A NOVEL NEURAL NETWORK ANALYSIS METHOD APPLIED TO BIOLOGICAL NEURAL NETWORKS

by

NATHAN A. DUNN

A DISSERTATION

Presented to the Department of Computer
and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

August 2006

"A Novel Neural Network Analysis Method Applied to Biological Neural Networks," a dissertation prepared by Nathan A. Dunn in partial fulfillment of the requirements for the Doctor of Philosophy degree in the Department of Computer and Information Science. This dissertation has been approved and accepted by:

Dr. John S. Conery, Co-chair of the Examining Committee

Dr. Shawn R. Lockery, Co-chair of the Examining Committee

8/18/06
Date

Committee in charge:	Dr. John S. Conery, Co-chair Dr. Shawn R. Lockery, Co-chair Dr. Allen D. Malony Dr. Reza Rejaie Dr. William M. Roberts

Accepted by:

Dean of the Graduate School

Copyright 2006 Nathan A. Dunn

An Abstract of the Dissertation of

Nathan A. Dunn for the degree of Doctor of Philosophy

in the Department of Computer and Information Science

to be taken August 2006

Title: A NOVEL NEURAL NETWORK ANALYSIS METHOD APPLIED

TO BIOLOGICAL NEURAL NETWORKS

Approved: ____________________________________

Dr. John S. Conery, Co-chair

____

Dr. Shawn R. Lockery, Co-chair

This thesis makes two major contributions: it introduces a novel method for analysis of artificial neural networks and provides new models of the nematode *Caenorhabditis elegans* nervous system. The analysis method extracts neural network motifs, or subnetworks of recurring neuronal function, from optimized neural networks. The method first creates models for each neuron relating network stimulus to neuronal response, then clusters the model parameters, and finally combines the neurons into multi-neuron motifs based on their cluster category. To infer biological function, this analysis method was applied to neural networks optimized to reproduce *C. elegans* behavior, which converged upon a small number of motifs. This allowed both a quantitative exploration of network function as well as discovery of larger motifs.

Neural network models of *C. elegans* anatomical connectivity were optimized to reproduce two *C. elegans* behaviors: chemotaxis (orientation towards a maximum chemical attractant concentration) and thermotaxis (orientation towards a set temperature). Three chemotaxis motifs were identified. Experimental evidence suggests that chemotaxis is driven by a differentiator motif with two important features. The first feature was a fast, excitatory pathway in parallel with one or more slow, inhibitory pathways. The second feature was inhibitory feedback on all self-connections and recurrent loops, which regulates neuronal response. Six thermotaxis motifs were identified. Every motif consisted of two circuits, each a previously discovered chemotaxis motif with most having a dedicated sensory neuron. One circuit was thermophilic (heat-seeking) and the other was cryophilic (cold-seeking). Experimental evidence suggests that the cryophilic circuit is a differentiator motif and the thermophilic circuit functions by klinokinesis.

CURRICULUM VITA

NAME OF AUTHOR: Nathan A. Dunn

PLACE OF BIRTH: Centralia, WA, U.S.A.

DATE OF BIRTH: June 9, 1974

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon
University of Washington
Oregon State University

DEGREES AWARDED:

Doctor of Philosophy in Computer and Information Science, 2006, University of Oregon
Master of Science in Chemical Engineering, 1998, University of Washington
Bachelor of Science in Chemical Engineering, 1997, Oregon State University

AREAS OF SPECIAL INTEREST:

Neural Networks
Computational Biology
Caenorhabditis elegans
Artificial Intelligence

PROFESSIONAL EXPERIENCE:

Graduate Research Fellow, Lockery Lab, University of Oregon, 2001 - 2006

Teaching Assistant, Computer and Information Science, University of Oregon, 2005 - 2006

Software Engineer, Tenzing Communications, Seattle, WA, 2000-2001, 2000 - 2001

Consultant, Emerald Solutions, Seattle, WA, 1999-2000

GRANTS:

National Science Foundation, IBN-0080068
Intl Joint Conf on Neural Networks, Student Travel Grant, 2006

PUBLICATIONS:

Nathan A. Dunn, Jon T. Pierce-Shimomura, John S. Conery, and Shawn R. Lockery. Clustered Neural Dynamics Identify Motifs for Chemotaxis in *C. elegans*. In *Neural Networks, 2006. Proceedings of the International Joint Conference on Neural Networks*, IEEE, 2006 (in press).

Nathan A. Dunn, Jon T. Pierce-Shimomura, John S. Conery, and Shawn R. Lockery. A Neural Network Model of Chemotaxis Predicts Functions of Synaptic Connections in the Nematode *C. elegans*. *Journal of Computational Neuroscience*, pp. 137–147, 17(2), 2004.

Nathan A. Dunn, John S. Conery, and Shawn R. Lockery. Circuit optimization Predicts Dynamic Network for Chemosensory Orientation in the Nematode C. elegans. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

Nathan A. Dunn, J S. Conery, and S R. Lockery. A Neural Network Model for Chemotaxis in *Caenorhabditis elegans*. In *Neural Networks, 2003. Proceedings of the International Joint Conference on Neural Networks*, volume 4, pp. 2574–2578. IEEE, 2003.

ACKNOWLEDGMENTS

I would like thank my co-advisors John Conery and Shawn Lockery for their guidance and knowledge. I thank Allen Malony, Reza Rejaie, Art Farley, Michal Young, and Bill Roberts for valuable insight.

I thank past and present Lockery Lab members Serge Faumont, Tod Thiele, Michael Ailion, and Adam Miller and many others for sharing their knowledge and advice. I also thank past and present Conery Lab members Julian Catchen, Joel Berghoff, and Bryan Kolaczkowski for providing a community for computer scientists who work in biology.

I thank Don Pate for invaluable technical assistance and Kevin Huck for many discussions on clustering and other technical matters.

This research was sponsored by the National Science Foundation, IBN-0080068.

Finally, I thank my family for their interest and encouragement. I also thank my wife, Elisa, for her love and support.

To family.

TABLE OF CONTENTS

Chapter Page

1. INTRODUCTION 1

1.1 Contributions 3
  1.1.1 Computer Science Contribution: Neural Dynamic Clustering 3
  1.1.2 Biological Modeling Contribution: Neural Networks Model *C. elegans* Behavior. 7
1.2 Organization of Dissertation 8

2. BACKGROUND 9

2.1 *C. elegans* 9
  2.1.1 Chemotaxis 10
  2.1.2 Thermotaxis 11
2.2 Biological Neural Networks 11
  2.2.1 Nervous System 11
  2.2.2 Measurement 15
  2.2.3 Modification 17
  2.2.4 Modeling 17
2.3 Artificial Neural Networks 22
  2.3.1 Network Evaluation 23
  2.3.2 Neuronal Activity Functions 27
  2.3.3 Connectivity 29
2.4 Analysis of Artificial Neural Networks 30
  2.4.1 Architectural Analysis 32
  2.4.2 Analytical Analysis 35
  2.4.3 Signal Analysis 38

3. A NEURAL NETWORK MODEL PREDICTS *C. ELEGANS* CHEMOTAXIS 40

3.1 Introduction 40
3.2 Methods 42

- 3.2.1 Assumptions 42
- 3.2.2 Network 42
- 3.2.3 Optimization 46
- 3.2.4 Generalization 49
- 3.3 Results 50
  - 3.3.1 Common Features of Networks 50
  - 3.3.2 Output Delay 53
  - 3.3.3 Analysis 56
- 3.4 Conclusion 58

4. NEURAL NETWORK MOTIFS FOR CHEMOTAXIS AND THERMOTAXIS IN *C. ELEGANS* 61

- 4.1 Introduction 62
- 4.2 Methods 63
  - 4.2.1 Assumptions 63
  - 4.2.2 Model Worm Movement 64
  - 4.2.3 Neural Network Model 65
  - 4.2.4 Model Evaluation 67
  - 4.2.5 Optimization 68
  - 4.2.6 Motif Identification 69
  - 4.2.7 Generalization 69
- 4.3 Results 70
  - 4.3.1 Chemotaxis Motifs 70
  - 4.3.2 Thermotaxis Motifs 76
- 4.4 Discussion 82

5. NEURAL DYNAMIC CLUSTERING 86

- 5.1 Introduction 87
- 5.2 Method 87
  - 5.2.1 Create Neural Models 88
  - 5.2.2 Cluster Neural Models into Single-neuron Motifs 88
  - 5.2.3 Combine Motifs 89
- 5.3 Case Study: Neural Networks for *C. elegans* Chemotaxis 90
  - 5.3.1 Results 90
  - 5.3.2 Summary 98
- 5.4 Case Study: Recurrent Neural Networks Reproduce *C. elegans* Pirouette Mechanism 98
  - 5.4.1 Results 99

5.5 Conclusion 102

6. RELATED WORK 103

6.1 *C. elegans* Computational Models 103
  6.1.1 Chemotaxis 104
  6.1.2 Thermotaxis 105
6.2 Analysis of Neural Networks 106
  6.2.1 Analytical Analysis 107
  6.2.2 Architectural Analysis 109
  6.2.3 Signal Analysis 111
  6.2.4 Conclusion 115

7. CONCLUSION 117

7.1 Contributions 117
7.2 Future Work 118
  7.2.1 Anatomically Constrained *C. elegans* Chemotaxis 119
  7.2.2 Multi-Sensory Evolution 119
  7.2.3 Neural Dynamic Clustering 120

BIBLIOGRAPHY 122

LIST OF FIGURES

Figure Page

2.1 Components of biological neurons. . . . 13
2.2 Live neurons. . . . 14
2.3 Model of dendritic cable. . . . 18
2.4 Dendritic cable model. . . . 18
2.5 Compartmental model of a neural network. . . . 20
2.6 A neuron modeled as a capacitor. . . . 20
2.7 An artificial neural network. . . . 24
2.8 Sigmoid function. . . . 28
2.9 Hinton diagram representation of a neural network. . . . 34
2.10 Parallel-axis plot representation of a neural network. . . . 35
2.11 Error surface plot. . . . 36
2.12 Control theory used to identify an unknown system. . . . 39
3.1 A model chemosensory network. . . . 44
3.2 An idealized sensorimotor transformation. . . . 45
3.3 Network performance following optimization. . . . 48
3.4 Generalization: model worm within a radial gradient. . . . 51
3.5 Two connectivity patterns observed for single-interneuron networks. . . . 52
3.6 The effect of output delay (Δt) on network parameters. . . . 54
3.7 Analysis of two linear neurons with recurrent synapses. . . . 55
3.8 The network of chemosensory interneurons in *C. elegans*. . . . 59
4.1 Model gradients used for the chemotaxis and thermotaxis optimization problems. . . . 64
4.2 Model worm movement. . . . 66
4.3 Interneuron-differentiator motif. . . . 73
4.4 Sensory-differentiator motif. . . . 74
4.5 Bounce-and-trap motif. . . . 75
4.6 Origin of the trapping behavior asymmetry. . . . 77
4.7 Dual-bounce-and-trap motif. . . . 78
4.8 Dual-interneuron-differentiator motif. . . . 79
4.9 Distributions of simulated worms driven by thermotaxis motifs. . . . 83
5.1 Neural dynamics modeled with an NL block model. . . . 92
5.2 Distribution of NL block parameters. . . . 94
5.3 Motifs that reproduce the pirouette mechanism. . . . 101
6.1 Neural networks for identification and control of a black-box system. . . . 112
7.1 Sannapp application implementing Neural Dynamic Clustering. . . . 121

LIST OF TABLES

Table Page

3.1 Common features of single-interneuron networks. 52
4.1 Thermotaxis motifs. 80
5.1 Labeled cluster. 95
5.2 Single-neuron motifs generated from labeled clusters. 96
5.3 Multi-neuron motifs generated from single-neuron motifs. 97
5.4 Motifs from recurrent networks optimized to exhibit the pirouette mechanism. 100

CHAPTER 1

Introduction

This thesis describes a novel method for analysis of neural networks and develops new models of the nematode *Caenorhabditis elegans* nervous system. The models of the *C. elegans* nervous system were generated by optimizing multiple idealized neural networks to reproduce features of *C. elegans* behavior. To make inferences into the *C. elegans* nervous system, we needed to determine the subnetworks in the idealized neural networks that drive transient behavior. These idealized neural networks modeled transient behavior response, which is the response directly following a stimulus. As such, we are interested in the relationship between stimulus and transient neuronal response, or ***neuron function***, and transient response of the network as a whole, or ***network function***. We have termed the subnetworks of neuron function observed across multiple networks ***neural network motifs***.

Motifs may be identified qualitatively by observing neuron activation in optimized networks during behavior. However, qualitative observation is ill defined, prone to human error, and time-consuming. We propose a method that quantitatively identifies motifs. This method should characterize neuron function across multiple networks. Current methods have the following limitations when applied to finding motifs. First, structural methods analyze network parameters whose nonlinear combinations produce effects in network function. Furthermore, they have a large number of parameters. A recurrent neural network with N neurons may have up to N^2 connections between neurons. Second, analytical methods are unable to make inferences into transient behavior due to the intractable nature of neural networks. This is especially true for recurrent networks with nonlinear activation functions, which are both common in practice. Third, signal analysis methods accurately describe network function, but focus on defining only a single neural network and are never applied to individual neurons or across multiple neural networks.

In this dissertation, we present the Neural Dynamic Clustering method to compare the features of neuron function across multiple neural networks. A pattern of neuron function repeated across multiple network solutions implies any or all of the following: that the neuron function is important, it is easy for the optimization algorithm to find, and that it has a robust function. Because we are comparing neuron function, not network similarity, we can also compare neural network solutions of different sizes.

Application of Neural Dynamic Clustering is demonstrated in Chapter 5, where it is applied within the context of idealized neural networks that model biological function from Chapters 3 and 4. Application of this method allowed us to overcome previous limitations. We were able to reproduce motifs that were previously identified by qualitatively observation in Chapter 4. Neural Dynamic Clustering also allowed analysis of four- and five-neuron networks produced during optimization in Chapter 3. Previously, only the three-neuron networks had been analyzed due to limitations of analytical and structural methods.

The motivation for development of Neural Dynamic Clustering was modeling *C. elegans* behavior using idealized neural networks. *C. elegans* provides several advantages as a biological organism for modeling. First, it has only 302 neurons. Thus, individual neurons can be associated with specific function with minimal differences between animals. Second, *C. elegans* anatomy has been entirely reconstructed such that the connectivity between every neuron is known [WSTB86]. Third, several *C. elegans* orientation behaviors in response to sensory stimuli have been characterized [TH03, GHB05].

We optimized neural networks to reproduce *C. elegans* behavior in four steps. First, we created an idealized neural network model. Second, we optimized our neural network model to reproduce characterized behavior, which generated a set of unique neural networks. Third, from the set of neural networks, we extracted neural network motifs to infer *C. elegans* biological function necessary for behavior. Fourth, we compared the motifs we found to previous experimental evidence to make predictions about neurophysiological connectivity and function. The most difficult step was the extraction of motifs from the optimized neural networks we generated. As such, we developed Neural Dynamic Clustering.

1.1 Contributions

This dissertation contains two major contributions in the fields of computer science and biological modeling, respectively. First, it introduces a novel method, Neural Dynamic Clustering, which extracts neural network motifs from a set of neural networks based on neuron function. Second, it provides models of the *C. elegans* nervous system. Application of Neural Dynamic Clustering to our biological modeling problems has allowed us to analyze networks we could not previously analyze and make more quantitative inferences into network function. Although Neural Dynamic Clustering is applied in the context of biological modeling in this dissertation, it is also a significant contribution to computer science. As such, we discuss Neural Dynamic Clustering in the context of computer science and *C. elegans* modeling in the context of biological modeling.

1.1.1 Computer Science Contribution: Neural Dynamic Clustering

We developed the Neural Dynamic Clustering analysis method [DPSCL06], which is the only method that extracts motifs from sets of neural networks based on neuron function, to make inferences into biological models. However, Neural Dynamic Clustering is also an important contribution to computer science because neural networks are an important computer science construct. Furthermore, Neural Dynamic Clustering may be applied to any other graph-based structure that has nodes with continuous activity and history.

Neural Dynamic Clustering is a unique method, which allows identification of neural network motifs. However, this leads to the question, "Why would one want to extract neural network motifs?". We present several reasons why extraction of motifs is desired, as well as some specific applications and examples. On a purely theoretical level, identifying motifs increases understanding of the common subnetworks that drive network function. Much as better microscopes have allowed biologists to further knowledge of biological systems, better analysis methods will further knowledge of how neural networks function. Given a set of optimized neural networks, Neural Dynamic Clustering allows for:

- inference into black-box system processes;
- assessment of neuron function contribution to network function; and
- more accurate assessment of network solution convergence.

By optimizing a neural network to reproduce the behavior of a black-box system, we can infer that the motifs of the optimized network relate to physical processes within the black-box system. In a black-box system, internal processes are not known. Assessment of neuron function contribution to overall network function indicates how separate sub-motifs function simultaneously or for separate objectives. By determining the number of functionally distinct network solutions using motifs, we can more accurately assess the convergent nature of a given optimization algorithm. We describe each of these applications in detail as they apply to various neural network applications.

Inference into Black-Box System Components

Motifs identified from neural networks optimized to reproduce the function of a black-box system represent important components within the black-box system. Inferences into black-box system components have the most direct applications to engineering and biology. As an example, if we identify motifs from a neural network that reproduces the behavior of a pneumatic controller for a robot arm [NRPH00], we can use those motifs to infer the dominant physical processes within the controller/robot system. If a set of neural networks identifies the response properties of the controller, then the motifs of those networks may identify physical properties of the controller such as valve pressure, friction, and weight.

Biological systems may be represented also as black boxes. We demonstrate inference into biological systems in two case studies (Chapter 5) where neural networks were optimized to model biological behavior. By inferring recurring network function, we were able to make inferences into dynamics that may drive biological behavior. There are many other examples of optimizing idealized neural networks to reproduce transient biological processes such as *C. elegans* touch response [WR95], zebra finch song learning [HDNL$^+$04], cricket auditory response [DG06], and human movement [MSF03]. Neural Dynamic Clustering allows a more direct assessment of network function than previous methods.

Assessment of Neuron Function Contribution to Network Function

There are two ways that assessment of network function may be applied. First, it can determine how individual sub-motifs combine to produce network behavior. Second, it identifies modes of failure predicted by individual sub-motifs. One example of determining how individual sub-motifs contribute to network behavior is multi-objective optimization (MOOP) applications that employ neural networks [Deb01, wIH04]. For example, artificial intelligence (AI) that controls non-player characters (NPC) in video games may be driven by neural networks that have been optimized to achieve multiple objectives, such as avoiding enemy fire and firing at opponents [SBM05b]. However, individual sub-motifs may be associated with individual optimization tasks. By identifying motifs associated with a single task, structure may be identified and used to stub neural network optimization solutions, as is done in some network optimization algorithms [BM06]. Identification of motifs in well-performing network solutions also allows an understanding of important network features.

Determining modes of failure predicted by sub-motifs is important in engineering and artificial intelligence applications. When sub-motifs have been identified, failure scenarios of an entire network can be broken down by assessing failure scenarios within individual motifs. Using the previous example of a pneumatic controller for a robot arm [NRPH00], motifs from the network that model this system could be used to determine the most likely positions in which the robot arm is likely to fail (not go to a desired position or simply break). This is accomplished by understanding the response of individual sub-motifs that contribute to network function.

More Accurate Assessment of Network Solution Convergence

To determine the number of functionally unique network solutions created during optimization, we classify each network according to the largest motif it contains as identified by the Neural Dynamic Clustering method. The number of unique network solutions is a measure of convergence. A single functionally unique solution represents complete convergence. Conversely, if each network is given a different classification, the solutions are completely divergent. Convergence is important for assessing and monitoring network optimization algorithms. In industrial applications, which often experience significant white-noise input, only a single highly accurate network solution may be desired. By classifying solutions according to their motifs, we can assess whether the optimization algorithm converges upon network solutions with the same network function.

Additionally, convergence is a way to determine network optimization progress. Many network optimization algorithms are stochastic in nature and rely on divergent network solutions to adequately cover network parameter search space. For example, simulated annealing emphasizes divergence at the initial stages of the algorithm, but convergence to a single solution towards the end of optimization [Mas93]. Conversely, there are many genetic algorithms [SBM05a] that enforce divergence following periods of convergent optimization. However, much of this enforced divergence is done by analysis of structure, or genetic parameters that encode structure, as opposed to network function. Similarly, with Tabu search methods [BT95], search space is eliminated based on similarity of structure, rather than function. Analysis of network function is the only accurate way to measure network similarity. Enforcing divergence according to network function should make better use of network parameter search space.

1.1.2 Biological Modeling Contribution: Neural Networks Model *C. elegans* Behavior.

The biological modeling contribution in this thesis can be divided into two contributions:

- we identified neural networks features that reproduced the *C. elegans* pirouette mechanism of chemotaxis; and
- we identified motifs that reproduced either *C. elegans* chemotaxis or thermotaxis.

Neural Network Features the Pirouette Mechanism

This contribution [DCL03, DCL04, DCPSL04] characterized important features of a motif that reproduced an observed *C. elegans* chemotaxis strategy, the pirouette mechanism [PSML99], which states that a worm is more likely to randomly reorient when attractant concentration decreases. The motif we found produced reorientation probability by differentiating chemosensory stimulus. The differentiator circuit consisted of a fast, direct, excitatory pathway in parallel with a slow, inhibitory pathway. An unexpected feature of this motif was the occurrence of inhibitory feedback on self-connections and recurrent loops. We show that the inhibitory feedback acted to regulate latency.

Motifs for *C. elegans* Chemotaxis and Thermotaxis

This contribution, portions of which are published in [DPSCL06], characterized motifs capable of reproducing *C. elegans* chemotaxis or thermotaxis behavior in a model worm by any strategy. For chemotaxis, we found three motifs: the interneuron-differentiator, the sensory-differentiator, and the bounce-and-trap motif. Step data

[MTF^{+}05] validates the differentiator motifs, verifying the structure of the motifs from Chapter 3. The thermotaxis motifs we found consisted of two sub-motifs, each with a dedicated sensory neuron. One sub-motif was thermophilic (heat-seeking) and the other was cryophilic (cold-seeking). The sub-motifs were composed of the previously identified chemotaxis motifs. The thermotaxis motifs we found support a dual-network model [IIM06] and suggest likely function of the neurons involved in this strategy.

1.2 Organization of Dissertation

The remainder of the thesis is organized as follows. Chapter 2 provides background, which includes *C. elegans* biology and behavior, biological neural networks, modeling biological neural networks with artificial neural networks, and an overview of neural network analysis methods. Chapters 3 and 4 present *C. elegans* biological modeling contributions. Chapter 5 presents the Neural Dynamic Clustering network analysis method and applies it to the networks produced during biological modeling in previous chapters. Chapter 6 describes related work in two parts: *C. elegans* modeling and signal analysis in neural networks related to Neural Dynamic Clustering. Chapter 7 summarizes the contributions of this thesis and suggests areas of future work.

CHAPTER 2

Background

This section provides background for the remainder of our thesis. First, we describe *C. elegans* history, biology, and behavior. Next, we discuss biological neural networks and how they may be modeled by artificial neural networks. Finally, we describe methods for neural network analysis in which to frame our own analysis method.

2.1 *C. elegans*

C. elegans is a soil-dwelling nematode, or flatworm, that has been used as a model organism for several decades. It was selected as a model organism because its reproduces quickly, it has well understood genetics [Bre74, WW90], and anatomical connectivity within its nervous system is completely known [WSTB86]. The nervous system consists of only 302 neurons connected with approximately 6000 synaptic connections and gap junctions. *C. elegans* also exhibits several characterized behavior including chemotaxis and thermotaxis. In chemotaxis, the worm orients towards a maximal concentration of chemical attractant. In thermotaxis, the worm orients towards a temperature that it associates with food, which is typically its cultivation temperature. The worm orients in response to other types of stimulus including O^2, pH, and odorants. Additionally, the worm also exhibits avoidance behavior in response to noxious and mechanosensory stimuli.

Worm movement can be characterized by forward swimming runs interrupted by bouts of random turning. Forward swimming consists of undulatory motion that generates movement in the direction the head is pointing. Random turns consist of combinations of reversals and Ω bends. During reversals, the worm reverses its undulatory motion and propels itself backwards. During Ω bends, the worm makes a bend resembling an Ω and exits headed towards a random orientation. Two pools of neurons are believed to control overall locomotion: one pool is associated with forward locomotion and the other with turning [ZBM+99, WKS04].

2.1.1 Chemotaxis

C. elegans has been shown to be attracted to a number of water-soluble chemicals, *including anions, cations, and small organic molecules* [BH91, Dus74, War73]. Of importance to the work in this thesis is the worm's attraction to the salt molecule, $NaCl$ composed of the anion Cl^- and cation Na^+. Additionally the nature of the attractive behavior in chemical gradients has been quantitatively described by the pirouette mechanism [PSML99]. During chemotaxis, *C. elegans* normally orients towards a maximum chemical attractant concentration. Pirouette probability is modulated by the rate of change of chemical attractant concentration $(dC(t)/dt)$ [PSML99, MTF+05]. When $dC(t)/dt < 0$, pirouette probability is increased whereas when $dC(t)/dt > 0$, pirouette probability is decreased. Thus, runs down the gradient are truncated and runs up the gradient are extended, resulting in net movement toward the gradient peak.

The *chemosensory neurons that are likely to be responsible for the input* representation are known [BH91], as are the premotor interneurons for turning behavior [CSW+85]. However, much less is known about the interneurons that link chemosensory input to behavioral output [TH03, WKS04, GHB05].

Because *C. elegans* has not been studied in the wild, the purpose of its chemotaxis behavior is unclear. *C. elegans* chemotaxis in the lab has been shown to occur for both feeding and finding a mate. As this behavior can be modified depending on exposure to food [GHB05], it is likely that taxis towards a chemical attractant most relevant to this research is part of a food-finding strategy.

2.1.2 Thermotaxis

During thermotaxis, *C. elegans* attempts to return to its cultivation temperature T_{cult} from a higher or lower temperature T [MO95]. As in chemotaxis, *C. elegans* also uses turning rate to orient towards T_{cult}[ZMFL03]. Turning probability is modulated by the rate of change of temperature, $dT(t)/dt$, with respect to T_{cult} [RS02, ZMFL03]. Once at T_{cult}, the worm often exhibits an isothermal tracking behavior [HR75, MO95]. Isothermal tracking uses a series of reorienting head movements to track isotherms of less than $0.05C$.

Many of the neurons responsible for thermotaxis have been identified [MO95] and activity has been measured in these neurons during applied thermal stimulus [KMMM04, SS05]. Furthermore, there is evidence that thermotaxis is driven by two circuits, one that is active above T_{cult}, and the other one that is active below T_{cult} [MO95].

2.2 Biological Neural Networks

To provide a background for modeling neural networks, this section describes biological neural network components, experimental measurement methods, and methods for modifying neural networks. We introduce neural network components and function in order to provide an understanding of the work presented here. We discuss neural network measurement techniques to provide a background into how biological neural networks may be modeled and analyzed. We discuss neural network modification techniques to show how biological and artificial neural networks may be altered to show corresponding in behavior.

2.2.1 Nervous System

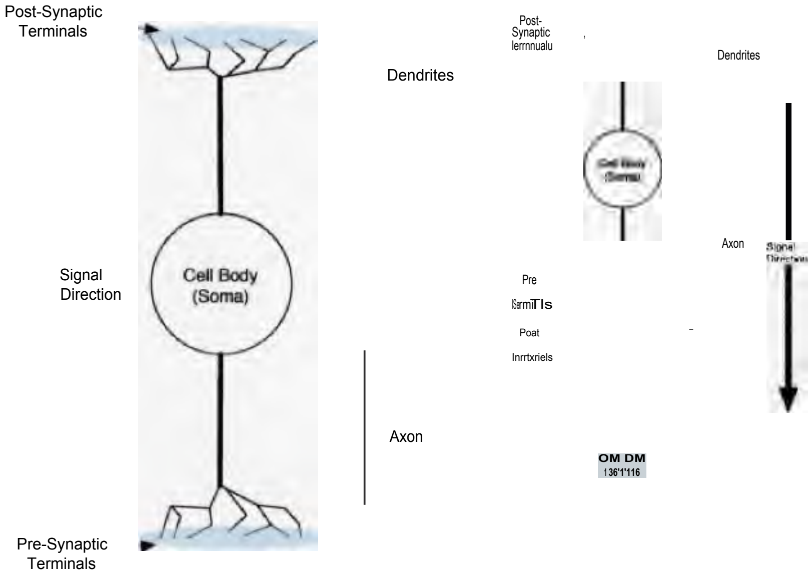
The nervous system is composed of interconnected neurons as shown by the cartoon in Figure 2.1. Figure 2.1(A) shows the three main components of a neuron: a cell body or soma, dendrites, and the axon. Synaptic input comes into the neuron through the postsynaptic terminals of the dendritic branches. Signal is propagated from the dendrites into the cell body. The cell body integrates the signal and redistributes it outward to the axon. The axon distributes signal from its presynaptic terminals onto the postsynaptic terminals of dendrites from another neuron as shown in Figure 2.1(B). The junction between the nerve cells is referred to as the synapse.

Two biological examples are shown in Figure 2.2. Figure 2.2(A) shows a rat hippocampal neuron growing on a silica substrate. In this picture, the dendrites and axons don't physically touch, but the axon (far, bottom left), dendrite, and cell body are clearly discernable. Figure 2.2(B) demonstrates neurons within a biological substrate (the rat cerebral cortex) that have connecting branches in three dimensions.

Cell Body: Integration of Neural Activity

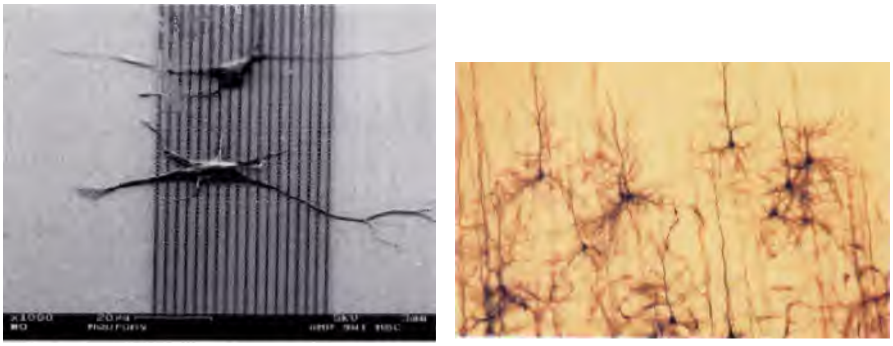
The cell body (or soma) contains the neuron's nucleus and is typically responsible for integrating incoming neuronal activity (presynaptic) and delivering outgoing synaptic activity (postsynaptic). Both dendrites and axons emanate from the cell body. If the cell body is destroyed, synaptic transmission and growth are quickly eliminated, as well.

There are two types of neural activity transmitted between neurons, *spiking* (or action potentials) and *graded* (or localized potentials) response. In spiking neurons, the cell body integrates synaptic input until a particular threshold is reached. When the threshold is reached, the neuron quickly self-activates and then quickly suppresses itself, creating a spike in neural activity that travels away from the neuron cell down the axon. Sufficient continuous external activation will create a population of discrete spikes. Discrete spikes are often analyzed in terms of their temporal frequency, which can be represented as a sigmoid by converting firing rate to a continuous population density. Voltage from spiking neurons may travel a much greater distance without significant attenuation of signal relative to local or graded activity. This is because there are pockets of voltage-gated channels along the dendrites and axons of neurons. These pockets emit voltage spikes when exposed to a sufficient amount of depolariza-



(A) (B)

FIGURE 2.1. (A) A simple neuron is composed of three parts: a dendrite, a cell body or soma, and an axon. A tree of dendrites receives incoming chemical or electrical signals from the presynaptic terminals on upstream neurons or sensory stimulus at its postsynaptic terminals. The dendrites electrically transmit their signals to the cell body. The cell body integrates the signals from the dendrite and creates a corresponding electrical signal that is sent out along the axon to the presynaptic terminal, sending a signal to the postsynaptic dendrite of downstream neurons. (B) The simple neuron connected to another simple neuron. The synapse is the junction between two cells through which cells transfer signal.



(A) (B)

FIGURE 2.2. (A) Rat hippocampal neurons are shown growing on a silicon substrate with clearly discernable neurons and branching dendrites (http: //www.news.cornell.edu/releases/July99/nanobiotech.hrs.html). (B) Neurons in the cerebral cortex show a web of integrated synapses (http://www.ics.uci.edu/ junkoh/alzheimer/neuron-synapse.html).

tion (activity), essentially acting as a signal repeater of activity. This makes spiking neurons ideal for transmitting a maximum amount of information over long distances. These are the most common type of neuron in the mammalian nervous system.

Neurons that exhibit graded response yield a nonlinear saturating response. Synaptic input within the center of the neuron's range yields a quasilinear response. However, for large excitatory synaptic input, the neuron saturates. Conversely, for large inhibitory synaptic input, the neuron is inactive. They are only able to transmit signals a short range (1-2 mm) without significant attenuation of signal. This type of activity is ideal for integrating highly sensitive input to be transmitted over a short distance. This is observed in some smaller organisms, such as *Ascaris* [DS89a], and is found in many mammalian sensory networks, such as the auditory and visual systems.

Synapses

Synapses are the junctions between nerve cells through which cells transfer signal. In general, signal is transmitted from the presynaptic axon to the postsynaptic dendrite (Figure 2.1). There are two common types of synapses: *gap junctions* (or an electrical synapse) and *chemical synapses.* A gap junction behaves like a passive wire, readily passing current in either direction. In a chemical synapse, chemical transmitter is released from the presynaptic axon terminals onto the receptors of the postsynaptic dendrites.

