eli Shlizerman. Neural interactome: Interactive simulation of a neuronal system. *Frontiers in computational neuroscience*, 13:8, 2019.

[JK7] Hexuan Liu, Jimin Kim, and Eli Shlizerman. Functional connectomics from neural dynamics: probabilistic graphical models for neuronal network of caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170377, 2018.

Bibliography

[1] Martyn Goulding. Circuits controlling vertebrate locomotion: moving in a new direction. *Nature Reviews Neuroscience*, 10(7):507–518, 2009.

[2] Sten Grillner. Neurobiological bases of rhythmic motor acts in vertebrates. *Science*, 228(4696):143–149, 1985.

[3] Shangbang Gao, Sihui Asuka Guan, Anthony D Fouad, Jun Meng, Taizo Kawano, Yung-Chi Huang, Yi Li, Salvador Alcaire, Wesley Hung, Yangning Lu, et al. Excitatory motor neurons are local oscillators for backward locomotion. *Elife*, 7:e29915, 2018.

[4] Anthony D Fouad, Shelly Teng, Julian R Mark, Alice Liu, Pilar Alvarez-Illera, Hongfei Ji, Angelica Du, Priya D Bhirgoo, Eli Cornblath, Sihui Asuka Guan, et al. Distributed rhythm generators underlie caenorhabditis elegans forward locomotion. *Elife*, 7:e29913, 2018.

[5] Quan Wen, Shangbang Gao, and Mei Zhen. Caenorhabditis elegans excitatory ventral cord motor neurons derive rhythm for body undulation. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170370, 2018.

[6] Primoz Ravbar, Neil Zhang, and Julie H Simpson. Behavioral evidence for nested central pattern generator control of drosophila grooming. *Elife*, 10:e71508, 2021.

[7] Harris S Kaplan, Oriana Salazar Thula, Niklas Khoss, and Manuel Zimmer. Nested neuronal dynamics orchestrate a behavioral hierarchy across timescales. *Neuron*, 105(3):562–576, 2020.

[8] Pierre A Guertin. The mammalian central pattern generator for locomotion. *Brain research reviews*, 62 (1):45–56, 2009.

[9] Konrad P Kording, Gunnar Blohm, Paul Schrater, and Kendrick Kay. Appreciating the variety of goals in computational neuroscience. *arXiv preprint arXiv:2002.03211*, 2020.

[10] Eviatar Yemini, Tadas Jucikas, Laura J Grundy, André EX Brown, and William R Schafer. A database of caenorhabditis elegans behavioral phenotypes. *Nature methods*, 10(9):877–879, 2013.

[11] Sydney Brenner. The genetics of caenorhabditis elegans. *Genetics*, 77(1):71–94, 1974.

[12] S Brenner. Nature's gift to science. nobel lecture, december 8, 2002. *Biosci. Rep*, 23:225–37, 2003.

[13] John G White, Eileen Southgate, J Nichol Thomson, Sydney Brenner, et al. The structure of the nervous system of the nematode caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci*, 314(1165):1–340, 1986.

[14] Lav R Varshney, Beth L Chen, Eric Paniagua, David H Hall, and Dmitri B Chklovskii. Structural properties of the caenorhabditis elegans neuronal network. *PLoS computational biology*, 7(2):e1001066, 2011.

[15] Gal Haspel and Michael J O'Donovan. A perimotor framework reveals functional segmentation in the motoneuronal network controlling locomotion in caenorhabditis elegans. *Journal of Neuroscience*, 31 (41):14611–14623, 2011.

[16] Balázs Szigeti, Padraig Gleeson, Michael Vella, Sergey Khayrulin, Andrey Palyanov, Jim Hokanson, Michael Currie, Matteo Cantarelli, Giovanni Idili, and Stephen Larson. Openworm: an open-science approach to modeling caenorhabditis elegans. *Frontiers in computational neuroscience*, 8:137, 2014.

[17] Steven J Cook, Travis A Jarrell, Christopher A Brittin, Yi Wang, Adam E Bloniarz, Maksim A Yakovlev, Ken CQ Nguyen, Leo T-H Tang, Emily A Bayer, Janet S Duerr, et al. Whole-animal connectomes of both caenorhabditis elegans sexes. *Nature*, 571(7763):63–71, 2019.

[18] Christopher A Brittin, Steven J Cook, David H Hall, Scott W Emmons, and Netta Cohen. A multi-scale brain map derived from whole-brain volumetric reconstructions. *Nature*, 591(7848):105–110, 2021.

[19] Daniel Witvliet, Ben Mulcahy, James K Mitchell, Yaron Meirovitch, Daniel R Berger, Yuelong Wu, Yufang Liu, Wan Xian Koh, Rajeev Parvathala, Douglas Holmyard, et al. Connectomes across development reveal principles of brain maturation. *Nature*, 596(7871):257–261, 2021.

[20] Ben Mulcahy, Daniel K Witvliet, James Mitchell, Richard Schalek, Daniel R Berger, Yuelong Wu, Doug Holmyard, Yangning Lu, Tosif Ahamed, Aravinthan DT Samuel, et al. Post-embryonic remodeling of the c. elegans motor circuit. *Current Biology*, 32(21):4645–4659, 2022.

[21] Hyunsoo Yim, Daniel T Choe, J Alexander Bae, Myung-kyu Choi, Hae-Mook Kang, Ken CQ Nguyen, Soungyub Ahn, Sang-kyu Bahn, Heeseung Yang, David H Hall, et al. Comparative connectomics of dauer reveals developmental plasticity. *Nature Communications*, 15(1):1546, 2024.

[22] ZF Altun and DH Hall. Muscle system, introduction. *WormAtlas. Edited for the web by Laura A. Herndon. Last revision: May*, 2:2012, 2009.

[23] ZF Altun and DH Hall. Handbook of c. elegans anatomy. *WormAtlas. http://www. wormatlas. org/handbook/contents. htm*, 2005.

[24] Qiang Liu, Philip B Kidd, May Dobosiewicz, and Cornelia I Bargmann. C. elegans awa olfactory neurons fire calcium-mediated all-or-none action potentials. *Cell*, 175(1):57–70, 2018.

[25] Jingyuan Jiang, Yifan Su, Ruilin Zhang, Haiwen Li, Louis Tao, and Qiang Liu. C. elegans enteric motor neurons fire synchronized action potentials underlying the defecation motor program. *Nature communications*, 13(1):2783, 2022.

[26] May Dobosiewicz, Qiang Liu, and Cornelia I Bargmann. Reliability of an interneuron response depends on an integrated sensory state. *Elife*, 8:e50566, 2019.

[27] Martina Nicoletti, Alessandro Loppini, Letizia Chiodo, Viola Folli, Giancarlo Ruocco, and Simonetta Filippi. Biophysical modeling of c. elegans neurons: Single ion currents and whole-cell dynamics of awcon and rmd. *PloS one*, 14(7):e0218738, 2019.

