# Neural Prosthetics for Paralysis: Algorithms and Low-Power Analog Architectures for Decoding Neural Signals

by

Benjamin Isaac Rapoport

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of

Master of Science in Physics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2007



Author .....
Department of Physics
5 February 2007

Certified by .....
Rahul Sarpeshkar
Associate Professor
Thesis Supervisor

Certified by .....
H. Sebastian Seung
Professor
Thesis Supervisor

Accepted by .....
Professor Thomas Greytak
Chairman, Department Committee on Graduate Students

# Neural Prosthetics for Paralysis: Algorithms and Low-Power Analog Architectures for Decoding Neural Signals

by

Benjamin Isaac Rapoport

Submitted to the Department of Physics
on 5 February 2007, in partial fulfillment of the
requirements for the degree of
Master of Science in Physics

## Abstract

This thesis develops a system for adaptively and automatically learning to interpret patterns of electrical activity in neuronal populations in a real-time, on-line fashion. The system is primarily intended to enable the long-term implantation of low-power, microchip-based recording and decoding hardware in the brains of human patients in order to treat neurologic disorders. The decoding system developed in the present work interprets neural signals from the parietal cortex encoding arm movement intention, suggesting that the system could function as the decoder in a neural prosthetic limb, potentially enabling a paralyzed person to control an artificial limb just as the natural one was controlled, through thought alone. The same decoder is also used to interpret the activity of a population of thalamic neurons encoding head orientation in absolute space. The success of the decoder in that context motivates the development of a model of generalized place cells to explain how networks of neurons adapt the configurations of their receptive fields in response to new stimuli, learn to encode the structure of new parameter spaces, and ultimately retrace trajectories through such spaces in the absence of the original stimuli. Qualitative results of this model are shown to agree with experimental observations. This combination of results suggests that the neural signal decoder is applicable to a broad scope of neural systems, and that a microchip-based implementation of the decoder based on the designs presented in this thesis could function as a useful investigational tool for experimental neuroscience and potentially as an implantable interpreter of simple thoughts and dreams.

Thesis Supervisor: Rahul Sarpeshkar
Title: Associate Professor

Thesis Supervisor: H. Sebastian Seung
Title: Professor

# Acknowledgments

It has been my privilege to pursue the research described in this thesis under the supervision of Professor Rahul Sarpeshkar, in the Analog VLSI and Biological Systems Group at the Research Laboratory of Electronics of the Massachusetts Institute of Technology. His enthusiasm and intuition have been extremely important in guiding my research.

I would also like to thank three advisors who have been instrumental in enabling me to move forward in my research and graduate studies. Professor Michael Feld, my Academic Advisor in the MIT Physics Department, has been very generous with his support and valuable advice. Professor Sebastian Seung has been my Physics Department Co-Supervisor as I have pursued an interdisciplinary project with a research group in the Department of Electrical Engineering and Computer Science. And Professor Philip Maini supervised my work at the Centre for Mathematical Biology at Oxford University during a wonderful fellowship year in which I developed modeling tools and intellectual interests that led to the present work.

The experimental data I used while pursuing the work reported here was very graciously provided by three collaborators: Dr. Sam Musallam and Professor Richard Andersen of the Division of Biology at the California Institute of Technology, and Hector Penagos of the Department of Brain and Cognitive Sciences at the Massachusetts Institute of Technology. I am grateful to them for their generosity and their interest in the present work.

While undertaking the research described in this thesis for the Physics Department, I have been honored to be a student jointly in the Department of Health Sciences and Technology at MIT, and in the MD-PhD Program at Harvard Medical School. I wish to express my sincere thanks to these programs as well as to the National Science Foundation Graduate Research Fellowship Program and the Hugh Hampton Young Fellowship. Their confidence in my potential as a scientist and future physician dedicated to designing and implementing therapeutic electronic interfaces with the brain and nervous system, as expressed through financial and administrative

support of my work and through experienced and concerned professional advising, has been essential.

Finally, I wish to express my deepest gratitude to my parents, my most influential moral and scientific mentors, for their inspiration and support.

# Contents

# List of Figures

# Chapter 1

# Introduction

An extraordinary set of experiments performed over the past several years have demonstrated the feasibility of constructing remarkable electronic interfaces with primate and human brains [44, 36, 40, 4, 25, 14, 32]. Such interfaces have proven capable of stably recording electrical signals from populations of cortical neurons in the brains of living subjects over periods of weeks or months, decoding the thoughts encrypted in those signals, and using the decoded information to control mechanical or computer-based interfaces in real-time. This kind of electronic interface with the brain holds great promise as a therapeutic approach to treating severe paralysis, and a dramatic demonstration of this promise received widespread public attention several months ago following the publication of the results of the first human trial of such a brain-machine interface in a severely paralyzed patient, along with a set of supplementary video clips demonstrating the system being used by its first subject to control a computer mouse by thought alone, and to manipulate a rudimentary robot prosthetic arm [14]. The same set of video clips, however, showed evidence of the long-term impracticability of the system being demonstrated. In particular, the prototype used to obtain such promising results in its first human trial required a transcranial, transdermal port (a port passing from the brain, through the bone of the skull, and through the overlying skin) to guide connections from an array of wire electrodes in the motor cortex of the patient's brain to an extensive external system for electrical signal processing. Following a one-year trial period, the implanted port

Figure 1-1: **Patient Connected to a State-of-the-Art Brain-Computer Interface.** The first patient to use the brain-computer interface described in [14], paralyzed from the neck down, is shown beside a robotic prosthetic hand whose fingers he was able to open and close using his thoughts alone. The transcranial, transdermal port used to transmit signals from neural recording electrodes to external signal processing hardware, is visible as a gray object on the right side of his head. From [14], Supplemental Information, 'Video 6: Neural Control, Prosthetic Hand.'

was removed and the system dismantled. Any implanted device physically bridging the intracranial cavity with the external environment provides a potential route for infectious agents to enter the brain, and in consideration of this potentially disastrous complication such a device cannot constitute a long-term solution to the problem of developing a therapeutic electronic interface with the brain.

Two major drawbacks are associated with state-of-the-art brain-machine interfaces. First, they require transcranial, transdermal links to connect electrodes implanted in the brain to external systems used to power and decode recorded neural signals. Second, the schemes used to decode neural signals are computationally intensive and are implemented using power-hungry digital signal processing hardware and software. In order to elevate present-generation devices in status from fascinating research tools to therapeutic interventions, the devices will have to be made sufficiently small and power-efficient to be implanted safely and completely within the body of a patient. Implantable electronic medical devices have been used therapeutically for years, including some prominent examples capable of recording and analyzing electronic signals. The cardiac pacemaker and deep brain stimulator, shown in Figure

1-2 are examples of such devices. The former delivers regular electrical stimuli to a diseased heart in order to maintain an appropriate heartbeat, and is also capable of detecting intrinsic heartbeats and changing its pacing signals in response to changes in native cardiac performance and physiologic requirements. The deep brain stimulator, on the other hand, has been used to deliver controlled electrical stimulation to the globus pallidus, a region of the brain implicated in mediating tremors and other abnormal, uncontrollable body movements associated with neurologic conditions such as Parkinson's disease [45]. As illustrated in the figures, each device consists of a comparatively large power supply and signal processing unit implanted in the chest wall, where space and heat dissipation are of minimal concern. This unit is connected to recording or stimulation electrodes implanted in the anatomic site of interest. In light of this existing paradigm for electronic medical implants, therapeutic brain-machine interfaces might be expected to develop along similar lines, with brain-implanted recording electrodes connected to wires that exit the skull but do not pierce the overlying skin, and connect to a power supply and signal processing unit implanted at a remote site within the body, such as the chest wall.