There are several fundamental differences between gap junctions and chemical synapses. First, transference of signal via chemical synapses may have a slower response. Second, chemical synapses may inhibit downstream neural activity, while gap junctions usually excite downstream neurons. Activity in an inhibitory synapse suppresses postsynaptic activity. This depends on the type of chemical neurotransmitter released and the corresponding channels in the postsynaptic receptor. Lastly, the downstream response to a small amount of synaptic transmitter may be greatly amplified due to nonlinear effects of neurotransmitter on neural activation. In contrast, transmission of current via a gap junction is roughly linear.

2.2.2 Measurement

To fully understand behavior of a system, it must be measured under a variety of different conditions. For a biological system, a stimulus (e.g., odorant, touch, visual presentation) is presented to an animal and a response is measured. Responses may consist of observed behavior and/or measured neural activity. Many different experimental methods are used to determine functionality in biological circuits. We introduce these methods and their limitations to understand how biologists explore the nervous system and how modeling can further contribute understanding of the biological system. Furthermore, many of the methods and concepts used by biologists to understand living networks may also be used when analyzing artificial networks.

Methods for measuring neural activity may be broken down into three categories: extracellular, intracellular, and optical. Extracellular methods measure activity by placing an electrode outside the cell. The electrode is able to relay information about whether or not a cell is depolarized (activated), and if activity is increasing or decreasing. One method of extracellular recording is noise analysis. When chemical transmitter is applied to a cell, the channels that are receptive to the transmitter begin to open more often. This has two effects. First, the cell is depolarized ("activated"). Second, the current fluctuates more rapidly as channels stochastically open and close. The amplitude of the noise relative to depolarization gives an indication of the channel's properties.

Intracellular recordings use microelectrodes that penetrate the cell. They give more detail of the excitatory and inhibitory processes of the cell by measuring the potential difference between the inside and outside of the cell. Typically, they involve applying different constant currents and measuring the response voltages (or vice versa). Often this is done in the background of stimulus or pharmacological agents, which alter the function of channels in the membrane. A common intracellular recording technique is the patch-clamp [KNM84]. The general method consists of creating suction on part of the neuron cell and applying an electrode. Depending on the type of recording, the membrane under suction may be separated from the cell. This method is able to show ion gates on the cell opening and closing, giving an indication of the types of channels and details of their activity. It also gives an indication of the density of a given channel on a cell's surface.

Optical methods are relatively new and often involve the use of genetic methods to embed proteins, which emit varying wavelengths of light relative to cell activity. One common method involves genetically adding a protein called *Cameleon* [CTE$^+$94, RBP$^+$95] to cells of interest. *Cameleon* is a protein composed of two fluorescent proteins that emit different wavelengths of light depending on the amount of available Ca^{2+} (calcium) in the neuron, an indication of neuronal activity [CCM$^+$01]. As more Ca^{2+} becomes available in the cell, the ratio of light emitted at the two wavelengths changes. The downside of this method is that the Cameleon protein must be inserted using a neuron-specific genetic promoter, which does not exist for every neuron, or for individual neurons. The primary advantage is that it leaves the animal intact during behavior.

Both extracellular and optical methods are ideal for making *in vivo* recordings during behavior, as neither requires any invasive surgery within the animal.

2.2.3 Modification

To test theories of neural function and structure, biological structure may be modified to assess changes in functionality. We discuss three strategies for modifying neural networks: pharmacology, physical surgery, and genetics.

With pharmacological modification, a chemical agent is applied to neurons that blocks channels, activates channels, or adds to the pool of free neurotransmitter within the neural network (channels are discussed in more detail in Section 2.2.4). Measurements of these modified channels gives an indication of what channels are available and their effect on the neuron. However, channel blockers may have non-specific effects on other cells, as it may be hard to control flow of the channel modifier. Additionally, neurons have multiple channel types, yielding different effects on different channels.

Neural networks may also be modified by physically ablating neurons. In ablation, the neuron is eliminated through manual surgery [SW80] or with the use of a focused laser [BH91], minimally impairing the rest of the animal. However, this method is very labor intensive, requires a great deal of expertise, and animals must be given time to heal.

Finally, neurons in the network may be altered through genetic modification [SNL$^+$99]. This either over-activates, or eliminates, individual neuronal response. This method is attractive because large populations may be produced for rapidly reproducing animals, and it is not physically disturbing to the animal. However, it can be difficult to specifically target neurons of interest and effects may be non-specific.

2.2.4 Modeling

We present two types of neural models: *cable* and *compartmental* modeling. Cable modeling treats current flowing through dendrite and axon trees as passive with uniform physical properties. Compartmental modeling extends cable theory by breaking up the neurons into a set of individual compartments, making it possible to represent more complex and heterogeneous models. We further show how to combine these models into an artificial neural network.

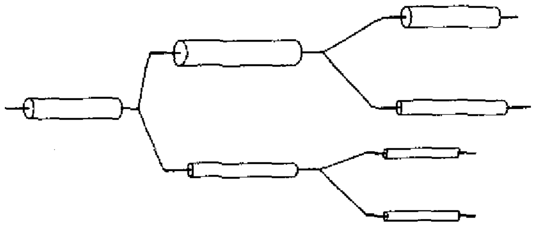


FIGURE 2.3. A model of a dendritic tree from cable theory. At every branch, the current splits and flows relative to the resistance in the downstream branch. In this model, electron flow is not amplified or repeated, but is roughly analogous to sending water through a leaky hose.

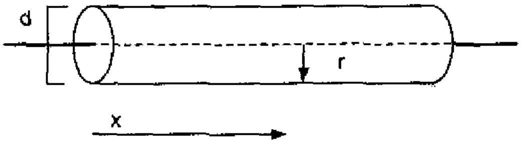


FIGURE 2.4. A model of a dendrite cable. The bulk of the current flows in the x direction. Current leaks radially through the cylinder surface. Larger diameter d reduces resistance (Equation 2.2).

Cable Theory Model

Cable theory models treat dendrite and axon trees as passive cables. Lord Kelvin developed much of the math used in cable theory. This math was originally for the transmission of electrical current, but is a general three-dimensional, cylindrical flow problem. This was first adapted to dendritic neurons in the late 50's[CEF55a][CEF55b] and continues to be refined. Though the dendrites and axons are not completely passive, this analysis gives insight into how biophysical properties of the dendritic tree affects activity (Figure 2.3).

Voltage $V(x)$ along a synaptic cable can be modeled as a leaky hose (Figure 2.4). Voltage at a distance x from a reference voltage V_0 can be calculated as:

$$V(x) = V_0 exp^{-x/\lambda} \tag{2.1}$$

$$\lambda = \sqrt{d/4\ R_m/R_i} \tag{2.2}$$

where λ is a decay constant related to resistance within the synaptic weight, d is cable diameter, R_m is membrane resistance, and R_i is intracellular resistance. This gives the intuitive solution that voltage decreases exponentially as it progresses along the cable in the x direction. A larger diameter (d) reduces resistance due to having less contact with the membrane surface, reducing the dissipation of voltage.

From Equation 2.2, two additional variables affect the dissipation of current. First, as the ratio of membrane length (x in Figure 2.4) to internal resistance increases, the dissipation of voltage decreases. This can be understood as less voltage being lost through leaks from the system. Secondly, a larger diameter implicitly yields a lower internal resistance due to less contact with the membrane surface, which reducing the dissipation of voltage.

Compartment Theory Model

Compartmental modeling allows a neuron with heterogeneous properties to be modeled as set of isopotential (single-voltage) compartments. An example of two sections of dendritic membrane is shown in Figure 2.5. Additional neural processes may be represented in the model by adding them in parallel within the capacitive compartment. This method is also able to represent somatic, dendritic, or axonal membrane.

An individual compartment in the model is described by the following equation:

$$I_m = I_{ion} + C_m \frac{dV}{dt} = I_{ion} + \frac{dQ}{dt} \tag{2.3}$$

where Q is total charge of the compartment, C_m represents the compartmental membrane capacitance, I_m represents the total current flow through the membrane, and I_{ion} represents ionic current that passes through the channels of the neuron. V is the

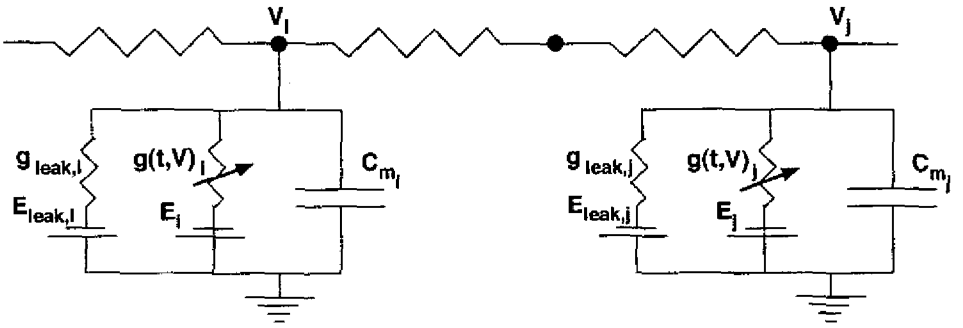


FIGURE 2.5. Compartmental model of a neural network. Two heterogeneous sections of excitatory dendrites are modeled as individual, connected compartments. Conductance g is $1/R$, where R is traditional electrical resistance measured in ohms. The surface of the dendrite (lipid bilayer) holds a capacitive charge represented by C_m. The varying conductance $g(t, V)$ represents voltage-gated channels within the dendrite, where V is voltage and t is time. Passive properties of the membrane are represented by the constant "leak" conductance and battery.

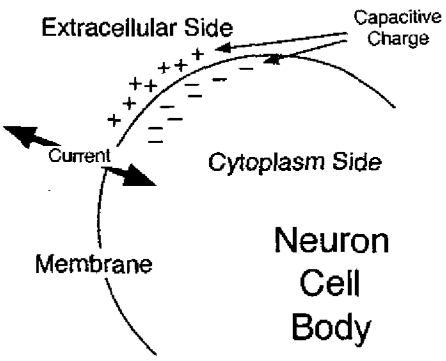


FIGURE 2.6. A neuron modeled as a capacitor. The neuron surface has a net positive external charge and a net negative internal charge, providing a constant voltage between the membrane surface and the inside of the neuron. When ion channels open within the membrane, ions, and therefore current, flows through the membrane, dynamically changing the current and voltage.

passive voltage of the compartment. Multiplying this equation by the input resistance R (or dividing by g), yields a more general equation in terms of voltages:

$$\tau \frac{dV}{dt} = RI_m - V \tag{2.4}$$

where τ is the time-constant of the compartment defined as $\tau = C_m R$ [SB98]. For neuron i this becomes:

$$\tau \frac{dV_i}{dt} = R_i I_{m_i} - V_i \tag{2.5}$$

Equation 2.5 can be expanded to represent multiple anatomical processes, as many neurons have several different types of processes (e.g. different channels) that may affect neural activity. As an example, we can combine the effect of chemical and electrical synapses into a single equation. In [LS93] a model is constructed from the general form of Equation 2.4 and chemical synapses are modeled separately from action potentials:

$$\tau_i \tfrac{dV_i}{dt} = -V_i + R_i \left[\underbrace{\sum_{j=0}^{n} g_{ji}(V_j - V_i))}_{electrical} + \underbrace{\sum_{j=0}^{n} h_{ji}}_{chemical} \right] \tag{2.6}$$

where h_{ji} represents the strength and sign of the chemical synapse from neuron j to neuron i.

A Combined Model Yields Artificial Neural Networks

A commonly used artificial neural network model is a simplification of Equation 2.6. The different connection types are lumped into a single weight term w_{ij}, which represents a general connection strength from neuron i to j that is the inverse of resistance. Additionally, because a combination of voltage and general chemical activity is used for neuron i, it is more correct to use a more general activity variable A, instead of voltage V, to describe the neural activity in neuron i. We also explicitly build saturation into neuronal activity. This gives:

$$\tau_i \frac{dA_i(t)}{dt} = S_i\left(I_i(t)\right) - A_i(t) \tag{2.7}$$

$$I_i(t) = \sum_{j=0}^{n} w_{ji} A_j(t) + b_i + u_i(t) \tag{2.8}$$

where S_i is the neural activity function, $I_i(t)$ is the integrated input to neuron i, b_i is the static bias of neuron i, and $u_i(t)$ is external network stimulus. S_i is typically one of the nonlinear saturating functions further discussed in Section 2.3. There are many variations on this form of the equation. The summation of terms assumes either that the integration in the neuron is a linear summation of components or that nonlinearities are calculated upstream.

2.3 Artificial Neural Networks

The use and development of artificial neural networks in a variety of fields outside of biology has greatly expanded the pool of available neural network models. In this section, we discuss evaluation methods (Section 2.3.1), neural activity models (Section 2.3.2), and connectivity (Section 2.3.3).

We use introduce a more generalized, vector form neural network of Equations 2.7 and 2.8:

$$\tau \frac{d\vec{A}(t)}{dt} = \vec{S}(\vec{I}(t)) - \vec{A}(t) \tag{2.9}$$

The input vector is defined with the assumption that I_i is defined by Equation 2.8 for a network with n neurons:

$$I(\vec{t}) = W\vec{A}(t) + \vec{b} + \vec{u}(t) \tag{2.10}$$

where W is the $n \times n$ weight matrix of connectivity. The vectors in Equations 2.9 and 2.10 are defined as:

$$\begin{aligned}
&\vec{I(t)} = (I(t)_0, I(t)_1, \cdots, I(t)_n)^T \\
&\vec{\tau} = (\tau_0, \tau_1, \cdots, \tau_n)^T \\
&\vec{b} = (b_0, b_1, \cdots, b_n)^T \\
&\vec{S}(\vec{I(t)}) = (s_0(I(t)_0), s_1(I(t)_1), \cdots, s_n(I(t)_n))^T \\
&\vec{A}(t) = (A(t)_0, A(t)_1, \cdots, A(t)_n)^T \\
&\vec{u}(t) = (u(t)_0, u(t)_1, \cdots, u(t)_n)^T
\end{aligned} \tag{2.11}$$

where ^{T} is the transposition operation.

There are several approaches for construction and evaluation of biological neural networks models. We will present some common approaches to provide background for the choices we used in developing the neural network models in Chapters 3 and 4.

2.3.1 Network Evaluation

Many methods can be used to evaluate artificial neural networks. Although most computational evaluation methods for neural networks are synchronous and discrete, it is helpful to understand the distinctions between the methods.

Time Model

Because the solution to Equation 2.9 is most often nonlinear, one of numerous estimation methods must be used to evaluate it such as Runge-Kutta, Gaussian Quadrature, or Euler's method [PFTV88]. Euler's method, or the Trapezoid rule, is one of the simplest methods and will be used to show basic discretization of the neural network shown in Figure 2.7. The vector form of Equation 2.10 discretized by Euler's method is given by:

$$\vec{A}(t + \Delta t) = \vec{A}(t) + \frac{\Delta t}{\vec{\tau}} \left[\vec{S}\left(\vec{x}(t)\right) - \vec{A}(t)\right] \tag{2.12}$$

Additional terms may be added to Equation 2.12 to increase accuracy. For example, if we used a 2nd-order Taylor's series expansion, we would calculate an additional midpoint, increasing our accuracy, but doubling the number of calculations necessary

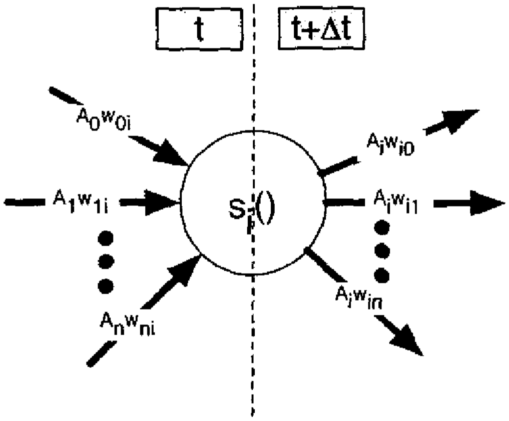


FIGURE 2.7. A simple neural network model from Equations 2.7, 2.8, 2.9, and 2.12. Activity from every neuron with a connection to it comes into neuron i at time t via its incoming connections. The integrated activity A_i of neuron i is related to other neurons as input via outgoing connections at time $t + \Delta t$. Unless otherwise restricted, incoming and outgoing connections may be self-connections.

to estimate a single time-point. Like most numerical estimations, tradeoffs are made between accuracy and efficiency.

Synchronicity

Neurons within a neural network may either be evaluated all at once (synchronously) or at different time-steps (asynchronously). During asynchronous evaluation, a stochastic evaluation function evaluates different neurons at each time-step within the simulation. Asynchronous models are more common for spiking neurons, allowing the system to randomly evaluate whether or not to fire. One asynchronous version of Equation 2.12 is:

$$\vec{A}(t + \overrightarrow{rand}(k)\Delta t) = \vec{A}(t) + \frac{\overrightarrow{rand}(k)\Delta t}{\vec{\tau}}\left[S\left(I(\vec{t})\right) - \vec{A}(t)\right] \qquad (2.13)$$

where k represents the iteration and $\overrightarrow{rand}(.)$ generates a uniform, stochastic vector of integers (a different value for each neuron). As the network continues to iterate, each neuron evaluates itself at its own stochastically determined time step, yielding asynchronous behavior.

Objective Function

An objective function defines the quality of the network solution and represents the task that the network was designed to perform. It is often used to guide optimization (Section 2.3.1), or to give a feel for the overall quality of the solution. In some domains, there may exist a "perfect" objective function value that the network may reach. However, objective functions often have unattainable goals and instead provide a direction for optimization. The objective function is often described as either error to be minimized or fitness to be maximized. In general, when fitness is at a maximum (e.g. 1), error is at a minimum (e.g. 0). Error and fitness are both commonly used in neural network terminology, though fitness is used more often in the biological and psychological sciences. In this paper, we will refer to error as an objective function.

Many methods may be used to calculate error. Most methods rely on comparing one or more output neurons of the network to a target set. Output is defined as the activity of one or more "output" neurons, or neurons in the "output layer". More succinctly, network output is $\vec{O}(t) = f(\vec{A}(t))$, where $\vec{O}(t)$ is a vector of size p (the number of outputs), p is $1 \leq p \leq n$, and n is the size of the network. $\vec{O}(t)$ is compared to a target vector $\vec{T}(t)$, also of size p that represents a desired or known solution. Targets may consist of a single final point (static) or a time-dependent series of points (dynamic). Network error can be represented in a general from as:

$$E = f_e(\vec{T}(t), \vec{O}(t)) \tag{2.14}$$

where f_e is an error function. f_e is typically an iterative function that compares the target point(s) to the output point(s) for all p neurons over every time-step of the target to calculate error. If the target is static, there will only be a single time-step to evaluate.

A common method for calculating the difference between target and output is sum-squared error, which sums the square of the difference between the target and output at each time point. This has the effect of emphasizing larger differences between the target and the output.

Optimization

The goal of optimizing a neural network is to connect a network in such a way that it performs a desired function. Although this may be done by hand-wiring, it is most often done using an optimization procedure, often referred to as "training". These methods are typically general-purpose parameter-optimization methods adapted to neural networks. The purpose of the optimization algorithms is to adjust the network objective function towards the desired value by changing network parameters such as weights.

There are several optimization methods that can be applied to neural networks [PFTV88, Mas93]. These fall into two categories: local and global. Local optimization methods are often referred to as hill-climbing methods, such as gradient descent, and are designed to quickly arrive at local optima. Global optimization methods, such as simulated annealing or genetic algorithms, are slower methods better suited for finding global optima. Typically, more complex neural networks, and especially recurrent neural networks, require global optimization algorithms.

We briefly describe primitive and simulated annealing, which were used to optimize neural networks in Chapters 3 and 4. Primitive annealing consists of a series of "temperature" steps whereby the temperature gradually decreases. At each step, the network parameters are stochastically perturbed an amount relative to the current temperature step from an initial starting set of network parameters. After each perturbation, the network's objective function is again evaluated. If the perturbed network is better than the original, then its parameters become the new starting network. After a set number of iterations at this temperature, the temperature is lowered and a new set of iterations is evaluated. Simulated annealing extends primitive annealing by making it possible to take a worse network as the next starting network. The probability of this is increased if the new point is not significantly worse and the search temperature is high. This provides a wider search of network parameters during initial exploration.

2.3.2 Neuronal Activity Functions

Similar to biological neural networks, artificial neural networks also integrate activity from incoming connections with their own internal activity, and produce an outgoing activity through outgoing connections. This section describes a few types of neuronal integration functions S. In this section, the neuronal activity function is described by:

$$y_i = S(x_i) \tag{2.15}$$

where y_i is neuronal output and x_i is neuronal input.

Linear Activation Functions

Linear activation functions map input signal to linearly scaled output by:

$$y_i = S(x_i) = k_i x_i \tag{2.16}$$

An advantage of linear systems is that they are much easier to analyze than nonlinear systems. Neural networks with this activity function may be solved analytically as ordinary differential equations [MM82], simplifying analysis. The downside of these functions is that they have an infinite range (i.e., they don't saturate), which makes them impractical in most applications because of the difficulty of optimization and instability. In addition, they are not biologically plausible, as biological neurons saturate and linear neurons do not.

Nonlinear Activation Functions

Nonlinear activation functions may be either discontinuous or continuous. In discontinuous activation functions, a threshold on internal activity produces a discrete set of outputs. A continuous activation function provides a continuous mapping between synaptic input and neuronal output. One common continuous, nonlinear activation function is the sigmoid σ a saturating, nonlinear function. For an unbounded

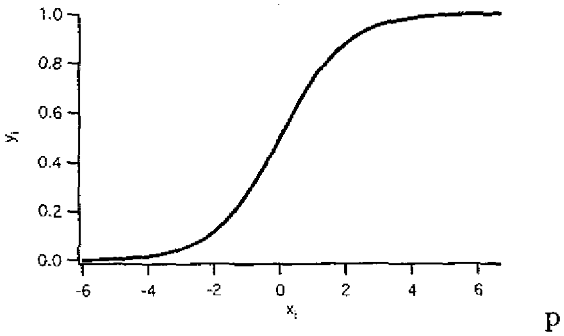


FIGURE 2.8. The plotted sigmoid function of Equation 2.18. y_i saturates to 1 as $x_i \rightarrow +\infty$ and saturates to 0 as $x_i \rightarrow -\infty$.

input x we give two examples. First, for the bounded output range $-1 < y < 1$ the sigmoid function is:

$$y = \sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.17}$$

Seconds, for the bounded range $0 < y_i < 1$ the common form of the sigmoid function is represented in Figure 2.8 and described by the following equation:

$$y = \sigma(x) = \frac{1}{1 + \exp(-x)} \tag{2.18}$$

In both functions, there is a smooth transition from an unsaturated range to a fully saturated state at the extreme ranges of x.

Conversely, a semilinear activation function is linear, but saturates at a given input boundary as described by:

$$sat(x) = \begin{cases} 1, & x \geq 1 \\ x, & 1 > x > -1 \\ -1, & x \leq -1 \end{cases} \tag{2.19}$$

The advantage of the semilinear neuronal activation function is that it has a more tractable, linear range, it is computationally more efficient, and still provides regions of saturation for the neuron.

Discontinuous Activation

Discontinuous activation functions are often used to model neuronal activity such as populations of spiking neurons or bistable, graded neurons. An activation function for a binary neuron is given by:

$$y = S(x) = sgn(x) \tag{2.20}$$

where *sgn* is a binary sign function expressed as:

$$sgn(x) = \begin{cases} 1, & x > 0 \\ -1, & x < 0 \end{cases} \tag{2.21}$$

An important implementation of a discontinuous neuronal activity function is the McCulloch-Pitts network introduced in 1943[MP43]. It used asynchronously firing neurons to generate associative memory. A discrete, spiking neural network model also uses a discontinuous neuronal activation function.

Stochastic Activity

Previous (Equation 2.13) model stochastic behavior as a series of individual events. Another way to model this type of behavior is by explicitly using event probability, ignoring discrete events. Many methods in this category are statistical sampling methods cast into neural networks. This is still a very active method of research and application, especially for associative memory (classifications) problems. The radial basis function is one example of a neuronal network that uses stochastic neural activity.

2.3.3 Connectivity

We present two general types of connectivity between neurons in a neural network: feedforward and recurrent.

Feedforward Networks (Perceptrons)

Neurons in a feedforward network only connect to neurons that are closer to the output neuron. As such, there are no backward, or recurrent connections, and therefore no graph cycles. Neurons that are the same number of connections from the input neuron are said to be in the same "layer" and are not connected to each other. The first layer is the input layer, where input is fed to the network. There may be one or more hidden layers where interneurons reside. The final layer is the output layer, where output is recorded. A network without hidden layers is referred to as a perceptron [Ros58], which perform well for linearly separable classification tasks. A network with multiple layers, often called a multi-layer perceptron or MLP, may be used to solve more difficult problems that are not linearly separable.

Recurrent Neural Networks (RNN)

Recurrent neural networks, sometimes referred to as feedback networks, are networks that allow cycles within the network, including connections from every neuron to every other neuron and connections that may enter and exit the same neuron (self-connections). Recurrent neural networks can solve problems that are complex, store more data, and more realistically model biological neural networks. However, there are two drawbacks to recurrent neural networks. First, they are difficult to optimize. They require computationally intensive global optimization techniques such as simulated annealing or genetic algorithms. Second, evaluation is not immediate. For example, in classification problems a determination of a result may not occur until the output neuron reaches a relatively stable point. Stability points and attractors are discussed in detail in [ML02], where analytical solutions may be found for many recurrent neural networks forms.

2.4 Analysis of Artificial Neural Networks

Artificial neural networks in computer science and engineering are typically not analyzed for function, but are more often treated as black boxes that perform the task for which they were optimized. However, in biological models, network function and structure are relevant because the network model must relate to the biological neural network. Therefore, to infer biological function from neural network models, analysis of neural network function and structure is necessary. In the research presented in Chapters 3 and 4, once a set of neural networks had been created, knowledge of how networks were similar and their function was necessary to generate predictions for the biological system we were attempting to model (the *C. elegans* neural network). This was accomplished by extracting a small set of unique functional network models, and then analyzing their function to generate models to be tested in the biological network. However, determining the function and similarity of artificial neural networks is nontrivial, especially as networks become larger and more highly connected.

This section describes three general analysis strategies: architectural analysis, analytical analysis, and signal analysis. In Architectural analysis (Section 2.4.1), connectivity is used to interpret and classify network solutions. In analytical analysis (Section 2.4.2), the equations that make up a neural network solution are solved and their solutions are characterized. In response analysis (Section 2.4.3), insights into network behavior are made using network response to stimulus in a traditional engineering approach.

Neural network response to sensory stimulus is divided into two types. The first type of network response is long-term neural network response, which consists of network solutions for associated memory, classifiers, or oscillators. Central pattern generators, which drive behaviors such as walking [HBG99, BHG99], are an example of an oscillating neural network. This type of solution is somewhat tractable as it may be evaluated out to an infinite timecourse. The second type of network response is transient, which consists of networks such as time-series predictors, differentiators, and many other biological models. These are difficult to solve and must be numerically estimated. An example of a transient neural network response is differentiators [MSF94].

2.4.1 Architectural Analysis

Architectural analysis attempts to find patterns of network structural components. Structural components consist of connectivity betweens neurons, including connection weights (or strength) and sign, as well as neuronal parameters such as bias and time-constant. Many of these methods are well suited for analyzing sets of similarly optimized neural networks.

Global properties of graph-based structures may be defined by their large-scale connectivity patterns [DM03]. These methods may be used to compare optimized networks with the assumption that neurons may be treated as nodes and connection weights as graph edges. Some measurable network properties are the distribution of degrees (the number of incoming and outgoing edges on a particular node), the clustering coefficient, the distribution of length between connections, and the average shortest-paths.

Understanding global network properties is helpful to understand possible network behavior and how large networks may be evolved or optimized. One relevant example from biology is the *C. elegans* neural network [SW98]. Here, a variety of connection types were abstracted from undirected connections to yield a highly clustered network with an average shortest path of $\bar{l} = 1.35$. A problem with this analysis is that its small size, less than 300 neurons, yields network properties that are not very statistically significant. This method only works for networks that have a sufficient number of known connections to provide statistical significance. Another biological example is the metabolic reaction in *E. coli* [WF01]. In one study, 282 biological substrates, or molecules undergoing reactions in the presence of enzymes, were considered. Substrates are analogous to neurons in a neural network. The reaction kinetics between substrates is analogous to directed network connections. These systems were highly clustered.

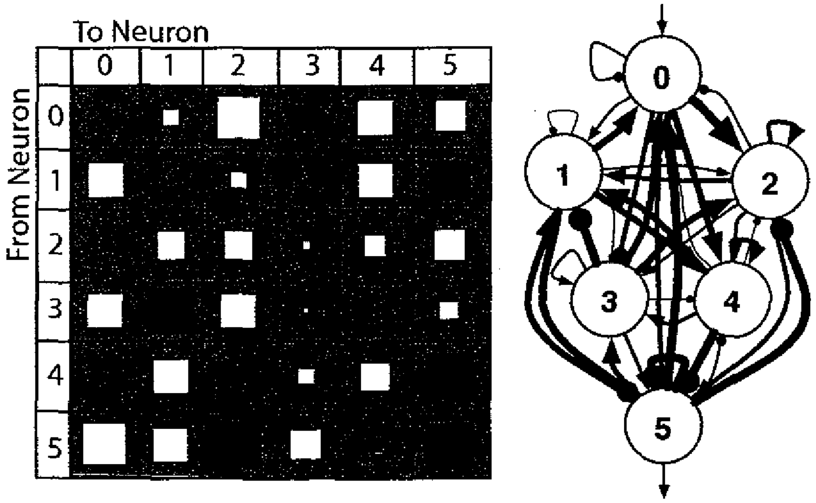
Statistical analysis of macro-level connectivity properties using connectivity patterns gives insight into the different classes of networks necessary to solve many problems. However, there are two main drawbacks to this method. First, it only explores connectivity patterns, and fails to analyze the sign and strength of connections or the dynamic function of the network. Second, this method may only be applied to networks that are sufficiently large to provide statistical significance, eliminating consideration of this type of analysis for many network models. Different tools are needed to perform more fine-grained architectural analysis of network structure and performance.

Another body of work in this area has focused on finding common biological modules in network structures according to connectivity. Milo [MSI$^+$02] characterized a number of these modules in different systems, which they termed motifs. We will refer to these modules as network connectivity motifs to distinguish them from our own definition of neural network motifs introduced in Chapter 5, which largely ignores connectivity. One of the evaluated biological systems was the *C. elegans* neural network, in which they found two, three, and four neuron network connectivity motifs. They compared the connectivity of each to a number of stochastically generated connectivity patterns with similar connectivity properties. This suggests that common patterns are re-used in natural systems, yielding potential insight into these systems.

An advantage of this type of analysis is that it is able to explore small, directed graphs. There are two main drawbacks to this method. First, connectivity strength and sign have not been analyzed using this method. Biological motifs are typically mixtures of both inhibitory and excitatory connections [MSF94]. Knowledge of connectivity without sign or strength gives little insight into probable network dynamics. Second, a large number of connected nodes are required to make analysis statistically significant.

Many of the tools used to analyze network functionality using connectivity allow network patterns to be explored by the human eye. Most of these methods attempt to represent the high dimensional space of neural networks (weights, biases, etc.) by abstracting them to two or three dimensions. This is often ideal for analysis of multiple networks.

Hinton diagrams (Figure 2.9) display connection sign and size on a two dimensional plot. A network with N nodes is represented by an $N \times N$ matrix of square blocks. Square blocks represent the size of each connection. The magnitude of the weight is the size of the square. The block in cell i, j represents the connection from neuron i



(A) (B)

FIGURE 2.9. (A) Hinton diagram of a (B) recurrent neural network with six neurons. Connections between neurons are represented by squares. The size of each weight is relative to the square size. Black squares represent inhibitory connections and white squares represent excitatory connections.

onto neuron j. Black squares represent negative weights and white squares represent positive weights. In other versions of the Hinton diagram, connections may instead be pooled by type (e.g., output, sensory, motor). This type of plot makes it easy to scan 10-20 connections in order to look for major connectivity patterns in one or more networks.

Another tool used to visualize connectivity properties is the parallel-axis plot (Figure 2.10). Connection descriptions are plotted on a two-dimensional plot, providing a global view of weight contributions across all networks. Network parameters are plotted along the x-axis with the respective values of each network plotted along the y-axis. This method is good for analyzing a few parameters in multiple networks, especially if the potential for reordering neurons within a network is small.