[28] Martina Nicoletti, Letizia Chiodo, Alessandro Loppini, Qiang Liu, Viola Folli, Giancarlo Ruocco, and

Simonetta Filippi. Biophysical modeling of the whole-cell dynamics of c. elegans motor and interneurons families. *PloS one*, 19(3):e0298105, 2024.

[29] Gal Haspel, Lan Deng, Maria Belen Harreguy, and Zainab Tanvir. Elegantly. In *The neural control of movement*, pages 3–29. Elsevier, 2020.

[30] Steven L Mclntire, Erik Jorgensen, Joshua Kaplan, and H Robert Horvitz. The gabaergic nervous system of caenorhabditis elegans. *Nature*, 364(6435):337–341, 1993.

[31] Cornelia I Bargmann. Neurobiology of the caenorhabditis elegans genome. *Science*, 282(5396): 2028–2033, 1998.

[32] Nancy J Kopell, Howard J Gritton, Miles A Whittington, and Mark A Kramer. Beyond the connectome: the dynome. *Neuron*, 83(6):1319–1328, 2014.

[33] Cornelia I Bargmann and Eve Marder. From the connectome to brain function. *Nature methods*, 10(6): 483, 2013.

[34] Eve Marder and Jean-Marc Goaillard. Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience*, 7(7):563–574, 2006.

[35] Lidia Ripoll-Sánchez, Jan Watteyne, HaoSheng Sun, Robert Fernandez, Seth R Taylor, Alexis Weinreb, Barry L Bentley, Marc Hammarlund, David M Miller, Oliver Hobert, et al. The neuropeptidergic connectome of c. elegans. *Neuron*, 111(22):3570–3589, 2023.

[36] Alejandro López-Cruz, Aylesse Sordillo, Navin Pokala, Qiang Liu, Patrick T McGrath, and Cornelia I Bargmann. Parallel multimodal circuits control an innate foraging behavior. *Neuron*, 102(2):407–419, 2019.

[37] Jennifer K Pirri, Adam D McPherson, Jamie L Donnelly, Michael M Francis, and Mark J Alkema. A tyramine-gated chloride channel coordinates distinct motor programs of a caenorhabditis elegans escape response. *Neuron*, 62(4):526–538, 2009.

[38] Eviatar Yemini, Albert Lin, Amin Nejatbakhsh, Erdem Varol, Ruoxi Sun, Gonzalo E Mena, Aravinthan DT Samuel, Liam Paninski, Vivek Venkatachalam, and Oliver Hobert. Neuropal: a multicolor atlas for whole-brain neuronal identification in c. elegans. *Cell*, 184(1):272–288, 2021.

[39] Vladislav Susoy, Wesley Hung, Daniel Witvliet, Joshua E Whitener, Min Wu, Core Francisco Park, Brett J Graham, Mei Zhen, Vivek Venkatachalam, and Aravinthan DT Samuel. Natural sensory context drives diverse brain-wide activity during c. elegans mating. *Cell*, 184(20):5122–5137, 2021.

[40] Kerem Uzel, Saul Kato, and Manuel Zimmer. A set of hub neurons and non-local connectivity features support global brain dynamics in c. elegans. *Current Biology*, 32(16):3443–3459, 2022.

[41] Isabel Beets, Sven Zels, Elke Vandewyer, Jonas Demeulemeester, Jelle Caers, Esra Baytemur, Amy Courtney, Luca Golinelli, İlayda Hasakioğulları, William R Schafer, et al. System-wide mapping of peptide-gpcr interactions in c. elegans. *Cell reports*, 42(9), 2023.

[42] Ugur Dag, Ijeoma Nwabudike, Di Kang, Matthew A Gomes, Jungsoo Kim, Adam A Atanas, Eric Bueno, Cassi Estrem, Sarah Pugliese, Ziyu Wang, et al. Dissecting the functional organization of the c. elegans serotonergic system at whole-brain scale. *Cell*, 186(12):2574–2592, 2023.

[43] Adam A Atanas, Jungsoo Kim, Ziyu Wang, Eric Bueno, McCoy Becker, Di Kang, Jungyeon Park, Talya S Kramer, Flossie K Wan, Saba Baskoylu, et al. Brain-wide representations of behavior spanning multiple timescales and states in c. elegans. *Cell*, 186(19):4134–4151, 2023.

[44] Sreekanth H Chalasani, Saul Kato, Dirk R Albrecht, Takao Nakagawa, LF Abbott, and Cornelia I Bargmann. Neuropeptide feedback modifies odor-evoked dynamics in caenorhabditis elegans olfactory neurons. *Nature neuroscience*, 13(5):615, 2010.

[45] Dean Pospisil, Max Aragon, and Jonathan Pillow. From connectome to effectome: learning the causal interaction map of the fly brain. *bioRxiv*, 2023.

[46] ZF Altun and Hall DH. Wormatlas. 2002–2006. *See http://www. wormatlas. org*, 2009.

[47] Harksun Lee, Myung-kyu Choi, Daehan Lee, Hye-sung Kim, Hyejin Hwang, Heekyeong Kim, Sungsu

Park, Young-ki Paik, and Junho Lee. Nictation, a dispersal behavior of the nematode caenorhabditis elegans, is regulated by il2 neurons. *Nature neuroscience*, 15(1):107, 2012.

[48] Cornelia I Bargmann. Chemosensation in c. elegans. *WormBook: The online review of C. elegans biology [Internet]*, 2006.

[49] Miriam B Goodman, David H Hall, Leon Avery, and Shawn R Lockery. Active currents regulate sensitivity and dynamic range in c. elegans neurons. *Neuron*, 20(4):763–772, 1998.

[50] Ithai Rabinowitch and William R Schafer. Engineering new synaptic connections in the c. elegans connectome. In *Worm*, volume 4, page e992668. Taylor & Francis, 2015.

[51] Jimin Kim, William Leahy, and Eli Shlizerman. Neural interactome: Interactive simulation of a neuronal system. *Frontiers in computational neuroscience*, 13:8, 2019.

[52] LF Abbott and Thomas B Kepler. Model neurons: from hodgkin-huxley to hopfield. In *Statistical mechanics of neural networks*, pages 5–18. Springer, 1990.

[53] Victor Lobato-Rios, Shravan Tata Ramalingasetty, Pembe Gizem Özdil, Jonathan Arreguit, Auke Jan Ijspeert, and Pavan Ramdya. Neuromechfly, a neuromechanical model of adult drosophila melanogaster. *Nature Methods*, 19(5):620–627, 2022.

[54] Jane Loveless, Konstantinos Lagogiannis, and Barbara Webb. Modelling the mechanics of exploration in larval drosophila. *PLoS computational biology*, 15(7):e1006635, 2019.