The research presented in this thesis is part of a collaborative effort to build a practical neural motor prosthesis, a brain-machine interface capable of enabling paralyzed patients to gain thought-based control of artificial limbs. The collaboration is based in the Analog VLSI and Biological Systems Group of the Research Laboratory of Electronics at the Massachusetts Institute of Technology. The ultimate design goal of the project is to enable a microchip-based prosthesis with brain-implanted electrodes, neural signal detection circuitry, neural signal decoding electronics, and a wireless transceiver to operate with power consumption low enough for the system to run on a small, implanted 100 milliampere-hour battery with 1000 wireless recharges for at least 10 years. Such a system will represent an important step toward making complex, brain-implantable motor prostheses a reality for paralyzed human patients. As indicated in the preceding paragraphs, a major impediment to developing complex, miniature, and power-efficient brain-machine interfaces is the extensive digital signal processing currently required to decode electrical recordings from populations


(a) (b)

Figure 1-2: **Cardiac Pacemaker and Deep Brain Stimulation Systems.** (a) A typical cardiac pacemaker with electrode leads implanted in the heart and a remotely located power supply and signal processing unit in the chest wall [23]. (b) A deep brain stimulation system with electrodes implanted in the brain and a remotely located power supply and signal processing unit in the chest wall [24].

of neurons into signals suitable for controlling an external device such as a computer interface or robotic artificial limb. With the high power-consumption rates of digital electronics comes ease of programmability, and so state-of-the-art neural signal decoding systems tend to use digital platforms almost exclusively [3]. It is possible, however, to design a neural signal decoding algorithm that can be implemented using ultra-low-power analog electronics. Such an algorithm could obviate the need for a remote power supply and signal processing unit, enabling a self-contained electronic interface with the brain to be implanted and to operate entirely within the skull cavity. A fully implantable, ultra-low-power, microchip-based brain-machine interface of this kind would represent a valuable therapeutic advance, as it could simplify procedures associated with implanting and maintaining the bioelectronic device, possibly reduce device-related complications by dramatically reducing the size and power dissipation of the device, and potentially enable the simultaneous implantation of multiple devices with a variety of brain-machine interface functions not limited to prosthesis control.

A major component of the challenge of developing a micropower neural prosthetic interface is designing a neural signal decoding algorithm that can be implemented using low-power analog electronics rather than the high-power, easily programmable digital systems in widespread use at present. Chapter Two of this thesis describes the theoretical basis for such an algorithm and discusses its practical implementation, providing results from simulations and performance tests using neural signal data recorded from trained experimental animals confirming the viability of the neural decoding technique described.

The problem of decoding the intentions of a thinking subject from electrical signals recorded from the brain essentially amounts to 'mind reading' in that it requires translating the patterns of electrical activity generated by populations of neurons (patterns that might reasonably be identified with the 'thoughts' of the subject) into signals comprehensible to others or to external devices in forms such as control commands to computer interfaces or robotic artificial limbs. On the surface, this problem of 'thought translation' might appear to belong to the realm of science fiction. In fact, however, such translation can be achieved with a relatively high probability of accurate decoding given a small amount of a priori knowledge concerning the information content of the brain region generating the recorded signals and a limited number of degrees of freedom specifying the intention to be decoded. The a priori knowledge typically amounts to knowing that the region of the brain under study is implicated in the function of interest to the decoder; for example, neural recordings from the motor cortex have been used for neural motor prostheses designed to decode arm reaching movements [5]. The limited number of degrees of freedom might be values of a set of kinematic variables describing the trajectory of a mouse cursor or artificial limb in one, two, or three-dimensional space (although the parameter space of kinematic variables describing the three-dimensional movement of a complex robotic limb can certainly have dimension greater than three) or discrete values corresponding to elements in a finite set of reach targets or perhaps the keys on a thought-controlled keyboard. In any such system the neural decoding problem amounts to optimizing a mapping of the form $W : N \rightarrow M$, where $N$ denotes the space of neural signals,

$M$ denotes the space of motor output parameters used to control a neural motor prosthesis, and $W$ transforms neural signals into sets of motor parameters. Chapter Two derives and discusses a decoding method in which $W$ is treated as an $m \times n$ matrix $\mathbf{W}$ of convolution kernels $W_{ij}$, $i \in \{1, \ldots m\}$, $j \in \{1, \ldots n\}$ (where $m$ and $n$ denote the dimensions of the spaces $M$ and $N$, respectively). A vector of motor parameters $\mathbf{M}(t) \in M$ at time $t$ is derived from a neural signal $\mathbf{N}(t') \in N$, $t' \in (0, t]$, by convolving $\mathbf{W}$ with $\mathbf{N}(t')$. In this sense, $\mathbf{W}$ is a matrix of active filters $W_{ij}$, whose inputs are neural signals and outputs are motor parameters. Each of the $W_{ij}$, in turn, depends on a set of $p$ filter parameters, $\left\{W_{ij}^{(1)}, \ldots, W_{ij}^{(p)}\right\}$. Optimizing the decoding performance achieved by $\mathbf{W}$ corresponds to tuning the parameters that define the $W_{ij}$ so as to achieve the most accurate results. Chapter Two derives and demonstrates a gradient-descent least-squares–based scheme by which the active filter $\mathbf{W}$ can adaptively train itself, using feedback from its performance during an initial training phase as in supervised on-line machine learning systems.

The final section of Chapter Two discusses how the neural decoding algorithm developed earlier in the chapter can be implemented in an ultra-low-power analog VLSI (very large scale integrated) microchip-based system. The actual design of such a chip is currently being pursued in the Analog VLSI and Biological Systems Group in the Research Laboratory of Electronics at the Massachusetts Institute of Technology by the author and fellow student Woradorn Wattanapanitch under the supervision of Professor Rahul Sarpeshkar.

After Chapter Two discusses a method of decoding information encrypted in signals produced by populations of neurons, Chapter Three extends the range of neural systems to which such a decoder is applicable by demonstrating that the decoding techniques used to interpret motor signals in Chapter Two can also be used to decode head direction from a population of thalamic neurons in a laboratory rat. Such on-line decoding of place-like cell activity is not typically possible using conventional methods of analysis. Motivated by this innovation, Chapter Three develops and investigates a theoretical model of the way in which information is encoded, internally represented, and internally manipulated by such neuronal populations as they process

and store sensory and higher-order information in the brain. It has been known for three decades that populations of cells within the hippocampus, now known as 'place cells,' collectively encode spatial maps as a result of each cell tuning to sensory parameters associated with a particular location [28, 2]. The robustness of the resulting neurally encoded maps have enabled experimenters to infer the time-varying position of experimental animals in controlled environments by observing the temporal dynamics of place cell electrical activity in real time. From the successes achieved in decoding hippocampal place cell activity emerged the concept of 'population vector coding,' which conceived of kinematic parameters describing spatial position as being encoded in the collective electrical activity patterns of populations of neurons [11]. And indeed, one successful effort to develop a neural prosthetic arm employed a population vector-based neural decoding strategy [41]. These developments further motivate the modeling undertaken in Chapter Three.

Using the hippocampal place cell paradigm as a starting point, Chapter Three develops a model of a generalized place cell, in effect a 'parameter space cell,' whose activity pattern is tuned to a point in a parameter space. In this sense the generalized place cell can be modeled by an activity function $f_i(\vec{x})$, $\vec{x} \in X$, $i \in \{1, \ldots, n\}$, where $X$ denotes a space of arbitrary topology and dimension $d$, parameterized by $(s_1(\vec{x}), \ldots, s_d(\vec{x}))$ and neurally encoded by a population of $n$ parameter space cells $\{f_1(\vec{x}), \ldots, f_n(\vec{x})\}$ (in hippocampal place cells, the parameters modeled by the $s_i$ are typically sensory inputs such as visual, olfactory, auditory, vestibular, or proprioceptive cues experienced in association with a particular point in parameter space, such as the location of a laboratory rat in an experimental maze [2]). The 'tuning' of a parameter space cell activity function $f_i$ to a point $\vec{x}_i \in X$ refers to the existence of a maximum of $f_i(\vec{x})$ at $\vec{x}_i$. Chapter Three develops the concept of a parameter space trajectory as a way of formalizing the notion of a temporal sequence of experiences learned and stored by a population of neurons, and later recalled in the absence of the original stimuli as a 'train of thought.' The chapter develops a model intended to explain how a network of neurons can learn to encode information about the structure of previously unfamiliar parameter spaces through exploration of those spaces,

experience-based adaptation of the functional forms of the $f_i$, and modification of the topology of the neural network. Some qualitative results generated by the model are compared with experimental observations.