Another visualization technique is plotting the error surface of the network (Figure 2.11). By adjusting only two parameters as independent variables, such as weights, the corresponding error surface may be mapped. This method is often used to get a

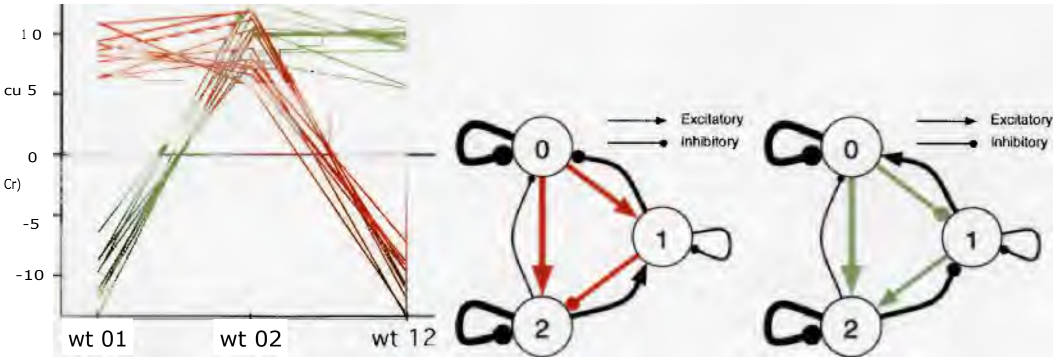


FIGURE 2.10. Parallel-axis plot. Each line represents weights from a single optimized neural network. *wt ij* represents the weight connecting neuron i to neuron *j*. Two primary patterns of connectivity are identified, which are colored red and green. An example network corresponding to each pattern is shown to the right. In the red pattern, wt01 is always positive and in the green pattern, wt01 is always negative. In all patterns, wt12 is always the opposite sign of wt01 and wt02 is always positive. This plot represents portions of the network shown in Figure 2.9 (a red pattern) for neurons 0, 1, and 2.

feel for the difficulty of optimization.

Architectural analysis is a quick way to extract information about the probable function of a set of neural networks. However, much of the information provided by looking only at connectivity may be misleading. For example, a neuron downstream of a large connection may be inactive and provide no function to the network. Knowledge of neuronal activity is essential to providing insight into network function. Visual methods are also limited by the number of dimensions they can analyze since the human brain is only capable of visualizing a finite number of dimensions at once. As an example, the number of connections in a fully connected, four-neuron, recurrent network is 12, which is far too many connections to visually analyze at once using current visualization tools.

2.4.2 Analytical Analysis

Analytical analysis, also known as white-box analysis, uses a *priori* information to model system behavior. In the case of neural networks, solutions to the equations that

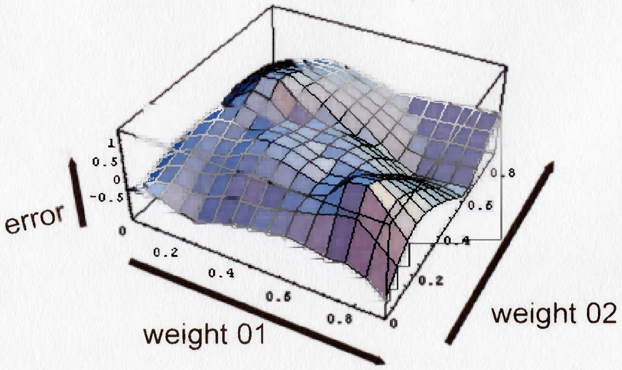


FIGURE 2.11. Error Surface Plot. All parameters (weights, biases, etc.) in a neural network are held constant while two weights within a neural network are changed ($wt01$ and $wt02$ from Figures 2.9 and 2.10). The error of the network is calculated for every pair of values of these weights, yielding a surface plot for a particular error function.

describe the neural network provide data for analysis. The advantage of analytical analysis is that we know exactly how each variable affects the solution. The drawback to this method is that solutions are often intractable, especially in the transient region. The transient response region is defined as the time period directly after a stimulus has been applied, but prior to the system arriving at steady-state.

One approach to analytical analysis is linearization, or simplifying the nonlinear equations that represent a neural network into their linear counterparts. Generating analytical solutions for neural networks is far easier than generating solutions for their nonlinear counterparts, which often have no solution. Generating solutions consists of solving sets of linear ordinary differential equations [MM82]. However, even with linearization, understanding dynamic response [DCPSL04] is difficult with as few as two nodes.

Using concepts from differential vector calculus, solutions can be generated for steady-state associative memories in recurrent neural networks. Additionally, many qualitative properties may be derived, such as the existence of stability points, the number and location of network equilibrium points, and domains of attractions of those points. Michel et al. [ML02] presented a series of results on the associative properties of recurrent neural networks. A number of properties that hold for continuous Hopfield networks are summarized below:

1. Hopfield networks possess unique solutions for all $t \geq 0$.
2. The energy function of a Hopfield network: $E = -\sum_{i<j} w_{ij} A_i A_j$, decreases monotonically along nonequilibrium solutions. So, as $t \rightarrow \infty$, the network tends towards equilibrium.
3. The number of equilibrium points is finite and exponentially proportional to the number of nodes.
4. If a solution $\vec{A}(t)$ is a stable equilibrium for a network, then it is asymptotically stable and must occur at a local minimum.

A problem with this type of result is that this analysis can only be used to analyze static attractors. This information is helpful, in terms of the goals of network classification and functional analysis, but doesn't provide a description of transient network dynamics. As an example, modifying bias or connection strength within a network could move an attractor significantly or modify its range without changing the network dynamics.

Beer [Bee95] showed that trying to conceptually understand the stable dynamics and relationships between weights in a recurrent neural network is problematic especially as the number of neurons increased. A multitude of stable ranges may be expressed with only two neurons in a recurrent two-neuron network with self-connections. The focus of this study was on "stable" points generated from different sets of weights, not on trajectories of transient response (except for stable, oscillating solutions). Pasemann [Pas02] further showed that Beer's results for continuous networks extend to discrete network models for the three-neuron case.

Analytical analysis provides exact solutions for neural network solutions, allowing analysis of a range of network properties and the effect of specific network properties on each. Analytical analysis may give some insight into the stability of network solutions. However, there are two drawbacks to this approach. The first drawback is that current analytical analysis methods only describe steady-state or equilibrium neural networks properties. This provides little help in understanding network solutions that rely on solutions within the transient region of neural network responses. The second drawback is that the entire input domain is considered since the goal is to attain the exact solution. This creates two additional problems. First, it is hard to generalize between solutions when the entire problem domain is considered. Second, biological systems, as well as most engineering applications, spend most, if not all of their time responding to a limited input range. In the analysis problems that were presented, one is less interested in responses of the system to input extremes than the normal operating environment observed in the biological system.

2.4.3 Signal Analysis

Signal analysis methods assume no knowledge of the internal function of a system. In other words, signal analysis methods model systems as black boxes. All inferences into system behavior are made by creating models of the system by measuring its response to external stimulus. Signal analysis is well suited to highly complex systems, such as large neural networks, where it is difficult to measure or separate the behavior of individual processes. Even though structural and neuronal activity is known in artificial neural networks, the nonlinear combinations that drive these parameters are not known.

Control theory represents a set of signal analysis techniques that can be used to analyze and control black-box systems. A black-box system can then be modeled by a transfer function that may be used to construct a controller. A linear transfer function is a linear differential equation, typically represented in the Laplace or Fourier domain, which describes a system's output response to any set of inputs. As an example, a thermostat that attempts to maintain a room at a set temperature can be modeled by a linear transfer function. Although the complex physics for heat dissipation and airflow may be calculated for a large room, it is often easier (and more accurate) to

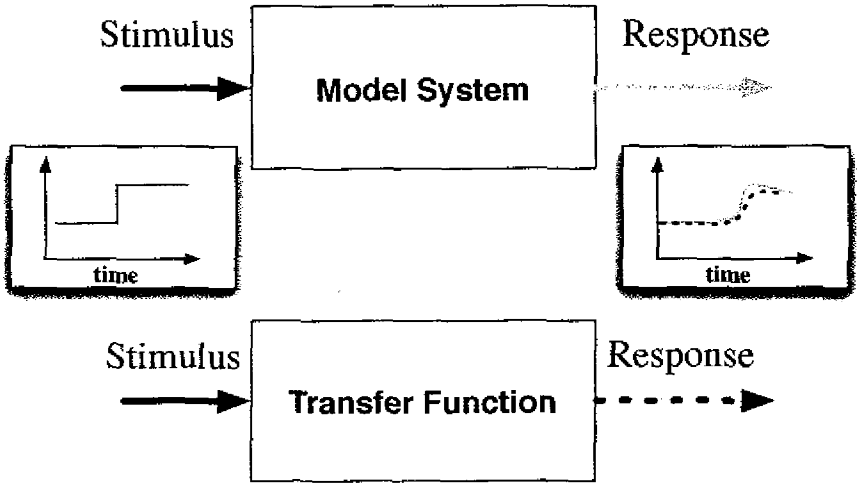


FIGURE 2.12. Using control theory to identify an unknown system. First, the system's response (gray) to a stimulus (black) is measured. Second, the linear transfer function is created using the stimulus/response pairs of the Model System. Third, the Transfer Function can be used to predict response (dashed line) to system stimuli to which it has not previously been exposed.

measure the room's temperature response to switching on or off a heating element. Figure 2.12 demonstrates how a set of stimulus response pairs may be used to extract a transfer function that accurately predicts the dynamic response for any set of stimuli. From a set of stimulus/response pairs, numerous methods may be used to generate a linear transfer function, including *curve fitting*, *cross-correlation*, *frequency response*, and linear regression [SEM03, WK03]. Although neurons in a neural network typically do not have a linear response, this provides a background for the other signal analysis methods that we introduce in Chapter 3.

CHAPTER 3

A Neural Network Model Predicts *C. elegans* Chemotaxis

Chapter 2 (Section 2.1) described *C. elegans* biology and demonstrated how a biological neural network may be abstracted into a model artificial neural network. In this chapter, we create a neural network model sufficient to reproduce the pirouette rule that drives *C. elegans* chemotaxis [PSML99].

The work in this chapter has been published in [DCL03, DCL04, DCPSL04]. Significant contributions have been made by the co-authors of these publications. Jon Pierce-Shimomura provided experimental data and analysis of *C. elegans* chemotaxis. John Conery provided the software framework, wrote the initial neural network and parallel optimization code, and contributed to the writing. Shawn Lockery was responsible for writing and the overall and direction of this project. Nathan Dunn was the principle contributor, providing the bulk of the coding, writing, and project direction and all of the network optimization and analysis.

3.1 Introduction

The complete description of the morphology and synaptic connectivity of all 302 neurons in *C. elegans* [WSTB86] raised the prospect of the first comprehensive understanding of the neuronal basis of an animal's entire behavioral repertoire. The advent of new electrophysiological and functional imaging techniques for *C. elegans* neurons [LG98, KLRB$^+$00] has made this project more realistic. Further progress would be accelerated, however, by understanding how the sensorimotor transformations underlying *C. elegans* could be implemented with *C. elegans*-like neuronal elements.

Previous work identified the main features of the sensorimotor transformation underlying *C. elegans* chemotaxis [Dus80, PSML99], one of the two forms of spatial orientation identified in this species [RBMP97]. Locomotion during chemotaxis consists of periods of sinusoidal forward movement, called "runs," which are punctuated by bouts of turning (occurring approximately twice a minute) [RC79] that have been termed "pirouettes" [PSML99]. Pirouette probability is modulated by the rate of change of chemical attractant concentration ($dC(t)/dt$). When $dC(t)/dt < 0$, pirouette probability is increased whereas when $dC(t)/dt > 0$, pirouette probability is decreased. Thus, runs down the gradient are truncated and runs up the gradient are extended, resulting in net movement toward the gradient peak.

The chemosensory neurons responsible for the input representation are known [BH91], as are the premotor interneurons for turning behavior [CSW$^+$85]. Much less is known about the interneurons that link chemosensory input to behavioral output [TH03]. To gain insight into how this transformation might be computed at the network level we used an unbiased parameter optimization algorithm to construct model neural networks capable of computing the transformation using *C. elegans*-like neurons. We found that networks with one or two interneurons were sufficient to produce this transformation. A common but unexpected feature of all networks was inhibitory feedback among all neurons. One explanation is that the main function of this feedback is to regulate the latency between sensory input and behavior [DCPSL04].

3.2 Methods

3.2.1 Assumptions

We used simulated annealing [PTVF92] to search for patterns of connectivity sufficient to compute the chemotaxis sensorimotor transformation. The model was constrained by three main assumptions:

1. Chemosensory neurons in *C. elegans* report attractant concentration at a single point in space.
2. Chemosensory interneurons converge on a network of locomotory command neurons to regulate turning probability.
3. The sensorimotor transformation in *C. elegans* is computed mainly at the network level, not at the cellular level.

Assumption (1) follows from the anatomy and distribution of chemosensory organs in *C. elegans* [War73, WTWB75, BH91]. Assumption (2) follows from anatomical reconstructions of the *C. elegans* nervous system [WSTB86], together with the fact that laser ablation studies have identified four pairs of pre-motor interneurons that are necessary for turning in *C. elegans* [CSW$^+$85]. Assumption (3) is heuristic.

3.2.2 Network

Neurons were modeled as passive, isopotential nodes according to the equation:

$$\tau_i \frac{dA_i(t)}{dt} = -A_i(t) + \sigma(I_i), \quad \texttt{with} \quad I_i = \sum_j (w_{ji} A_j(t)) + b_i \tag{3.1}$$

where A_i is the activation level of neuron i in the network, I_i is the sum of all inputs (synaptic and/or otherwise) to neuron i, $\sigma(I_i)$ is the sigmoidal logistic function $1/(1+exp(-I_i))$, w_{ji} is the synaptic input from neuron j to neuron i, and b_i is static bias. The time constant τ_i determines how rapidly the activation approaches its steady-state value for constant I_i.

In conceptual terms, Equation 3.1 represents a neuron whose inputs combine by simple linear addition to set the steady-state activation level. Here, however, steady-state activation is a nonlinear function of net input by virtue of the function σ. This function provides an idealized and compact representation of a variety of saturating nonlinearities that are likely to be present in real *C. elegans* neurons. Chief among these are a regenerative current that has been identified in *C. elegans* chemosensory neurons [GHAL98], and a sigmoidal relationship between presynaptic voltage and transmitter release that has been observed in electrophysiological recordings from *Ascaris*, another species of nematode which is likely to have similar physiology [DS89a, DS89b]. Although synaptic transfer functions have not yet been measured by electrophysiology in *C. elegans* neurons, it is likely that *C. elegans* neurons are similar to *Ascaris* neurons in this respect, because neither species appears to signal by classical spiking neurotransmitter release.

The model of the chemosensory network had one input neuron, eight interneurons, and one output neuron (Figure 3.1). The input neuron ($i = 0$) was a lumped representation of all the chemosensory neurons in the real animal. Sensory input to the network was based on the timecourse of attractant concentration experienced by a real worm in an actual chemotaxis assay [PSML99]. The relationship between attractant concentration and sensory input current in *C. elegans* chemosensory neurons is not yet known; for simplicity, we assumed a linear relationship. Accordingly, attractant concentration in the model ($C(t)$) was scaled and shifted into the range 0 to 1 and simply added to the sensory neuron's net input ($I_{i=0}$ in Equation 3.1). This convention assumes that increases in concentration depolarize the chemosensory neurons in *C. elegans*. However, we also obtained solutions under the opposite assumption (hyperpolarization). The interneurons in the model ($1 \leq i \leq 8$) represented all the chemosensory interneurons in the real animal. The output neuron ($i = 9$) abstractly represented the set of four command neurons that regulate turning behavior: AVA, AVB, AVD, and PVC [CSW$^+$85, ZBM$^+$99]. The activity level of the output neuron ($A_9(t)$) determined the behavioral state of the model, i.e. its turning probability P according to the piecewise function:

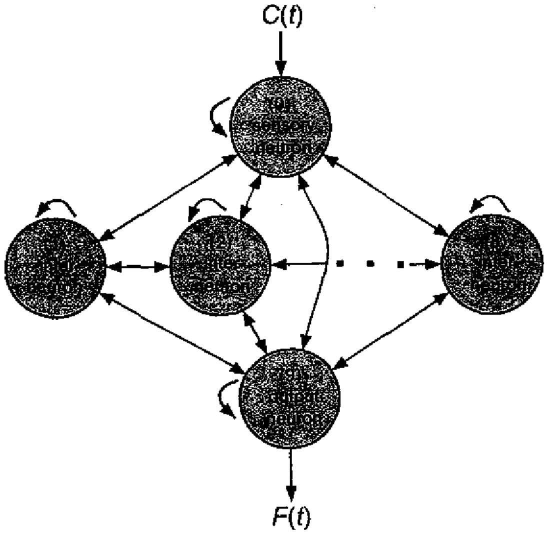


FIGURE 3.1. Model chemosensory network. Model neurons were passive, isopotential nodes. The network contained one sensory neuron, one output neuron, and eight interneurons. Input to the sensory neuron was the timecourse of chemoattractant concentration $C(t)$. The activation of the output neuron was mapped to turning probability by the function $F(t)$ given in Equation 3.2. The network was fully connected and self-connections were allowed. Numerals indicate the index numbers by which neurons were identified.

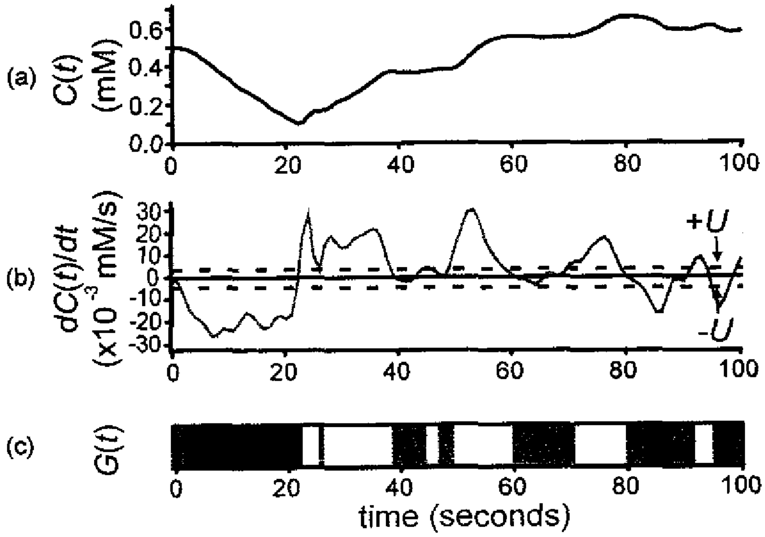


FIGURE 3.2. Construction of the idealized sensorimotor transformation used during network optimization. (A) The timecourse of concentration for a real worm during chemotaxis in a radial gradient of the attractant NH_4Cl. (B) The derivative of the concentration timecourse. (C) Desired turning probability P as a function of time. Black indicates P_{high}, gray indicates P_{rest}, and white indicates P_{low}. Desired turning probability was determined by applying Equation 3.3 to the trace in (B) after being run through a low-pass filter. Thresholds ($\pm U$) are indicated by dashed lines.

$$F(t) = \begin{cases} P_{high} & A_9(t) \leq \alpha_1 \\ P_{rest} & \alpha_1 < A_9(t) < \alpha_2 \\ P_{low} & A_9(t) \geq \alpha_2 \end{cases} \tag{3.2}$$

where α_1 and α_2 are arbitrary thresholds determined during optimization as described in Section 3.2.3. We chose a piecewise function because a previous statistical analysis of intervals between turns in real worms ([PSML99], Figure 5(b)) indicated the existence of distinct turning probability states.

3.2.3 Optimization

We optimized the chemosensory network to compute an idealized version of the true sensorimotor transformation linking $C(t)$ to turning probability [PSML99]. Figure 3.2 shows how the idealized transformation was constructed. First, we computed the instantaneous derivative of the concentration timecourse ($C(t)$, Figure 3.2(A)) recorded from a real worm during chemotaxis by subtracting neighboring points in the timecourse ($dC(t)/dt$, Figure 3.2(B)). The subtraction procedure increased the noise due to experimental error in the original $C(t)$ trace (by additivity of variances between neighboring points). The obvious way to remove added noise is to low-pass filter the $dC(t)/dt$ trace at a corner frequency that is consistent with the time scale of chemosensory processing in *C. elegans*. To find the optimum corner frequency, we plotted the cross-correlation between the timecourse of pirouette-initiation probability and the sign of the low-pass filtered $dC(t)/dt$ timecourse over a range of corner frequencies (0.055-6.05 Hz). The cross-correlation reached a maximum at 1.35 Hz. This value was used for the results presented below. However, the main results of this chapter were insensitive to corner frequency over the range we examined. It was reasonable to identify the optimum corner frequency via the correlation between pirouettes and the sign of $dC(t)/dt$ because previous work has shown this correlation to be the basis of *C. elegans* chemotaxis [PSML99].

We then mapped $dC(t)/dt$ to desired turning probability $G(t)$ according to the relationship:

$$G(t) = \begin{cases} P_{high} & dC(t)/dt \leq -U \\ P_{rest} & -U < dC(t)/dt < +U \\ P_{low} & dC(t)/dt \geq +U \end{cases} \tag{3.3}$$

where U is a threshold derived from previous behavioral observations (Figure 7B in [PSML99]). The result is shown in Figure 3.2(C). The goal of optimization was to make the network's turning probability $F(t)$ equal to the desired turning probability $G(t)$ at all t. This was done by minimizing the average network error per time point, given by:

$$E_{avg} = \frac{1}{T}\int_0^T E(t)dt \tag{3.4}$$

with T equal to the number of time steps in the simulation and $E(t)$ given by:

$$\begin{array}{lll}
\text{case 1}: & \left.\begin{array}{l} F(t) = P_{high} \ \wedge \ G(t) = P_{low} \vee \\ F(t) = P_{low} \ \wedge \ G(t) = P_{high} \end{array}\right\} & E(t) = 1 \\[2ex]
\text{case 2}: & \left.\begin{array}{l} F(t) = P_{high} \ \wedge \ G(t) = P_{rest} \vee \\ F(t) = P_{rest} \ \wedge \ G(t) = P_{high} \end{array}\right\} & E(t) = 1/h \\[2ex]
\text{case 3}: & \left.\begin{array}{l} F(t) = P_{low} \ \wedge \ G(t) = P_{rest} \vee \\ F(t) = P_{rest} \ \wedge \ G(t) = P_{low} \end{array}\right\} & E(t) = 1/h \\[2ex]
\text{case 4}: & F(t) = G(t) & E(t) = 0
\end{array} \tag{3.5}$$

where $h \geq 1$. We introduced the parameter h simply to improve optimization efficiency. This parameter allowed us to penalize networks less for smaller deviations from the desired output (cases 2 and 3) than for larger deviations (case 1). In pilot studies with $h = 2$, we found that 87% of optimization runs converged to a solution, whereas with $h = 1$ only 14% of runs converged. Setting $h = 1$ had no effect on the main results shown in Table 3.1.

Optimization of the network was carried out by annealing three parameter types from Equation 3.1: weights, time constants, and biases. Optimized networks were fully connected and self-connections were allowed. In pilot runs of the optimization algorithm, we noted that many candidate networks performed poorly (high E_{avg}) despite the fact that the activity of the output neuron ($A_9(t)$) rose and fell correctly (i.e. in anti-phase) with respect to $dC(t)/dt$. Further analysis revealed that the output neuron of the poorly performing network often had an incorrect offset, low gain, or both. To circumvent these problems, we added to the performance evaluation routine the ability to minimize E_{avg} by adjusting the thresholds α_1 and α_2 (Equation 3.2). This more generous assessment of network performance increased the efficiency

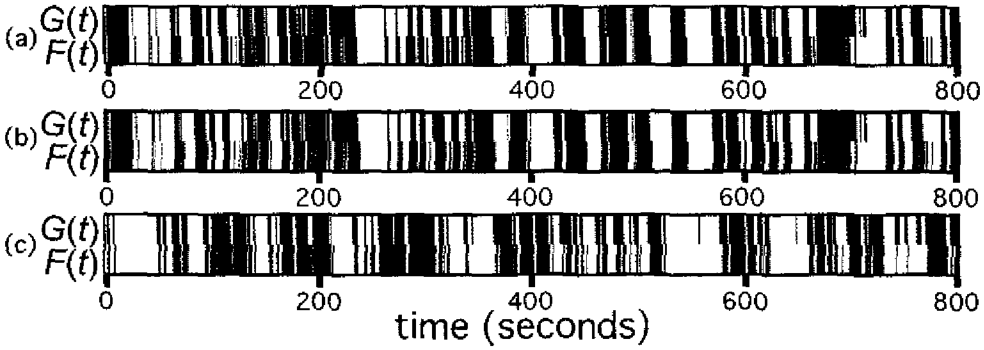


FIGURE 3.3. Network performance after optimization. In each panel ((A)-(C)), the upper trace represents $G(t)$, the desired turning probability in response to a particular $C(t)$ timecourse (not shown), whereas the lower trace represents $F(t)$, the resulting network turning probability. Shading signifies turning probability (black = P_{high}, gray = P_{rest}, white = P_{low}). (A) Performance of a typical network after optimization. (B) Performance of the same network after pruning away inactive interneurons. (C) Performance of the pruned network when stimulated by a different $C(t)$ timecourse. Note that in ((A)-(C)), network turning probability is delayed relative to desired turning probability because of the time required for sensory input to affect behavioral state in the model.

of the optimization routine by an order of magnitude.

We ran the optimization algorithm 50 times on $C(t)$ data from 10 different real worms for a total of 500 networks. Although each of the 10 worms demonstrated exemplary chemotaxis (in that they went directly to the peak of the gradient and stayed there for the remainder of the assay), their $C(t)$ timecourses were unique, exhibiting a wide range of individual behavior. The result of one optimization run on one worm is illustrated in Figure 3.3(A), which shows good agreement between network ($F(t)$) and desired turning probabilities ($G(t)$). We noted that in most networks many interneurons had a constant offset and showed little or no response to changes in sensory input. These interneurons were eliminated using a pruning procedure in which the tonic effect of the offset was absorbed into the bias term of postsynaptic neurons. In general, pruning had little or no effect on network performance (Figure 3.3(A) versus Figure 3.3(B)), suggesting that the eliminated neurons were indeed nonfunctional. Even after pruning, most of the 500 networks performed well as judged by eye. For example, 449 of the 500 networks (90%) performed at least as well as the network shown in Figure 3.3(B). This result indicates that the optimization algorithm was an efficient means of finding networks capable of reproducing the chemotaxis sensorimotor transformation.

Of the 449 top-performing networks, 234 (52%) had one interneuron, 170 (38%) had two interneurons, and the remainder had three or four interneurons. We conclude that computation of the chemotaxis sensorimotor transformation does not require a large number of interneurons. For the remainder of this study, we focused on the single-interneuron networks because this was the largest class of networks.

3.2.4 Generalization

A key test of an optimized network is whether it responds correctly to inputs that were not presented during optimization. We measured generalization using two independent methods.

In the first method, we challenged each of the 449 top-performing networks with $C(t)$ data from a new worm, i.e. one that was not a member of the set of 10 worms used during optimization. The $G(t)$ function (Equation 3) for this worm is shown in the top panel of Figure 3.3(C). There was generally good agreement between network output and desired turning probability as judged by eye. We used Equation 5 to quantify the degree of agreement, and ranked networks accordingly; 80% of the networks generalized at least as well as the network whose output $F(t)$ is shown in the lower panel of Figure 3.3(C).

In the second method, we asked whether the network was able to direct locomotion of a model worm to the center of a virtual gradient. The virtual gradient was identical in shape to the real gradient that was used in actual behavioral assays [PSML99]. A worm was represented as a point whose position was updated at one second intervals. Point displacement was computed from the instantaneous speed (v) and head angle (θ). Like real worms, the model worm could exist in two states: running or turning. At the beginning of each interval, the behavioral state was updated according to the current turning probability (Equation 3.2), where $P_{high} = 0.42$ and $P_{low} = 0.04$.

The values for P_{high} and P_{low} were taken from fits to the exponential distributions of intervals between turns in the original statistical analysis ([PSML99], Figure 5(b)). The value for P_{rest} (0.08) was taken from the average turning rate of real worms in the absence of a chemical gradient. Instantaneous speed was 0.15 mm/sec during runs and 0.1 mm/sec during turns [PSML99]. Direction was updated as follows. If the worm was in the run state, the simulation sampled from a uniform distribution of $\Delta\theta$ values where $\Delta\theta < \pm 5°$. If the worm was in the turn state, the simulation sampled from a uniform distribution where $\Delta\theta > \pm 50°$. Like real worms, the model worms assayed for 1200 seconds.

As with real worms (Figure 3.4(A)), most networks were able to direct the locomotion of a model worm up a virtual gradient (Figure 3.4(B)). We quantified the performance of each of the 449 well-optimized networks by computing its success rate in reaching the near-vicinity of the gradient's peak (circle in Figure 3.4(B)) in 1000 attempts (Figure 3.4(C)). 98% of the networks performed above the chance level, defined as the success rate for a model worm in which turning probability was fixed at P_{rest}. We conclude that the optimized networks showed an acceptable level of generalization in response to novel inputs.

3.3 Results

3.3.1 Common Features of Networks

All single-interneuron networks, regardless of which worm the $C(t)$ data came from, had three common features, summarized in Table 3.1 and Figure 3.5. First, the direct pathway from sensory neuron to the output neuron was excitatory, whereas the parallel, indirect pathway via the interneuron was inhibitory. Such a circuit computes an approximate derivative of its input by subtracting a delayed version of the input from its present value [MSF94]. Second, all neurons had significant inhibitory self-connections. We noted that inhibitory self-connections were strongest on the input and output neurons, the two neurons comprising the direct pathway representing current sensory input. We hypothesized that the function of larger inhibitory

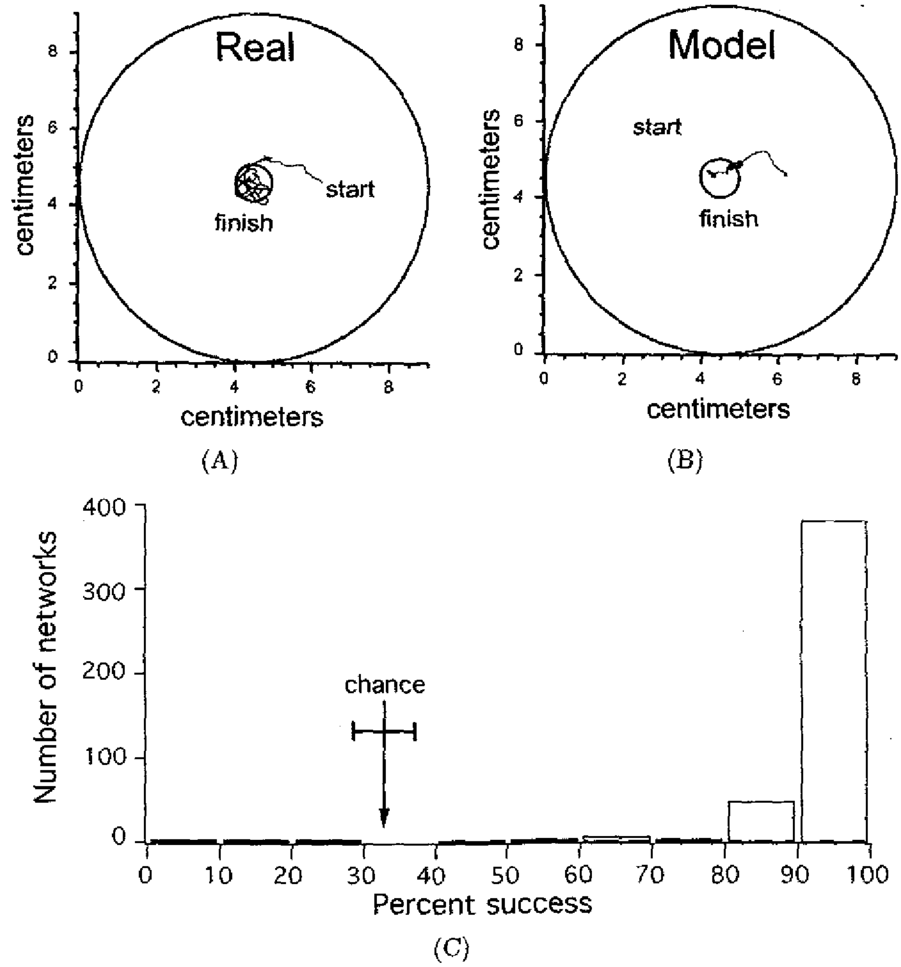


FIGURE 3.4. Test of generalization in a radial gradient. (A) Track of a real worm. (B) Track of a model worm in a virtual gradient. (C) Distribution of success rates for single neuron networks ($N = 382$). Success rate for each network was computed as the number of times a model worm reached the gradient peak (circle in (B)) in 1000 trials. The arrow indicates the average success rate (32.4% ± 5%, 99.9% confidence interval, 200,000 trials) for a model worm in which turning probability was fixed at P_{rest}.

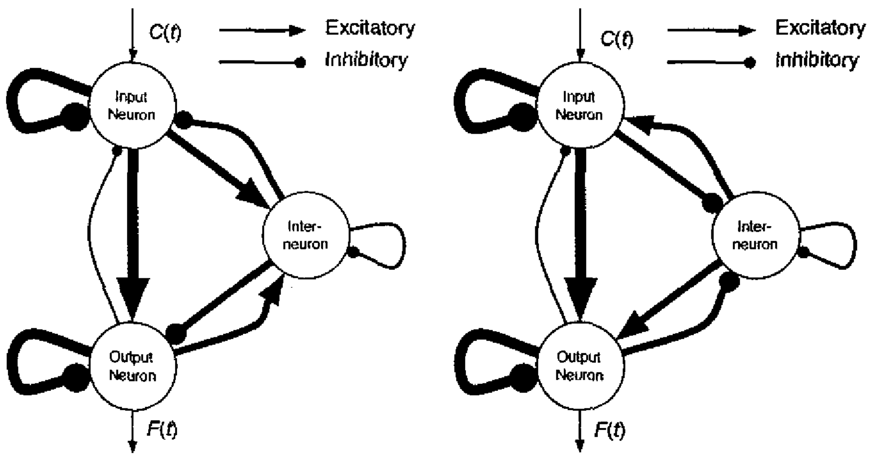


FIGURE 3.5. The two connectivity patterns observed for single-interneuron networks. Connection strength is proportional to arrow thickness. The two networks differ only in the polarity of connections to and from the interneuron, which are reversed. They are functionally equivalent because in both networks the net effect of the disynaptic pathway from input to output is inhibitory.