[55] Roman Vaxenburg, Igor Siwanowicz, Josh Merel, Alice A Robie, Carmen Morrow, Guido Novati, Zinovia Stefanidi, Gwyneth M Card, Michael B Reiser, Matthew M Botvinick, et al. Whole-body simulation of realistic fruit fly locomotion with deep reinforcement learning. *bioRxiv*, pages 2024–03, 2024.

[56] Sibo Wang-Chen, Victor Alfred Stimpfling, Pembe Gizem Özdil, Louise Genoud, Femke Hurtak, and Pavan Ramdya. Neuromechfly 2.0, a framework for simulating embodied sensorimotor control in adult drosophila. *bioRxiv*, pages 2023–09, 2023.

[57] Philip K Shiu, Gabriella R Sterne, Nico Spiller, Romain Franconville, Andrea Sandoval, Joie Zhou, Neha Simha, Chan Hyuk Kang, Seongbong Yu, Jinseop S Kim, et al. A leaky integrate-and-fire computational model based on the connectome of the entire adult drosophila brain reveals insights into sensorimotor processing. *bioRxiv*, 2023.

[58] Sven Dorkenwald, Arie Matsliah, Amy R Sterling, Philipp Schlegel, Szi-Chieh Yu, Claire E McKellar, Albert Lin, Marta Costa, Katharina Eichler, Yijie Yin, et al. Neuronal wiring diagram of an adult brain. *bioRxiv*, 2023.

[59] Hengji Wang, Joshua Swore, Shashank Sharma, John R Szymanski, Rafael Yuste, Thomas L Daniel, Michael Regnier, Martha M Bosma, and Adrienne L Fairhall. A complete biomechanical model of hydra contractile behaviors, from neural drive to muscle to movement. *Proceedings of the National Academy of Sciences*, 120(11):e2210439120, 2023.

[60] Örjan Ekeberg and Sten Grillner. Simulations of neuromuscular control in lamprey swimming. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 354(1385):895–902, 1999.

[61] Eric D Tytell, Chia-Yu Hsu, Thelma L Williams, Avis H Cohen, and Lisa J Fauci. Interactions between internal forces, body stiffness, and fluid environment in a neuromechanical model of lamprey swimming. *Proceedings of the National Academy of Sciences*, 107(46):19832–19837, 2010.

[62] Martin Wadepuhl and Wolf-Jürgen Beyn. Computer simulation of the hydrostatic skeleton. the physical equivalent, mathematics and application to worm-like forms. *Journal of theoretical biology*, 136(4): 379–402, 1989.

[63] BA Skierczynski, RJA Wilson, WB Kristan Jr, and R Skalak. A model of the hydrostatic skeleton of the leech. *Journal of theoretical biology*, 181(4):329–342, 1996.

[64] Josh Merel, Diego Aldarondo, Jesse Marshall, Yuval Tassa, Greg Wayne, and Bence Ölveczky. Deep neuroethology of a virtual rodent. *arXiv preprint arXiv:1911.09451*, 2019.

[65] Stephen R Wicks, Chris J Roehrig, and Catharine H Rankin. A dynamic network simulation of the

nematode tap withdrawal circuit: predictions concerning synaptic function using behavioral criteria. *Journal of Neuroscience*, 16(12):4017–4031, 1996.

[66] Oleg Tolstenkov, Petrus Van der Auwera, Wagner Steuer Costa, Olga Bazhanova, Tim M Gemeinhardt, Amelie CF Bergs, and Alexander Gottschalk. Functionally asymmetric motor neurons contribute to coordinating locomotion of caenorhabditis elegans. *Elife*, 7:e34997, 2018.

[67] Yung-Chi Huang, Jinyue Luo, Wenjia Huang, Casey M Baker, Matthew A Gomes, Bohan Meng, Alexandra B Byrne, and Steven W Flavell. A single neuron in c. elegans orchestrates multiple motor outputs through parallel modes of transmission. *Current Biology*, 33(20):4430–4445, 2023.

[68] Jun Meng, Tosif Ahamed, Bin Yu, Wesley Hung, Sonia Ei Mouridi, Zezhen Wang, Yongning Zhang, Quan Wen, Thomas Boulin, Shangbang Gao, et al. A tonically active master neuron modulates mutually exclusive motor states at two timescales. *Science Advances*, 10(15):eadk0002, 2024.

[69] Eduardo J Izquierdo and Randall D Beer. Connecting a connectome to behavior: an ensemble of neuroanatomical models of c. elegans klinotaxis. *PLoS computational biology*, 9(2):e1002890, 2013.

[70] Mohan Chen, Dazheng Feng, Hongtao Su, Tingting Su, and Meng Wang. Neural model generating klinotaxis behavior accompanied by a random walk based on c. elegans connectome. *Scientific Reports*, 12(1):3043, 2022.

[71] Johannes Larsch, Steven W Flavell, Qiang Liu, Andrew Gordus, Dirk R Albrecht, and Cornelia I Bargmann. A circuit for gradient climbing in c. elegans chemotaxis. *Cell reports*, 12(11):1748–1760, 2015.

[72] Sarah G Leinwand and Sreekanth H Chalasani. Neuropeptide signaling remodels chemosensory circuit composition in caenorhabditis elegans. *Nature neuroscience*, 16(10):1461–1467, 2013.

[73] Sreekanth H Chalasani, Nikos Chronis, Makoto Tsunozaki, Jesse M Gray, Daniel Ramot, Miriam B Goodman, and Cornelia I Bargmann. Dissecting a circuit for olfactory behaviour in caenorhabditis elegans. *Nature*, 450(7166):63–70, 2007.

[74] Andrew Gordus, Navin Pokala, Sagi Levy, Steven W Flavell, and Cornelia I Bargmann. Feedback from network states generates variability in a probabilistic olfactory circuit. *Cell*, 161(2):215–227, 2015.

[75] Eduardo J Izquierdo, Paul L Williams, and Randall D Beer. Information flow through a model of the c. elegans klinotaxis circuit. *PloS one*, 10(10):e0140397, 2015.

[76] Ikue Mori and Yasumi Ohshima. Neural regulation of thermotaxis in caenorhabditis elegans. *Nature*, 376(6538):344–348, 1995.

[77] David Biron, Sara Wasserman, James H Thomas, Aravinthan DT Samuel, and Piali Sengupta. An olfactory neuron responds stochastically to temperature and modulates caenorhabditis elegans thermotactic behavior. *Proceedings of the National Academy of Sciences*, 105(31):11002–11007, 2008.

[78] Yen-Chih Chen, Hung-Jhen Chen, Wei-Chin Tseng, Jiun-Min Hsu, Tzu-Ting Huang, Chun-Hao Chen, and Chun-Liang Pan. A c. elegans thermosensory circuit regulates longevity through crh-1/creb-dependent flp-6 neuropeptide signaling. *Developmental cell*, 39(2):209–223, 2016.