Given the success in understanding the neural encoding scheme of hippocampal place cells and the extension of population vector-based decoding techniques to neuronal populations relevant to the operation of neural motor prostheses, it seems natural to wonder whether the parameter space cell generalization might provide a reasonable model of the neural encoding dynamics and internal representations of information stored and processed by a variety of neuronal populations. Future experience with neural motor prostheses and other brain-machine interface systems may provide insight into the strategies used by the brain to encode information and enable the parameter space cell model to be further refined and compared in greater detail with experimental observations.

Chapter Four summarizes the results presented in this thesis and discusses several directions for further research.

# Chapter 2

# An Algorithm and Analog Architecture for Decoding Neural Signals

## 2.1 Overview

As discussed in the Introduction, the research described in this thesis is part of a collaborative research project whose goal is to design, build, and test a miniature, ultra-low-power system of analog chips, suitable for long-term implantation in the brain, and capable of adaptive, real-time decoding of movement intention signals to be used in the control of prosthetic limbs in animal models, and eventually applied in the treatment of paralyzed human patients. In order to ensure that the system is fully implantable, its overall power budget requires that the system will be able to run on a small, implanted 100 mAh battery with 1000 wireless recharges for at least 10 years. This low power budget is made possible in large part by the ability of carefully designed low-power analog signal processing to perform the functions currently implemented by high-power digital electronics in state-of-the-art systems. This chapter discusses the development of a neural decoding and learning algorithm optimized for implementation in customized analog electronics.

A variety of decoding techniques for neural signals have been developed and implemented successfully in rodents [6], monkeys [44, 40, 36, 4, 25] and humans [17, 8]. The methods employed in these systems have been reviewed in the literature and include two primary strategies: adaptive linear filtering and probabilistic methods [43, 49].

Testing of such algorithms typically involves observing an experimental animal trained to perform a motor task in a stimulus-reward behavioral paradigm. In linear decoding algorithms, a linear projection of the neuronal firing-rate variables is mapped to a smaller set of motor variables over a certain time window. The projection matrix is optimized by applying linear regression techniques to training data obtained from simultaneous recordings of neural signals and limb position in experimental animals, or in the case of a paralyzed human patient by simultaneous recordings of neural signals and the motor command the subject is instructed to execute mentally [14]. In Bayesian probabilistic decoding algorithms, data of a similar form is used to generate a database of training trials from which maximum likelihood estimates of an intended motor behavior are estimated conditioned on observing a particular neural signal [37].

The actual signals used to decode motor intentions in most systems reported to date have been neuronal action potential 'spikes.' In addition, local field potential (LFP) signals have proven to contain predictive information relevant to decoding neural motor signals, with gamma band (25–90 Hz) activity containing information predictive of arm-reach direction and lower-frequency spectral activity predictive of movement onset [37]. Neural prostheses designed to decode primarily from LFP activity rather than from intrinsically high-frequency spikes may be able to operate at reduced power due to their lower bandwidth requirements. Furthermore, LFPs are likely to be more robust than spike signals since they represent the aggregate activity of a population of neurons rather than that of a single unit, and so are less sensitive to system perturbations such as electrode movement.

In spite of dramatic preliminary successes reported in the field of neuromotor prosthetics, all existing systems accomplish neural decoding through the use of massive amounts of signal processing hardware and digital post processing. This chapter

describes a system designed to achieve real-time neural decoding in a miniature, implantable, ultra-low-power context. The scheme is based on analog circuitry that implements a continuous-time, adaptive linear filtering algorithm to map neural signal inputs onto motor command outputs. The following sections describe the mathematical foundations of this method, present results from simulations of the system described, and discuss how it can be implemented in a low power analog electronic setting.

## 2.2 A Gradient-Descent Least-Squares Approach to Decoding Neural Signals

The function of neural decoding is to map neural signals onto the motor commands to which those signals correspond. In a neuromotor prosthetic system, the neural signals are obtained from electrode interfaces with populations of cortical neurons. The decoding system must transform those raw data into the control signals for manipulation of a prosthetic limb. Such a system typically has two modes of operation: a training mode in which it optimizes (or 'learns') the mapping it must implement, and an operational mode in which it uses the learned mapping to control a prosthesis. This section presents the mathematical foundations of a modified gradient-descent least-squares algorithm that operates in real time and automatically learns how to perform an optimized translation of raw neural signals into motor control parameters.

The gradient-descent least-squares algorithm is a method for optimizing a linear transformation of the form $\mathbf{WN}(t) = \mathbf{M}(t)$, where $\mathbf{N}(t)$ is an $n$-dimensional vector containing the neural signal data (neuronal firing rates, analog signal values, or local field potentials, for example) at time $t$, $\mathbf{M}(t)$ is a corresponding $m$-dimensional vector containing the motor output parameters (limb or joint positions, velocities, or accelerations, for example) generated at time $t$, and $\mathbf{W}$ is a matrix of linear weights (formally analogous to a matrix of synaptic weights encountered in the context of related problems in the field of artificial neural networks) used to convert the collec-

tion of neural signals $\mathbf{N}(t)$ into each of the motor control parameters in $\mathbf{M}(t)$. The present section first describes a traditional gradient-descent least-squares procedure for optimizing a static weight matrix, $\mathbf{W}$ [48, 13]. It then explains the construction of a modified method that enables a time-dependent weighting kernel, $\mathbf{W}(t)$, to optimize itself dynamically through real-time learning.

Suppose the neural signal data consist of time-dependent voltage waveforms collected from $n$ neurons. Then $\mathbf{N}(t)$ denotes a column vector of dimension $n$ whose $i$th component, $N_i(t)$, corresponds to the amplitude of the voltage waveform measured from the $i$th neuron at time $t$. Similarly, suppose that $m$ motor control parameters are to be extracted from the neural signal data. These parameters might include variables corresponding to linear speeds of motion in each of the principal directions and angular velocities at limb joints [31]. Then $\mathbf{M}(t)$ denotes a column vector of dimension $m$, whose $i$th element, $M_i(t)$, corresponds to the value of the $i$th motor parameter at time $t$. Under these circumstances, $\mathbf{W}$ is an $m \times n$ matrix of time-independent, linear weights.

A solution to this static neural decoding problem is achieved by optimizing $\mathbf{W}$ at time $t$ based on all data obtained by the system since a previous time, $t - \tau$, where $\tau$ is a tunable time constant of the decoding system. (The problem is 'static' in the sense that $\mathbf{W}$ the decoded motor output signal at time $t$ is assumed to depend only on the neural input signals received at time $t$.) This process of optimization takes place during the learning mode, during which $\mathbf{M}(t)$ is known. In experimental systems on animals with functioning limbs, as mentioned earlier, $\mathbf{M}(t)$ can be obtained by observing the motion of the limb, while in clinical settings $\mathbf{M}(t)$ can be obtained by instructing a patient to execute a prescribed set of commands mentally [8] or mentally to mirror a sequence of actions presented in a video sequence. During the operational mode, by contrast, the function of the system is precisely to predict $\mathbf{M}(t)$ using $\mathbf{WN}(t) \approx \mathbf{M}(t)$ and the matrix $\mathbf{W}$ optimized during the training mode. In words, one can state that $\mathbf{W}$ is considered optimized, in a least-squares sense, when the sum of the squares of the deviation of its predicted motor control parameters is minimized over a given time interval. This statement can be expressed mathematically, as follows.