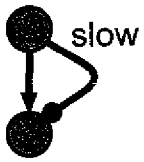
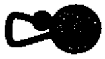
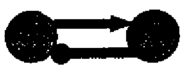
Feature	Figure	Function
direct excitatory delayed inhibitory		differentiation
self-connection		hypothesis: regulation of response latency
inhibitory recurrent connection		

TABLE 3.1. Common features of single-interneuron networks.

self-connections was to decrease response latency in the direct pathway, perhaps by reducing the effective time constant of these neurons. Such a decrease would be a means of compensating for the fact that $G(t)$ was an instantaneous function of $C(t)$, whereas the neuronal time constant τ_i tends to introduce a delay between $C(t)$ and the network's output $F(t)$. Third, the net effect of all disynaptic recurrent connections was also inhibitory. By analogy to inhibitory self-connections, we hypothesized that the function of the recurrent pathways was also to regulate response latency.

3.3.2 Output Delay

To test the hypothetical functions of the self-connections and recurrent connections, we introduced an explicit time delay (Δt) between $dC(t)/dt$ and the desired turning probability $G(t)$ such that:

$$G'(t) = G(t - \Delta t) \tag{3.6}$$

$G'(t)$ was substituted for $G(t)$ during optimization. We then repeated the optimization procedure with a range of Δt values (0 to 3 seconds) and looked for systematic effects on weights and time-constants (Figure 3.6).

Effects on self-connections. We found that the strengths of self-connections on all three neurons were inversely related to Δt (Figure 3.6(A), ANOVA: input neuron $F_{3,309} = 51.98$, $p < 0.001$; interneuron $F_{3,309} = 9.441$, $p < 0.001$; output neuron $F_{3,309} = 63.54$, $p < 0.001$). This result is consistent with the hypothesis that the function of these self-connections is to regulate response latency.

Effects on recurrent connections. We quantified the strengths of recurrent connections by taking the products of the two weights along each of the three recurrent loops in the network. We found that the strengths of the two recurrent loops that included the interneuron were inversely related to Δt (Figure 3.6(B), ANOVA: input neuron to interneuron $F_{3,309} = 4.14$, $p < 0.005$; output neuron to interneuron $F_{3,309} = 2.755$, $p < 0.05$). This result suggests that the function of these loops is

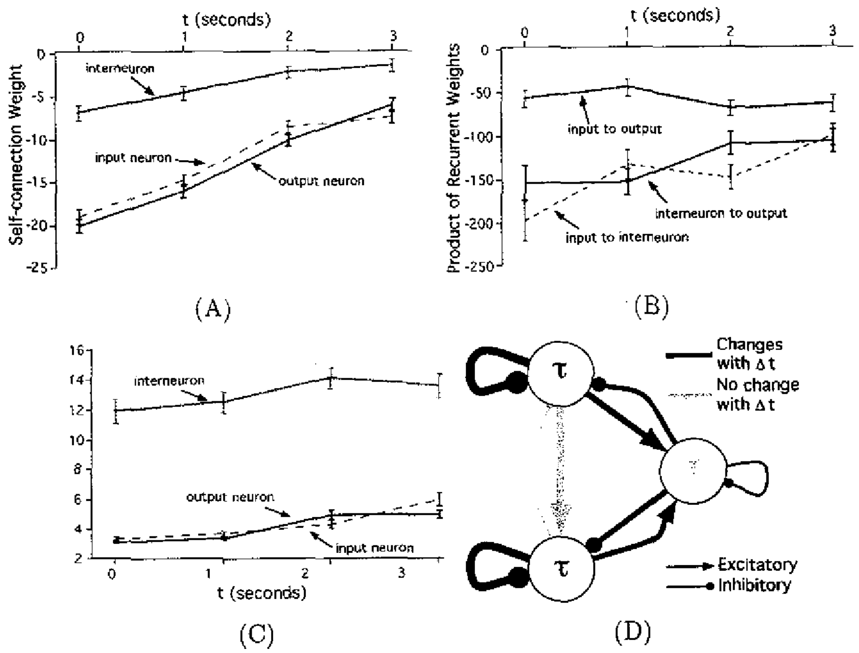


FIGURE 3.6. The effect of output delay (Δt) on network parameters. Plotted points are averages with error bars representing standard error. (A) Self-connections. (B) Recurrent connections. Recurrent connection strength was quantified by taking the product of the weights along each of the three recurrent loops in Figure 3.5. (C) Time constants. (D) Summary of effects. Parameters affected by changes in output delay are shown in black. Unaffected parameters are shown in gray. The average time constant is proportional to the font-size of τ shown in the neuron. Average connection magnitude is proportional to line width. Parameter magnitudes are shown only for $\Delta t = 0$.

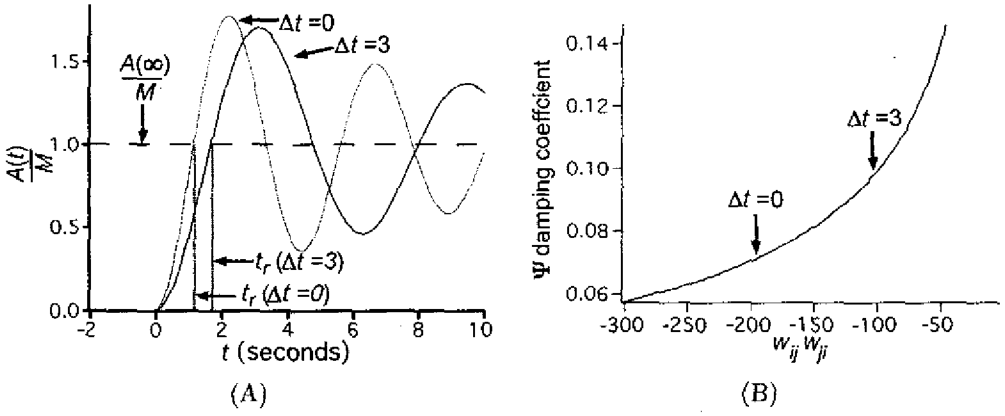


FIGURE 3.7. Analysis of two linear neurons with recurrent synapses. (A) Response of the system to an instantaneous step of magnitude M occurring at $t = 0$. The gray line shows the response $A(t)$ when the product of the recurrent connections $(w_{ij}w_{ji})$ is the average value of the input to interneuron loop obtained for networks optimized for output delay $\Delta t = 0$. The dark line shows the response when the recurrent product optimized with $\Delta t = 3$. Rise time (t_r), the time it takes $A(t)$ to first cross its eventual steady-state output, increases as Δt increases. (B) Plot of the relationship of damping coefficient to the product of weights on recurrent synapses. Arrows indicate the values of the damping coefficient for average values of the recurrent product. These averages were obtained at the indicated output delay Δt.

to regulate response latency and supports the hypothetical function of the recurrent connections. Interestingly, however, the strength of the recurrent loop between input and output neurons was not affected by changes in Δt ($F_{3,309} = 1.361$, $p = 0.255$). Thus, effects on recurrent loop products were limited to two of the three loops in the network.

Effects on time constants. We found that the time constants of the input neuron and the output neuron increased with output delay (Figure 3.6(C), ANOVA: input neuron $F_{3,309} = 19.9$, $p < 0.001$; output neuron $F_{3,309} = 21.56$, $p < 0.001$). This result suggests that time constants also contributed to the regulation of response latency. However, we noted that there was no effect of optimization delay on the time constant of the interneuron ($F_{3,309} = 1.791$, $p = 0.149$). Thus, effects on time constants were restricted to two of the three neurons in the network.

Figure 3.6(D) summarizes the effects of output delays. The changes reflect two distinct mechanisms for regulating latency. Changes in self-connections and time constants represent cellular level mechanisms, whereas changes in the strengths of recurrent connections represent network level mechanisms. The two mechanisms were distributed differently within the network. The cellular level mechanism was confined to the input and output neurons, whereas the network level mechanism was confined to synaptic pathways involving the interneuron. The difference between the distributions of the two mechanisms could reflect performance trade-offs. Alternatively, differences could also reflect properties of the optimization algorithm.

3.3.3 Analysis

Self-connections. To provide a theoretical explanation for the effects of time delays on the magnitude of self-connections, we analyzed the step response of Equation 3.1 for a reduced system containing a single linear neuron with a self-connection, which is given by:

$$\tau_i \frac{dA_i(t)}{dt} = w_{ii}A_i(t) - A_i(t) + h(t) \tag{3.7}$$

where $h(t)$ represents a generic external input (sensory or synaptic). Solving Equation 3.7 for $h(t)$ equal to an instantaneous step of amplitude M at $t = 0$ with $A_1(0) = 0$ gives:

$$A_i(t) = \left(\frac{M}{1 - w_{ii}}\right)\left[1 - exp\left[-\left(\frac{1 - w_{ii}}{\tau_i}t\right)\right]\right] \tag{3.8}$$

From Equation 3.8, when $w_{ii} = 0$ (no self-connection) the neuron relaxes to steady-state at the rate $1/\tau_i$, whereas when $w_{ii} < 0$ (inhibitory self-connection) the neuron relaxes at the higher rate of $(1 + |w_{ii}|)/\tau_i$. Thus, single-neuron response latency drops as the strength of the inhibitory self-connection increases and, conversely, response latency rises as self-connection strength decreases. This result explains the effect on self-connection strength of introducing a delay between $dC(t)/dt$ and turning probability (Figure 3.6(A)). This result is also consistent with the fact that the magnitudes of the interneuron self-connection were consistently smaller than the magnitude of the other two self-connections. Smaller interneuron self-connections lead to longer response latencies in this neuron, whose function is to present a delayed version of the input to the output neuron.

Recurrent inhibitory connections. We made a similar analysis of the effects of time delays on the recurrent connections. Here, however, we studied a reduced system of two linear neurons with recurrent synapses and an external input to one of the neurons.

$$\tau_i \frac{dA_i(t)}{dt} = w_{ji} A_j(t) - A_i(t) + h(t) \tag{3.9}$$

$$\tau_j \frac{dA_j(t)}{dt} = w_{ij} A_i(t) - A_j(t) \tag{3.10}$$

We solved this system for the case where the external input $h(t)$ is an instantaneous step of size M that occurs at time $t = 0$, with $A_i(0) = A_j(0) = 0$, and $\tau_i = \tau_j = \tau$. For the case where $w_{ij} w_{ji} < 0$, the solution to Equations 3.9 and 3.10 are 2nd-order differential equations with complex roots:

$$\begin{aligned} A_i(t) = 1 - & \left(\frac{\sqrt{1 - w_{ij} w_{ji}}}{\sqrt{-w_{ij} w_{ji}}} \right) \exp\left(-t/\tau\right) \\ \times & \left[\sin\left(\left(\tfrac{\sqrt{-w_{ij} w_{ji}}}{\tau} \left(1 + \sqrt{-w_{ij} w_{ji}}\,\right) \right) t + \phi \right) \right] \end{aligned} \tag{3.11}$$

$$\begin{aligned} A_j(t) = & \left(\frac{M w_{ij}}{1 - w_{ij} w_{ji}} \right) \\ \times & \left[1 - \left(\tfrac{\sqrt{1 - w_{ij} w_{ji}}}{\sqrt{-w_{ij} w_{ji}}} \right) \exp\left(-t/\tau\right) \left[\sin\left(\tfrac{\sqrt{-w_{ij} w_{ji}}}{\tau} t + \phi \right) \right] \right] \end{aligned} \tag{3.12}$$

$$\phi = \arctan\left(\sqrt{-w_{ij} w_{ji}}\,\right) \tag{3.13}$$

The step response of the two-neuron system is an underdamped oscillation (Figure 3.7(A)). The extent to which the oscillation is damped is given by the damping coefficient:

$$\Psi = \frac{1}{\sqrt{1 - w_{ij}w_{ji}}} \tag{3.14}$$

Damping affects several key aspects of the step response including rise time (t_r), which is defined as the latency between step onset and the time it takes output to first cross its eventual steady-state value. From Equation 3.14 it is evident that as the strength of the recurrent loop (i.e. $|w_{ij}w_{ji}|$) decreases, damping increases (Figure 3.7(B)). Thus, response latency rises as the strength of the recurrent loop decreases. This result explains the effect on recurrent loop strength of introducing a delay (Δt) between $dC(t)/dt$ and turning probability (Figure 3.6(B)).

3.4 Conclusion

We used simulated annealing to search for networks capable of computing an idealized version of the chemotaxis sensorimotor transformation in *C. elegans*. We found that one class of such networks is the three-neuron differentiator with inhibitory feedback. The appearance of differentiator networks was not surprising [MSF94] because the networks were optimized to report, in essence, the sign of $dC(t)/dt$ (Equation 3.3). The prevalence of inhibitory feedback, however, was unexpected. Inhibitory feedback was found at two levels: self-connections and recurrent connections. Combining an empirical and theoretical approach, we have argued that inhibitory feedback at both levels could function to regulate the response latency of the system's output relative to its input. Such regulation would be significant in the *C. elegans* nervous system if the membrane time constant were a limiting factor in response latencies.

There are intriguing parallels between our three-neuron network models and the biological network. Figure 3.8 shows the network of interneurons interposed between the chemosensory neuron class ASE, the main chemosensory neurons for salt chemotaxis, and the locomotory command neurons classes AVA and AVB. The interneu-

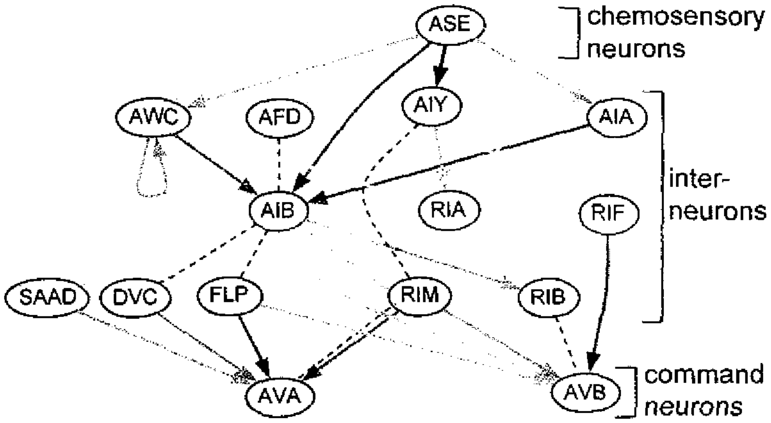


FIGURE 3.8. The network of chemosensory interneurons in *C. elegans*. Shown are the interneurons interposed between the chemosensory neuron ASE and the two locomotory command neurons AVA and AVB based on their anatomical connections. These interneurons are hypothetical members of the chemotaxis network. Whether or not they actually play a role in chemotaxis remains to be tested. The diagram is restricted to interneuron pathways with at most three synapses. Arrows represent chemical synapses. Dashed lines represent gap junctions. Pathways containing synaptic partners with fewer than two presynaptic densities, or fewer than three gap junctions, were omitted. Connectivity is inferred from the anatomical reconstructions [WSTB86]. Left and right symmetrical pairs are collapsed into single neurons and the number of connections to the pair is summed. Connections between asymmetrical pairs are drawn as self-connections.

rons in Figure 3.8 are candidates for computing the sensorimotor transformation for chemotaxis in *C. elegans.* Three parallels are prominent. First, there are four candidate differentiator circuits, as noted previously [Whi85]. These circuits are formed by the neuronal triplets ASE-AIY-RIA, ASE-AIA-AIB, ASE-AFD-AIB and ASE-AWC-AIB. Second, there are self-connections on three neuron classes in the circuit, including AWC and RIA, which may both be involved in the differentiator circuits. These self-connections represent anatomically identified connections between left and right members of the respective neuron pair; we represented them here as self-connections because each member of a pair is pre- and post-synaptic to analogous neurons and so the two neurons probably function as a single unit. It remains to be seen, however, whether these connections are inhibitory in the biological network. Self-connections could also be implemented at the cellular level by voltage dependent currents. A voltage-dependent potassium current, for example, can be functionally equivalent to an inhibitory self-connection because depolarization of the neuron recruits a hyperpolarizing current. Electrophysiological recordings from ASE and other neurons in *C. elegans* confirm the presence of such currents [GHAL98, NPBK02]. Thus, it is conceivable that many neurons in the biological network have the cellular equivalent of self-connections. Third, there are reciprocal connections between ASE and three of its six postsynaptic targets. These connections could provide recurrent inhibition if they have the appropriate signs.

Common patterns of connectivity between the model and biological networks suggest new functionality for previously identified connections in the *C. elegans* nervous system. It should be possible to test these functions through electrophysiological recordings, calcium imaging, and neuronal ablations.

CHAPTER 4

Neural Network Motifs for Chemotaxis and Thermotaxis in *C. elegans*

In Chapter 3, the neural network that drives *C. elegans* chemotaxis was modeled assuming a single mechanism for behavior. This behavior model was the pirouette mechanism [PSML99], which stated that the worm was more likely to perform a reorienting turn when chemoattractant concentration was decreasing. In contrast, the work in this chapter generates neural network models that drive a model worm to exhibit chemotaxis without assuming the pirouette mechanism. In addition to modeling chemotaxis behavior, this chapter also explores neural network models that reproduce thermotaxis behavior, in which the worm moves along a temperature gradient. In contrast to Chapter 3, the neural network model in this study contained two sensory neurons instead of one, had one interneuron instead of eight, the activity of the output neuron was binary, and there was no recurrence.

Portions of the chemotaxis modeling work have already been published [DPSCL06]. Additionally, we used the results from chemotaxis modeling as a case study in Chapter 5. John Conery contributed to coding, analysis, and writing. Shawn Lockery provided major contributions to editing, analysis, and the direction of this work. Nathan Dunn was the major contributor to this work, providing most of the writing and direction as well as the majority of the coding, network optimization, and network analysis.

4.1 Introduction

The nearly complete description of the morphology and synaptic connectivity of all 302 neurons in the nematode *Caenorhabditis elegans* [WSTB86, CHC06] raised the prospect of the first comprehensive understanding of the neuronal basis of an animal's entire behavioral repertoire. New electrophysiological and functional imaging techniques for *C. elegans* neurons [LG98, KLRB^{+}00] has made this project more realistic. Knowing the neural function of the neurons that drive the sensorimotor transformations in *C. elegans* would accelerate further progress.

Previous experimental work has identified the main features of the sensorimotor transformation of two important spatial orientation behaviors in *C. elegans* [RBMP97]: chemotaxis [Dus80, PSML99] and thermotaxis [HR75, MO95]. Locomotion during both behaviors consists of periods of sinusoidal forward movement, called "runs," punctuated by bouts of turning (occurring approximately twice a minute) [RC79] that have been termed "pirouettes" [PSML99, ZMFL03].

During chemotaxis, *C. elegans* normally orients towards a maximum chemical attractant concentration. Pirouette probability is modulated by the rate of change of chemical attractant concentration ($dC(t)/dt$) [PSML99, MTF^{+}05]. When $dC(t)/dt < 0$, pirouette probability is increased whereas when $dC(t)/dt > 0$, pirouette probability is decreased. Thus, runs down the gradient are truncated and runs up the gradient are extended, resulting in net movement toward the gradient peak. The likely chemosensory neurons responsible for the input representation are known [BH91], as are the premotor interneurons for turning behavior [CSW^{+}85]. However, much less is known about the interneurons that link chemosensory input to behavioral output [TH03, WKS04, GHB05].

During thermotaxis, *C. elegans* attempts to return to its cultivation temperature T_{cult} from a higher or lower temperature T. Pirouette probability is modulated by the rate of change of temperature $dT(t)/dt$ with respect to T_{cult} [RS02, ZMFL03]. Many of the neurons responsible for thermotaxis have been identified [MO95] and activity has been measured in these neurons during applied thermal stimulus [KMMM04, SS05]. Furthermore, there is evidence that thermotaxis is driven by two sub-motifs, one that is active above T_{cult}, and the other one that is active below T_{cult} [MO95].

To gain insight into how chemotaxis and thermotaxis sensorimotor transformations might be computed at the network level, we used an unbiased stochastic parameter optimization algorithm to construct model neural networks capable of computing either the chemotaxis or thermotaxis sensorimotor transformations using *C. elegans*-like neurons. Through analysis of neuronal activation and architecture of the neural network solutions, we found several motifs (recurring patterns of neural response function) for each sensorimotor transformation problem. By comparing these motifs to experimental data, we are able to make predictions about possible function and connectivity of the chemotaxis and thermotaxis networks in *C. elegans*. For chemotaxis, we found a previously identified neural network motif [MSF94, DCPSL04] as well as two other motifs that are unexpected. Thermotaxis neural network solutions consisted of opposing sub-motifs of identified chemotaxis motif pairs, one thermophilic (heat-seeking) and the other cryophilic (cold-seeking). At any time, only one of the two sub-motifs was active. For most neural network solutions, each sub-motif relied on its own sensory neuron.

4.2 Methods

4.2.1 Assumptions

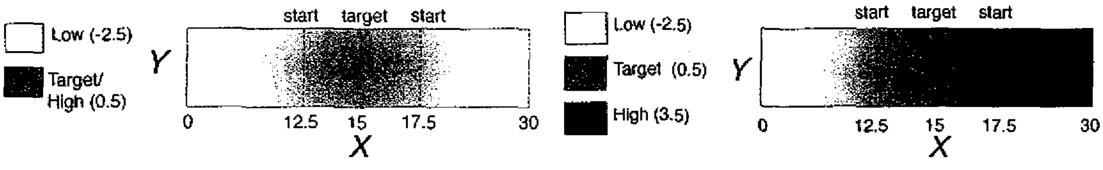
Our model *C. elegans* neural networks were constrained by three main assumptions:

1. *C. elegans* reports attractant concentration from a single point in space.
2. *C. elegans* reports temperature from a single point in space.
3. Interneurons from both sensory neurons converge on a network of locomotory command neurons to regulate turning probability.

Assumptions 1 and 2 follow from the anatomy and distribution of chemosensory organs in *C. elegans* [War73, WTWB75, BH91, MO95]. Assumption (3) follows from anatomical reconstructions of the *C. elegans* nervous system [WSTB86], together with the fact that laser ablation studies have identified four pairs of premotor interneurons that are necessary for turning in *C. elegans* [CSW$^+$85, GHB05].

4.2.2 Model Worm Movement

The model gradients (Figure 4.1) had similar geometries to those used in actual behavioral assays for chemotaxis [LPS99] and thermotaxis [YO03]. The model worms were assayed over 1200 one-second time-steps to match experimental tracking time [PSML99]. One-second intervals gave a satisfactory balance of computational efficiency and resolution.



(A) (B)

FIGURE 4.1. Model gradients used for the (A) chemotaxis and (B) thermotaxis optimization problems. The worm travels in two dimensions along a gradient that changes linearly in only the X direction. In the chemotaxis gradient (A), the worm attempts to maximize its attractant concentration by moving to the maximal attractant concentration of 0.5 at $X = 15$. The worm begins from alternating starting points of $X = 12.5$ and $X = 17.5$, which both have values of 0. In the thermotaxis gradient (B), the worm attempts to return its position to the target value (T_{cult}) of 0.5 from either a higher or lower temperature. The positions and values for the thermotaxis gradient are the same as for chemotaxis except that the high starting value at $X = 17.5$ is 1 instead of 0. Gradient values are arbitrary. In both problems, worms were initially oriented away from the target.

In our model, a worm was represented as a point in two-dimensional space whose position is updated at one-second intervals. The point displacement was computed at every time-step as shown by Figure 4.2(B). First, network input $u(t)$, which represents either chemical attractant concentration or temperature, is calculated from the worm's x, y position on the gradient. Second, the neural network determines the movement state of the worm as either a forward moving "run" or randomly reorienting "turn". Third, the instantaneous velocity v and head angle θ are determined based on the worm's movement state and *C. elegans* movement data [PSML99]. θ is updated at each time-step by $\Delta t d\theta/dt$. If in the run state, $v = 0.15mm/s$ and $d\theta/dt = 0$. If in the turn state, $v = 0.11mm/s$ and $d\theta/dt$ is sampled from a uniform distribution of $|d\theta/dt| > 50°/\text{s}$. Finally, the head angle is adjust to θ and the worm is moved forward $v\Delta t$ to a new x, y position.

4.2.3 Neural Network Model

The neural network model that governed the worm's behavior (Figure 4.2(A)) had two input (sensory) neurons, a single interneuron, and an output neuron, which determined the worm's turn probability. The sensory neurons sampled the attractant concentration or temperature at the worm's position on the gradient $u(t)$. We used a pair of sensory neurons for two reasons. First, evidence from thermotaxis [MO95] suggests that at least two thermosensory neuron are involved in *C. elegans* thermotaxis. Second, we were unable to optimize neural networks for the thermotaxis problem with only a single sensory neuron, suggesting that an additional sensory neuron was necessary for this behavior. Only a single interneuron was included in this model to simplify analysis as well as to find the simplest motifs required for behavior. For computational efficiency, the forward and reverse command neuron pools [CSW$^+$85, WR95, ZBM$^+$99] will be modeled as a single stochastic (Boltzmann/Hopfield) neuronal unit.

Sensory neurons and interneurons were modeled as passive, isopotential nodes according to the equation:

$$\tau_i \frac{dA_i(t)}{dt} = -A_i(t) + \sigma(I_i) \tag{4.1}$$

where A_i is the activation level of neuron i in the network, I_i is the input to neuron i, and $\sigma(I_i)$ is the sigmoidal logistic function $1/(1 + exp(-I_i))$. The time constant τ_i determined how rapidly the activation approaches its steady-state value for constant I_i. Input I_i is defined as the sum of all inputs (synaptic or otherwise) to neuron i:

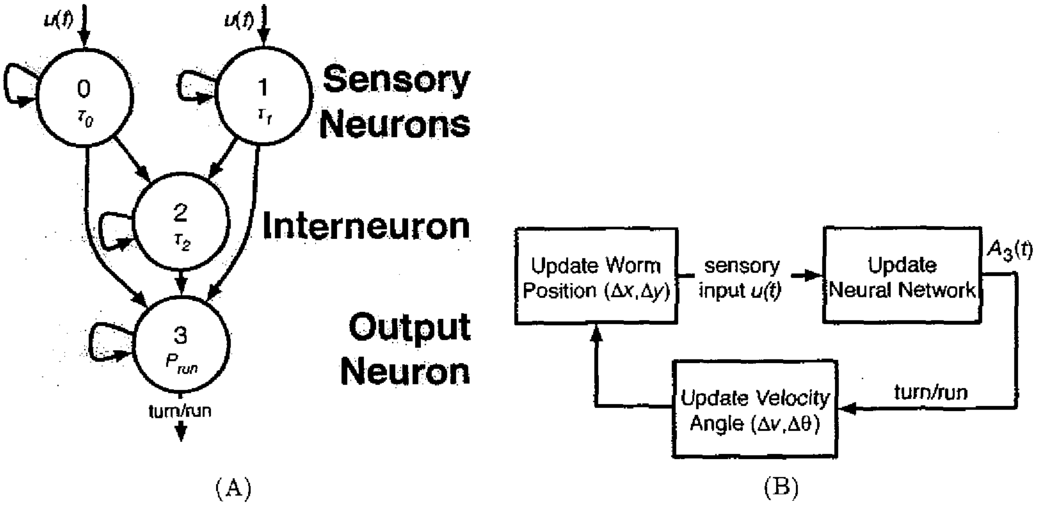


FIGURE 4.2. Model worm movement. (A) Model four-neuron neural network. The network is governed by Equations 4.1-4.3. Black arrows represent synaptic weight. Network output is defined as the activation of the output neuron $A_3(t)$. (B) Update procedure for model worm. At each time step, sensory input $u(t)$ (attractant concentration or temperature) is sampled by the sensory neurons of the model neural network (A) at the worm's position on the gradient (x, y). Next, the neural network calculates network output, $A_3(t)$, which determines the worm's movement state, either run (1) or turn (0). Based on the movement state, the worm's velocity v and head-angle Θ are adjusted. Finally, the worm's position (x,y) is updated according to v and Θ and the procedure is repeated.

$$I_i = \sum_j (w_{ji} A_j(t)) + b_i + (w_{ui} u(t)) \tag{4.2}$$

where w_{ji} is the synaptic weight from neuron j to neuron i, b_i is bias, $u(t)$ sensory input, and w_{ui} is the gain of $u(t)$ onto neuron i, which is 0 for non-sensory neurons. The gain parameter was added to amplify changes in the gradient.

The output Neuron 3 is a binary, stochastic neuron that dictates whether the worm is in the run or turn state. The on-state ($A_3(t) = 1$) of the output neuron indicates forward locomotion, whereas the off-state ($A_3(t) = 0$) indicates turning. The probability of the worm being in the run state is a sigmoidal function of net synaptic input $I_3(t)$ into the output neuron:

$$P_{run}(t) = 1 - P_{turn}(t) = \sigma(I_3(t)k) \tag{4.3}$$

where k is a constant that amplifies network output and $P_{turn}(t)$ is the probability of being in the turn state. When $A_3(t) = 0$, the self-connection on the output is ineffectual. k was set to 30 to hasten optimization. If k was too small (e.g., 1), the output neuron was insensitive to changes in the network. If it was too great (e.g., $k = 100$), the output neuron would act as a deterministic, bistable switch.

4.2.4 Model Evaluation

Networks were optimized to minimize error by reducing the model worm's distance from its target attractant concentration or temperature. The error for trial n of an individual worm $E_{worm,n}$ was the sum of the absolute distance from the worm's x position to the target x position in only the x direction (Figure 4.1) over T time-steps:

$$E_{worm,n} = \Delta t \sum_{t=0}^{T-1} |x(t)_{worm} - x_{target}| \tag{4.4}$$

where $x(t)_{worm}$ is the worm's horizontal position at time t, and x_{target} is the target horizontal position. The vertical position y is irrelevant in this analysis as the gradient doesn't change in that orientation. For a neural network $E_{network}$ was calculated as the average error for a neural network run over N trials by:

$$E_{network} = \frac{\sum_{n=1}^{N} E_{worm,n}}{NT} \tag{4.5}$$

Optimized neural networks were deemed valid neural network solutions when $E_{network} < 1.25$, since $E_{network} = 1.25$ if a worm only bounces between opposing starting points.

4.2.5 Optimization

The neural network model from Figure 4.2(A) was optimized to perform the orientation tasks described in Figure 4.1 by minimizing $E_{network}$ in Equation 4.5. The optimization was done using simulated annealing, a widely used stochastic optimization algorithm [PTVF92, Mas93] that decreases its search range over a series of decreasing "temperature steps". The following parameters from Equations 4.1 and 4.2 were adjusted during optimization: w_{ij}, τ_i, and b_i. The parameters ranged from -30 to 30. When the range was smaller, the algorithm was unable to find good solution, whereas when it was larger, it inhibited refinement of solutions. Each neural network optimization examined 100,000 parameter sets — 500 temperature steps with 200 iterations per temperature step. In Equation 4.5, the number of worms per evaluation of $E_{network}$ was $N = 100$, evaluated over $M = 1200$ one-second time-steps.

Worm runs were evenly distributed over an IBM p690[1] 16-node, shared-memory server or an 11-node, Beowulf cluster. Software was written in C++ using MPI[2] for parallelization, and the blitz[3] library was used for array processing. Each optimization run took approximately 5 hours to complete on either computer system.

---

[1] http://www-1.ibm.com/servers/eserver/pseries/hardware/highend/p690.html

[2] http://www-unix.mcs.anl.gov/mpi/mpich/

[3] http://www.oonumerics.org/blitz/

4.2.6 Motif Identification

We defined motifs as patterns of neuronal activation $A_i(t)$ relative to input $u(t)$ that occurred in multiple network solutions. Motifs were identified through analysis of optimized network solutions where the simulated worms performed either chemotaxis or thermotaxis.

Neuronal activation response to network input most relevant for identification of motifs for a single neuron i were (1) the range of $u(t)$ over which $A_i(t)$ responded, (2) the rate of change of $A_i(t)$ relative to changing $u(t)$, and (3) the sign of the pathway from $u(t)$ to the output neuron. The rate of change of $A_i(t)$ relative to $u(t)$ is referred to as the "speed" of the neuron. Observation of these parameters was done during generalization (Section 4.2.7). Combining the individual identified neurons according to our three criteria led to motif discovery. The speed of the neuron was set during optimization. The speed of a neuron could be explicitly increased by reducing the time-constant τ_i (Equation 4.1) or increasing the weight of inhibitory self-connections w_{ii} (Equation 4.2). Even though these parameters specifically affect speed, other factors, such as steady-state network response and up-stream connectivity, also contribute to the neuron's function.

4.2.7 Generalization

To verify that the network solutions we generated were not artifacts of the initial optimization conditions, we performed a generalization test that evaluated networks ($E_{network}$, $N = 100$ and $T = 1200$, Equation 4.5) with model worms that had random starting orientation and position. This test provided incremental differences in geometry, position, and orientation from the original optimization conditions (Figure 4.1) to detect over-optimized network solutions.