[79] Martin Chalfie, John E Sulston, John G White, Eileen Southgate, J Nicol Thomson, and Sydney Brenner. The neural circuit for touch sensitivity in caenorhabditis elegans. *Journal of Neuroscience*, 5(4):956–964, 1985.

[80] Hagar Setty, Yehuda Salzberg, Shadi Karimi, Elisheva Berent-Barzel, Michael Krieg, and Meital Oren-Suissa. Sexually dimorphic architecture and function of a mechanosensory circuit in c. elegans. *Nature communications*, 13(1):6825, 2022.

[81] Miriam B Goodman and Erich M Schwarz. Transducing touch in caenorhabditis elegans. *Annual review of physiology*, 65(1):429–452, 2003.

[82] Wei Li, Lijun Kang, Beverly J Piggott, Zhaoyang Feng, and XZ Shawn Xu. The neural circuits and sensory channels mediating harsh touch sensation in caenorhabditis elegans. *Nature communications*, 2 (1):315, 2011.

[83] Jordan H Boyle, Stefano Berri, and Netta Cohen. Gait modulation in c. elegans: an integrated neuromechanical model. *Frontiers in computational neuroscience*, 6:10, 2012.

[84] Eduardo J Izquierdo and Randall D Beer. The whole worm: brain–body–environment models of c. elegans. *Current opinion in neurobiology*, 40:23–30, 2016.

[85] Eduardo J Izquierdo and Randall D Beer. From head to tail: a neuromechanical model of forward locomotion in caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170374, 2018.

[86] Kazuma Sakamoto, Zu Soh, Michiyo Suzuki, Yuichi Iino, and Toshio Tsuji. Forward and backward locomotion patterns in c. elegans generated by a connectome-based model simulation. *Scientific Reports*, 11(1):13737, 2021.

[87] Erick Olivares, Eduardo J Izquierdo, and Randall D Beer. A neuromechanical model of multiple network rhythmic pattern generators for forward locomotion in c. elegans. *Frontiers in computational neuroscience*, 15:572339, 2021.

[88] Thomas Maertens, Eckehard Schöll, Jorge Ruiz, and Philipp Hövel. Multilayer network analysis of c. elegans: Looking into the locomotory circuitry. *Neurocomputing*, 427:238–261, 2021.

[89] Eduardo J Izquierdo and Shawn R Lockery. Evolution and analysis of minimal neural circuits for klinotaxis in caenorhabditis elegans. *Journal of Neuroscience*, 30(39):12908–12917, 2010.

[90] Xin Deng, Jian-Xin Xu, Jin Wang, Guo-yin Wang, and Qiao-song Chen. Biological modeling the undulatory locomotion of c. elegans using dynamic neural network approach. *Neurocomputing*, 186: 207–217, 2016.

[91] Zu Soh, Kazuma Sakamoto, Michiyo Suzuki, Yuichi Iino, and Toshio Tsuji. A computational model of internal representations of chemical gradients in environments for chemotaxis of caenorhabditis elegans. *Scientific Reports*, 8(1):17190, 2018.

[92] Padraig Gleeson, David Lung, Radu Grosu, Ramin Hasani, and Stephen D Larson. c302: a multiscale framework for modelling the nervous system of caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170379, 2018.

[93] James Kunert, Eli Shlizerman, and J Nathan Kutz. Low-dimensional functionality of complex network dynamics: Neurosensory integration in the caenorhabditis elegans connectome. *Physical Review E*, 89 (5):052805, 2014.

[94] James M Kunert-Graf, Eli Shlizerman, Andrew Walker, and J Nathan Kutz. Multistability and long-timescale transients encoded by network structure in a model of c. elegans connectome dynamics. *Frontiers in computational neuroscience*, 11:53, 2017.

[95] Andrey Palyanov, Sergey Khayrulin, and Stephen D Larson. Three-dimensional simulation of the caenorhabditis elegans body and muscle cells in liquid and gel environments for behavioural analysis. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170376, 2018.

[96] Matteo Cantarelli, Boris Marin, Adrian Quintana, Matt Earnshaw, Robert Court, Padraig Gleeson, Salvador Dura-Bernal, R Angus Silver, and Giovanni Idili. Geppetto: a reusable modular open platform for exploring neuroscience data and models. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170380, 2018.

[97] Anthony Zador, Sean Escola, Blake Richards, Bence Ölveczky, Yoshua Bengio, Kwabena Boahen, Matthew Botvinick, Dmitri Chklovskii, Anne Churchland, Claudia Clopath, et al. Catalyzing next-generation artificial intelligence through neuroai. *Nature communications*, 14(1):1597, 2023.

[98] Megan Morrison and Lai-Sang Young. A data-driven biophysical network model reproduces c. elegans premotor neural dynamics. *arXiv preprint arXiv:2501.00278*, 2024.

[99] Lu Mi, Richard Xu, Sridhama Prakhya, Albert Lin, Nir Shavit, Aravinthan Samuel, and Srinivas C Turaga. Connectome-constrained latent variable model of whole-brain neural activity. In *International Conference on Learning Representations*, 2021.

[100] Janne K Lappalainen, Fabian D Tschopp, Sridhama Prakhya, Mason McGill, Aljoscha Nern, Kazunori Shinomiya, Shin-ya Takemura, Eyal Gruntman, Jakob H Macke, and Srinivas C Turaga. Connectome-constrained deep mechanistic networks predict neural responses across the fly visual system at single-neuron resolution. *bioRxiv*, pages 2023–03, 2023.

[101] Ruxandra Barbulescu, Gonçalo Mestre, Arlindo L Oliveira, and Luís Miguel Silveira. Learning the dynamics of realistic models of c. elegans nervous system with recurrent neural networks. *Scientific Reports*, 13(1):467, 2023.

[102] Zhongyu Chen, Yuguo Yu, and Xiangyang Xue. A connectome-based digital twin caenorhabditis elegans capable of intelligent sensorimotor behavior. *Mathematics*, 11(11):2442, 2023.

[103] Mengdi Zhao, Ning Wang, Xinrui Jiang, Xiaoyang Ma, Haixin Ma, Gan He, Kai Du, Lei Ma, and Tiejun Huang. An integrative data-driven model simulating c. elegans brain, body and environment interactions. *Nature Computational Science*, pages 1–13, 2024.

[104] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[105] Jimin Kim, Jeremy T Florman, Julia A Santos, Mark J Alkema, and Eli Shlizerman. Modular integration of neural connectomics, dynamics and biomechanics for identification of behavioral sensorimotor pathways in caenorhabditis elegans. *arXiv preprint arXiv:2504.18073*, 2025.

[106] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

[107] Allan R Willms. Neurofit: software for fitting hodgkin–huxley models to voltage-clamp data. *Journal of neuroscience methods*, 121(2):139–150, 2002.