The prediction error at time $t$ is the vector

$$\mathbf{e}(t) = \mathbf{M}(t) - \mathbf{WN}(t), \tag{2.1}$$

so the sum of the squared deviations of all of the motor parameters from their predicted values at time $t$ is

$$|\mathbf{e}(t)|^2 = (\mathbf{M}(t) - \mathbf{WN}(t))^T (\mathbf{M}(t) - \mathbf{WN}(t)), \tag{2.2}$$

where the superscripted $T$ denotes the transpose operation. The optimized weight matrix must minimize this quantity integrated over a specified time interval $\tau$, so the actual quantity to be minimized is

$$E(W, t, \tau) \equiv \int_{t-\tau}^{t} |\mathbf{e}(t)|^2 dt \tag{2.3}$$

$$= \int_{t-\tau}^{t} (\mathbf{M}(t) - \mathbf{WN}(t))^T (\mathbf{M}(t) - \mathbf{WN}(t))\, dt. \tag{2.4}$$

The gradient descent scheme for optimizing $\mathbf{W}$ operates as follows. Since $E$ is quadratic in $\mathbf{W}$, it possesses a unique extremum $E(\mathbf{W}^*)$ when considered as a function of $\mathbf{W}$ or its elements, $W_{ij}$. The $m \times n$ elements of $\mathbf{W}$ can be taken to lie in an $m \times n$-dimensional parameter space, within which the optimized $\mathbf{W}$, $\mathbf{W}^*$, corresponds to a single point. The negated gradient of $E$ with respect to the $W_{ij}$,

$$-\vec{\nabla} E(\mathbf{W}, t, \tau) = \frac{\partial E(\mathbf{W}, t, \tau)}{\partial W_{ij}} \tag{2.5}$$

$$= 2 \int_{t-\tau}^{t} \mathbf{N}(t) (\mathbf{M}(t) - \mathbf{WN}(t))\, dt, \tag{2.6}$$

is then a vector (tangent to the parameter space of the $W_{ij}$) with two important

properties. First, the vector $-\vec{\nabla}E(\mathbf{W}, t, \tau)$ is directed from the present value of $\mathbf{W}$ toward a better approximation of its optimum value $\mathbf{W}^*$. Second, the quadratic nature of $E$ ensures that the magnitude of $-\vec{\nabla}E(\mathbf{W}, t, \tau)$ depends linearly on the distance in $W_{ij}$-space between $\mathbf{W}$ and $\mathbf{W}^*$, since the quantity $\mathbf{M}(t) - \mathbf{WN}(t)$ in the integrand of Equation 2.6 can be expressed as

$$\mathbf{M}(t) - \mathbf{WN}(t) = \mathbf{W}^*(t)\mathbf{N}(t) - \mathbf{WN}(t) \tag{2.7}$$

$$= (\mathbf{W}^* - \mathbf{W})\,\mathbf{N}(t). \tag{2.8}$$

These properties facilitate the construction of a decoding system in which an initially arbitrary $\mathbf{W}$ can be induced to converge automatically toward the optimum weight matrix, $\mathbf{W}^*$, after suitably many $\tau$ have elapsed. Such a system is constructed by permitting $-\vec{\nabla}E(\mathbf{W}, t, \tau)$ to serve as a feedback signal to modify each $W_{ij}$ in $\mathbf{W}$ in proportion to $\frac{\partial E(\mathbf{W},t,\tau)}{\partial W_{ij}}$ on a time scale set by $\tau$ during the training mode.

The accuracy of predictions made by the static decoding scheme just described is likely to be limited because the predictions it makes are based only on present inputs. The information content and predictive value of recent data are therefore lost to the system. Effective decoding should exploit the information content of prior data in interpreting present inputs. The performance of the decoding system just described can be improved by relaxing the static assumption made during its construction, but doing so adds a level of complexity to the system. Such an improvement can be achieved by eliminating the static $\mathbf{W}$ in favor of a dynamically optimized time-dependent weighting kernel, $\mathbf{W}(t)$. If the elements of $\mathbf{W}$ are permitted to vary in time, and if data at earlier times are permitted to influence present predictions, the decoding scheme can be modified from its original form as a matrix product, $\mathbf{M}(t) \approx \mathbf{WN}(t)$, to a convolution product:

$$\mathbf{M}(t) = \int_{u=0}^{t} \mathbf{W}(t-u)\mathbf{N}(u)du, \tag{2.9}$$

in which the operator $\mathbf{W}(t)$ is reinterpreted as a matrix of time-dependent weighting kernels. In this system, the prediction error at time $t$ is the vector

$$\mathbf{e}(t) = \mathbf{M}(t) - \int_{u=0}^{t} du \, \{\mathbf{W}(t-u)\mathbf{N}(u)\}. \tag{2.10}$$

Optimizing the kernel, in a least-squares sense, still corresponds to minimizing a quantity of the form $E \equiv \int |\mathbf{e}(t)|^2 dt$. However, bearing in mind that the aim of this analysis is ultimately to design an effective active filter, further insight into the system can be obtained by applying the theorem of Parseval, which provides that $\int |\mathbf{e}(t)|^2 dt = \int |\mathbf{e}(\omega)|^2 d\omega$, indicating that the total error evaluated in the time domain is equal to the total error evaluated in the frequency domain. The decoding scheme can therefore be expressed in frequency space as

$$\overline{\mathbf{e}(t)} = \overline{\mathbf{M}(t) - \mathbf{W}(t) \circ \mathbf{N}(t)} \rightarrow \overline{\mathbf{e}}(t) = \overline{\mathbf{M}}(\omega) - \overline{\mathbf{W}}(\omega)\overline{\mathbf{N}}(\omega), \tag{2.11}$$

where $\omega$ denotes the frequency space variable, and the overbars and open circle denote the Fourier transform and convolution product operations, respectively.

The gradient descent scheme for optimizing the weighting kernels can then proceed in analogy with the procedure used in the initial system, although the meaning of certain quantities must be reinterpreted in accordance with the transformation to the frequency domain. The vectors $\mathbf{N}(\omega)$ and $\mathbf{M}(\omega)$ correspond to the Fourier transforms of the time-dependent vectors $\mathbf{N}(t)$ and $\mathbf{M}(t)$, respectively, but are of no special interest as such because the decoding system will handle only the corresponding time-domain signals and will not be required to perform any Fourier transforms explicitly. The matrix $\mathbf{W}(\omega)$, however, can usefully be reinterpreted as an array of frequency-domain filters, corresponding to the time-domain convolution with the time-dependent matrix of weighting kernels, $\mathbf{W}(t)$. The filters can be constructed so as to depend on a set of $n$ parameters for each of the $m$ motor control variables, so that $\mathbf{W}(t)$ is specified by a total of $m \times n$ parameters, $W_{ij}(t)$, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$, which can

therefore be thought of as lying in an $m \times n$-dimensional parameter space.

Since $E(\mathbf{W}(t), t)$ remains quadratic in $\mathbf{W}(t)$, it still possesses a unique extremum, $\mathbf{W}^*(t)$, when considered as a function of the parameters $W_{ij}(t)$. The negated gradient of $E$ with respect to the $W_{ij}(t)$,

$$-\vec{\nabla}E(t) = \frac{\partial E(\mathbf{W}(t), t)}{\partial W_{ij}(t)} \tag{2.12}$$

$$= 2\int_{t'=0}^{t} \mathbf{M}(t) - \mathbf{N}(t)\left(\mathbf{W}(t) \circ \mathbf{N}(t)\right) dt, \tag{2.13}$$

is then a vector tangent to the $m \times n$-dimensional parameter space containing the elements of the filter $\mathbf{W}(t)$ that operates on the incoming neural signals in order to decode them into motor commands. Once again, $-\vec{\nabla}E(t)$ is directed from the present value of $\mathbf{W}(t)$ toward a better approximation of its optimum value $\mathbf{W}^*(t)$, and its magnitude depends linearly on the parameter space distance between $\mathbf{W}(t)$ and $\mathbf{W}^*(t)$. As indicated earlier and as discussed in the next section, by using $-\vec{\nabla}E(t)$ as a feedback signal to modify $\mathbf{W}(t)$, each of the parameters $W_{ij}$ can be modified in proportion to $\frac{\partial E(t)}{\partial W_{ij}}$ over a predetermined timescale, so that an initially arbitrary $\mathbf{W}(t)$ can be induced to converge toward an optimized matrix of convolution kernels; or, as seen in frequency space, an optimized set of filters for the input neural signals.