For both chemotaxis and thermotaxis, values on the gradient (concentration or temperature) were linear with respect to radius. The maximum radius was 4.5 cm. The target radius was 2.5 cm, and had value of 0.5 for both chemotaxis and thermotaxis. In the chemotaxis generalization gradient, concentration was -0.125 on the edge and 0 at the center. The chemotaxis generalization gradient is shown in Figures

4.3(B), 4.4(B), and 4.5(B). Thermotaxis generalization involved two radial gradients. The first gradient had a minimum temperature of -0.125 on the edge and a maximum temperature of 1 at the center as shown in Figures 4.7(B) and 4.8(B). The second gradient had a maximum temperature of 1.125 at the edge and a minimum temperature of 0 at the center as shown in Figures 4.7(E) and 4.8(E).

To test robustness, we optimized neural networks with varying amounts of white noise added to the output neuronal activation (Equation 4.1) of the sensory neurons and interneurons. We also evaluated networks trained without noise in noisy environments and those trained with noise in noiseless environments.

4.3 Results

4.3.1 Chemotaxis Motifs

The chemotaxis motifs we found in our network solutions could be divided into two types. In the first type, which we refer to as the "differentiator motif", behavior depended only on the worm's network input history. The second type, which we refer to as the "bounce-and-trap motif", behavior depended only on the worm's current sensory input. Of the 50 solutions, we identified 21 differentiator motifs and 29 bounce-and-trap motifs. The optimized differentiator solutions had a 16% lower $E_{network}$ value[4] than the bounce-and-trap motifs.

Differentiator Motifs

We found two differentiator motifs: the "interneuron-differentiator" (Figure 4.3) and the "sensory-differentiator" (Figure 4.4). They both worked by crudely differentiating input $u(t)$ using a "fast", excitatory neuron and "slow", inhibitory neuron in parallel. Because both differentiators are functionally similar, we will refer to the patterns of excitatory and inhibitory connections in parallel as the "differentiator motif".

---

[4] Significant at 99.5% confidence using a two-tailed t-test.

Interneuron-Differentiator Motif The interneuron-differentiator (Figure 4.3) relied on a fast, excitatory pathway (Neuron 1 to Neuron 3) in parallel with a slow, inhibitory pathway through the interneuron (Neuron 1 to Neuron 2 to Neuron 3). The chemotaxis behavior of the network is shown in Figure 4.3(B) and the timecourse of neuronal activation is shown in Figure 4.3(C). The reduced speed of the slow, inhibitory pathway was due to the large time constant on Neuron 2 (τ_2). We noted that turning events (labeled $a-f$) were restricted to intervals in which concentration, $u(t)$, decreased rapidly. For example, at turning event a, the activation of Neuron 1 dropped well below that of Neuron 2. Thus, the contribution of the excitatory pathway to Neuron 3 dropped faster than the contribution of the inhibitory pathway to Neuron 3. This difference caused Neuron 3 to desaturate, resulting in a spike in P_{turn}. Conversely, turns were absent from intervals in which concentration dropped slowly. For example, between turning events a and b, the difference between the contribution of the excitatory and inhibitory pathway was insufficient to desaturate Neuron 3.

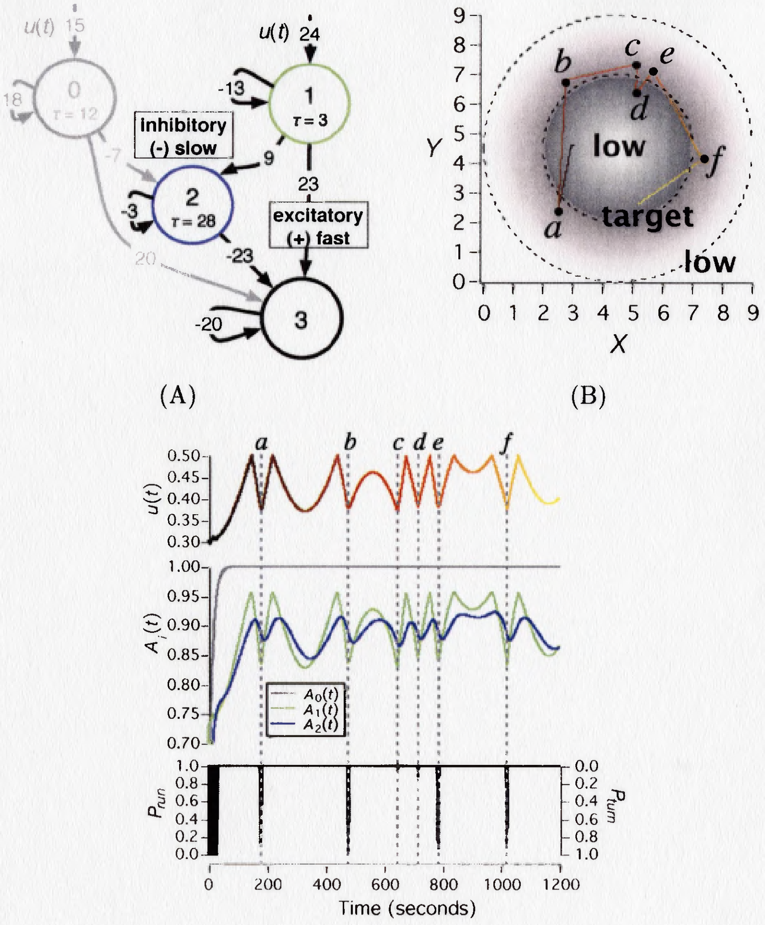
Overall, we found 12 network solutions with the interneuron-differentiator motif. Although interneuron-differentiator networks differed in details from each other, they functioned in essentially the same way. This motif has been identified in other neural network models [Fet93] as well as in a model *C. elegans* chemotaxis neural network [DCPSL04].

Sensory-Differentiator Motif The sensory-differentiator motif (Figure 4.4) relied on a fast, sensory neuron in an excitatory pathway (Neuron 0 to Neuron 3) in parallel with a slower, sensory neuron in an inhibitory pathway (Neuron 1 to Neuron 3). The chemotaxis behavior of the network is shown in Figure 4.4(B) and the timecourse of neuronal activation is shown in Figure 4.4(C). The reduced speed of Neuron 1 was due to its time constant (τ_1) being significantly greater than the time constant of Neuron 0 (τ_0). We noted that turning events (labeled $a-e$) were restricted to intervals in which concentration, $u(t)$, decreased rapidly. For example, at turning event a, the activation of Neuron 0 increased quickly relative to the decreased activation of Neuron 1. Thus, the contribution of the excitatory pathway to Neuron 3 dropped faster than the contribution of the inhibitory pathway to Neuron 3. This difference caused Neuron 3 to desaturate, resulting in a spike in P_{turn}. Conversely, turns were absent from intervals in which concentration dropped slowly. For example, between turning events d and e, the difference between the contribution of the excitatory and inhibitory pathway was insufficient to desaturate Neuron 3.

Overall, we found 9 network solutions with the sensory-differentiator motif. Although sensory-differentiator networks differed in details from each other, they functioned in essentially the same way.

Bounce-and-Trap Motif

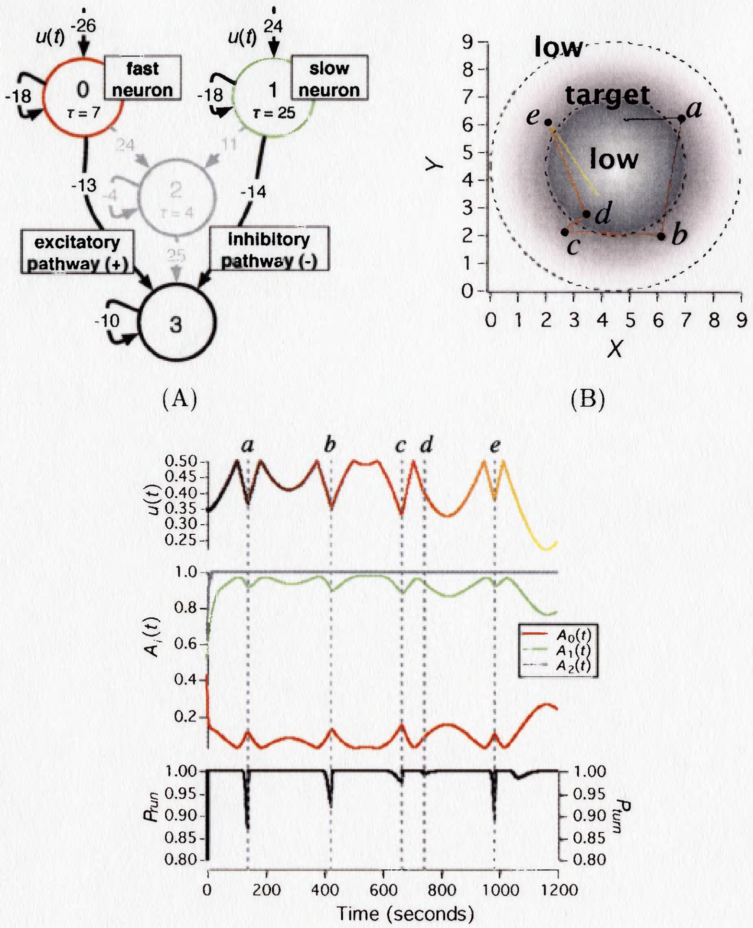
The *bounce-and-trap motif* (Figure 4.5) relied on a "bounce neuron" that was part of an excitatory pathway (Neuron 0 to Neuron 3) and a "trap neuron" that was part of an inhibitory pathway (Neuron 1 to Neuron 3). The chemotaxis behavior of the network is shown in Figure 4.5(B) and the timecourse of neuronal activation is shown in Figure 4.5(C). We noted that turning events (labeled $a - c$) were restricted to intervals in which the worm was at a low concentration, which we termed the "bounce threshold", or to intervals in which the worm was near the target concentration. In contrast to networks that used the differentiator motifs, networks that used the bounce-and-trap motif turned in response to absolute concentration levels rather than the rate of change in concentration. Dependence on absolute concentration was observed at both the bounce threshold and the target concentration. At the bounce threshold (turning events a and b), the activation of Neuron 0 decreased such that it was insufficient to keep Neuron 3 saturated (Figure 4.5(C), bottom), resulting in a spike in P_{turn}. Upon reaching the target concentration (turning event c), the trap neuron suddenly shifted from fully off to fully on. Here, both sensory neurons were activated, but the inhibitory connection from Neuron 1 was stronger than the excitatory connection from Neuron 0. This imbalance caused Neuron 3 to inactivate, resulting in sustained turning that tended to keep the animal at the target concentration because, over time, sustained turning reduces average speed to near zero. Although sustained turning is theoretically sufficient to produce trapping, it is a potentially weak trapping mechanism because of its susceptibility to occasional



(A) (B)

(C)

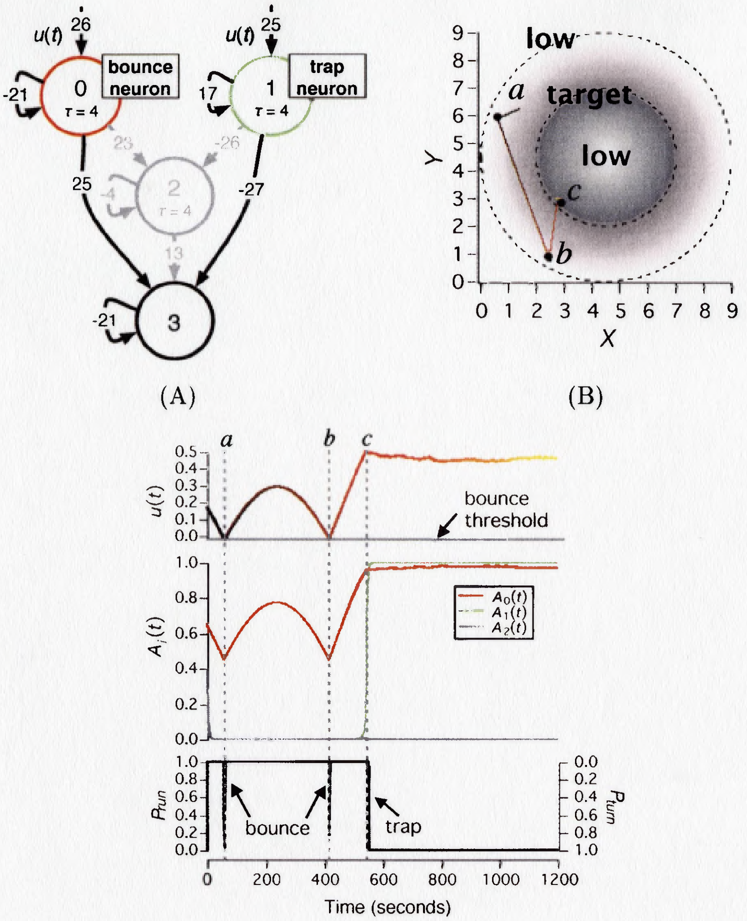
FIGURE 4.3. Interneuron-differentiator motif. (A) Motif diagram. Neuron 0 (gray) is inactive and not part of the motif; Neuron 1 (green) is part of a fast, excitatory pathway; Neuron 2 (blue) is part of a slow, inhibitory pathway; and Neuron 3 (black) is the output neuron. The bias of both active Neurons 1 and 2 is 3. (B) Track of model worm showing chemotaxis behavior. Track color represents time. Turning events are labeled a-f. (C) Sensory input ($u(t)$), neuronal activation ($A_i(t)$), and behavioral probability (P) versus time for the track shown in (B). Turning events $a - f$ are triggered when the activation of Neuron 3 drops, leading to an increase in P_{turn}. Activation of Neuron 3 drops because the contribution from the excitatory pathway ($1 \longrightarrow 3$) decreases more rapidly than the contribution from the inhibitory pathway ($1 \longrightarrow 2 \longrightarrow 3$).



(A) (B)

(C)

FIGURE 4.4. Sensory-differentiator-motif. (C) Motif diagram. Neuron 0 (red) is part of a fast, excitatory pathway; Neuron 1 (green) is part of a slow, inhibitory pathway; Neuron 2 (blue) is inactive and not part of this motif; and Neuron 3 (black) is the output neuron. Bias of both active Neurons 0 and 1 is 10. (B) Track of model worm showing chemotaxis behavior. Track color represents time. Turning events are labeled *a*-*e*. (C) Sensory input ($u(t)$), neuronal activation ($A_i(t)$), and behavioral probability (P) versus time for the track shown in (B). Turning events $a - e$ are triggered when the activation of Neuron 3 drops, leading to an increase in P_{turn}. Activation of Neuron 3 drops because the contribution from the excitatory pathway ($0 \rightarrow 3$) decreases more rapidly than the contribution from the inhibitory pathway ($1 \rightarrow 3$).



(A) (B)

(C)

FIGURE 4.5. Bounce-and-trap motif. (A) Motif diagram. Neuron 0 (red) is a bounce neuron, Neuron 1 (green) is a trap neuron, Neuron 2 (gray) is inactive and not part of this motif, and Neuron 3 (black) is the output neuron. Bias for the active Neurons 0 and 1 is 10 and -15, respectively. (B) Track of model worm showing chemotaxis behavior. Track color represents time. Turning events are labeled $a - c$. (C) Sensory input ($u(t)$), neuronal activation ($A_i(t)$), and behavioral probability (P) versus time for the track shown in (B). Turning events $a - c$ are triggered when the activation of Neuron 3 drops, leading to an increase in P_{turn}. Activation of Neuron 3 drops because sensory input falls to the “bounce threshold” so that the contribution from Neuron 0 fails to saturate Neuron 3 at a and b. At c, sensory input becomes high enough to saturate Neuron 1, which suppresses Neuron 3, trapping the worm at the target concentration.

short runs in the down-gradient direction, which would allow the worm to escape the target region. We found that trapping was enhanced by an asymmetry such that it is easier to enter the target region than it is to leave it. This asymmetry was revealed by an analysis of trap neuron equilibria (Figure 4.6). This analysis showed that for a worm moving up the gradient, the point at which the trap neuron shifts from off to on (trapping) is much closer to the target zone than is the point at which, for a worm moving down the gradient, the trap neuron shifts from on to off (escape). We found the trap neuron located at both the sensory neuron and interneuron positions. However, the bounce neuron was always located at one of the two sensory neurons.

4.3.2 Thermotaxis Motifs

We obtained 59 thermotaxis network solutions, which fell into six distinct thermotaxis motifs (Table 4.1). Each thermotaxis motif was a combination of two sub-motifs, a thermophilic sub-motif and a cryophilic sub-motif. Each of the sub-motifs was analogous to one of three previously found chemotaxis motifs: an interneuron-differentiator motif (Table 4.1, ID), a sensory-differentiator motif (Table 4.1, SD), or a bounce-and-trap motif (Table 4.1, BT). The synaptic connectivity of the thermophilic sub-motifs matched the connectivity of the chemotaxis sub-motifs, which is expected because both sub-motifs drive movement up the gradient. In contrast, the signs of the synapses in the cryophilic sub-motifs were reversed relative to the chemotaxis sub-motifs, which is expected because these sub-motifs drive movement down the gradient. The most common thermotaxis motifs were the dual-bounce-and-trap motif ($N = 31$) and the dual-interneuron-differentiator motif ($N = 19$). Other combinations of sub-motifs were rare, and three were never seen (zeros in Table 4.1).

Dual-Bounce-and-Trap Motif. The dual-bounce-and-trap thermotaxis motif was comprised of two bounce-and-trap sub-motifs, a thermophilic sub-motif (Figure 4.7(A)) and a cryophilic sub-motif (Figure 4.7(D)). The bounce-and-trap sub-motifs functioned in the same manner as the chemotaxis bounce-and-trap motif (Section 4.3.1), except that the trap neuron was always an interneuron instead of a sensory neuron.

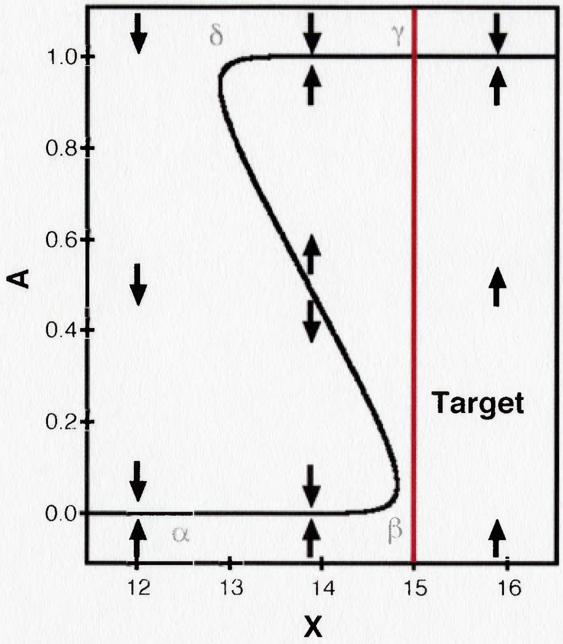
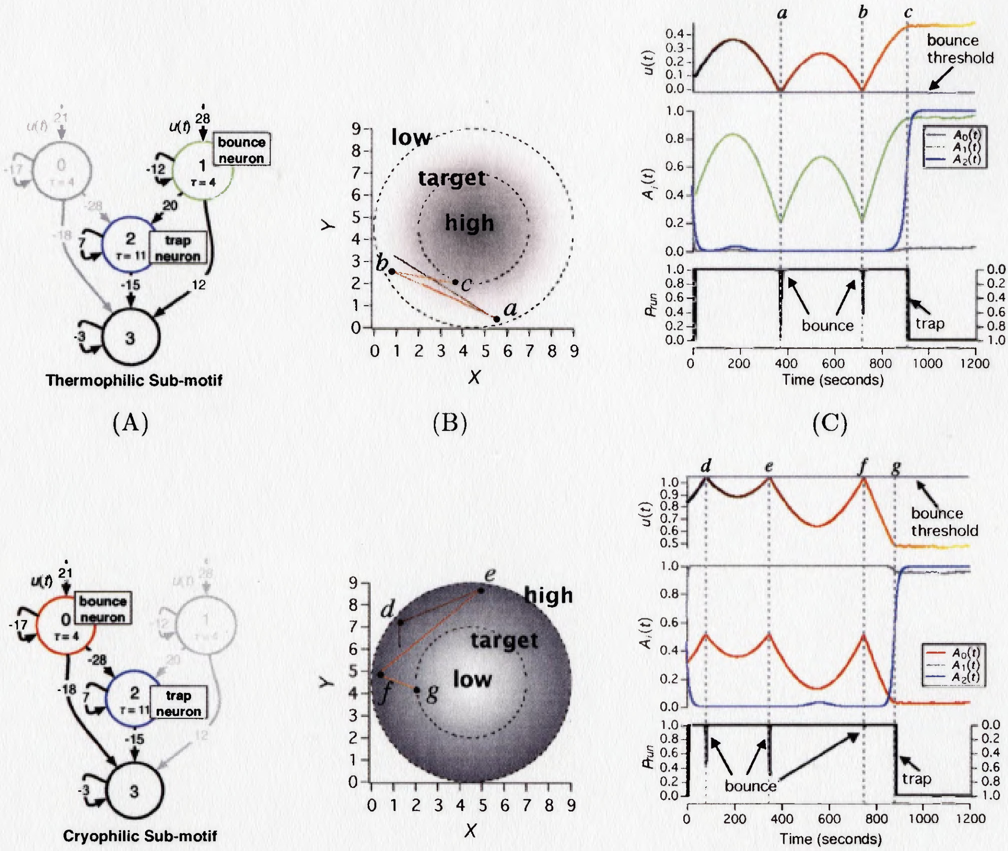


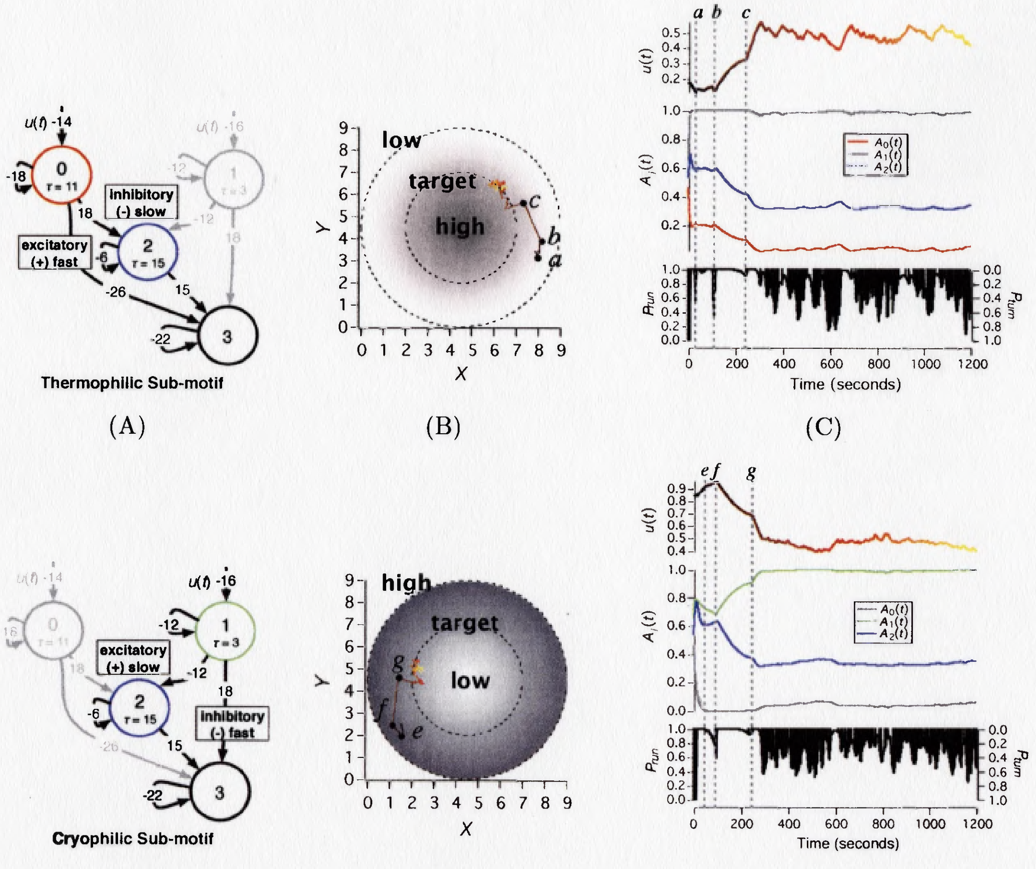
FIGURE 4.6. Origin of the trapping behavior asymmetry. Arrows indicate the sign of $dA(t)/dt$ (up arrows, positive) indicated by Equation 4.1 for the specified values of A and X. A is the instantaneous value of the trap neuron activation and X defines sensory input $u(t)$ according to the linear relationship shown in Figure 4.1(B). The black line shows the $A - X$ nullcline ($dA(t)/dt = 0$), which represents all possible equilibrium points of Equation 4.1. The target concentration of 0.5 is located at the position $X = 15$ (red line). A worm that starts at a low concentration and moves slowly up the gradient traces the nullcline from α to β. These equilibrium points are stable because deviations from the line produce restorative values of $dA(t)/dt$. The shift from off to on (trapping, see text) takes place just to the right of β, where $dA(t)/dt$ becomes positive, driving the trap neuron to the on state. A worm that starts near the target and moves down the gradient traces another stable region of the nullcline, from γ to δ. The shift from on to off (escape, see text) takes place just to the left of δ, where $dA(t)/dt$ becomes negative, driving the trap neuron into the off state. Trapping behavior is enhanced by the fact that the escape point (δ) is further from the target zone than the trap point (β). This asymmetry can be ascribed to the strong excitatory self-connection on the trap neuron, which produces a pronounced z-shaped fold in the nullcline along the X-axis; neurons with weak self-connections have no such fold [Bee95].



(A) (B) (C)

(D) (E) (F)

FIGURE 4.7. Dual-bounce-and-trap motif. The bias of Neurons 0, 1, and 2 are -13, 2, and -20, respectively. In both worm tracks ((B),(E)), track color represents time. In both traces ((C),(F)), sensory input is $u(t)$, neuronal activation is $A_i(t)$, and behavioral probability is P. (A) Heat-seeking circuit. Neuron 0 (gray) is inactive and not part of the motif, Neuron 1 (green) is a bounce neuron, Neuron 2 (blue) is a trap neuron, and Neuron 3 (black) is the output neuron. (B) Track of model worm showing heat-seeking behavior and corresponding trace (C). Turning events a and b are triggered by the bounce neuron, Neuron 1. Turning event c is triggered by the trap neuron, Neuron 2. (D) Cold-seeking circuit. Neuron 0 (red) is a bounce neuron, Neuron 1 (gray) is inactive and not part of this motif, Neuron 2 (blue) is a trap neuron, and Neuron 3 (black) is the output neuron. (E) Track of model worm showing cold-seeking behavior and corresponding trace (F). Turning events $d-f$ are triggered by the bounce neuron, Neuron 0. Turning event g is triggered by the trap neuron, Neuron 2.



(A) (B) (C)

(D) (E) (F)

FIGURE 4.8. Dual-interneuron-differentiator motif. The bias of Neurons 0, 1, and 2 are 4, 24, and 13, respectively. In both worm tracks ((B),(E)), track color represents time. In both traces ((C),(F)), sensory input is $u(t)$, neuronal activation is $A_i(t)$, and behavioral probability is P. (A) Heat-seeking circuit. Neuron 0 (red) is part of a fast, excitatory pathway; Neuron 1 (gray) is inactive and not part of this circuit; neuron 2 (blue) is part of a slow, inhibitory pathway; and Neuron 3 (black) is the output neuron. (B) Track of model worm showing heat-seeking behavior and corresponding trace (C). Turning events $a - d$ are triggered by the differentiator circuit. (D) Cold-seeking circuit. Neuron 0 (gray) is inactive and not part of this circuit; Neuron 1 (green) is part of an inhibitory, fast pathway; Neuron 2 (blue) is part of a slow, excitatory pathway; and Neuron 3 (black) is the output neuron. (E) Track of a model worm showing cold-seeking behavior and corresponding trace (F). Turning events $e - g$ by the differentiator circuit. At h, the worm reverts to the heat-seeking circuit (A).

		$T > T_{cult}$ (cryophilic)		
		BT	SD	ID
$T < T_{cult}$ (thermophilic)	BT	31	2	3
	SD	2	0	2
	ID	0	0	19

TABLE 4.1. The number of identified thermotaxis motifs from 59 thermotaxis network solutions. Each thermotaxis motif is composed of pairs of previously identified chemotaxis motifs. T and T_{cult} are the worm's current and cultivation (or target) temperatures, respectively. Rows represent chemotaxis motifs that increased T when $T < T_{cult}$ similar to the optimization in the chemotaxis problem. Columns represent motifs that decreased temperature when $T > T_{cult}$. The abbreviated motifs are: ID=Interneuron-Differentiator, SD=Sensory-Differentiator, BT=Bounce-and-Trap.

Dual-Interneuron-Differentiator Motif. The dual-interneuron-differentiator thermotaxis motif was comprised of two interneuron-differentiator sub-motifs, a thermophilic sub-motif (Figure 4.8(A)) and a cryophilic sub-motif (Figure 4.8(D)). The sub-motifs functioned in essentially the same manner as the chemotaxis interneuron-differentiator motif (Section 4.3.1).

Less Frequent Thermotaxis Motifs. Three thermotaxis motifs occurred less frequently than the dual-bounce-and-trap and dual-interneuron-differentiator motifs (Table 4.1). The first was the bounce-and-trap, sensory-differentiator thermotaxis motif. Here, the bounce-and-trap sub-motif functioned as either cryophilic or thermophilic with the sensory-differentiator exhibiting the opposite function. We obtained four such networks, two of each configuration. The second was the bounce-and-trap, interneuron-differentiator motif. This motif was composed of a thermophilic bounce-and-trap sub-motif and a cryophilic interneuron-differentiator sub-motif. We obtained three such networks. The third was the sensory-differentiator, interneuron-differentiator motif, which had a thermophilic sensory-differentiator sub-motif and a cryophilic interneuron-differentiator sub-motif. We obtained two such networks. There is no reason that motifs not found in Table 4.1 should not occur if given a sufficiently large sample of optimized networks, with the exception of the dual-sensory-differentiator motif.

Coordination of Cryophilic and Thermophilic Motifs. Thermotaxis involves migration to T_{cult} from above or below this temperature. The existence of mutations and neuronal ablations that produce cryophilic or thermophilic phenotypes indicates that the circuits for cryophilic and thermophilic migration are at least partly distinct. This distinction is the basis for the so-called Opposing Drives model of *C. elegans* thermotaxis [HR75, MO95]. According to this model, the cryophilic and thermophilic circuits are active on both sides of T_{cult}, but with an asymmetry such that below T_{cult}, the thermophilic circuit is dominant, whereas above T_{cult} the cryophilic circuit is dominant. At T_{cult}, however, the effects of the two circuits cancel, eliminating migration and keeping the animal at the correct temperature.

One of our thermotaxis motifs provides an *existence proof for an alternative model*. In particular, we found that in the dual-interneuron-differentiator networks, only the thermophilic circuit is active below T_{cult}, whereas only the cryophilic network is active above T_{cult}. Thus, the two drives represented by these circuits never act in opposition. Interestingly, in virtual ablation experiments in planar thermal gradients (Figure 4.9), we found that killing the cryophilic circuit's sensory neuron caused a thermophilic phenotype and killing the thermophilic circuit's sensory neuron cause a cryophilic phenotype. These results qualitatively reproduce effects of thermophilic and cryophilic mutations tested in real gradients [IIM06]. We conclude that the ability to produce distinct cryophilic and thermophilic phenotypes may support thermotaxis models in addition to the Opposing Drives model.

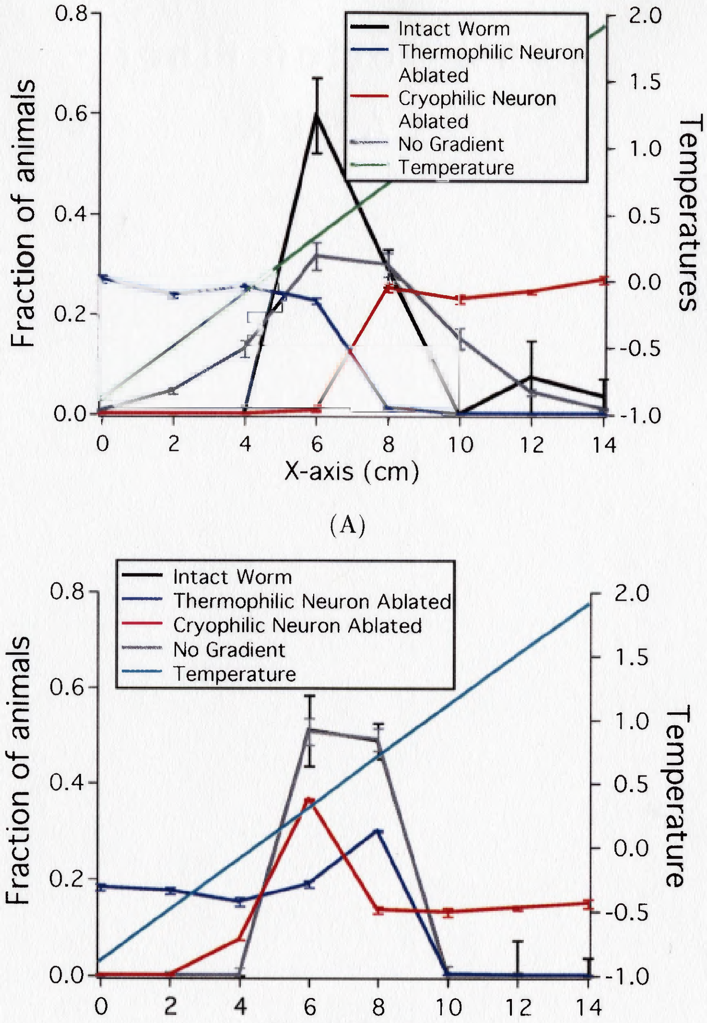
Sensory Organization of Thermotaxis. The thermotaxis motifs we found indicate that having a dedicated cryophilic and thermophilic sensory neuron is a *plausible* feature of the nervous system components that drive *C. elegans* thermotaxis. Most of the motifs we found utilized a separate, dedicated sensory neuron for the thermophilic circuit and the cryophilic circuit. The exception to this was the dual-interneuron-differentiator, which must share a sensory neuron with another motif by definition. This is consistent with an earlier proposal [MO95], which suggested that AFD was the dedicated thermophilic sensory neuron and an unknown neuron was the dedicated cryophilic sensory neuron.