[108] Allan R Willms, Deborah J Baro, Ronald M Harris-Warrick, and John Guckenheimer. An improved parameter estimation method for hodgkin-huxley models. *Journal of computational neuroscience*, 6: 145–168, 1999.

[109] Laure Buhry, Michele Pace, and Sylvain Saïghi. Global parameter estimation of an hodgkin–huxley formalism using membrane voltage recordings: Application to neuro-mimetic analog integrated circuits. *Neurocomputing*, 81:75–85, 2012.

[110] Juan Luis Jiménez Laredo, Loïs Naudin, Nathalie Corson, and Carlos M Fernandes. A methodology for determining ion channels from membrane potential neuronal recordings. In *Applications of Evolutionary*

Computation: 25th European Conference, EvoApplications 2022, Held as Part of EvoStar 2022, Madrid, Spain, April 20–22, 2022, Proceedings, pages 15–29. Springer, 2022.

[111] Loïs Naudin, Juan Luis Jiménez Laredo, Qiang Liu, and Nathalie Corson. Systematic generation of biophysically detailed models with generalization capability for non-spiking neurons. *PloS one*, 17(5): e0268380, 2022.

[112] Y Curtis Wang, Johann Rudi, James Velasco, Nirvik Sinha, Gideon Idumah, Randall K Powers, Charles J Heckman, and Matthieu K Chardon. Multimodal parameter spaces of a complex multi-channel neuron model. 2022.

[113] Jemy A Mandujano Valle and Alexandre L Madureira. Parameter identification problem in the hodgkin-huxley model. *Neural Computation*, 34(4):939–970, 2022.

[114] Pedro J Gonçalves, Jan-Matthis Lueckmann, Michael Deistler, Marcel Nonnenmacher, Kaan Öcal, Giacomo Bassetto, Chaitanya Chintaluri, William F Podlaski, Sara A Haddad, Tim P Vogels, et al. Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*, 9: e56261, 2020.

[115] Lautaro Estienne. Towards an hybrid hodgkin-huxley action potential generation model. In *2021 XIX Workshop on Information Processing and Control (RPIC)*, pages 1–6. IEEE, 2021.

[116] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[117] Gal Haspel, Edward S Boyden, Jeffrey Brown, George Church, Netta Cohen, Christopher Fang-Yen, Steven Flavell, Miriam B Goodman, Anne C Hart, Oliver Hobert, et al. To reverse engineer an entire nervous system. *arXiv preprint arXiv:2308.06578*, 2023.

[118] Eduardo J Izquierdo. Role of simulation models in understanding the generation of behavior in c. áelegans. *Current Opinion in Systems Biology*, 13:93–101, 2019.

[119] Netta Cohen and Jack E Denham. Whole animal modeling: piecing together nematode locomotion. *Current Opinion in Systems Biology*, 13:150–160, 2019.

[120] Eric Brown, Jeff Moehlis, and Philip Holmes. On the phase reduction and response dynamics of neural oscillator populations. *Neural computation*, 16(4):673–715, 2004.

[121] Jordan Snyder, Anatoly Zlotnik, and Aric Hagberg. Stability of entrainment of a continuum of coupled oscillators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103108, 2017.

[122] Jeff Moehlis, Eric Shea-Brown, and Herschel Rabitz. Optimal inputs for phase models of spiking neurons. *Journal of Computational and Nonlinear Dynamics*, 2006.

[123] Gábor Orosz, Jeff Moehlis, and Richard M Murray. Controlling biological networks by time-delayed signals. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1911):439–454, 2010.

[124] Per Danzl, João Hespanha, and Jeff Moehlis. Event-based minimum-time control of oscillatory neuron models. *Biological cybernetics*, 101(5-6):387, 2009.

[125] BA Mitchell and LR Petzold. Control of neural systems at multiple scales using model-free, deep reinforcement learning. *Scientific reports*, 8(1):1–12, 2018.

[126] Gabriella Panuccio, Arthur Guez, Robert Vincent, Massimo Avoli, and Joelle Pineau. Adaptive control of epileptiform excitability in an in vitro model of limbic seizures. *Experimental neurology*, 241:179–183, 2013.

[127] Arthur Guez, Robert D Vincent, Massimo Avoli, and Joelle Pineau. Adaptive treatment of epilepsy via batch-mode reinforcement learning. In *AAAI*, pages 1671–1678, 2008.

[128] Joelle Pineau, Arthur Guez, Robert Vincent, Gabriella Panuccio, and Massimo Avoli. Treating epilepsy via adaptive neurostimulation: a reinforcement learning approach. *International journal of neural systems*, 19(04):227–240, 2009.

[129] Sho Shirasaka, Nobuhiro Watanabe, Yoji Kawamura, and Hiroya Nakao. Optimizing stability of mutual synchronization between a pair of limit-cycle oscillators with weak cross coupling. *Physical Review E*, 96(1):012223, 2017.

[130] Eli Shlizerman, Jeffrey A Riffell, and J Nathan Kutz. Data-driven inference of network connectivity for modeling the dynamics of neural codes in the insect antennal lobe. *Frontiers in computational neuroscience*, 8:70, 2014.

[131] Ithai Rabinowitch, Bishal Upadhyaya, Aaradhya Pant, and Jihong Bai. Repairing neural damage in a c. elegans chemosensory circuit using genetically engineered synapses. *bioRxiv*, 2020.

[132] Franciszek Rakowski, Jagan Srinivasan, Paul W Sternberg, and Jan Karbowski. Synaptic polarity of the interneuron circuit controlling c. elegans locomotion. *Frontiers in computational neuroscience*, 7: 128, 2013.

[133] Eli Shlizerman. Driving the connectome by-wire: Comment on “what would a synthetic connectome look like?” by ithai rabinowitch. *Physics of life reviews*, 2019.

[134] Ithai Rabinowitch. What would a synthetic connectome look like? *Physics of life reviews*, 2019.

[135] Brooke L Sinnen, Aaron B Bowen, Jeffrey S Forte, Brian G Hiester, Kevin C Crosby, Emily S Gibson, Mark L Dell’Acqua, and Matthew J Kennedy. Optogenetic control of synaptic composition and function. *Neuron*, 93(3):646–660, 2017.

[136] Hexuan Liu, Jimin Kim, and Eli Shlizerman. Functional connectomics from neural dynamics: probabilistic graphical models for neuronal network of caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170377, 2018.

[137] Jimin Kim, Julia A Santos, Mark J Alkema, and Eli Shlizerman. Whole integration of neural connectomics, dynamics and bio-mechanics for identification of behavioral sensorimotor pathways in caenorhabditis elegans. *bioRxiv*, page 724328, 2019.

[138] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

[139] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau,

Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[140] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007. doi: 10.1109/MCSE.2007.55.

[141] Christopher Rackauckas and Qing Nie. Differentialequations. jl–a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of open research software*, 5(1):15–15, 2017.