## 2.3 An Algorithm for Decoding Neural Signals Using a Gradient-Descent Least-Squares Approach

This section describes a practical method of using the modified gradient-descent least-squares approach developed in the preceding section to optimize a matrix $\mathbf{W}$ of convolution kernels for estimating the motor intention $\mathbf{M}(t)$ encoded in a neural signal $\mathbf{N}(t)$ as $\mathbf{W} \circ \mathbf{N}(t) \approx \mathbf{M}(t)$. During the learning period in which this optimization takes place, $\mathbf{W}$ can be considered as a function of time, $\mathbf{W}(t)$. The optimization algorithm is designed to induce $\mathbf{W}(t)$ to converge to $\mathbf{W}^*$, defined by $\mathbf{W}^*\mathbf{N}(t) = \mathbf{M}(t)$,

to within a suitable margin of tolerance over a timescale shorter than the duration of the learning period. In the least-squares framework, the goal of the learning phase is to adapt $\mathbf{W}(t)$ so that the least-squares error over a specified integration interval, $\tau$, defined as

$$E(\mathbf{W}, t, \tau) = \int_{t-\tau}^{t} |\mathbf{e}(u)|^2 du \tag{2.14}$$

$$= \sum_{i=1}^{m} \int_{t-\tau}^{t} |\mathbf{e}_i(u)|^2 du \tag{2.15}$$

$$\equiv \sum_{i=1}^{m} E_i, \tag{2.16}$$

is minimized. The independence of each of the $m$ terms in 2.15 is due to the independence of the $m$ sets of parameters $W_{ij}$, $j \in \{1, \ldots, n\}$ associated with generating each $\mathbf{M}_i(t)$ and implies that the error $E_i$ associated with each motor output can be treated independently of the others. Since

$$\mathbf{e}(t) = \mathbf{M}(t) - \int_{u=0}^{t} \mathbf{W}(t-u)\mathbf{N}(u) du \tag{2.17}$$

$$\equiv \mathbf{M}(t) - \mathbf{W}(t) \circ \mathbf{N}(t), \tag{2.18}$$

as in Equation 2.10, a given motor control output, $M_i(t)$, is given by

$$M_i(t) = \sum_{j=1}^{n} W_{ij}(t) \circ N_j(t), \tag{2.19}$$

where each $W_{ij}(t)$ can be thought of as the impulse-response function corresponding to a filter applied to $N_j(t)$. The problem of designing a practical algorithm for optimizing $\mathbf{W}$ can therefore be construed as a problem of defining appropriate functional forms for the filters $W_{ij}$. If the functional form of each $W_{ij}$ is taken to depend on a set of $p$ parameters, $W_{ij}^{(k)}$ $k \in \{1, \ldots, p\}$, then the gradient-descent method provides an

approach to modifying those filter parameters toward a least-squares optimum. A practical learning strategy for optimizing the matrix of filter kernels is to modify each of the filter parameters continuously and in parallel, on a timescale set by $\tau$, in the direction indicated by the negated gradient of $E_i(\mathbf{W}, t, \tau)$ with respect to each $W_{ij}^{(k)}$:

$$-\vec{\nabla}_{ij}^{(k)} E(\mathbf{W}, t, \tau) \equiv \frac{\partial E}{\partial W_{ij}^{(k)}} \tag{2.20}$$

$$= -\sum_{i=1}^{m} \int_{t-\tau}^{t} du \left\{ 2 \left( M_i(u) - \sum_{j=1}^{n} W_{ij}(u) \circ N_j(u) \right) \times \left( -\frac{\partial W_{ij}(u)}{\partial W_{ij}^{(k)}} \circ N_j(u) \right) \right\} \tag{2.21}$$

$$= \sum_{i=1}^{m} 2 \int_{t-\tau}^{t} e_i(u) \left( \frac{\partial W_{ij}(u)}{\partial W_{ij}^{(k)}} \circ N_j(u) \right) du. \tag{2.22}$$

Expressed in words, the learning algorithm refines $\mathbf{W}$ in a continuous-time fashion on a timescale set by $\tau$. At each time step, each of the parameters $W_{ij}^{(k)}$ is incremented by a term proportional to $-\vec{\nabla}_{ij}^{(k)} E(\mathbf{W}, t, \tau)$, where the proportionality constant, $\epsilon$, is a suitably small numerical constant whose value can be chosen empirically. The quantity $-\vec{\nabla}_{ij}^{(k)} E(\mathbf{W}, t, \tau)$ used to increment each filter parameter can be described in words as the product of the error in the filter output and a filtered version of the filter input, averaged over a time interval $\tau$. The error term is identical for the parameters of all filters contributing to a given component of the motor output, $M_i(t)$. The filtered version of the filter input is generated by a convolution kernel $\frac{\partial W_{ij}(u)}{\partial W_{ij}^{(k)}}$, which depends on the functional form of each filter and in general differs for each filter parameter. Considered in these terms, the learning algorithm is analogous to the 'delta rule' of artificial neural network theory [13]. Figure 2-1 shows a block diagram of a system for implementing the neural signal decoding algorithm described in this section. As discussed more fully in Section 2.6, the averaged product is implemented by enabling the outputs of a set of product blocks alter the voltages on capacitors designated for storing the voltage-encoded parameter values. These stored values are modified through exponentially weighted averaging using low-pass filters.