4.4 Discussion

We were able to optimize neural networks to perform either chemotaxis or thermotaxis. We discovered a limited number of functionally distinct motifs from these networks. For chemotaxis, we identified and experimentally validated a previously identified differentiator motif—a fast, excitatory and a slow, inhibitory sensorimotor transformation in parallel [Fet93, DCPSL04]. We also discovered the previously unknown bounce-and-trap mechanism, which had not been seen before. The thermotaxis motifs we found combined thermophilic and cryophilic sub-motifs. The sub-motifs were analogous to the motifs found for chemotaxis.

Chemotaxis. The response of *C. elegans* to small chemoattractant steps of 40-50 mM NaCl and 50-40 mM NaCl [MTF$^+$05] qualitatively matches responses produced by the differentiator motifs we found for chemotaxis. Both our differentiator neural networks models and *C. elegans* respond to positive changes in concentration with transient decreases in turn probability and negative changes in concentration with transient increases in turn probability. Additionally, both *C. elegans* [PSML99] and our neural network differentiator motifs orient the worm independent of absolute attractant concentration. Although we have modeled the differentiator motif using neural networks, our motifs may instead explain processes at the intracellular level, similar in function to those observed in *E. coli* chemotaxis [FBB$^+$97, GS98].

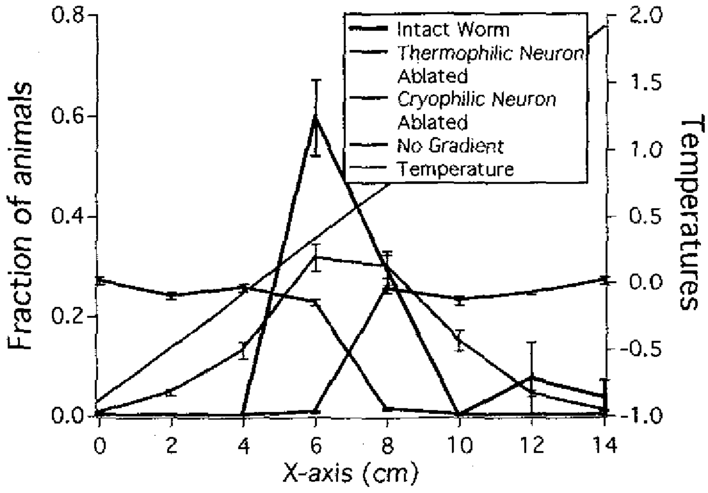
Thermotaxis. The thermotaxis motifs we found make several predictions about the neurophysiology that drives *C. elegans* thermotaxis. The first prediction is that two sensory neurons are necessary to exhibit thermotaxis. All of the thermotaxis motifs we found required use of both sensory neurons. Almost always, one neuron was dedicated to thermophilic movement and the other was dedicated to cryophilic movement. This is consistent with the cryophilic behavior exhibited when AFD is removed [MO95,



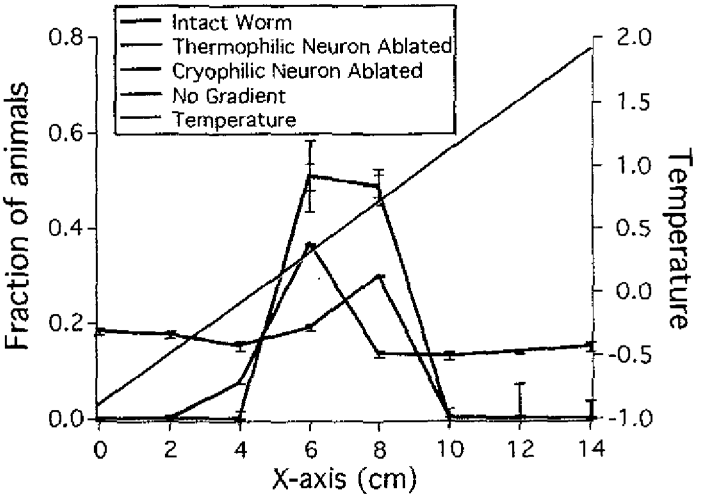
(A)

(B)

FIGURE 4.9. Distributions of simulated worms driven by thermotaxis motifs. (A) Distributions of simulated worms created by a dual-interneuron-differentiator motif. Elimination of the thermophilic sensory neuron yielded a distribution towards the cold region. Elimination of the cryophilic sensory neuron yielded a distribution towards the warm region. For an intact worm, the absence of a gradient yielded a flatter distribution. (B) Distributions of simulated worms created by a dual-bounce-and-trap motif accumulate at T_{cult}. Elimination of the thermophilic sensory neuron yielded a shallow distribution towards the cold region. Elimination of the cryophilic sensory neuron yielded a shallow distribution towards the warm region. The distribution in the absence of a gradient was identical to the intact worm because the worm constantly turned at T_{cult}. Distributions were of 400 worms taken after 60 minutes over nine trials. Error bars indicate standard error. Temperature was set to T_{cult} when no gradient was present. The height (Y direction) was 10 cm and isothermal. Worms reoriented randomly away from the gradient edge upon contact.



(A)



(B)

FIGURE 4.9. Distributions of simulated worms driven by thermotaxis motifs. (A) Distributions of simulated worms created by a dual-interneuron-differentiator motif. Elimination of the thermophilic sensory neuron yielded a distribution towards the cold region. Elimination of the cryophilic sensory neuron yielded a distribution towards the warm region. For an intact worm, the absence of a gradient yielded a flatter distribution. (B) Distributions of simulated worms created by a dual-bounce-and-trap motif accumulate at T_{cult}. Elimination of the thermophilic sensory neuron yielded a shallow distribution towards the cold region. Elimination of the cryophilic sensory neuron yielded a shallow distribution towards the warm region. The distribution in the absence of a gradient was identical to the intact worm because the worm constantly turned at T_{cult}. Distributions were of 400 worms taken after 60 minutes over nine trials. Error bars indicate standard error. Temperature was set to T_{cult} when no gradient was present. The height (Y direction) was 10 cm and isothermal. Worms reoriented randomly away from the gradient edge upon contact.

CCG$^+$06, IIM06], which suggests that another sensory neuron is able to transmit thermosensory information to a cryophilic circuit.

The second prediction is that sub-motifs share neurons (though not the sensory neuron) due to the observation of the networks we found. This observation was likely enhanced due to the limited size of the model. While this prediction is certainly not required for thermotaxis, the interneurons AIY, AIZ, and RIA may be involved in both thermophilic and cryophilic behaviors.

The third prediction is that thermotaxis motifs have functionally distinct sub-motifs dedicated to either thermophilic or cryophilic behavior. Although possible, this does not imply two separate sets of neurons, but instead suggests neurons that are functionally distinct. As opposing-circuit strategy based on laser and genetic ablation has been proposed[MO95, IIM06]. Additionally, asymmetry of behavior observed above and below the cultivation temperature [RS02, ZMFL03] also indicate that different functional circuits drive thermophilic and cryophilic behavior. However, at constant temperature, worms exhibited longer run durations at colder temperatures (Table 1, [RS02]), providing a sufficient difference in run duration to generate klinokinesis, or turn probability based on average local sensory stimulus, towards warmer temperatures [SBBP90]. The thermophilic klinokinesis model is independent of cultivation temperature, which is consistent with the opposing drive model.

The fourth prediction is that only one sub-motif is functional at any given time. This contradicts the opposing drive model, as well as the klinokinesis model. However, we believe that we did not observe the klinokinesis model, because it is not as aggressive at orienting the worm as the differentiator or bounce-and-trap motifs [Dus01].

Although the size of our model neural network was limited, the neural network model (Figure 4.2(A)) was chosen to facilitate analysis and produce the simple models of behavior. Neural network models with more than four neurons will likely provide identical optimization results through two mechanisms: (1) similar solutions may be provided in parallel (e.g., two trap neurons induce trapping behavior in response to the same sensory input) or (2) unnecessary neurons will be eliminated. We found this to be true for the optimized chemotaxis neural networks, which consistently eliminated an unnecessary neuron or produced parallel motifs (e.g., one bounce neuron and two trap neurons). Furthermore, we optimized 5-neuron neural networks to perform thermotaxis, which converged to a viable solution 13% (18/140) of the time, similar to the training results for the 4-neuron network model 11%(59/528). This suggests that the size of our network model was not responsible for reducing the number of motifs that we found.

CHAPTER 5

Neural Dynamic Clustering

The work in both Chapters 3 and 4 created a set of optimized neural networks. To make inferences into these results, we need to be able to extract patterns of functionality from these networks. The work in Chapters 3 and 4 provide examples of the limitations of current methods. Network analysis in Chapter 3 was done using both architectural (Section 2.4.1) and analytical (Section 2.4.2) analysis techniques. Due to the limitations of these techniques, we were unable to analyze networks with greater than three neurons. The network analysis in Chapter 4 relied upon qualitative signal analysis (Section 2.4.3). Although we were able to obtain satisfactory neural network results, a more quantitative analysis strategy is necessary, especially for more complex neural network models and behaviors.

The Neural Dynamic Clustering method and one of the case studies (Section 5.3) have been published in [DPSCL06]. The case studies in this work used neural networks optimized to perform chemotaxis from Chapters 3 and 4. Jon Pierce-Shimomura provided *C. elegans* experimental chemotaxis data and editing. Shawn Lockery contributed to the writing and direction of the case studies. John Conery provided some coding and a significant amount of writing, analysis, and overall direction. Nathan Dunn was the primary contributor, providing the bulk of the coding, writing, analysis, as well as developing the Neural Dynamic Clustering method.

5.1 Introduction

The Neural Dynamic Clustering method was developed to extract neural network motifs from a set of neural networks. A neural network motif is neuronal function observed in one or more neurons over multiple neural networks. This allows a more thorough understanding of the neural dynamics that drive network function as well as a way to compare neural networks in a meaningful way. Currently, no methods exist for this type of analysis.

Neural Dynamic Clustering generates neural network motifs by clustering models of the network solutions' neural dynamics. Motifs are extracted from neural dynamic model parameters generated from the neuronal responses of multiple network solutions.

Neural Dynamic Clustering is a generic framework that could be adapted to many different neural network models and optimization problems. It may be applied to any evolved graph-based system with multiple solutions and nodes that are recordable. For example, bandwidth through individual routers on the Internet could be analyzed by this method.

5.2 Method

Neural Dynamic Clustering is a general framework that allows extraction of neural network motifs from a set of neural networks. The method works by clustering the neurons into single-neuron motifs and combining the single-neuron motifs in larger motifs. The neurons are then clustered based on the parameters from their neural dynamic models. Any model of neural dynamics will work that effectively reproduces individual neuronal output given a reasonable range of external network input.

Here we provide an outline of the steps used to identify motifs. These are divided into three steps: (1) *create neural models*, (2) *cluster neural model into single-neuron motifs*, and (3) *combine motifs* into multi-neuron motifs.

5.2.1 Create Neural Models

In this step, system identification techniques [PN92, WK03, SEM03] are used to create block models that simulate neural dynamics. A block model consists of a linear combination of simpler models. Several different types of models may be used to relate network input to neuronal response. We used block models because they robustly modeled the nonlinear dynamics of the neuron, had a sufficient number of consistent model parameters, and were relatively simple.

There are two simple models often used to create block models, which we will use here: the linear transfer function, L, and the static nonlinear function, N. A linear transfer function is a linear differential equation, typically represented in the Laplace or Fourier domain, which describes a system's time-dependent response to stimulus. Conversely, the static nonlinear function has no time-dependence, and instantaneously evaluates stimulus. Any nonlinear function may be used for N. Although high order polynomial series, such as the Hermite polynomial, are a more common way to define N, we used a sigmoid function to more closely approximate the nonlinearity of our neurons. There are many standard block models for nonlinear system identification built with the L and N components such as the Wiener model (LN), the Hammerstein model (NL), the Sandwich model (LNL), and the cascade model (NLN) [WK03]. However, in both case studies we use the NL model, as it gives the most accurate approximation of neuronal response.

5.2.2 Cluster Neural Models into Single-neuron Motifs

In this step, the neural parameters from our model were clustered using k-means [Eve01] to generate single-neuron motifs. There were multiple parts to this step. First, neural networks were pruned by eliminating neurons that had no dynamic activity. Second, we determined significant parameters by examining their distribution. Third, we added network-normalized parameters to our model. Network-normalized parameters are neural parameters normalized within each network. Fourth, we normalized values not already network-normalized and then scaled all parameters so that the weight of each parameter during clustering could be explicitly set. Last, we used

R [EH06] to cluster (k-means) neurons by their NL model parameters.

The results of the clustering step were a set of motifs, each of which occurred in at least two networks, and their associated neurons. Each neuron is assigned to only a single motif. All of the neurons from an individual neural network may belong to the same motif. A motif A that contains neuron i of network x and neuron j of network y is designated as $A = \{x[i]; y[j]\}$. As an example, if we analyze a set of five neural networks with four neurons each, one possible motif is

$$A = \{1[0]; 1[2]; 2[1]; 3[2]; 3[3]; 5[1]\}$$

which contains neurons from networks 1, 2, 3, and 5, where the designated neuron for each network is in brackets.

5.2.3 Combine Motifs

From the single-neuron motifs, larger motifs are constructed by recursively combining the motifs using code written in Python [Pil04]. First, we collapsed the single-neuron motifs into larger clusters of single-neuron motifs by examining cluster centers. For example, if the parameters of the cluster centers for motifs A and B are similar, than we may combine the two by relabeling all of the neurons identified belonging to motif B as belonging to motif A, instead. Second, we collapsed the single-neuron motifs into multi-neuron motifs by first combining the single-neuron motifs into larger motifs by combining single-neuron motifs from within the same network and then combining the larger motifs that have similar combinations of single-neuron motifs. This process is repeated with the larger motifs, until an entire network can be constructed from a single motif. We illustrate this in the following example. Using the single-neuron motif A from our previous example and motif B defined as:

$$B = \{1[1]; 2[0]; 3[1]; 4[1]\}$$

the combined motifs are:

$$A/B = \{1[0,1]; 1[2,0]; 2[1,0]; 3[2,1]; 3[3,1]\}$$

$$A/A = \{1[0,2]; 3[2,3]\}$$

$$A/A/B = \{1[0,2,1]; 3[2,3,1]\}$$

In this notation, X/Y indicates a combined motif of the motifs X and Y, in order. Neuron order is preserved and network 3 appears once for each neuron that it matches in A/B. A minimum of two motifs must be included to combine a motif. $A/A/B$ is a combination of $A/\mathbf{A}$ and $\mathbf{A}/B$ across the bold-faced A's. The position of the neuron in its combined motif corresponds to its single-neuron motif designations. For example, in $1[0,2,1]$, $1[0]$ and $1[2]$ belong to motif A and $1[1]$ belongs to motif B.

5.3 Case Study: Neural Networks for *C. elegans* Chemotaxis

We used Neural Dynamic Clustering to reproduce the qualitative chemotaxis results of Chapter 4 that produced two fundamentally different neural network motifs, a bounce-and-trap motif and a differentiator motif. Chapter 4 used simulated annealing to construct model neural network solutions capable of computing *C. elegans* chemotaxis using *C. elegans*-like neurons. The result was 50 structurally unique neural networks. Previously, inferring common function from a set of network solutions was done by human observation of neural activity. However, Neural Dynamic Clustering was able to quantitatively identify neural network motifs.

5.3.1 Results

Using Neural Dynamic Clustering, we identified the three motifs identified in Section 4.3.1—two finite memory differentiator motifs and one zero memory bounce and trap motif found previously in Chapter 4. Additionally, we identically matched 48 of the 50 chemotaxis motifs. We describe the three specific steps of Neural Dynamic

Clustering used to identify the motifs.

Create Neural Models

We were able to successfully create models of neural dynamics for the neurons in all 50 neural networks. Most of the neural models were able to fit network input not used during optimization. Neural response to network input was modeled most accurately as the sequential nonlinear/linear (NL) block model shown in Figure 5.1(A). This was because the only non-zero elements of the second-order Wiener kernel (a series approximation method) that fit the neurons from the optimized neural networks, occurred along the diagonal [WK03]. Analysis of the numerical estimation of Equation 4.1 yielded a similar insight. The nonlinear block was a zero memory (steady-state or static) input/output function. Because our neural model was a sigmoid (Equation 4.1), we also used the sigmoid function of Figure 5.1(B) for our static nonlinearity instead of a polynomial series:

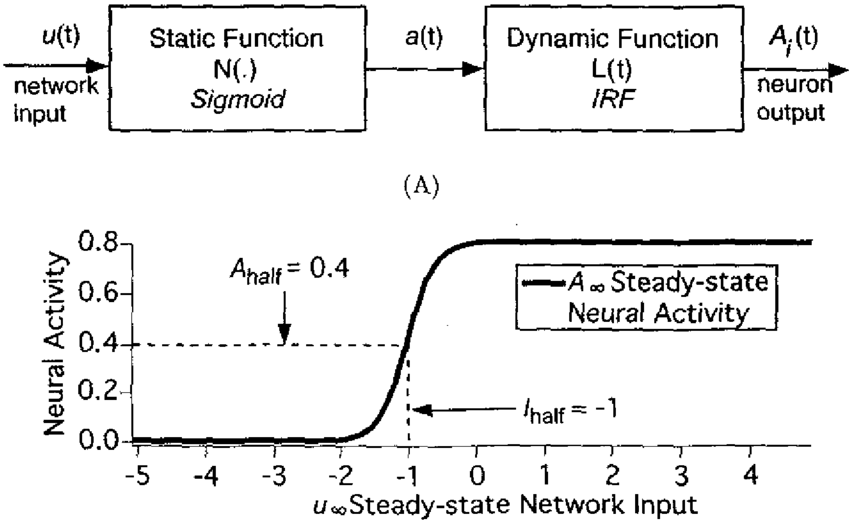
$$A_\infty = \frac{M}{1 + exp(-(\alpha(u_\infty - I_{half})))} + A_{offset} \tag{5.1}$$

where u_∞ is *steady-state network input*, A_∞ is *steady-state* neural output, M is the amplitude of the sigmoid, α is the slope of the sigmoid, I_{half} is the neural input half-saturation point, and A_{offset} represents a neural activity offset.

The linear portion of the NL block was a first-order impulse response function (IRF). The IRF function is fit to a decaying exponential with the assumption that the zero-order portion of the kernel is 0 (i.e., $a_0 = 0$):

$$Ampl(p) = a_1 exp(-r\ p) + a_0 \tag{5.2}$$

where p is lag in *seconds*, $Ampl(p)$ is the amplitude at lag p, r is rate in $seconds^{-1}$, a_1 is a linear scaling constant, and a_0 represents the mean, which is typically set to 0. Because scaling of amplitude is largely driven by the nonlinear portion of the block, we ignore a_1 for our analysis, and instead concentrate on r, which represents the neuron's speed. A larger r value indicates a neuron whose output activity $A_i(t)$



(A)

(B)

FIGURE 5.1. Neural dynamics are modeled as an NL block model (A), which consists of a zero memory, nonlinear sigmoid (B) followed by a finite memory, linear impulse response function (IRF). The sigmoid (B) is fit by Equation 5.1 with $M = 0.8$, $\alpha = 5$, $I_{half} = -1$, and $A_{offset} = 0$. A_{half} is the half-saturation output that corresponds to I_{half}.

responds more rapidly due to changes in neural network input $u(t)$. r was primarily set during optimization by adjusting the time-constant τ_i (Equation 4.1) and self-connections w_{ii} (Equation 4.2) for any neuron i.

Identification of parameters for each neuron's block model occurred over several steps using Matlab [HL00] code adapted from [WK03] and network input $u(t)$ versus individual neural output $A_i(t)$. First, the nonlinear block is fit to Equation 5.1 using steady-state network input u_∞. Next, a linear IRF is identified by applying Gaussian input (standard deviation of 0.5, mean of 0) to the network after being transformed through the identified sigmoid. The zero-order offset from the linear kernel is added to Equation 5.1 as A_{offset}. Finally, the linear IRF kernel is fit to Equation 5.2.

After each neuron has been fit to an NL model, it was generalized having it match unseen network input $u(t)$ to measured neural output $A_i(t)$. For the generalization, the unseen network input was generated from several different model worms performing chemotaxis.

For our particular model, the last neuron $A_3(t)$ is not analyzed because it is binary (Equation 4.3). Although we could have applied a filter or analyzed the input to the neuron, for simplicity we disregarded it from analysis.

Clustering Motifs

The first part of the clustering step was pruning. We found that 84% of our network solution pruned away a single neuron due to inactivity. In the remaining neural network solutions, motifs either existed in parallel, for example a bounce and two trap neurons, or the remaining neuron was dynamically active, but had no effect on network function. To determine whether a neuron had an effect, we saturated the neuron either on or off to eliminate its dynamic activity, and then observed the effect on the neural network. This method was chosen because there was no default activity level at which to set the neurons and it was a more robust test than fixing neurons at a defined neural activity level.

In the second part of the clustering step, we determined the parameters most relevant for clustering by examining the distribution of parameters. Figure 5.2 demon-

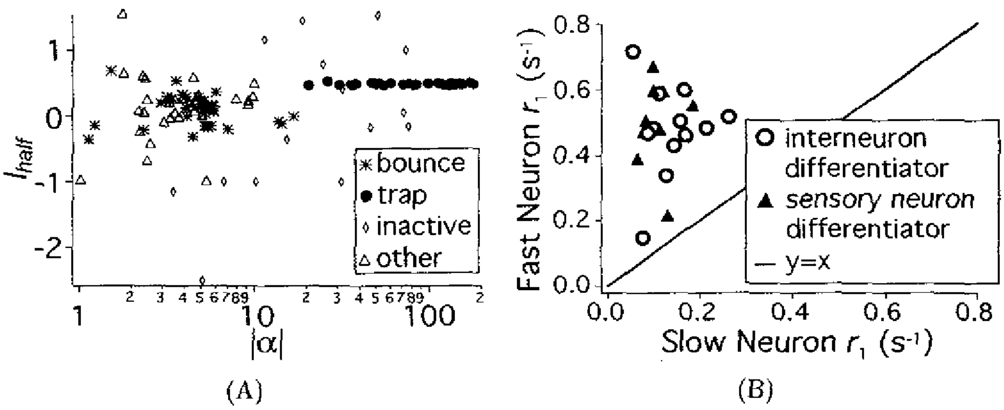


FIGURE 5.2. Distribution of (A) nonlinear sigmoid and (B) linear IRF parameters from the NL block model of Figure 5.1(A). Three neurons identified as inactive are not shown (A). In (B), r indicates the first order rate constants of Equation 5.2. Each point represents two neurons in a single neural network. Only neurons identified as either belonging to sensory or interneuron differentiators are shown.

strates the most relevant parameters of the NL model. These were α and I_{half} from Equation 5.1, and r from Equation 5.2. Trap neurons were tightly clustered around $I_{half} = 0.5$, with large slopes (high $|\alpha|$) while the bounce neurons were loosely clustered around $I_{half} = 0$, with a larger active range (small $|\alpha|$). r was greater for neurons labeled as fast than neurons labeled as slow.

In the third part, we added network-normalized values from our neural dynamic models to the values that will be clustered. A network-normalized value for an individual neuron is that value normalized between the maximum and minimum values of that parameter for the neuron's network. Network-normalized values are used to support observations such as the one demonstrated in Figure 5.2(B), which suggests that the network-normalized value of r, $net(r)$, evaluated in the context of its own network, is more relevant than r evaluated in the context of all neural network solutions.

In the fourth part, we normalized α and I_{half}, and scaled $net(r)$, so that each parameter would have equivalent influence during clustering. We primarily tried varying the scale of $net(r)$, but found that a scale near unity yielded the best results.

motif	α	I_{half}	$net(r)$
slow1	**-0.04280287**	4.0875735	**-0.5000000**
slow2	**-0.01047920**	0.6860025	**-0.4988943**
trap1	**2.57955471**	**1.5451524**	-0.5000000
fast1	**0.02256107**	-0.7802657	**0.3750000**
fast2	**-0.01442384**	2.1194284	**0.1698700**
trap2	**-1.38082003**	**1.5529448**	-0.4972258
fast3	**0.03362721**	0.4911198	**0.5000000**
trap3	**0.88749762**	**1.5303960**	-0.4820260

TABLE 5.1. Labeled cluster centers generated from normalized NL model parameters using k-means. Bold values indicate parameters used for naming the label, which are assigned according to the proximity of their centers. Each cluster center is named for a motif following the clustering step.

In the last part of this step, we clustered the parameters α, I_{half}, and $net(r)$ using k-means and eight random starting centers. Eight centers gave the most consistent clustering results over multiple trials, though five to nine centers also gave decent results. Cluster centers were used to generate labels (Table 5.1) for the single-neuron motifs (Table 5.2). The parameters most responsible for defining motif names in Table 5.1 are in bold. Both fast and slow neurons had small α in order to provide a sufficient range of network input where they were unsaturated and active. Fast neurons had higher $net(r)$ values, while the slow neurons had $net(r)$ near the minimum value of -0.5. Trap motifs had high absolute α. Additionally, the trap motifs had consistent biases of -1.54, which corresponded to a network input value -0.5 when denormalized. The effect of the high α and set bias resulted in the switch-like effect observed in Figures 4.5.

Combine Motifs

The motif combination step united single-neuron motifs into multi-neuron motifs using the method outlined in Section 5.2.3. First, we collapsed Table 5.2 by the motif designations of fast, slow, and trap from Table 5.1. These labels were applied based on the parameters of Table 5.1 in bold. For example, in Table 5.1, the first and second centers have $net(r)$ values near the minimum, which indicate slow neuronal response,

Motif	Network Set
slow1	1[2];40[2]
slow2	4[1];5[0];6[2];8[2];9[1];21[2];22[2];24[1];25[1];26[0];27[2] 29[1];36[2];39[2];43[2];44[0]
trap1	0[0];5[2];10[0];15[0];16[0];18[0];33[0];37[1];41[0];47[0]
trap2	2[0];3[2];11[1];12[1];19[2];23[1];28[1];30[0];31[1];31[2] 42[0];46[0];48[0]
trap3	13[2];17[1];20[0];30[2];32[1];34[1];38[1];38[2];45[0];49[1]
fast1	4[0];4[2];7[2];8[0];17[0];23[0];30[1];32[0];34[0];35[1] 36[1];38[0];40[0];44[1];48[1];49[0]
fast2	1[0];12[0];14[0];14[2];22[0];35[0];45[1]
fast3	0[1];2[1];3[0];5[1];6[1];7[0];9[0];10[1];11[0];13[1];14[1] 15[1];16[1];18[1];19[0];20[1];21[1];24[0];25[0];26[1];27[0] 28[0];29[0];31[0];33[1];37[0];39[1];41[1];42[1];43[1];46[1] 47[1]

TABLE 5.2. Single-neuron motifs generated from clusters of neural dynamic parameters. Motif labels were derived from the cluster centers in Table 5.1.

and small absolute α values, which indicate a sigmoid with a large linear range. As such, we apply the same label, slow, to both cluster centers and treat them as a single cluster for subsequent steps shown in Table 5.2 and 5.3. Unfortunately, unlike other steps, combining single-neuron motifs requires human intervention. Although a smaller number of cluster centers could have been used, this method seemed to give results that are more consistent because it allows clusters to ignore certain parameters during the clustering step. For fewer cluster centers, it would be unlikely that neurons in slow1 and slow2 would belong to the same motif since they had to have very different I_{half} values to initially fall into separate clusters. This is acceptable as neurons with a smaller (less steep) α value are likely to have less specific I_{half} values.

Next, we combined the single-neuron motifs from Table 5.2 into the multi-neuron motifs of Table 5.3. For example, the first row of Table 5.3, fast/slow, contains several networks, including 1 and 4. In Table 5.2, we see that neuron 0 of network 1 is part of motif fast2 and neuron 2 is part of motif slow1. These are combined into the multi-neuron fast/slow motif $1[0,2]$ as shown in the top row of Table 5.3. Similarly, for network 4, neurons 0 and 2 belong to fast1 and neuron 1 belongs to slow2. This yields the combinations 4[0, 1] and 4[0, 2] in fast/slow. Network 4 may have been further combined into a larger motif, such as fast/fast/slow, but no other fast/fast/slow multi-neuron motif occurred.

Of the networks identified, 96%, or all but two networks (networks 13 and 36), matched our qualitative observations. This was because 13[2] was labeled as trap instead of fast and 36[2] was labeled as slow instead of trap. Neurons labeled as fast may function as either bounce neurons in the bounce and trap motif or fast neurons in the differentiator motif. Neural networks that belonged to the fast/fast category were included as differentiators (7,14, and 35), because it was more likely that neurons labeled fast should have been labeled as slow than as trap.

We found (Table 5.3) one 3-neuron network, the fast/trap/trap, which combined the fast/trap and trap/trap motifs into a combined motif where two trap neurons acted in parallel with a single bounce neuron. Additional 3-neuron networks were combined, but in insufficient numbers (< 2) to form a motif.

Motif	**Network Set**
fast/slow (differentiator)	1[0,2];4[0,1];4[2,1];5[1,0];6[1,2];8[0,2];9[0,1] 21[1,2];22[0,2];24[0,1];25[0,1];26[1,0];27[0,2] 29[0,1];36[1,2];39[1,2];40[0,2];43[1,2];44[1,0]
fast/trap (bounce& trap)	0[1,0];2[1,0];3[0,2];5[1,2];10[1,0];11[0,1];12[0,1] 13[1,2];15[1,0];16[1,0];17[0,1];18[1,0];19[0,2] 20[1,0];23[0,1];28[0,1];30[1,0];30[1,2];31[0,1] 31[0,2];32[0,1];33[1,0];34[0,1];37[0,1];38[0,1] 38[0,2];41[1,0];42[1,0];45[1,0];46[1,0];47[1,0] 48[1,0];49[0,1]
trap/trap	30[0,2];31[1,2];38[1,2]
fast/fast	4[0,2];7[0,2];14[0,1];14[0,2];14[1,2];35[0,1]
fast/trap/trap	30[1,0,2];31[1,0,2];38[1,0,2]

TABLE 5.3. Multi-neuron motifs combined from single-neuron motifs of Table 5.2.

5.3.2 Summary

A drawback of this study was the small size of our neural network model. Optimizations with additional neurons could be performed to determine if models of greater complexity could be found. However, we believe that even with additional neurons, identical optimization solutions will result through two previously observed mechanisms. First, similar motif components may be provided in parallel as seen by the fast/trap/trap motif in Table 5.3, where two trap neurons simultaneously induce trapping behavior. Second, unnecessary neurons will have their dynamic activity eliminated, as was seen in 84% of the neural network solutions.

5.4 Case Study: Recurrent Neural Networks Reproduce *C. elegans* Pirouette Mechanism

This case study extracted neural network motifs from the optimized neural networks of Chapter 3. Simulated annealing was used to construct model neural network solutions capable of reproducing the pirouette mechanism for *C. elegans* chemotaxis [PSML99], which required networks to differentiate sensory stimulus. We analyzed 86 neural networks that varied from three to five neurons with a total of 302 neurons. Networks originally had eight interneurons, but inactive neurons were removed (pruned), reducing the network to the minimal set of necessary neurons.

This case study was performed for several reasons. First, this method allows us to analyze networks we were previously unable to analyze. Second, this case study provides further evidence that Neural Dynamic Clustering works. Last, it shows that Neural Dynamic Clustering performs well on highly recurrent neural networks, which was not shown in the previous case study.

Previous analysis used structural and analytical techniques. For structural techniques, we used parallel-axis plots (Figure 2.10) and Hinton diagrams (Figure 2.9). For analytical analysis, we analyzed linear, recurrent pairs of neurons (Section 2.4.2). These methods limited our analysis to only three-neuron networks. The Neural Dynamic Clustering method was able to verify the predicted function of previously identified three-neuron neural networks and identify the function of not yet analyzed four and five-neuron networks. To do this, the analysis had to answer two questions. First, do we observe the differentiator function predicted for the three-neuron differentiators of [DCPSL04] (Chapter 3)? Second, what is the function of the four and five-neuron networks we were previously unable to analyze [DCL03, DCL04, DCPSL04]?

5.4.1 Results

The methodology we used in this case study was the same as in Section 5.3, but with a few key differences. First, we analyzed networks using the output neuron. The continuous activity of the output neuron was analyzed without the threshold being applied (Figure 3.2(B)). Second, we explicitly analyzed neural networks of different sizes. Third, due to the constrained nature of the optimization problem, the only static nonlinearity necessary was the mean of neuronal activity, a_0 in Equation 5.2. As such, we generated motifs using only the relative decay rate $rel(r)$ from Equation 5.2, looking for two k-means clusters. The motifs we found are shown in Table 5.4. For brevity, we only show motifs three neurons or larger and only the largest motif for each network. For example, network 1 belongs to both the fast/slow/slow/slow motif (1[0,1,2,3]) and the fast/slow/slow/slow/slow motif (1[0,1,2,3,4]); however, it was only shown in the larger of the two motifs.