[142] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017. doi: 10.1137/141000671. URL `https://epubs.siam.org/doi/10.1137/141000671`.

[143] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, et al. Jupyter notebooks–a publishing format for reproducible computational workflows. In *Positioning and power in academic publishing: Players, agents and agendas*, pages 87–90. IOS press, 2016.

[144] Padraig Gleeson, Sharon Crook, Robert C Cannon, Michael L Hines, Guy O Billings, Matteo Farinella, Thomas M Morse, Andrew P Davison, Subhasis Ray, Upinder S Bhalla, et al. Neuroml: a language for describing data driven models of neurons and networks with a high degree of biological detail. *PLoS computational biology*, 6(6):e1000815, 2010.

[145] Ankur Sinha, Padraig Gleeson, Boris Marin, Salvador Dura-Bernal, Sotirios Panagiotou, Sharon Crook, Matteo Cantarelli, Robert C Cannon, Andrew P Davison, Harsha Gurnani, et al. The neuroml ecosystem for standardized multi-scale modeling in neuroscience. *bioRxiv*, pages 2023–12, 2023.

[146] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[147] Christopher Rackauckas and Qing Nie. Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *The Journal of Open Research Software*, 5(1),

2017. doi: 10.5334/jors.151. URL `https://app.dimensions.ai/details/publication/pub.1085583166andhttp://openresearchsoftware.metajnl.com/articles/10.5334/jors.151/galley/245/download/`. Exported from https://app.dimensions.ai on 2019/05/05.

[148] Jimin Kim and Eli Shlizerman. Deep reinforcement learning for neural control. *arXiv preprint arXiv:2006.07352*, 2020.

[149] Barry Bentley, Robyn Branicky, Christopher L Barnes, Yee Lian Chew, Eviatar Yemini, Edward T Bullmore, Petra E Vértes, and William R Schafer. The multilayer connectome of caenorhabditis elegans. *PLoS computational biology*, 12(12):e1005283, 2016.

[150] Shay Stern, Christoph Kirst, and Cornelia I Bargmann. Neuromodulatory control of long-term behavioral patterns and individuality across development. *Cell*, 171(7):1649–1662, 2017.

[151] Olaf Sporns and Edward T Bullmore. From connections to function: the mouse brain connectome atlas. *Cell*, 157(4):773–775, 2014.

[152] T McMillen and P Holmes. An elastic rod model for anguilliform swimming. *Journal of mathematical biology*, 53(5):843–886, 2006.

[153] David M Egelman and P Read Montague. Calcium dynamics in the extracellular space of mammalian neural tissue. *Biophysical journal*, 76(4):1856–1867, 1999.

[154] HJ Schatzmann. The calcium pump of the surface membrane and of the sarcoplasmic reticulum. *Annual review of physiology*, 51(1):473–485, 1989.

[155] Quan Wen, Michelle D Po, Elizabeth Hulme, Sway Chen, Xinyu Liu, Sen Wai Kwok, Marc Gershow, Andrew M Leifer, Victoria Butler, Christopher Fang-Yen, et al. Proprioceptive coupling within motor neurons drives c. elegans forward locomotion. *neuron*, 76(4):750–761, 2012.

[156] Brenner S Wood WB. The nematode caenorhabditis elegans. *Cold Spring Harbor Laboratory*, 1:988, 1988.

[157] H Hatze. A myocybernetic control model of skeletal muscle. *Biological cybernetics*, 25(2):103–119, 1977.

[158] H Hatze. A general myocybernetic control model of skeletal muscle. *Biological cybernetics*, 28(3): 143–157, 1978.

[159] Robert Riener and Jochen Quintern. A physiologically based model of muscle activation verified by electrical stimulation. *Bioelectrochemistry and Bioenergetics*, 43(2):257–264, 1997.

[160] Örjan Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biological cybernetics*, 69(5-6):363–374, 1993.

[161] Ernst Niebur and Paul Erdös. Theory of the locomotion of nematodes: dynamics of undulatory progression on a surface. *Biophysical journal*, 60(5):1132–1146, 1991.

[162] Graham Bowtell and Thelma Williams. Anguilliform body dynamics: modelling the interaction between muscle activation and body curvature. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 334(1271):385–390, 1991.

[163] Graham Bowtell and Thelma L Williams. Anguilliform body dynamics: a continuum model for the interaction between muscle activation and body curvature. *Journal of mathematical biology*, 32(2):83–91, 1994.

[164] JC Carling, G Bowtell, and TL Williams. Swimming in the lamprey: Modeling the neural pattern generation, the body dynamics and the fluid dynamics. *Mechanics and physiology of animal swimming*, 1994.

[165] Greg J Stephens, Bethany Johnson-Kerner, William Bialek, and William S Ryu. Dimensionality and dynamics in the behavior of c. elegans. *PLoS computational biology*, 4(4):e1000028, 2008.

[166] André EX Brown, Eviatar I Yemini, Laura J Grundy, Tadas Jucikas, and William R Schafer. A dictionary of behavioral motifs reveals clusters of genes affecting caenorhabditis elegans locomotion. *Proceedings of the National Academy of Sciences*, 110(2):791–796, 2013.

[167] Beverly J Piggott, Jie Liu, Zhaoyang Feng, Seth A Wescott, and XZ Shawn Xu. The neural circuits and synaptic mechanisms underlying motor initiation in c. elegans. *Cell*, 147(4):922–933, 2011.

[168] Saul Kato, Yifan Xu, Christine E Cho, LF Abbott, and Cornelia I Bargmann. Temporal responses of c. elegans chemosensory neurons are preserved in behavioral dynamics. *Neuron*, 81(3):616–628, 2014.

[169] Saul Kato, Harris S Kaplan, Tina Schrödel, Susanne Skora, Theodore H Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor command sequence of caenorhabditis elegans. *Cell*, 163(3):656–669, 2015.

[170] Jeffrey P Nguyen, Frederick B Shipley, Ashley N Linder, George S Plummer, Mochi Liu, Sagar U Setru, Joshua W Shaevitz, and Andrew M Leifer. Whole-brain calcium imaging with cellular resolution in freely behaving caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 113(8): E1074–E1081, 2016.

[171] Jonathan T Pierce-Shimomura, Beth L Chen, James J Mun, Raymond Ho, Raman Sarkis, and Steven L McIntire. Genetic analysis of crawling and swimming locomotory patterns in c. elegans. *Proceedings of the National Academy of Sciences*, 105(52):20982–20987, 2008.

[172] Tianqi Xu, Jing Huo, Shuai Shao, Michelle Po, Taizo Kawano, Yangning Lu, Min Wu, Mei Zhen, and Quan Wen. Descending pathway facilitates undulatory wave propagation in caenorhabditis elegans through gap junctions. *Proceedings of the National Academy of Sciences*, 115(19):E4493–E4502, 2018.