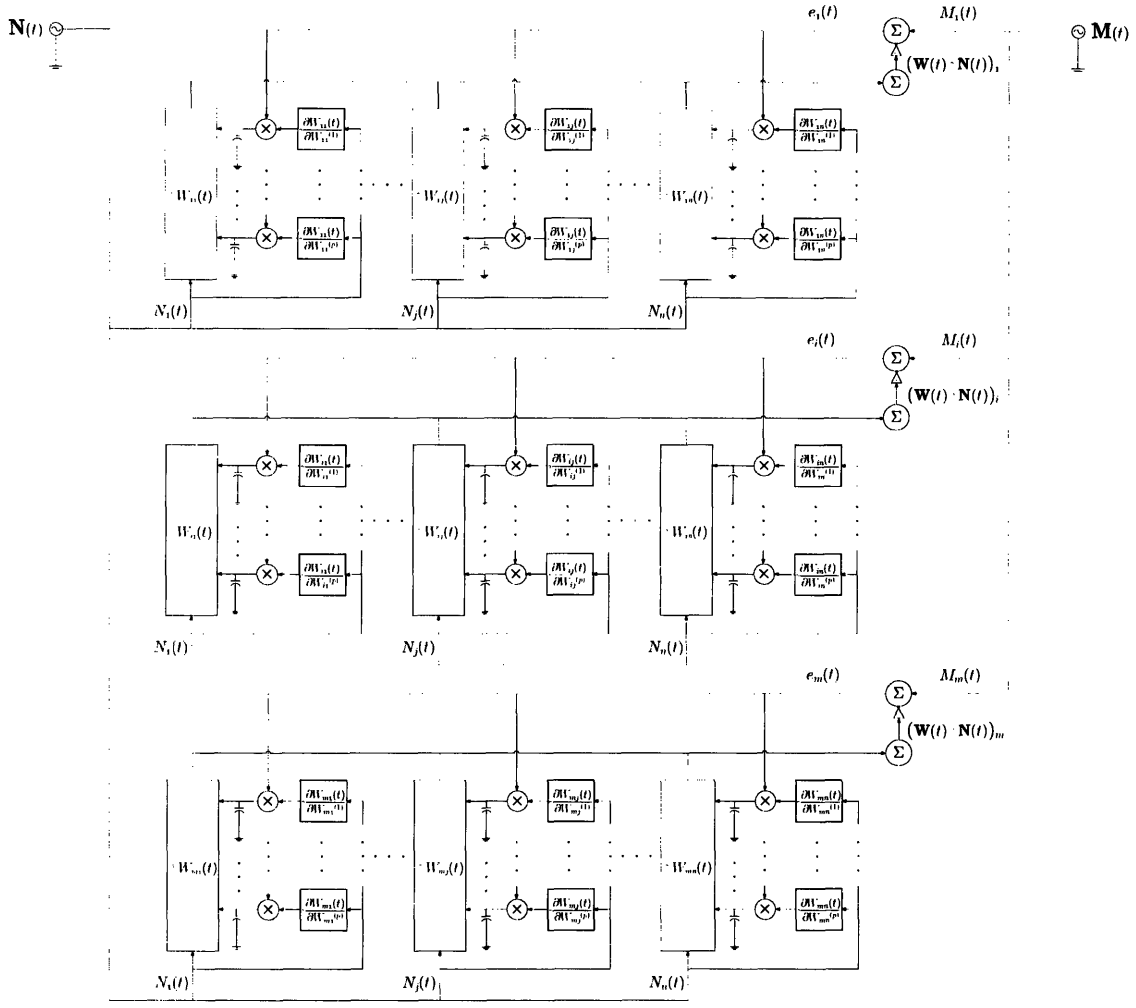


Figure 2-1: **Block Diagram of a System for Implementing a Continuous-Time Modified Gradient-Descent Least-Squares Neural Decoding Scheme.** This figure diagrams a system for implementing the continuous-time modified gradient-descent least-squares neural signal decoding algorithm described in Sections 2.2 and 2.3.

As discussed in detail in Sections 2.4 and 2.5, this decoding system was implemented and tested using a set of first-order low-pass filters having transfer functions of the form

$$H_{ij}(s) = \frac{A_{ij}}{1 + \tau_{ij} s}, \tag{2.23}$$

where $s \equiv i\omega$. The corresponding impulse-response kernels therefore have the form

$$W_{ij}(t) = \frac{A_{ij}}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}}, \tag{2.24}$$

where $A_{ij} \equiv W_{ij}^{(k=1)}$ and $\tau_{ij} \equiv W_{ij}^{(k=2)}$, so that

$$\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k=1)}} = \frac{1}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}} \tag{2.25}$$

$$\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k=2)}} = \frac{A_{ij}}{\tau_{ij}^2} e^{-\frac{t}{\tau_{ij}}} \left( \frac{t}{\tau_{ij}} - 1 \right). \tag{2.26}$$

Intuitively, this form of impulse-response kernel is analogous to multiplication of the neural signal $\mathbf{N}(t)$ by an $m \times n$ matrix with elements $A_{ij}$. However, the additional parameter $\tau_{ij}$ associated with each $A_{ij}$ defines a characteristic time over which previous inputs $\mathbf{N}(t')$, $t' < t$, influence the estimation $\tilde{\mathbf{M}}(t) = \mathbf{W} \circ \mathbf{N}(t)$ with contributions that decay exponentially in $t - t'$ at a rate set by $\tau_{ij}$.

Equation 2.25 indicates that the partial derivative of the low-pass kernel used to generate the kernel for modifying the zero-frequency gain constants $A_{ij}$ is a scaled version of the original impulse response. Equation 2.26, on the other hand, indicates that the partial derivative of the low-pass kernel used to generate the kernel for modifying the time constants $\tau_{ij}$ is a band-pass filter with a tunable pass band. The simulated performance of the neural signal decoding system as implemented using filters of this form is discussed in Sections 2.4 and 2.5, and a scheme for implementing

the system using a set of low-power analog circuits is discussed in Section .

## 2.4 Performance of Linear Decoding Algorithm on Simulated Neural Signals

As one way of testing the neural signal decoding algorithm described in Sections 2.2 and 2.3, a system was devised for simulating neural signals of the kind recorded in a collaborating laboratory from the primate parietal cortex during experiments that required the animals to perform directed arm or eye movement tasks [29, 1, 25]. Such experiments have demonstrated that the signals emitted by parietal neurons just before arm or eye movement contain information predicting the direction of the impending movement, and that this information is encoded primarily in the 25–90 Hz spectral component of the recorded local field potential (the 25–90 Hz regime is referred to as the $\gamma$-band) [29, 1]. In particular, individual parietal neurons tend to exhibit electrical activity predictive of arm or eye movement in a single preferred direction. Increases in $\gamma$-band activity of such tuned parietal neurons anticipate movement in the preferred directions of those neurons. This phenomenon is exhibited in Figure 2-2(a). A potentially useful signal for decoding intended movement from neural activity is therefore an envelope curve describing the modulated amplitude of the power transmitted in the gamma pass-band. Such a curve can be constructed by tracing the peaks of the rectified output of a band-pass filter tuned to the $\gamma$-band. A band-pass filter and envelope detector (peak-detecting rectifier) operating in low-power analog circuitry has been developed and described by other members of the research group in which the present work is being pursued, so the feasibility of this signal processing step has been taken for granted [34].

In order to test the performance of the neural decoding system, a set of simulated $\gamma$-band power envelopes were generated in order to model the local field potentials recorded by a set of $n$ neural recording electrodes. One simulated spectrogram is shown together with its corresponding $\gamma$-band power envelope in Parts (b) and (c)