In answer to the first question we proposed, whether or not we will observe a previously predicted three-neuron differentiator, we attempted to find differentiators within the three-neuron networks by looking for a fast sensory neuron in parallel with a slow interneuron. Additionally, the fast sensory neuron should also be part of an excitatory pathway to the output neuron, while the slow interneuron should be part of an inhibitory pathway to the output neuron. This occurred for all 50 three-neuron networks, which previous findings [DCPSL04].

To answer the second question we proposed, to determine the function of the four and five-neuron networks, we hypothesized that, as part of a differentiator circuit, the sensory neuron should be fast and that all remaining interneurons should be slow. Additionally, we again suggest that the pathway from the sensory neuron to the

Motif	Network Set
fast/ slow/ slow	2[0,1,2]; 11[0,1,2]; 15[0,1,2]; 18[0,1,2]; 21[0,1,2]; 22[0,1,2]; 24[0,1,2]; 28[0,1,2]; 34[0,1,2]; 35[0,1,2]; 36[0,1,2]; 37[0,1,2]; 42[0,1,2]; 46[0,1,2]; 49[0,1,2]; 50[0,1,2]; 55[0,1,2]; 56[0,1,2]; 57[0,1,2]; 58[0,1,2]; 64[0,1,2]; 66[0,1,2]; 71[0,1,2]; 72[0,1,2]; 76[0,1,2]; 78[0,1,2]; 80[0,1,2]; 83[0,1,2]; 85[0,1,2]; 87[0,1,2]; 90[0,1,2]; 92[0,1,2]; 94[0,1,2]; 97[0,1,2]; 99[0,1,2]; 101[0,1,2]; 103[0,1,2]; 108[0,1,2]; 117[0,1,2]; 118[0,1,2]; 119[0,1,2]; 120[0,1,2]; 127[0,1,2]; 129[0,1,2]; 132[0,1,2]; 134[0,1,2]; 136[0,1,2]; 140[0,1,2]; 145[0,1,2]; 147[0,1,2]
fast/ slow/ slow/ slow	8[0,1,2,3]; 14[0,1,2,3]; 17[0,1,2,3]; 26[0,1,2,3]; 31[0,1,2,3]; 38[0,1,2,3]; 39[0,1,2,3]; 40[0,1,2,3]; 44[0,1,2,3]; 48[0,1,2,3]; 51[0,1,2,3]; 60[0,1,2,3]; 62[0,1,2,3]; 65[0,1,2,3]; 68[0,1,2,3]; 73[0,1,2,3]; 77[0,1,2,3]; 81[0,1,2,3]; 82[0,1,2,3]; 86[0,1,2,3]; 89[0,1,2,3]; 115[0,1,2,3]; 125[0,1,2,3]; 126[0,1,2,3]; 135[0,1,2,3]; 138[0,1,2,3]; 146[0,1,2,3]; 148[0,1,2,3]
fast/ slow/ slow/ slow/ slow	1[0,1,2,3,4]; 3[0,1,2,3,4]; 6[0,1,2,3,4]; 47[0,1,2,3,4]; 111[0,1,2,3,4]; 124[0,1,2,3,4]; 128[0,1,2,3,4]; 130[0,1,2,3,4]

TABLE 5.4. Motifs from recurrent networks optimized to exhibit the pirouette mechanism. Only motifs three neurons are greater are shown. Only the largest motif of each network is shown.

output neuron should be excitatory and the pathway through the slow interneurons should be inhibitory. This would suggest that the slow interneurons functioned in parallel with each other. Of the 36 four and five-neuron networks, the sensory neuron was always labeled as fast and the direct connection from the sensory neuron to the output neuron was always excitatory. All of the interneurons and the output neuron belonged to the slow motif. Every network examined had at least one slow interneuron that was part of an inhibitory pathway from the sensory to the output neuron. Additionally, in greater than half of the networks, every slow interneuron was part of an inhibitory pathway, indicating that one or more parallel inhibitory pathways existed.

Part of the reason that the sensory neuron was labeled as fast, was because it reacted to network stimulus without delay, whereas all other neurons were exposed to network stimulus one time-step later. This largely accounted for why the decay constant, r, on the sensory neuron was, on average, more than 70 times greater than on the interneurons. The network also explicitly optimized the sensory neuron for greater speed. By hand-wiring one of the optimized three-neuron networks to be

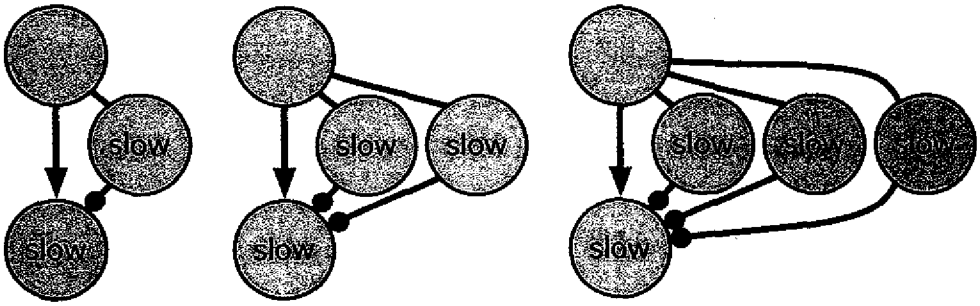


FIGURE 5.3. Motifs that reproduced the pirouette mechanism from Table 5.4. Arrows represent excitatory connections. Filled circles represent inhibitory connections. The motifs are composed of a fast sensory neuron directly connected by an excitatory connection to the output neuron at bottom, in parallel with one or more inhibitory pathways through a slow interneuron.

slower, we were able to easily approach an r ratio of 1.7 between the sensory neuron and interneuron. This suggests that the network explicitly optimized for a faster response on the sensory neuron versus the interneuron.

To more explicitly determine if parameters other than delay were relevant, we eliminated delay from the analysis. This was done by assuming that a_1 was always positive and the maximum correlated amplitude was always 0 (shifting Equation 5.2 to the left). In this study, the speed of the output neuron was noticeably faster than the interneurons. As such, the output neuron was excluded from analysis to make clustering easier. Two additional networks were also removed, 66 and 83, because they were difficult to fit. After we generated the neuronal models, we saw that the sensory neuron had the highest relative amplitude $rel(a_1)$ (a_1 from Equation 5.2) and the highest $rel(r)$ relative to the interneurons. These criteria generated motifs without output neurons that were identical to those of Table 5.4, with the exception of two networks: 2 and 56. This matched our previous motifs and showed that the sensory neuron responded both faster and with greater amplitude than the interneurons, which is consistent with a differentiator motif.

We were able to quickly extract neural network motifs from neuronal function. This analysis took only two days and yielded significant results compared with previous analysis, which took several weeks. Additionally, we were able to identify function that would have been available only by qualitative analysis.

5.5 Conclusion

In Chapter 3, analysis relied on analytical and structural methods to extract motifs from a set of neural networks. While it was adequate for much of the analysis, we were unable to characterize neural networks larger than three neurons. In Chapter 4, we relied on qualitative observation of neural activity (signal analysis) of each network during behavior to extract motifs.

Thee Neural Dynamic Clustering method presented in this chapter was able to quantitatively identify motifs for chemotaxis that were previously identified in both Chapters 3 and 4. It had the advantage that it was able to identify motifs from a set of neural networks using a generic response model to represent the function of individual neurons. Additionally, we were able to locate and define distinct functional motifs as well as describe their function.

The disadvantage of Neural Dynamic Clustering is its reliance on human intervention and inference during the parameter clustering step. Two additional problems with our case studies were the small number and size of the networks. However, the number of networks analyzed provided sufficient resolution to demonstrate convergence upon a small set of motifs. Testing on larger recurrent neural networks with uniform neural activity function would provide greater complexity and present an opportunity for multiple motifs to emerge within a single network.

CHAPTER 6

Related Work

Here, we present work related to the contributions of this thesis. For Chapters 3 and 4, we discuss previous models of chemotaxis and thermotaxis in *C. elegans.* For Chapter 5, we describe other neural network analysis strategies.

6.1 *C. elegans* Computational Models

Several neural network level models of *C. elegans* chemotaxis and thermotaxis behavior have been proposed. Relevant sensory neurons [BH91, MO95] and motor command neurons [ZBM+99] have been identified and anatomical connectivity of the entire animal is known [WSTB86]. However, specific interneurons that contribute to behavior, as well as the specific function of individual neurons are unknown. Because of this, a model of the nervous system is required.

As with the model of any system, stimulus/response pairs and internal measurement of the system define the system model. Stimulus is often just sensory input, such as chemical concentration or temperature. Response, however, is a metric corresponding to behavioral response. For *C. elegans* chemotaxis, the behavior model is the pirouette mechanism [PSML99] (Section 2.1.1). Possible metrics for this behavior model would be turn frequency or turn probability. For thermotaxis, two behavior models (Section 2.1.2) have been observed. In the first behavior model, when the worm is near the cultivation temperature it tracks isotherms (areas of identical temperature) using head movements, making small corrective changes to its orientation [MO95]. In the second behavior model, the worm is a significant distance away from its cultivation temperature and uses a model similar to the pirouette mechanism [ZMFL03] to orient towards its cultivation temperature. The latter behavior model is the model for thermotaxis explored in Chapter 4. Imaging technology discussed in Section 2.2.2 provides the most accurate measurement of neural activity during behavior [KMMM04, FL05, HAK$^+$05]. However, even with imaging technology, modeling is important for three reasons. First, an entire system may be simulated and observed simultaneously. Second, it describes all of the internal dynamics that drive behavior. Third, a model provides testable predictions.

6.1.1 Chemotaxis

Generation of direct anatomical predictions for the chemotaxis circuit has been difficult due to the number of sensory neurons that respond to individual chemoattractants [Dus80, BH91] and the unknown activation of the sensory neurons to stimulus. Chapter 3 [DCL04, DCPSL04] presented the first recurrent, nonlinear neural network model that implemented the pirouette mechanism of chemotaxis. Prior to the existence of the pirouette model [PSML99], a head-turning model was the dominant model for *C. elegans* chemotaxis. As such, previous *C. elegans* chemotaxis models employed recurrent, linear neural network models that assumed the head-turning model of *C. elegans* chemotaxis [MFL98]. However, Chapter 3 optimized neural network models to exhibit the pirouette rule, instead. The neural network motif we observed was a differentiator motif with inhibitory self-connections that regulate neuronal response and inhibitory, recurrent connections to regulate network stability.

The model in Chapter 4 was a simpler, feed-forward neural network with two sensory neurons instead of one. As opposed to Chapter 3, this model allowed any behavioral model that reliably reproduced chemotaxis. This assumption produced an additional motif that relied on absolute chemoattractant concentration as opposed to a differentiator circuit. However, evidence of a differentiator circuit has been further supported by chemotaxis step assays [MTF$^+$05]. Conversely, the bounce-and-trap motif may describe other behaviors, such as aversion to noxious stimuli [TB04] or the tap withdrawal response [TB04].

6.1.2 Thermotaxis

Using laser-ablation, a circuit had been proposed for *C. elegans* thermotaxis that consists of a pair of sensory neuron (AFD and an unknown neuron), and several interneurons [MO95]. Additionally, specific interneurons have been implicated in heat-seeking (AIY) and cold-seeking (AIZ) behavior. Assuming no behavioral restrictions, Chapter 4 modeled thermotaxis during instances when the worm was significantly above or below its cultivation temperature. Network models required two-circuit neural network model that requires a separate sensory input for each circuit, one for increasing the worm's temperature and one for decreasing the worm's temperature. Each circuit employed one of the chemotaxis motifs also discovered in Chapter 4. However, previous work [RS02, ZMFL03] demonstrated that the worm relied on a differentiator mechanism at both regions above and below the cultivation temperature. Chapter 4 validates previous evidence of the dual-functioning circuit [MO95, RS02, SSM03], including Ca^{2+} response during behavior [KMMM04].

The dominant neurophysiological model of *C. elegans* thermotaxis is the opposing drive model [MO95, IIM06], which suggests opposing cryophilic and thermophilic circuits that are in equilibrium at the cultivation temperature. Above the cultivation temperature, the cryophilic circuit dominates behavior, but below the cultivation temperature, the thermophilic circuit dominates behavior. Based on laser and genetic ablations, it is believed that AIY and AFD belong to the thermophilic circuit while AIZ and an unknown sensory neuron belong to the cryophilic circuit [MO95]. This is because deficits in AIY and AFD lead to cryophilic behavior and deficits in AIZ lead to thermophilic behavior.

By measuring turning rate relative to sinusoidal temperature changes, a switch model was proposed whereby the activation of AFD above the cultivation temperature [KMMM04] activates the cryophilic circuit (AIZ) and suppresses the thermophilic circuit and below the cultivation temperature, AFD is inactive and fails to suppress the thermophilic circuit (AIY) or activate the cryophilic one [CCG$^+$06]. Another model that used both constant and linearly changing temperature only found a mechanism for cryophilic behavior [RS02]. However, extended run duration at lower constant temperatures suggests a thermophilic response involving klinokinesis with temperature averages [SBBP90] that was not suggested in the paper. Response to temperature steps suggests a pirouette mechanism for both the upstep and downstep responses [ZMFL03]. However, steps were made from the cultivation temperature, which may also reflect unknown responses associated with isothermal tracking.

Measurement of AFD Ca^{2+}[KMMM04] and synaptic release [SSM03] relative to temperature has resulted in several paradoxes. Ca^{2+} activity only increased with increasing temperature for temperatures greater than the cultivation temperature. This creates a paradox, as AFD is active in a region where cryophilic drive is dominant, but a deficit in AFD creates cryophilic behavior. Also, hyperactivation of AFD creates a thermophilic bias [KIKM02]. Furthermore, synaptic release is only recorded at temperatures other than the cultivation temperature. However, the temporal resolution of the Ca^{2+} response was much higher (0.2 seconds) than that of the measured synaptic release activity ($>$ 1 minute).

The work in Chapter 4 supports the hypothesis of a dual-circuit with dedicated sensory neurons and shared neurons. It largely supports previous models that suggest opposing drives for thermophilic and cryophilic behavior.

6.2 Analysis of Neural Networks

The work of Chapters 3 and 4 produced multiple structurally unique neural network solutions with equivalent performance that exhibit a small set of qualitatively different behaviors. This effect will occur in any under-constrained optimization problem, which defines most neural network optimization problems. To understand how our neural networks solve the problems for which they were optimized, neural network motifs must be extracted from the generated sets of neural networks. A neural network motif is neuronal function involving one or more neurons that occurs over multiple neural networks.

Section 2.4 describes three different network analysis strategies: analytical analysis, structural analysis, and signal analysis. Analytical analysis was unable to describe network dynamics because of the difficulty of mathematically expressing the dynamics of interconnected, nonlinear networks in the transient domain. Architectural analysis assumes a direct correlation between connectivity properties and functional importance that may be invalid. Additionally, pure architectural analysis makes few predictions about the dynamic behavior of individual neurons. Furthermore, in biological systems, dynamic behavior is most often measured as neural activity. As such, the focus of our methods is signal analysis. Signal analysis provides the greatest insight into inter- and intraneuronal transient behavior, but has only been applied to model global properties of individual neural networks and has yet to be adapted to extract motifs of functional neurons (neural network motifs) from multiple neural networks. We developed Neural Dynamic Clustering, which used existing signal analysis techniques, but applied them to individual neurons over multiple networks.

Here, we describe methods closely related to the extraction of neural network motifs. These include methods that extract patterns from a large number of networks as well as methods that attempt to quantify the function of a single neural network.

6.2.1 Analytical Analysis

Analytical analysis methods attempt to determine how neural networks function based solely on solutions to the equations that define neural network function. Note that the solutions for almost all of the neural networks described in Section 2.3 are not tractable and as such are simulated using numerical estimation techniques such as Euler's method or Runga-Kutta. An advantage of this type of analysis is that properties of the neural network related to equilibria and stability may be derived exactly from the parameters of the network. While this is desirable, it does not provide insight into the transient properties or neuronal function that are most relevant to the work in Chapters 3 and 4. Instead, analytical analysis methods focus on steady-state and equilibrium solutions.

Properties of recurrent neural networks with from one to three neurons have been characterized using dynamical systems theory for continuous time [Bee95] and discrete systems [Pas02]. This work demonstrates the difficulty of generating general results for networks with greater than two neurons. Within these small networks, an abundance of attractors were found. These attractors were stationary (stable), oscillating, or chaotic. This analysis suggests likely regions of parameter space to focus optimization, depending on the desired behavior. Although this type of analysis is only possible for networks with three or fewer neurons, it was suggested that larger neural networks could be decomposed into smaller networks for analysis [Bee95]. While these methods are helpful in understanding the individual behavior of neurons, for example the trap neuron in Figure 4.6, it offers little insight into transient dynamics.

Recurrent neural networks have several interesting properties related to equilibrium and steady-state solutions [SMR96, ML02]. Conditions such as symmetrical recurrence (Hopfield networks) provide further simplifications, but are not necessary to analyze networks [MF89]. This type of analysis allows analysis of arbitrarily sized networks. These results rely on Lyapunov stability calculations for nonlinear systems. The premise of Lyapunov stability is that if a system with initial neuronal activity $\vec{A}$ remains near $\vec{A}$, then the system is Lyapunov stable. Furthermore, if all points near $\vec{A}$ end up at $\vec{A}$, then $\vec{A}$ is asymptotically stable. If the Lyapunov function is negative, the system is stable, otherwise, the system is unstable. Energy functions for neural networks [Hop82] are used as Lyapunov function. The energy function depends upon the assumptions of the neural network it represents. Typically, the energy function contains the interconnected terms of the network and an integral term that involves the nonlinearities.

This analysis led to several general conclusions. The energy function of a variety of neural networks was then shown to decrease monotonically along nonequilibrium points, reaching equilibrium as $t \to \infty$ [ML02]. It was further shown that there are only a finite number of equilibrium points, which are exponentially proportional to the number of neurons. This analysis allows determination of stable attractors for a recurrent neural network, as well as synthesis of networks with known attractors [TH86]. However, few properties of transient function are revealed using such methods.

6.2.2 Architectural Analysis

Architectural analysis methods are methods that directly analyze the structural properties of neural networks. Most often, these are connection weights between neurons.

Visual neural network analysis methods graphically represent properties of network structure. An advantage of visual methods is that they allow a quick, qualitative analysis of a large set of parameters, such as weights of multiple neural networks. A disadvantage is that they fail to provide insight into network function.

The purpose of visualization analysis methods is to provide an interactive environment that allows users to analyze network parameters, a high dimensional set of data. One such method is the parallel-axis plot, which may be used to analyze a number of high-dimensional data instances at once. Analysis with this type of tool allows users to cluster and prune from visual inspections [FWR99]. The parallel-axis plot has also been applied to visualize the response of spiking neurons similar to the signal analysis methods [SMC05] we discussed.

Properties of global connectivity may be statistically analyzed to characterize neural network structure. An advantage of statistical methods is that they provide a quick, quantitative measure of a network properties. A disadvantage of statistical methods is that the size of the network analyzed must be sufficiently large. Furthermore, the properties they yield give little insight into the transient dynamics of the system.

Connectivity in any type of network may be characterized according to the distribution of connectivity between nodes [DM03]. This gives an indication of how the neural network may have evolved. This type of analysis has been applied to a variety of biological networks including *C. elegans* [SW98] and metabolic interactions [WF01]. Real-world networks often have short average path lengths between neurons and are highly clustered (nearest neighbors are likely to be neighbors). These methods provide structural insight to network evolution, but provide little insight into function or behavior.

To search for connectivity patterns, which we call connection motifs, network connectivity may be compared to connectivity from randomly generated networks with identical global properties. Thus, overrepresented patterns of connectivity are connectivity motifs within a network. This type of analysis has been done within *C. elegans* [RAC04], the human brain [SK04], and a variety of other processes [MSI$^+$02]. It has been effective for finding networks with less than four nodes and with directed connectivity. Although comparison of networks to random networks allows for analysis of smaller networks, the generated connectivity motif yields little insight into network function. Furthermore, the connectivity motifs that are dedicated currently report no particular weight or sign values.

Another method that characterizes neural network function is ablation of network structure concurrent with measured response. A difficulty of ablation analysis, however, is assigning importance to an ablated connection or neuron. The non-specific effect of structure may lead to false predictions. As an example, a single ablation of a redundant connection may imply that the connection had contributed no overall function to the network. Two approaches to this are functional contribution analysis [ASMR03] and multi-lesion Shapley value analysis [KSH$^+$04]. The multi-lesion Shapley value analysis uses game theory [Sha53] to assess significant network structure. Lesions are analogous to the ablations referred to in experimental work performed on the *C. elegans* nervous system using lasers and genetics [MO95, BH91, GHB05, CCG$^+$06]. In other words during a lesion, the contribution of a neuron or connection is removed. This method statistically determines whether the contribution of a network structure leads to a greater overall network fitness for different lesions, including multiple lesions. As such, the multi-lesion Shapley method has been used to quantify the contribution of sensory neurons in *C. elegans* chemotaxis [KKM$^+$05].

Ablation response methods are able to determine the functional importance of neurons and connections within a neural network. However, there are several disadvantages to this method for extracting neural network motifs. First, ablation may have several non-specific effects even if multiple lesions are considered. Second, they provide no insight into transient function. Third, they provide little advantage for analysis of multiple neural networks. Fourth, a combinatorial explosion of experiments may be necessary to provide adequate statistical significance. In artificial environments, such an analysis method is relatively inexpensive. However, performing the same quantity of biological ablations may be time prohibitive.

6.2.3 Signal Analysis

Signal analysis methods use neuronal activity response to generate a quantitatively described neural network function. In Chapter 4, this was done by qualitatively observing multiple neural networks over a large number of behavior observations. Although several methods exist that quantify network function, none of these methods was designed to extract patterns of transient neuronal function observed over multiple neural networks as Neural Dynamic Clustering does. Here, we describe several signal analysis methods related to Neural Dynamic Clustering.

Neural Networks in Nonlinear System Identification

Neural networks have been primarily used in system identification to model nonlinear black-box systems off-line. A black-box system is one where knowledge of the processes within a system is unknown. This often occurs when a system is highly nonlinear and is difficult to apply stimulus/response pairs to on-line. In these cases, a neural network is used to model the system as shown in Figure 6.1. Neural networks are optimized to reproduce responses recorded from a running system over a period of time. Once the neural network model has been fully optimized, it can then be analyzed off-line in place of the black-box system (Figure 6.1(B))[NRPH00] to implement a control structure.

Using neural networks to model and control nonlinear systems as demonstrated in Figure 6.1 is an accepted method in control theory [NRPH00, SEM03]. Neural Dynamic Clustering attempts to reverse-engineer this system identification process by taking the optimized neural network model and creating system models of the individual neurons instead of the entire network.

The nonlinear Volterra series model has been used to model neural network function [MZ97]. Volterra series models may also be extracted from neural network models [Ste05]. Both tasks are accomplished analytically. For our purposes, the extraction of

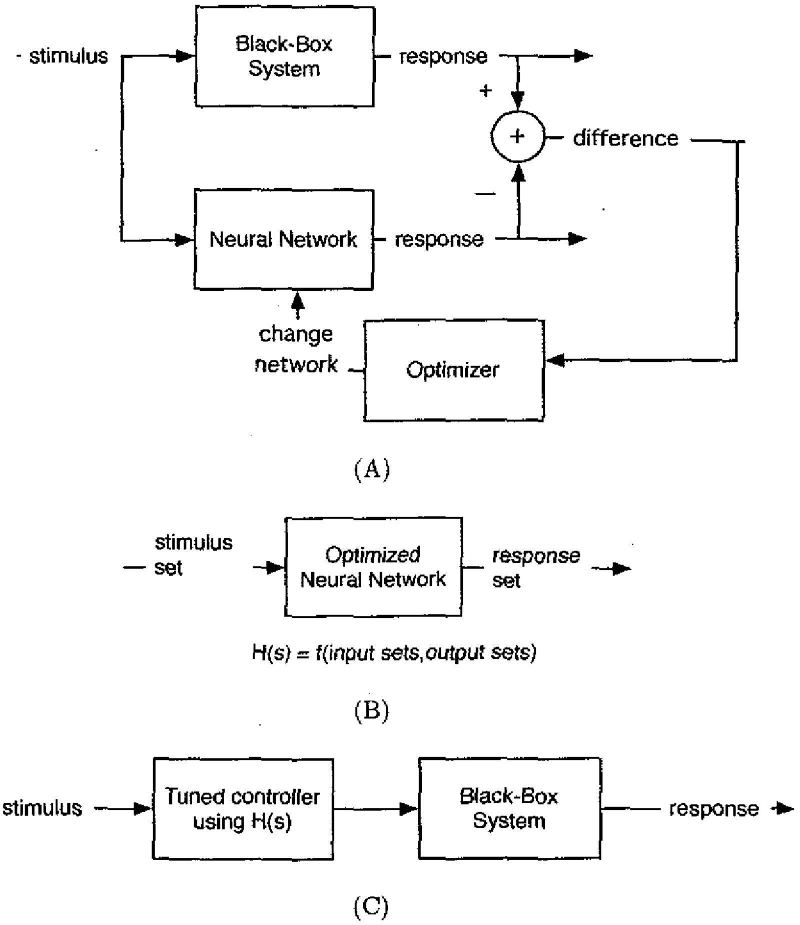


FIGURE 6.1. Neural networks for identification and control of a black-box system. (A) A neural network optimized to mimic a black-box system. Stimulus is fed into both the neural network and the black-box system. The difference between response of the black-box system and the neural network error is used to guide the neural network optimization algorithm. (B) Extracting a transfer function $H(s)$ from the optimized neural network using multiple stimulus/response pairs. (C) A transfer function is used to design a controller for the black-box system. Both the transfer function and the neural network are able to reproduce the response of the black-box system to any stimulus, depending on the quality of the network optimization and transfer function fit.

the Volterra series model is useful, especially at the level of individual neurons. However, we chose not to use a Volterra series to represent neuronal models for Neural Dynamic Clustering because the series may be of variable size and several representations that are more compact are available from which to model neuronal function [WK03].

Instantaneous Linearization

A set of linear transfer functions may describe a nonlinear system over a range of operation that exhibits nonlinear response. This is because linear stability around an equilibrium point corresponds to local nonlinear stability at the same point [NL96, Sr98, NRPH00, CN04]. Thus, several locally stable linear systems extracted over a range of stimuli from a nonlinear black-box system provide equivalent function to the original nonlinear black-box system. Additional local linear models are created only when the range of stimuli cause a significant error for the closest linear model. The poles and zeros associated with the set of linear transfer functions may be plotted to show the different models that are active over a range of stimuli [Sr98]. Instantaneous linearization has been successful for identifying the dynamics of entire neural networks, but was not used in our Neural Dynamic Clustering algorithm, as more direct methods were readily available that measured nonlinearity in a single, nonlinear model instead of several linear models. This was advantageous primarily because we wanted to reduce the number of terms that were clustered.

Stimulus Mapping

Another way to quantify how a neural network performs is to map all of its steady-state input responses. This has been done for multi-layer perceptrons applied to categorization tasks [Duc03]. In this method, optimized neural networks are exposed to a range of multi-variables stimuli where no more than three of the stimuli change. The input variables are plotted as points in three-dimensional space where the color for each point represents the assigned category for each network, which provides insight into how an individual neural network will cluster its classification results.

This gives an indication of potential problem areas of stimulus combinations where categories of neighboring input regions may overlap, suggesting further optimization or verification in that region. However, this method only quantifies steady-state solutions instead of the transient solutions that we are primarily interested in for Chapters 3 and 4.

Rule Extraction

After a neural network model has been created, analysis may be performed by inferring a set of logical rules with respect to network function. These methods reduce the function of a recurrent neural network to a set of logical rules. These rules may take the form of propositional logic (if... then ... else), fuzzy logic, or finite state machines. For recurrent neural networks, most rule extraction methods are finite state machines [JZ05]. All of these methods work by mapping the continuous output states of the network to discrete output states, applying stimuli, and observing the transitions between the discretized output states. Although many of these methods use deterministic classifications to build state machines, the CrySSMEx method may extract finite state machine from a recurrent neural network using input sequences such as a genetic sequence [JZ05]. This method is able to represent a state machine for either deterministic or stochastic data.

The rule extraction paradigm lends itself well to clustering. Networks that contain similar state machine rules would be more likely to belong to the same cluster. Although this method is applied only at the network level, it should be simple to extract state machines from neuronal activity, as well. However, it is unclear how continuous transient function could be discretized to represent important transient function. For the bounce neuron in the bounce-and-trap motif, such a representation may work, as the response of this motif is similar to a deterministic finite state machine. For the differentiators, however, the discrete states would have to be very small to reproduce neuronal and network function.

Time Series Analysis

A set of methods has been proposed that extract protein reaction rates through analysis of concentration time-series data while the system is being perturbed [SKKss, SCJ05, CLC05]. In all methods, rates are represented by a Jacobian matrix. A Jacobian matrix is a matrix of partial derivatives with respect to all values of the matrix. In this case, the values of the matrix are protein concentrations from all of the protein substrates. Protein concentration is analogous to neuronal activity in neural networks and reaction rates are analogous to weights in neural networks. Modification of every node in the network is not necessary, as only a few perturbations are necessary to extract the rate constants. Furthermore, proposed versions of this method also do not require steady-state values to extract rates [CLC05].

An appealing aspect of this method is that it explicitly estimates parameters that drive transient function. Applying this method to neural networks would be problematic, though, because the reaction rates contribute linearly to protein concentration. Additionally, the rate constants are insufficient to reproduce the transient dynamics necessary to replicate neural network or *C. elegans* behavior.

6.2.4 Conclusion

In our contribution to modeling chemotaxis and thermotaxis behavior in *C. elegans*, we optimized multiple neural networks to reproduce a single behavior, either chemotaxis or thermotaxis. Optimization of multiple neural networks allowed us to perform an exhaustive search of possible motifs for a given neural network model. However, adequate tools to extract neural network motifs, or patterns of neuronal function, from a set of neural networks do not exist. The most important period of network function was the transient response directly following stimulus (as opposed to steady-state response). Current network analysis tools took one of three approaches. First, they considered equilibrium or steady-state network solutions (analytical analysis). Equilibrium and steady-state analysis is only helpful for associated-memory (classifiers) or oscillating solutions. Second, they considered features of connectivity (architectural analysis). Connectivity analysis is only helpful for determining the possible importance of connections and neurons, but gives little insight into function, transient or otherwise. Third, they considered features of network response (signal analysis). This type of analysis considered transient response, but only for a single neural network in its entirety. However, Neural Dynamic Clustering analyzes the transient function of neurons over a set of neural networks in order to extract neural network motifs.

CHAPTER 7

Conclusion

7.1 Contributions

This work introduces a new method for analyzing neural networks and contributes to the understanding of the neurophysiological basis behind *C. elegans* chemotaxis and thermotaxis. In Chapter 3, we optimized neural networks to perform *C. elegans* chemotaxis according to the pirouette mechanism. We found one neural network motif, the three-neuron differentiator with inhibitory feedback. Inhibitory feedback was found on self-connections and recurrent connections. Inhibitory feedback at both levels functioned to regulate the response latency of the system's output relative to its input. Self-connections could represent anatomically identified connections between left and right members of the respective neuron pair or voltage dependent currents. This neural network motif suggests new functionality that is testable through electrophysiological recording, calcium imaging, and neuronal ablation.

In Chapter 4, we optimized neural networks to perform either chemotaxis or thermotaxis using any behavior mechanism. We discovered a limited number of functionally distinct neural network motifs from the set of optimized neural networks. We compared the response of chemotaxis motifs we found to biological observations and matched a previously identified differentiator motif—a fast, excitatory and a slow, inhibitory sensorimotor transformation in parallel [Fet93, DCPSL04]. The thermotaxis motifs we found were combinations of chemotaxis motifs in opposing circuits that were either thermophilic (heat-seeking) or cryophilic (cold-seeking). From these results and other observations [RS02], we make several predictions about the neurophysiology that drives *C. elegans* thermotaxis. First, two sensory neurons are necessary to exhibit thermotaxis, with one neuron being dedicated to the thermophilic circuit and one being dedicated to the cryophilic circuit. Second, sub-motifs may share interneurons. Third, thermotaxis motifs have functionally distinct sub-motifs dedicated to either thermophilic or cryophilic behavior. From differences in run duration at constant and changing temperature [RS02], we suggest that the thermotaxis circuit is composed of a differentiator motif for the cryophilic circuit and a slow sensory neuron may drive klinokinesis in the thermophilic circuit.

In Chapter 5, we developed the Neural Dynamic Clustering method to analyze sets of neural networks. We applied Neural Dynamic Clustering to two case studies. In one case study, Neural Dynamic Clustering was applied to the chemotaxis neural networks of Chapter 4. Neural Dynamic Clustering was able to quantitatively identify motifs for chemotaxis that were previously identified without consideration or specific knowledge of individual neuronal models. In the other case study, Neural Dynamic Clustering was applied to the neural networks of Chapter 3. We verified the predicted function from previous analysis, as well as determined the function of larger neural networks that we were previously unable to analyze.

Neural Dynamic Clustering allowed us to analyze a set of networks based on their neuronal function without use of structure. An additional advantage of this method is that it could be applied to any set of networks with continuous nodes, such as the Internet, intracellular processes, or food webs.

7.2 Future Work

The contributions of this thesis suggest several areas for further research. Among these are a more anatomically constrained *C. elegans* chemotaxis model and further research on the Neural Dynamic Clustering method to support *C. elegans* modeling and generic network analysis.