[173] Eli Shlizerman, Konrad Schroder, and J Nathan Kutz. Neural activity measures and their dynamics. *SIAM Journal on Applied Mathematics*, 72(4):1260–1291, 2012.

[174] Jesse M Gray, Joseph J Hill, and Cornelia I Bargmann. A circuit for navigation in caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 102(9):3184–3191, 2005.

[175] Christopher Fang-Yen, Mark J Alkema, and Aravinthan DT Samuel. Illuminating neural circuits and behaviour in caenorhabditis elegans with optogenetics. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1677):20140212, 2015.

[176] Gang Yan, Petra E Vértes, Emma K Towlson, Yee Lian Chew, Denise S Walker, William R Schafer, and Albert-László Barabási. Network control principles predict neuron function in the caenorhabditis elegans connectome. *Nature*, 550(7677):519–523, 2017.

[177] Wei Li, Zhaoyang Feng, Paul W Sternberg, and XZ Shawn Xu. A c. elegans stretch receptor neuron revealed by a mechanosensitive trp channel homologue. *Nature*, 440(7084):684–687, 2006.

[178] Xiao N Shen, J Sznitman, P Krajacic, T Lamitina, and PE Arratia. Undulatory locomotion of caenorhabditis elegans on wet surfaces. *Biophysical journal*, 102(12):2772–2781, 2012.

[179] Christopher Fang-Yen, Matthieu Wyart, Julie Xie, Risa Kawai, Tom Kodger, Sway Chen, Quan Wen, and Aravinthan DT Samuel. Biomechanical analysis of gait adaptation in the nematode caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 107(47):20323–20328, 2010.

[180] Jack E Denham, Thomas Ranner, and Netta Cohen. Signatures of proprioceptive control in caenorhabditis elegans locomotion. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373 (1758):20180208, 2018.

[181] Elizabeth R Sawin, Rajesh Ranganathan, and H Robert Horvitz. C. elegans locomotory rate is modulated by the environment through a dopaminergic pathway and by experience through a serotonergic pathway. *Neuron*, 26(3):619–631, 2000.

[182] Netta Cohen and Tom Sanders. Nematode locomotion: dissecting the neuronal–environmental loop. *Current opinion in neurobiology*, 25:99–106, 2014.

[183] Michael Krieg, Aleksandra Pidde, and Ravi Das. Mechanosensitive body–brain interactions in caenorhabditis elegans. *Current Opinion in Neurobiology*, 75:102574, 2022.

[184] Hongfei Ji, Anthony D Fouad, Zihao Li, Andrew Ruba, and Christopher Fang-Yen. A proprioceptive feedback circuit drives caenorhabditis elegans locomotor adaptation through dopamine signaling. *Proceedings of the National Academy of Sciences*, 120(20):e2219341120, 2023.

[185] Heather L More and J Maxwell Donelan. Scaling of sensorimotor delays in terrestrial mammals. *Proceedings of the Royal Society B*, 285(1885):20180613, 2018.

[186] Navin Pokala, Qiang Liu, Andrew Gordus, and Cornelia I Bargmann. Inducible and titratable silencing of caenorhabditis elegans neurons in vivo with histamine-gated chloride channels. *Proceedings of the National Academy of Sciences*, 111(7):2770–2775, 2014.

[187] Jamie L Donnelly, Christopher M Clark, Andrew M Leifer, Jennifer K Pirri, Marian Haburcak, Michael M Francis, Aravinthan DT Samuel, and Mark J Alkema. Monoaminergic orchestration of motor programs in a complex c. elegans behavior. *PLoS biology*, 11(4):e1001529, 2013.

[188] Jennifer K Pirri, Diego Rayes, and Mark J Alkema. A change in the ion selectivity of ligand-gated ion channels provides a mechanism to switch behavior. *PLoS Biology*, 13(9):e1002238, 2015.

[189] Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[190] Netta Cohen and Thomas Ranner. A new computational method for a model of c. elegans biomechanics: Insights into elasticity and locomotion performance. *arXiv preprint arXiv:1702.04988*, 2017.

[191] Benny HH Cheung, Merav Cohen, Candida Rogers, Onder Albayram, and Mario De Bono. Experience-dependent modulation of c. elegans behavior by ambient oxygen. *Current Biology*, 15(10):905–917, 2005.

[192] Andy J Chang, Nikolas Chronis, David S Karow, Michael A Marletta, and Cornelia I Bargmann. A distributed chemosensory circuit for oxygen preference in c. elegans. *PLoS Biol*, 4(9):e274, 2006.

[193] Karl Emanuel Busch, Patrick Laurent, Zoltan Soltesz, Robin Joseph Murphy, Olivier Faivre, Berthold Hedwig, Martin Thomas, Heather L Smith, and Mario De Bono. Tonic signaling from o2 sensors sets neural circuit activity and behavioral state. *Nature neuroscience*, 15(4):581–591, 2012.

[194] Jesse M Gray, David S Karow, Hang Lu, Andy J Chang, Jennifer S Chang, Ronald E Ellis, Michael A Marletta, and Cornelia I Bargmann. Oxygen sensation and social feeding mediated by a c. elegans guanylate cyclase homologue. *Nature*, 430(6997):317–322, 2004.

[195] Manuel Zimmer, Jesse M Gray, Navin Pokala, Andy J Chang, David S Karow, Michael A Marletta,

Martin L Hudson, David B Morton, Nikos Chronis, and Cornelia I Bargmann. Neurons detect increases and decreases in oxygen levels using distinct guanylate cyclases. *Neuron*, 61(6):865–879, 2009.

[196] Candida Rogers, Annelie Persson, Benny Cheung, and Mario de Bono. Behavioral motifs and neural pathways coordinating o2 responses and aggregation in c. elegans. *Current biology*, 16(7):649–659, 2006.

[197] Patrick Laurent, Zoltan Soltesz, Geoffrey M Nelson, Changchun Chen, Fausto Arellano-Carbajal, Emmanuel Levy, and Mario de Bono. Decoding a neural circuit controlling global animal state in c. elegans. *Elife*, 4:e04241, 2015.

[198] Ashley E Raba, Jonathan M Cordeiro, Charles Antzelevitch, and Jacques Beaumont. Extending the conditions of application of an inversion of the hodgkin–huxley gating model. *Bulletin of mathematical biology*, 75:752–773, 2013.

[199] Loïs Naudin. Different parameter solutions of a conductance-based model that behave identically are not necessarily degenerate. *Journal of Computational Neuroscience*, pages 1–6, 2023.

[200] Eve Marder and Adam L Taylor. Multiple models to capture the variability in biological neurons and networks. *Nature neuroscience*, 14(2):133–138, 2011.

[201] Astrid A Prinz, Cyrus P Billimoria, and Eve Marder. Alternative to hand-tuning conductance-based models: construction and analysis of databases of model neurons. *Journal of neurophysiology*, 2003.