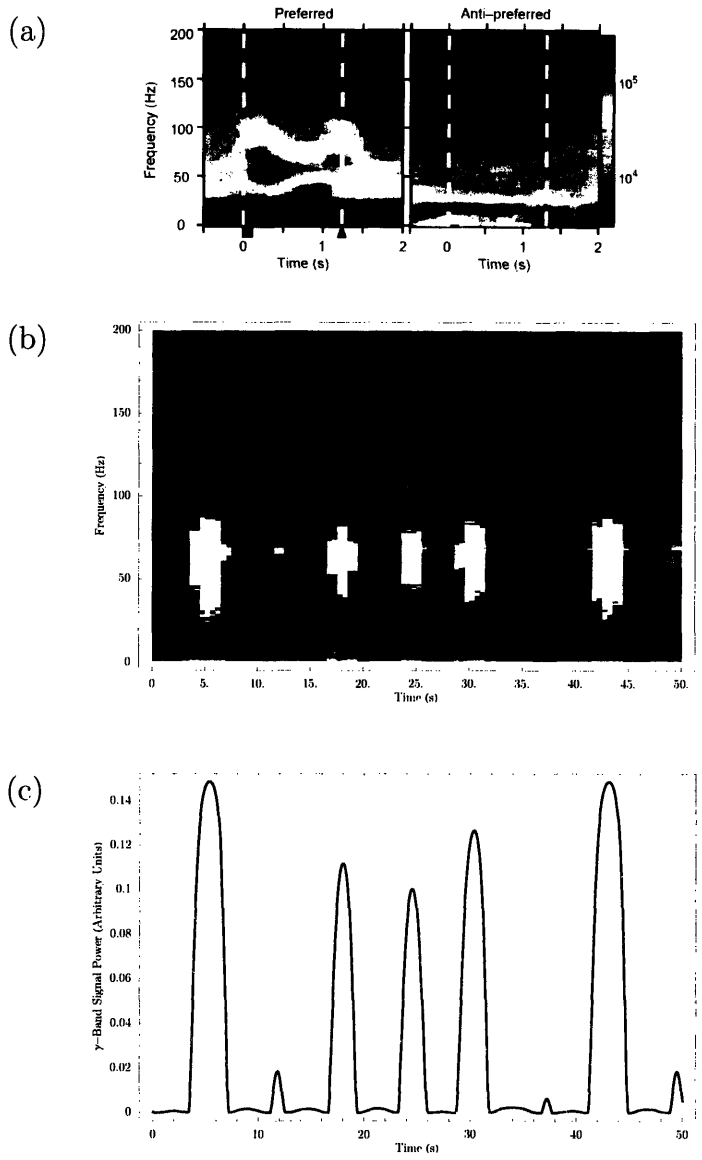



Figure 2-2: **Spectrograms of Recorded and Simulated Local Field Potentials from the Primate Motor Cortex During Arm Movement.** (a) Spectrogram of local field potentials recorded in the macaque intraparietal cortex during eye movement. Spectral activity in the $\gamma$-band (25–90 Hz) increases during the time interval between the dashed vertical lines, when a cue is present instructing the animal to anticipate moving its eyes in the preferred direction associated with the local field potential at a single site in the intraparietal cortex. Gamma-band activity is absent at the same site when the animal is cued for eye movement in the opposite direction. Similar neural tuning and spectral behavior are found in association with arm movement. Modified from [29]. (b) Spectrogram of simulated local field potentials associated with arm movement over an extended interval. Alternating intervals of activity and inactivity reflect periods in which the arm is or is not moved in a preferred direction associated with the neural site generating the simulated local field potentials. (c) Envelope curve describing the modulation of spectral power transmitted in the $\gamma$ pass band of the simulated local field potential.

of Figure 2-2. In the main set of performance tests, the $\gamma$-band power envelopes were modeled using a set of sinusoids with randomized amplitudes and phases and a constant offset term, and the corresponding waveforms were stored in the vector $\mathbf{N}(t)$. A random $m \times n$ matrix, $\mathbf{W}^*$, was then generated and used to construct a set of $m$ motor control parameters constituting the vector $\mathbf{M}(t)$. The vector $\mathbf{N}(t)$ was used as the input to the neural decoding algorithm, which was also permitted to observe $\mathbf{M}(t)$ during a learning period of variable length. Over this learning period, the algorithm sought to optimize an $m \times n$-dimensional convolution kernel internal to the decoding system, $\mathbf{W}$, of the form described in Section 2.3. The parameter $\epsilon$ was set to 0.1 during these simulations. The neural decoding algorithm generated consistently accurate results, reliably learning to produce an output that converged to $\mathbf{M}(t)$, when tested in this manner. In this sense the neural signal decoding scheme proved to be an effective algorithm for supervised on-line machine learning. The decoding algorithm and the simulations just described were implemented using *Mathematica* running on a 1.2 GHz Pentium processor (Intel). Qualitative results are presented in Figures 2-3, 2-4, and 2-5. A quantitative approach to evaluating the results is then presented and associated computations are summarized in Figure 2-6. The quantitative method facilitates a comparison of the neural decoding system presented here to others described in the context of neural prosthetics for limb paralysis, summarized in Table 2.1 and highlighted in Figure 2-7.

Figure 2-3 shows gradient-descent least-squares learning of a three-dimensional trajectory in real time, achieved by the neural decoding algorithm implemented as discussed in Section 2.3 in a typical test simulation of the form just described. The simulation was performed using $(n, m) = (11, 3)$ (where $n = 11$ corresponds to 10 sinusoids supplemented by a constant offset), so the motor control parameters are displayed in a three-dimensional plot, with $\mathbf{M}(t)$ normalized so that the trajectory is bounded by a unit cube. The input trajectory $\mathbf{M}(t)$ is plotted in gray over a learning interval $t \in [0, 40]$ s, while the decoded trajectory, $\tilde{\mathbf{M}}(t) \equiv \mathbf{W} \circ \mathbf{N}(t)$, is plotted in color over the same interval, both for visual clarity and to parameterize time in order to illustrate the convergence of $\tilde{\mathbf{M}}(t)$ to $\mathbf{M}(t)$ as $t$ increases over the learning interval.

As $t$ increases, $\tilde{\mathbf{M}}(t)$ evolves through red, orange, yellow, green, light blue, dark blue, violet, and finally magenta. Figure 2-3 illustrates qualitatively that $\tilde{\mathbf{M}}(t)$ converges toward $\mathbf{M}(t)$ reasonably quickly on the timescale set by full-scale variations in the trajectory.

Figure 2-4 shows the time evolution of three typical sets of parameters $\{W_{ij}^{(k)}\} = \{A_{ij}, \tau_{ij}\}$ corresponding to the filters associated with $W_{ij}$, the $ij$-component of $\mathbf{W}$, for three choices of $ij$. The plots of $A_{ij}(t)$ and $\tau_{ij}(t)$ illustrate two important phenomena. First, both $A_{ij}$ and $\tau_{ij}$ exhibit convergence to steady-state parameter values over the learning interval shown, as required for stability of the decoding system and as expected on the basis of the trajectory shown in Figure 2-3. Second, the range of values explored by $A_{ij}(t)$ and $\tau_{ij}(t)$ during learning is small, despite the random initialization of all $A_{ij}(t = 0)$ and $\tau_{ij}(t = 0)$. The reason for these small excursions is that the parameter space defining $\mathbf{W}$ has dimension $m \times n \times p$, whereas $\mathbf{W}^*$ is only $m \times n$-dimensional; furthermore, $n = 11 > m = 3$. The decoding system is evidently underconstrained and as a result many sets of parameters will generate accurate decoding performances. It is important to recognize that this underconstraining is not simply an artifact of the simplified neural encoding model used here to test the decoding system. Rather, real neural encoding is known to be highly redundant in the sense that populations of neurons store related information, and the time-dependent electrical activity waveforms of individual neurons within a population of neighboring cells convey a significant amount of mutual information [25]. This redundancy phenomenon facilitates decoding of neural signals but does not by any means trivialize the problem of implementing a practical system for real-time decoding of neural signals, particularly such a system that is constrained by a power budget that can ensure safe operation of the decoder when it is chronically implanted within the brain.

An important caveat for the decoding system involves constraints placed on the form of $\mathbf{N}(t)$ and $\mathbf{M}(t)$ as seen by the algorithm, and constraints on the values assumed by the filter parameters $W_{ij}^{(k)}$. The former constraints correspond to routine issues involving data normalization encountered in association with many machine learning