7.2.1 Anatomically Constrained *C. elegans* Chemotaxis

Current models of chemotaxis relate sensory stimulus to a simple stochastic output model that represents movement. However, recent work has provided additional knowledge of sensory neuron function and refinements to models of *C. elegans* locomotion that will allow a network model to more closely reflect anatomical connectivity. Sensory neurons may be recorded during behavior by measuring Ca^{+2} activity in neurons, an indication of cellular activity [KLRB$^+$00, HAK$^+$05, FL05]. Worm movement may be more accurately represented by a newer model under development, which involves two pools of neurons, one for forward locomotion and one for backward locomotion (turning). This combined model should provide both greater detail and accuracy than previous models, giving further insight into *C. elegans* behavior.

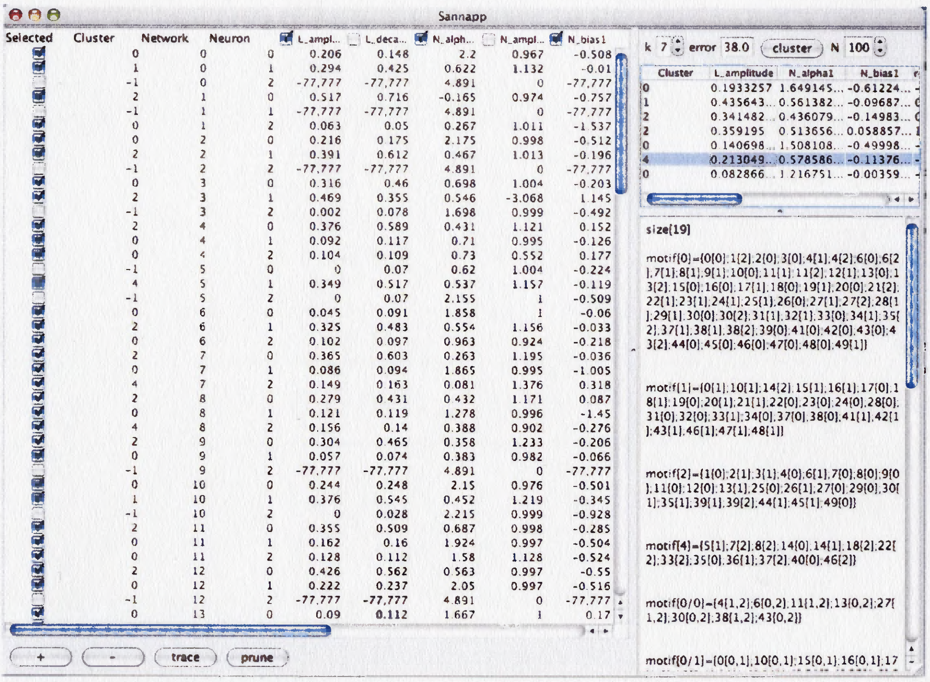
7.2.2 Multi-Sensory Evolution

In this work, we wish to discover how memory may evolve for *C. elegans* type behavior. During thermotaxis, *C. elegans* migrates towards a preferred temperature from either a higher or lower temperature. Although we have referred to this as the cultivation temperature, the temperature "set-point" [CCG$^+$06] for *C. elegans* is easily set and unset by exposing the animal to food at a given temperature [MKK$^+$05]. To explore how this mechanism may occur in *C. elegans*, we model *C. elegans* movement according to the movement model presented in Chapter 4. The model worm is given food that it is able to orient towards at the feeding temperature. Performing thermotaxis to get to a feeding temperature provides an evolutionary advantage, such that an evolutionary algorithm such as rtNEAT [SBM05b] can be used to optimize neural networks. During optimization, feeding temperature would change multiple times over the life of an animal. This would allow us to understand how the *C. elegans* neural network might integrate chemosensory function with thermotactic memory.

7.2.3 Neural Dynamic Clustering

In addition to supporting other future work, Neural Dynamic Clustering will continue to be refined as both an application and a method. We have begun to refactor code written in Python, R, Matlab, and Igor into an application called Sannapp (Signal Analysis of Neural Network Application). Sannapp can currently perform all of the Neural Dynamic Clustering method except for creating neuronal models, which is still done using Matlab scripts adapted from [WK03]. Weka [WE01, FHT$^+$04] was used for the clustering step. All other code was ported or rewritten from the original source used in [DPSCL06]. Figure 7.1 shows Sannapp clustering and combining a set of networks with identified neuron models. Although Sannapp was not used in [DPSCL06], it was able to replicate the results shown in Table 5.3.

Refinements of the method would include additional neuron models, neuron filtering, and clustering methods. These refinements will be added as needed when analyzing additional neural network sets. Additional network analysis should standardize the analysis setting for networks as well as create a catalog of identified motifs. Standardization should also automate steps that currently involve human intervention. A catalog of motifs will allow additional inferences into the effects of optimization algorithms, neural network structure, and objective functions on network behavior and structure.



Selected	Cluster	Network	Neuron	☑ L_ampl...	☐ L_deca...	☑ N_alph...	☐ N_ampl...	☑ N_bias1
☑	0	0	0	0.206	0.148	2.2	0.967	-0.508
☑	1	0	1	0.294	0.425	0.622	1.132	-0.01
☐	-1	0	2	-77,777	-77,777	4.891	0	-77,777
☑	2	1	0	0.517	0.716	-0.165	0.974	-0.757
☐	-1	1	1	-77,777	-77,777	4.891	0	-77,777
☑	0	1	2	0.063	0.05	0.267	1.011	-1.537
☑	0	2	0	0.216	0.175	2.175	0.998	-0.512
☑	2	2	1	0.391	0.612	0.467	1.013	-0.196
☐	-1	2	2	-77,777	-77,777	4.891	0	-77,777
☑	0	3	0	0.316	0.46	0.698	1.004	-0.203
☑	2	3	1	0.469	0.355	0.546	-3.068	1.145
☐	-1	3	2	0.002	0.078	1.698	0.999	-0.492
☑	2	4	0	0.376	0.589	0.431	1.121	0.152
☑	0	4	1	0.092	0.117	0.71	0.995	-0.126
☑	0	4	2	0.104	0.109	0.73	0.552	0.177
☐	-1	5	0	0	0.07	0.62	1.004	-0.224
☑	4	5	1	0.349	0.517	0.537	1.157	-0.119
☐	-1	5	2	0	0.07	2.155	1	-0.509
☑	0	6	0	0.045	0.091	1.858	1	-0.06
☑	2	6	1	0.325	0.483	0.554	1.156	-0.033
☑	0	6	2	0.102	0.097	0.963	0.924	-0.218
☑	2	7	0	0.365	0.603	0.263	1.195	-0.036
☑	0	7	1	0.086	0.094	1.865	0.995	-1.005
☑	4	7	2	0.149	0.163	0.081	1.376	0.318
☑	2	8	0	0.279	0.431	0.432	1.171	0.087
☑	0	8	1	0.121	0.119	1.278	0.996	-1.45
☑	4	8	2	0.156	0.14	0.388	0.902	-0.276
☑	2	9	0	0.304	0.465	0.358	1.233	-0.206
☑	0	9	1	0.057	0.074	0.383	0.982	-0.066
☐	-1	9	2	-77,777	-77,777	4.891	0	-77,777
☑	0	10	0	0.244	0.248	2.15	0.976	-0.501
☑	1	10	1	0.376	0.545	0.452	1.219	-0.345
☐	-1	10	2	0	0.028	2.215	0.999	-0.928
☑	2	11	0	0.355	0.509	0.687	0.998	-0.285
☑	0	11	1	0.162	0.16	1.924	0.997	-0.504
☑	0	11	2	0.128	0.112	1.58	1.128	-0.524
☑	2	12	0	0.426	0.562	0.563	0.997	-0.55
☑	0	12	1	0.222	0.237	2.05	0.997	-0.516
☐	-1	12	2	-77,777	-77,777	4.891	0	-77,777
☑	0	13	0	0.09	0.112	1.667	1	0.17

k 7 error 38.0 N 100

Cluster	L_amplitude	N_alpha1	N_bias1
0	0.1933257	1.649145...	-0.61224...
1	0.435643...	0.561382...	-0.09687...
2	0.341482...	0.436079...	-0.14983...
2	0.359195	0.513656...	0.058857...
0	0.140698...	1.508108...	-0.49998...
4	0.213049...	0.578586...	-0.11376...
0	0.082866...	1.216751...	-0.00359...

```
size[19]

motif[0]={0[0];1[2];2[0];3[0];4[1];4[2];6[0];6[2
];7[1];8[1];9[1];10[0];11[1];11[2];12[1];13[0];1
3[2];15[0];16[0];17[1];18[0];19[1];20[0];21[2];
22[1];23[1];24[1];25[1];26[0];27[1];27[2];28[1
];29[1];30[0];30[2];31[1];32[1];33[0];34[1];35[
2];37[1];38[1];38[2];39[0];41[0];42[0];43[0];4
3[2];44[0];45[0];46[0];47[0];48[0];49[1]}


motif[1]={0[1];10[1];14[2];15[1];16[1];17[0];1
8[1];19[0];20[1];21[1];22[0];23[0];24[0];28[0];
31[0];32[0];33[1];34[0];37[0];38[0];41[1];42[1
];43[1];46[1];47[1];48[1]}


motif[2]={1[0];2[1];3[1];4[0];6[1];7[0];8[0];9[0
];11[0];12[0];13[1];25[0];26[1];27[0];29[0];30[
1];35[1];39[1];39[2];44[1];45[1];49[0]}


motif[4]={5[1];7[2];8[2];14[0];14[1];18[2];22[
2];33[2];35[0];36[1];37[2];40[0];46[2]}


motif[0/0]={4[1,2];6[0,2];11[1,2];13[0,2];27[
1,2];30[0,2];38[1,2];43[0,2]}


motif[0/1]={0[0,1];10[0,1];15[0,1];16[0,1];17
```

FIGURE 7.1. Sannapp application implements the Neural Dynamic Clustering method. The left panel represents neuron and neuron model instances. The neuron models (rows) and their parameters (columns) may be selected or excluded from analysis. The cluster value (second column) is assigned during the k-means clustering step. The upper-right panel represents cluster centers. *k* is the number of new initial random centers, error is sum-squared error, and *N* is the number of centers to attempt. The bottom-right panel represents combined motifs. These are recalculated automatically following the clustering step. As shown here, clusters may be reassigned, which forces a recalculation of combined motifs.

BIBLIOGRAPHY

[ASMR03] R. Aharonov, L. Segev, I. Meilijson, and E. Ruppin. Localization of function via lesion analysis. *Neural Comput*, 15(4):885–913, 2003.

[Bee95] R.D. Beer. On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behavior*, 3:469–509, 1995.

[BH91] C.I. Bargmann and H.R. Horvitz. Chemosensory neurons with overlapping functions direct chemotaxis to multiple chemicals in *C. elegans*. *Neuron*, 7:729–742, 1991.

[BHG99] R.D. Beer, J.C. Hillel, and J.C. Gallagher. Evolution and analysis of model cpgs for walking: II. General principles and individual variability. *J Comp Neurosci*, 7(2):99–118, 1999.

[BM06] B.D. Bryant and R. Miikkulainen. Evolving stochastic controller networks for intelligent game agents. In *Proceedings of the IEEE 2006 Congress on Evolutionary Computating*, Piscataway, NJ, 2006. IEEE.

[Bre74] S. Brenner. The genetics of *C. elegans*. *Genetics*, 77:71–94, 1974.

[BT95] Roberto Battiti and Giampietro Tecchiolli. Training neural nets with the reactive tabu search. *IEEE Transactions on Neural Networks*, 6(5):1185–1200, September 1995.

[CCG+06] S.H. Chung, D.A. Clark, C.V. Gabel, E. Mazur, and D.A. Samuel. The role of the afd neuron in c. elegans thermotaxis analyzed using femtosecond laser ablation. *BMC Neuroscience*, 7(30), 2006.

[CCM+01] A. W. Chan, K. Y. Chong, C. Martinovich, C. Simerly, and G. Schatten. Transgenic monkeys produced by retroviral gene transfer into mature oocytes. *Science*, 291(5502):309–12, 2001.

[CEF55a] J.S. Coombs, J.C. Eccles, and P. Fatt. The electrical properties of the motoneurone membrane. *Journal of Physiology*, 139:291–325, 1955.

[CEF55b] J.S. Coombs, J.C. Eccles, and P. Fatt. The specific ionic conductances and the ionic movements across the motoneurone membrane that produce the inhibitory post-synaptic potential. *Journal of Physiology*, 139:326–373, 1955.

[CHC06] Beth L. Chen, David H. Hall, and Dmitri B. Chklovskii. Wiring optimization can relate neuronal structure and function. *PNAS*, 103(12):4723–4728, 2006.

[CLC05] Wen-Chieh Chang, Chang-Wei Li, and Bor-Sen Chen. Quantitative inference of dynamic regulatory pathways via microarray data. *BMC Bioinformatics*, 6(1):44, 2005.

[CN04] Lingji Chen and Kumpati S. Narendra. Identification and control of a nonlinear discrete-time system based on its linearization: A unified framework. *IEEE Transactions on Neural Networks*, 15(3):663–673, 2004.

[CSW⁺85] M. Chalfie, J.E. Sulston, J.G. White, E. Southgate, J.N. Thomson, and S. Brenner. The neural circuit for touch sensitivity in *C. elegans*. *J Neurosci*, 5:956–964, 1985.

[CTE⁺94] M. Chalfie, Y. Tu, G. Euskirchen, W.W. Ward, and D.C. Prasher. Green fluorescent protein as a marker for gene expression. *Science*, 263(5148):802–805, Feb 1994.

[DCL03] N.A. Dunn, J.S. Conery, and S.R. Lockery. A neural network model for chemotaxis in *C. elegans*. In *Neural Networks, 2003. Proceedings of the International Joint Conference on Neural Networks*, volume 4, pages 2574–2578. IEEE, July 2003. paper 708.

[DCL04] N.A. Dunn, J.S. Conery, and S.R. Lockery. Circuit optimization predicts dynamic network for chemosensory orientation in the nematode c. elegans. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[DCPSL04] N.A. Dunn, J.S. Conery, J.T. Pierce-Shimomura, and S.R. Lockery. A neural network model of chemotaxis predicts functions of synaptic connections in the nematode *C. elegans*. *J Comput Neurosci*, 17(2), 2004.

[Deb01] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* John Wiley & Sons, New York, 2001.

[DG06] A. Dimitrov and T. Gedeon. Effects of stimulus transformations on estimates of sensory neuron selectivity. *Journal of Computational Neuroscience*, 20(3):265–283, 2006.

[DM03] S.N. Dorogovtsev and J.F. Mendes. *Evolution of Networks: From Biological Nets to the Internet and www.* Oxford Press, 2003.

[DPSCL06] N.A. Dunn, J.T. Pierce-Shimomura, J.S. Conery, and S.R. Lockery. Clustered neural dynamics identify motifs for chemotaxis in *C. elegans*. In *International Joint Conference on Neural Networks*. IEEE, 2006.

[DS89a] R. E. Davis and A. O. Stretton. Passive membrane properties of motorneurons and their role in long-distance signaling in the nematode *Ascaris*. *J Neurosci*, 9:403–414, 1989.

[DS89b] R. E. Davis and A. O. W. Stretton. Signaling properties of *Ascaris* motorneurons: graded active response, graded synaptic transmission, and tonic transmitter release. *J Neurosci*, 9:415–425, 1989.

[Duc03] W Duch. Coloring black boxes: visualization of neural network decisions. In *International Joint Conference on Neural Networks*, volume 1, pages 1735–1740. IEEE, 2003.

[Dus74] D.B. Dusenbery. Analysis of chemotaxis in the nematode *C. elegans* by counter current separation. *J Experimental Zoology*, 188:41–47, 1974.

[Dus80] D.B. Dusenbery. Responses of the nematode *C. elegans* to controlled chemical stimulation. *J Comp Physiol*, 136:127–331, 1980.

[Dus01] D.B. Dusenbery. Performance of basic strategies for following gradients in two dimensions. *J Theoretical Biology*, 208:345–360, 2001.

[EH06] B. Everitt and T. Hothorn. *A Handbook of Statistical Analyses using R*. Chapman & Hall/CRC, Boca Raton, FL, 2006. ISBN 1-584-88539-4.

[Eve01] B.S. Everitt. *Cluster Analysis*. Oxford University Press, New York, NY, 4 edition, 2001.

[FBB$^+$97] J.J. Falke, R.B. Bass, S.L. Butler, S.A. Chervitz, and M.A. Danielson. The two-component signaling pathway of bacterial chemotaxis: A molecular view of signal transduction by receptors, kinases, and adaptation enzymes. *Annual Review of Cell and Developmental Biology*, 13(1):457–512, 1997.

[Fet93] E.E. Fetz. Dynamic neural network models of sensorimotor behavior. In D. Gardner, editor, *The Neurobiology of Neural Networks*, pages 165–190. Bradford Books, MIT Press, Cambridge, 1993.

[FHT$^+$04] E. Frank, M. Hall, L. Trigg, G. Holmes, and I. H. Witten. Data mining in bioinformatics using weka. *Bioinformatics*, 20(15):2479–2481, October 2004.

[FL05] S. Faumont and S.R. Lockery. The awake behaving worm: simultaneous imaging of neuronal activity and behavior in intact animals at millimeter scale. *J Neurophysiol*, Nov 2005.

[FWR99] Ying-Huey Fua, Matthew O. Ward, and Elke A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *VIS '99: Proceedings of the conference on Visualization '99*, pages 43–50, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press.

[GHAL98] M.B. Goodman, D.H. Hall, L. Avery, and S.R. Lockery. Active currents regulate sensitivity and dynamic range in *C. elegans* neurons. *Neuron*, 20:763–772, 1998.

[GHB05] Jesse M. Gray, Joseph J. Hill, and Cornelia I. Bargmann. Inaugural Article: A circuit for navigation in Caenorhabditis elegans. *PNAS*, 102(9):3184–3191, 2005.

[GS98] T.W. Grebe and J. Stock. Bacterial chemotaxis: The five sensors of a bacterium. *Current Biology*, 8(5):R154–R157, February 1998.

[HAK$^+$05] Massimo A Hilliard, Alfonso J Apicella, Rex Kerr, Hiroshi Suzuki, Paolo Bazzicalupo, and William R Schafer. In vivo imaging of c. elegans ash neurons: cellular response and adaptation to chemical repellents. *The EMBO Journal*, 24:63–72, December 2005.

[HBG99] J.C. Hillel, R.D. Beer, and J.C. Gallagher. Evolution and analysis of model cpgs for walking: I. dynamical modules. *J Comput Neurosci*, 7(2):43–62, 1999.

[HDNL$^+$04] C. Huetz, C. Del Negro, K. Lehongre, P. Tarroux, and J. Edeline. The selectivity of canary hvc neurons for the bird's own song: Rate coding, temporal coding, or both? *Journal of Physiology-Paris*, 98(4-6):395–406, 2004.

[HL00] D. Hanselman and B. Littlefield. *Mastering Matlab 6 (Paperback)*. Prentice Hall, Upper Saddle River, NJ, 2000. ISBN 0130194689.

[Hop82] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc Natl Acad Sci U S A*, 79(8):2554–8, 1982.

[HR75] E.M. Hedgecock and R.L. Russell. Normal and mutant thermotaxis in the nematode *C. elegans*. *Proc Natl Acad Sci U S A*, 72:4061–4065, 1975.

[IIM06] H. Ito, H. Inada, and I. Mori. Quantitative analysis of thermotaxis in the nematode caenorhabditis elegans. *J Neurosci Methods*, 154:45–52, 2006.

[JZ05] H. Jacobsson and T. Ziemke. Cryssmex, a novel rule extractor for recurrent neural networks : Overview and case study. *Lecture Notes in Computer Science*, 3697:503–508, 2005. ICANN 2005 : 15th International Conference, Warsaw, Poland, September 11-15, 2005 : proceedings).

[KIKM02] A. Kuhara, H. Inada, I. Katsura, and I. Mori. Negative regulation and gain control of sensory neurons by the *C. elegans* calcineurin tax-6. *Neuron*, 33(5):751–763, 2002.

[KKM⁺05] A. Kaufman, A. Keinan, I. Meilijson, M. Kupiec, and E. Ruppin. Quantitative analysis of genetic and neuronal multi-perturbation experiments. *PloS Computational Biology*, 2005.

[KLRB⁺00] R. Kerr, V. Lev-Ram, G. Baird, P. Vincent, R. Y. Tsien, and W. R. Schafer. Optical imaging of calcium transients in neurons and pharyngeal muscle of *C. elegans*. *Neuron*, 26(3):583–94, 2000.

[KMMM04] Koutarou D Kimura, Atsushi Miyawaki, Kunihiro Matsumoto, and Ikue Mori. The *C. elegans* thermosensory neuron AFD responds to warming. *Curr Biol*, 14(14):1291–1295, 2004.

[KNM84] Stephen W. Kuffler, John G. Nicholls, and A. Robert Martin. *From Neuron to Brain*. Sinauer, Sunderland, Massachusetts, 2nd edition, 1984.

[KSH⁺04] A. Keinan, B. Sandbank, C.C. Hilgetag, I. Meilijson, and E. Ruppin. Fair attribution of functional contribution in artificial and biological networks. *Neural Computation*, 16(9):1887–1915, 2004.

[LG98] S.R. Lockery and M.B. Goodman. Tight-seal whole-cell patch clamping of *C. elegans* neurons. *Methods Enzymol*, 293:201–17, 1998.

[LPS99] S.R. Lockery and J.T. Pierce-Shimomura. Unpublished *C. elegans* chemotaxis in a radial gradient. 31 worm tracks and concentrations recorded performing chemotaxis in a radial gradient of $NH_4Cl.$, 1999.

[LS93] S.R. Lockery and T.J. Sejnowski. Realistic network models of distributed processing in the leech. In D. Gardner, editor, *The Neurobiology of Neural Networks*, pages 107–136. Bradford Books, MIT Press, Cambridge, 1993.

[Mas93] T. Masters. *Practical Neural Network Recipes in C++*. Morgan Kaufmann, New York, 1993.

[MF89] A.N. Michel and J.A. Farrell. Qualitative analysis of neural networks. *IEEE Transactions on Circuits and Systems*, 36(2):229–243, 1989.

[MFL98] T. M. Morse, T. C. Ferree, and S. R. Lockery. Robust spatial navigation in a robot inspired by chemotaxis in *C. elegans*. *Adaptive Behavior*, 6:763–772, 1998.

[MKK$^+$05] Akiko Mohri, Eiji Kodama, Koutarou D. Kimura, Mizuho Koike, Takafumi Mizuno, and Ikue Mori. Genetic Control of Temperature Preference in the Nematode Caenorhabditis elegans. *Genetics*, 169(3):1437–1450, 2005.

[ML02] A.N. Michel and D Liu. *Qualitative Analysis and Synthesis of Recurrent Neural Networks*. Macel Dekker, Inc, 2002.

[MM82] A.N. Michel and R.K. Miller. *Ordinary Differential Equations*. Academic Press, 1982.

[MO95] I. Mori and Y. Ohshima. Neural regulation of thermotaxis in *C. elegans*. *Nature*, 376:344–348, 1995.

[MP43] W.S. McCulloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bull. Math. Biophys.*, 5:115–133, 1943.

[MSF94] E.E. Munro, L.E. Shupe, and E.E Fetz. Integration and differentiation in dynamical recurrent neural networks. *Neural Comput*, 6:405–419, 1994.

[MSF03] M.A. Maier, L.E. Shupe, and E.E. Fetz. Recurrent neural networks of integrate-and-fire cells simulating short-term memory and wrist movement tasks derived from continuous dynamic networks. *Journal of Physiology-Paris*, pages 601–612, 2003.

[MSI$^+$02] R. Milo, Shen-Orr S., S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.

[MTF$^+$05] A.C. Miller, T.R. Thiele, S. Faumont, M.L. Moravec, and S.R. Lockery. Step-response analysis of chemotaxis in *C. elegans*. *J Neurosci*, 2005.

[MZ97] V.Z. Marmarelis and Z. Zhao. Volterra models and three-layer perceptrons. *IEEE Transactions on Neural Networks*, 8:1421–1433, 1997.

[NL96] Kumpati S. Narendra and Sai-Ming Li. *Mathematical Perspectives on Neural Networks*, chapter 11: Neural Networks in Control Systems, pages 347–393. Lawrence Erlbaum Associates, Mahwah, New Jersey, 1996.

[NPBK02] W.T. Nickell, R.Y. Pun, C.I. Bargmann, and S.J. Kleene. Single ionic channels of two *C. elegans* chemosensory neurons in native membrane. *J Membrane Biology*, 189(1):55–66, 2002.

[NRPH00] M. Norgaard, O. Ravn, N.K. Poulsen, and L.K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems.* Springer, Great Britian, 2000.

[Pas02] F. Pasemann. Complex dynamics and the structure of small neural networks. *Network: Computation in Neural Systems*, 13:195–216, 2002.

[PFTV88] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, Cambridge, 1988.

[Pil04] Mark Pilgrim. *Dive into Python (Paperback).* Apress, 2004.

[PN92] Robert Pinter and Bahram Nabet. *Nonlinear Vision: Determination of Neural Receptive Fields, Functions, and Networks.* CRC Press, Boca Raton, Florida, 1992.

[PSML99] J.T. Pierce-Shimomura, T.M. Morse, and S.R. Lockery. The fundamental role of pirouettes in *C. elegans* chemotaxis. *J Neurosci*, 19(21):9557–69, 1999.

[PTVF92] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C.* Cambridge University Press, New York, 2nd edition, 1992.

[RAC04] Markus Reigl, Uri Alon, and Dmitri Chklovskii. Search for computational modules in the c. elegans brain. *BMC Biology*, 2(1):25, 2004.

[RBMP97] D.L. Riddle, T. Blumenthal, B.J. Meyer, and J.R. Priess, editors. *Genome sequencing: The worm revealed.* Cold Spring Harbor Laboratory Press, 1997.

[RBP$^+$95] R. Rizzuto, M. Brini, P. Pizzo, M. Murgia, and T. Pozzan. Chimeric green fluorescent protein as a tool for visualizing subcellular organelles in living cells. *Curr Biol*, 5(6):635–642, 1995.

[RC79] T.A. Rutherford and N.A. Croll. Wave forms of *C. elegans* in a chemical attractant and repellent and in thermal gradients. *J Nematology*, 11:232–240, 1979.

[Ros58] F. Rosenblatt. The perceptron: A probabalistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[RS02] W. S. Ryu and A. D. Samuel. Thermotaxis in *C. elegans* analyzed by measuring responses to defined thermal stimuli. *J Neurosci*, 22(13):5727–33, 2002.

[SB98] I. Segev and R.E. Burke. Compartmental models of complex neurons. In C. Koch and I. Segev, editors, *Methods in neuronal modeling*, pages 93–136. MIT Press, Cambridge, 2 edition, 1998.

[SBBP90] M.J. Schnitzer, S.M. Block, H. Berg, and E.M. Purcell. Strategies for chemotaxis. In J.P. Armitage and J.M. Lackie, editors, *Symp. Soc. Gen. Biol.*, volume 46, pages 15–34. Cambridge Univ. Press, Cambridge, U.K., 1990. Biology of the Chemotactic Response.

[SBM05a] K.O. Stanley, B.D. Bryant, and R. Miikkulainen. Evolving neural network agents in the nero video game. In *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games*, Piscataway, NJ, 2005. IEEE.

[SBM05b] K.O. Stanley, B.D. Bryant, and R. Miikkulainen. Real-time neuroevolution in the nero video game. *IEEE Transactions on Evolutionary Computation*, 9(6):653–668, 12 2005.

[SCJ05] H. Schmidt, K. Cho, and E.W. Jacobsen. Identification of small scale biochemical networks based on general type system perturbations. *FEBS Journal*, 272(9):2141–2151, 2005.

[SEM03] D.E. Seborg, T.F. Edgar, and D.A. Mellichamp. *Process Dynamics and Control.* Wiley Text Books, 2 edition, 2003.

[Sha53] L.S. Shapley. A value for n-person games. In H.W. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games*, pages 307–317. Princeton University Press, Princeton, 1953.

[SK04] O. Sporns and R. Kotter. Motifs in brain networks. *PloS Biology*, 2(11), 2004.

[SKKss] E. Sontag, A. Kiyatkin, and B. Kholodenko. Inferring dynamic architecture of cellular networks using time series of gene expression, protein and metabolite data. *Bioinformatics*, in Press.

[SMC05] L. Stuart, D. Marocco, and A. Cangelosi. Information visualization for knowledge extraction in neural networks. *Lecture Notes in Computer Science*, 3697:515–520, 2005. ICANN 2005 : 15th International Conference, Warsaw, Poland, September 11-15, 2005 : proceedings).

[SMR96] P. Smolensky, M.C. Mozer, and D.E. Rumelhart, editors. *Mathematical Perspectives on Neural Networks.* Lawrence Erlbaum Associates, Mahwah, New Jersey, 1996.

[SNL+99] Y. Sambongi, T. Nagae, Y. Liu, T. Yoshimizu, K. Takeda, Y. Wada, and M. Futai. Sensing of cadmium and copper ions by externally exposed adl, ase, and ash neurons elicits avoidance response in *C. elegans. Neuroreport*, 10(4):753–7, 1999. 0959-4965Journal Article.

[SS05] AD Samuel and P Sengupta. Sensorimotor integration: locating locomotion in neural circuits. *Curr Biol*, 15(9):R341–R343, 2005.

[SSM03] AD Samuel, RA Silva, and VN Murthy. Synaptic activity of the AFD neuron in *Caenorhabditis elegans* correlates with thermotactic memory. *J Neurosci*, 23(2):373–376, 2003.

[Ste05] G. Stegmayer. Comparison of volterra models extracte from a neural network for nonlinear systems modeling. *Lecture Notes in Computer Science*, 3697:457–463, 2005. ICANN 2005 : 15th International Conference, Warsaw, Poland, September 11-15, 2005 : proceedings).

[SW80] J. Sulston and J. White. Regulation and cell autonomy during postembryonic develop- ment of *C.elegans. Develop. Biol.*, 78:577–597, 1980.

[SW98] S. H. Strogatz and D. J. Watts. Collective dynamics of 'small-world' networks. *nature*, 393:440–442, 1998.

[Sr98] O. Srensen. System identification, prediction, simulation and control with neural networks. In *Fourth International Conference on Neural Networks and their Applications*, Marseilles, France, March 1998.

[TB04] D.M. Tobin and C.I. Bargmann. Invertebrate nociception: behaviors, neurons and molecules. *J Neurobiol*, 61(1):161–174, 2004.

[TH86] D. Tank and J. Hopfield. Simple 'neural' optimization networks: An a/d converter, signal decision circuit, and a linear programming circuit. *IEEE Transactions on Circuits and Systems*, 33(5):533–541, 5 1986.

[TH03] L.T. Tsalik and O. Hobert. Functional mapping of neurons that control locomotory behavior in *C. elegans. J Neurobiol*, 56(2):178–197, 2003.

[War73] S. Ward. Chemotaxis by the nematode *C. elegans:* identification of attractants and analysis of the response by use of mutants. *Proc of the Natl Acad Sci USA*, 70:817–821, 1973.

[WE01] Ian H. Witten and Frank Eibe. *Data Mining.* Hanser Fachbuch, January 2001.

[WF01] A. Wagner and D.A. Fell. The small world inside large metabolic networks. *Proc R Soc Lond B Biol Sci*, 268(1478):1803–1810, Sep 2001.

[Whi85] J.G. White. Neuronal connectivity in *C. elegans*. *Trends in Neuroscience*, 8:277–283, 1985.

[wIH04] S. wiegand, C. Igel, and U. Handmann. Evolutionary multi-objective optimization of neural networks for face detection. *International Journal of Computational Intelligence and Applications*, 4(3):237–253, 2004.

[WK03] D.T. Westwick and R.E. Kearney. *Identification of Nonlinear Physiological Systems.* IEEE Press, Hoboken, NJ, 2003.

[WKS04] T. Wakabayashi, I. Kitagawa, and R. Shingai. Neurons regulating the duration of forward locomotion in *C.elegans*. *Neuroscience Research*, 50(1):103–111, September 2004.

[WR95] S. R. Wicks and C. H. Rankin. Integration of mechanosensory stimuli in *C. elegans*. *J Neurosci*, 15(3):2434–2344, 1995.

[WSTB86] J. G White, E. Southgate, J. N. Thomson, and S. Brenner. The structure of the nervous system of the nematode *C. elegans*. *Phil Trans of the R Soc Lond [Biol]*, 314:1–340, 1986.

[WTWB75] S. Ward, N. Thomson, J. G. White, and S. Brenner. Electron microscopical reconstruction of the anterior sensory anatomy of the nematode *C. elegans*. *J Comp Neurol*, 160:313–338, 1975.

[WW90] E. Wolinsky and J. Way. The behavioral genetics of *C. elegans*. *Behavior Genetics*, 20:169–189, 1990.

[YO03] Y Yamada and Y Ohshima. Distribution and movement of *C. elegans* on a thermal gradient. *J Exp Biol*, 206:2581–2593, Aug 2003.

[ZBM+99] Y. Zheng, P. J. Brockie, J. E. Mellem, D. M. Madsen, and A. V. Maricq. Neuronal control of locomotion in *C. elegans* is modified by a dominant mutation in the glr-1 ionotropic glutamate receptor. *Neuron*, 24(2):347–61, 1999.

[ZMFL03] Z.A. Zariwala, A.C. Miller, S. Faumont, and S.R. Lockery. Step response analysis of thermotaxis in *C. elegans*. *J Neurosci*, 2003.