[202] Astrid A Prinz, Dirk Bucher, and Eve Marder. Similar network activity from disparate circuit parameters. *Nature neuroscience*, 7(12):1345–1352, 2004.

[203] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63 (11):139–144, 2020.

[204] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[205] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[206] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11315–11325, 2022.

[207] Xiulong Liu, Kun Su, and Eli Shlizerman. Tell what you hear from what you see–video to audio generation through text. *arXiv preprint arXiv:2411.05679*, 2024.

[208] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[209] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[210] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[211] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[212] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[213] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[214] Loïs Naudin, Nathalie Corson, MA Aziz-Alaoui, Juan Luis Jimenez Laredo, and Thibaut Démare. On the modeling of the three types of non-spiking neurons of the caenorhabditis elegans. *International Journal of Neural Systems*, 31(02):2050063, 2021.

[215] Eugene M Izhikevich. *Dynamical systems in neuroscience*. MIT press, 2007.

[216] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In *Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, September 18–20, 2000 Proceedings 6*, pages 849–858. Springer, 2000.

[217] Etay Hay, Sean Hill, Felix Schürmann, Henry Markram, and Idan Segev. Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. *PLoS computational biology*, 7(7):e1002107, 2011.

[218] Werner Van Geit, Erik De Schutter, and Pablo Achard. Automated neuron model optimization techniques: a review. *Biological cybernetics*, 99:241–251, 2008.

[219] Tea Robič and Bogdan Filipič. Differential evolution for multiobjective optimization. In *Evolutionary Multi-Criterion Optimization: Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings 3*, pages 520–533. Springer, 2005.

[220] Saku Kukkonen and Jouni Lampinen. Gde3: The third evolution step of generalized differential evolution. In *2005 IEEE congress on evolutionary computation*, volume 1, pages 443–450. IEEE, 2005.

[221] Rakesh Angira and BV Babu. Non-dominated sorting differential evolution (nsde): An extension of differential evolution for multi-objective optimization. In *IICAI*, pages 1428–1443, 2005.

[222] Timothy H Rumbell, Danel Draguljić, Aniruddha Yadav, Patrick R Hof, Jennifer I Luebke, and Christina M Weaver. Automated evolutionary optimization of ion channel conductances and kinetics in models of young and aged rhesus monkey pyramidal neurons. *Journal of computational neuroscience*, 41:65–90, 2016.

[223] J Christopher Octeau, Mohitkumar R Gangwani, Sushmita L Allam, Duy Tran, Shuhan Huang, Tuan M Hoang-Trong, Peyman Golshani, Timothy H Rumbell, James R Kozloski, and Baljit S Khakh. Transient, consequential increases in extracellular potassium ions accompany channelrhodopsin2 excitation. *Cell reports*, 27(8):2249–2261, 2019.

[224] Laure Buhry, Audrey Giremus, Eric Grivel, Sylvain Saïghi, and Sylvie Renaud. New variants of

the differential evolution algorithm: application for neuroscientists. In *2009 17th European Signal Processing Conference*, pages 2352–2356. IEEE, 2009.

[225] Julian Blank and Kalyanmoy Deb. Pymoo: Multi-objective optimization in python. *IEEE Access*, 8: 89497–89509, 2020.

[226] Loïs Naudin, Juan Luis Jiménez Laredo, and Nathalie Corson. A simple model of nonspiking neurons. *Neural Computation*, 34(10):2075–2101, 2022.

[227] Jinming Zou, Yi Han, and Sung-Sau So. Overview of artificial neural networks. *Artificial neural networks: methods and applications*, pages 14–22, 2009.

[228] Oliver Hobert. The neuronal genome of caenorhabditis elegans. *WormBook: The online review of C. elegans biology [Internet]*, 2018.

[229] Paul Brooks, Andrew Champion, and Marta Costa. Mapping of the zebrafish brain takes shape. *Nature Methods*, pages 1–2, 2022.

[230] Michael Winding, Benjamin D Pedigo, Christopher L Barnes, Heather G Patsolic, Youngser Park, Tom Kazimiers, Akira Fushiki, Ingrid V Andrade, Avinash Khandelwal, Javier Valdes-Aleman, et al. The connectome of an insect brain. *Science*, 379(6636):eadd9330, 2023.

[231] Seung Wook Oh, Julie A Harris, Lydia Ng, Brent Winslow, Nicholas Cain, Stefan Mihalas, Quanxin Wang, Chris Lau, Leonard Kuan, Alex M Henry, et al. A mesoscale connectome of the mouse brain. *Nature*, 508(7495):207–214, 2014.

[232] Nicholas James Sofroniew, Daniel Flickinger, Jonathan King, and Karel Svoboda. A large field of view two-photon mesoscope with subcellular resolution for in vivo imaging. *elife*, 5:e14472, 2016.

[233] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[234] Juan Wu, Seabyuk Shin, Cheong-Gil Kim, and Shin-Dug Kim. Effective lazy training method for deep

q-network in obstacle avoidance and path planning. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1799–1804. IEEE, 2017.

[235] Fabio Pardo, Arash Tavakoli, Vitaly Levdik, and Petar Kormushev. Time limits in reinforcement learning. *arXiv preprint arXiv:1712.00378*, 2017.

[236] Eyal Itskovits, Rotem Ruach, Alexander Kazakov, and Alon Zaslaver. Concerted pulsatile and graded neural dynamics enables efficient chemotaxis in c. elegans. *Nature communications*, 9(1):2866, 2018.

[237] Fleur L Strand. *Neuropeptides: regulators of physiological processes*. MIT press, 1999.

[238] Chiara Salio, Laura Lossi, Francesco Ferrini, and Adalberto Merighi. Neuropeptides as synaptic transmitters. *Cell and tissue research*, 326(2):583–598, 2006.

[239] Matthew Beverly, Sriram Anbil, and Piali Sengupta. Degeneracy and neuromodulation among thermosensory neurons contribute to robust thermosensory behaviors in caenorhabditis elegans. *Journal of Neuroscience*, 31(32):11718–11727, 2011.

[240] Atsushi Kuhara, Masatoshi Okumura, Tsubasa Kimata, Yoshinori Tanizawa, Ryo Takano, Koutarou D Kimura, Hitoshi Inada, Kunihiro Matsumoto, and Ikue Mori. Temperature sensing by an olfactory neuron in a circuit controlling behavior of c. elegans. *Science*, 320(5877):803–807, 2008.

[241] Christopher V Gabel, Harrison Gabel, Dmitri Pavlichin, Albert Kao, Damon A Clark, and Aravinthan DT Samuel. Neural circuits mediate electrosensory behavior in caenorhabditis elegans. *Journal of Neuroscience*, 27(28):7586–7596, 2007.

[242] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[243] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.