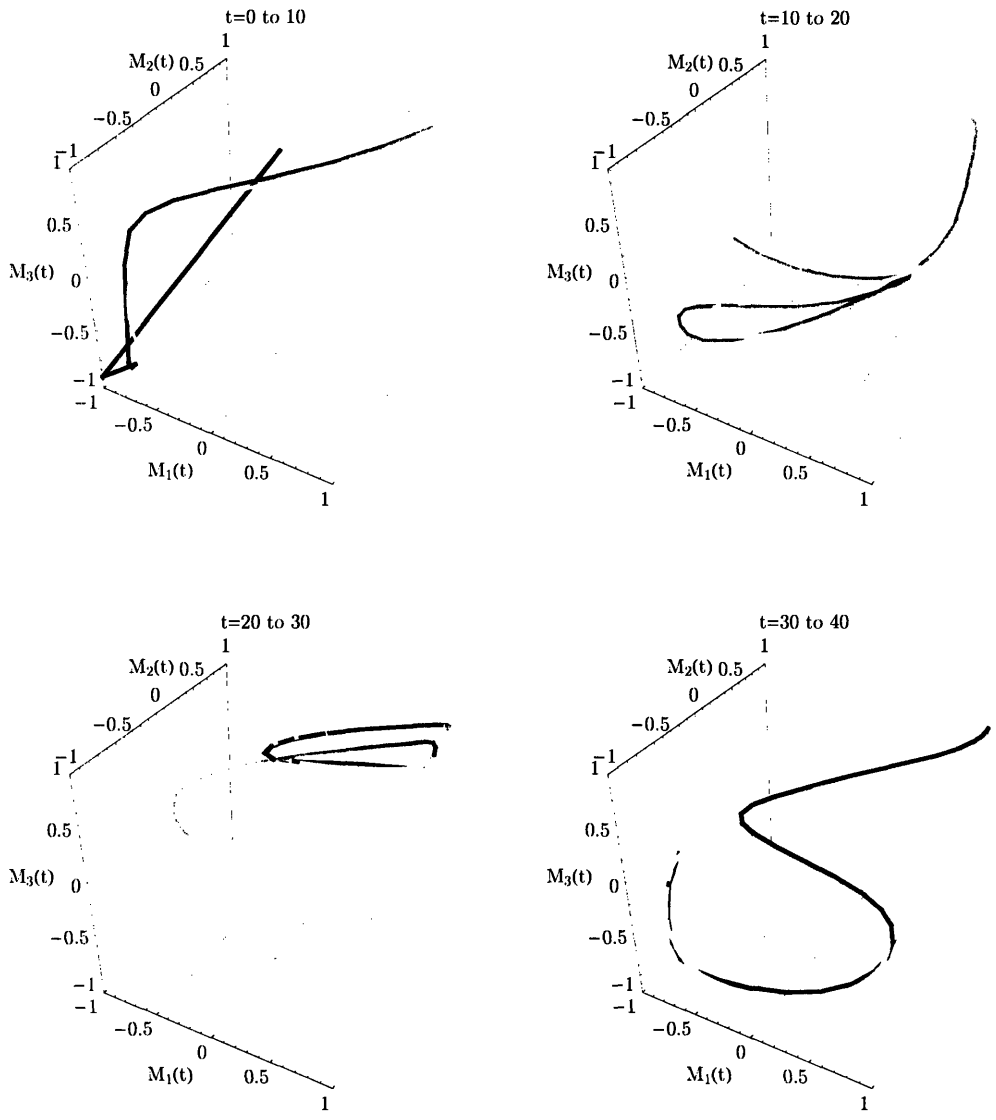



Figure 2-3: **Gradient-Descent Least-Squares Learning of a Three-Dimensional Trajectory in Real Time.** The input trajectory $\mathbf{M}(t)$ is plotted in gray over a learning interval $t \in [0, 40]$ s (divided into four segments of 10 s each for visual clarity), while the decoded trajectory, $\tilde{\mathbf{M}}(t)$, is plotted in color over the same interval, with time parameterized by the shifting of hue from red through orange, green, blue, violet, and finally magenta. $\tilde{\mathbf{M}}(t)$ converges toward $\mathbf{M}(t)$ reasonably quickly on the timescale set by full-scale variations in the trajectory.

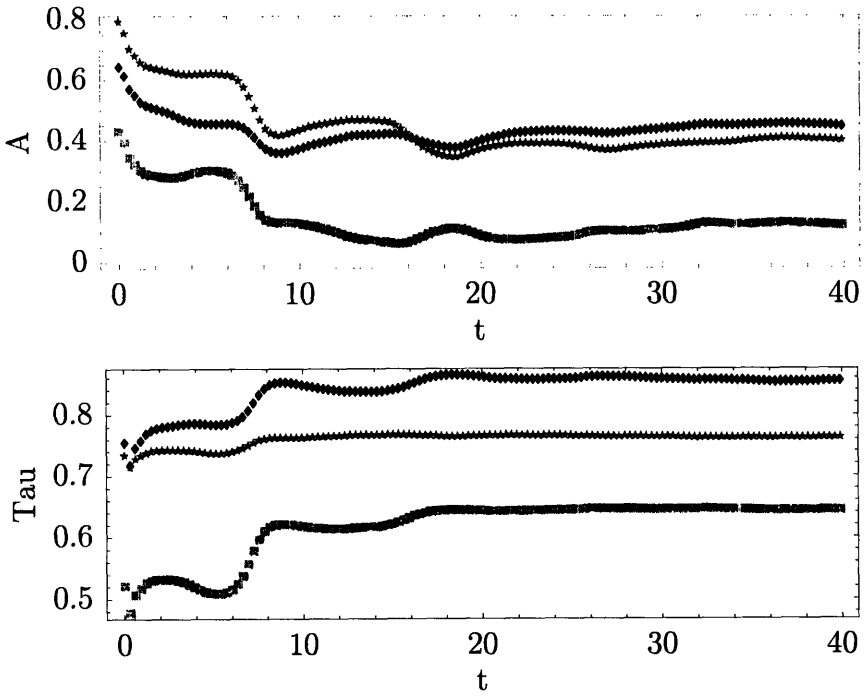



Figure 2-4: **Convergence of Filter Parameters During Learning.** The paired plots illustrate the time evolution of three typical sets of parameters $\{W_{ij}^{(k)}\} = \{A_{ij}, \tau_{ij}\}$, corresponding to the filters associated with $W_{ij}$, the $ij$-component of $\mathbf{W}$, for three choices of $ij$ (indicated by tracing color in the corresponding plots). $A_{ij}$ and $\tau_{ij}$ converge to steady-state parameter values over the learning interval.

systems [30]. In the simulations described here, $\mathbf{N}(t)$ and $\mathbf{M}(t)$ are transformed using a hyperbolic tangent function before being used as inputs to the decoding algorithm, so that the system sees $N_i(t)$ and $M_i(t)$ constrained to the interval $[-1, 1]$. This normalization was selected in part due to the convenience of its implementation in analog circuitry [22].

The constraints on $A_{ij}$ and $\tau_{ij}$ are more particular to the present implementation. These parameters will be constrained by physical device characteristics in an analog circuit-based implementation of the decoder, but in numerical simulations they need not necessarily be constrained. If they are not, however, conditions may arise in which $\tau_{ij}$ decreases during learning to a small positive value, making the ratios $\frac{1}{\tau_{ij}}$ and $\frac{A_{ij}}{\tau_{ij}}$, which appear in Equations 2.25 and 2.26 used to adapt $A_{ij}$ and $\tau_{ij}$, large in magnitude. Small fluctuations in $\tau_{ij}$ about zero will then cause the terms $\frac{1}{\tau_{ij}}$ and $\frac{A_{ij}}{\tau_{ij}}$ to oscillate rapidly between large positive and large negative values (oscillations of this form will be damped by the value of $\epsilon$). Such behavior can lead to a certain degree of instability in unconstrained systems. This potential for instability can be avoided by constraining the values of $A_{ij}$ and $\tau_{ij}$, for example by restricting $\tau_{ij}$ to positive values. Empirically, however, it was found that normalization of $\mathbf{N}(t)$ and $\mathbf{M}(t)$ was sufficient to ensure system stability and robust convergence of $\tilde{\mathbf{M}}(t)$ to $\mathbf{M}(t)$.

Figure 2-5 provides a final qualitative demonstration of the neural decoding system, comparing each $\tilde{M}_i(t)$ (plotted in black) to its corresponding $M_i(t)$ waveform (plotted in red) after a typical learning period has ended, feedback of $\mathbf{M}(t)$ into the system has stopped, and the filter parameters $W_{ij}^{(k)}$ have been fixed. The time intervals over which these waveform tracings are plotted is equal in length to the learning period, and the paired tracings show that the $\tilde{M}_i(t)$ continue to track the corresponding $\tilde{M}_i(t)$ in the absence of feedback. Traditional performance metrics for machine learning systems are typically based on the quality of estimations made by the machine learning system on unseen data after learning has ended. The qualitative comparisons illustrated in Figure 2-5 therefore suggest a quantitative method of evaluating the performance of the decoding algorithm. A scale-invariant and dimen-

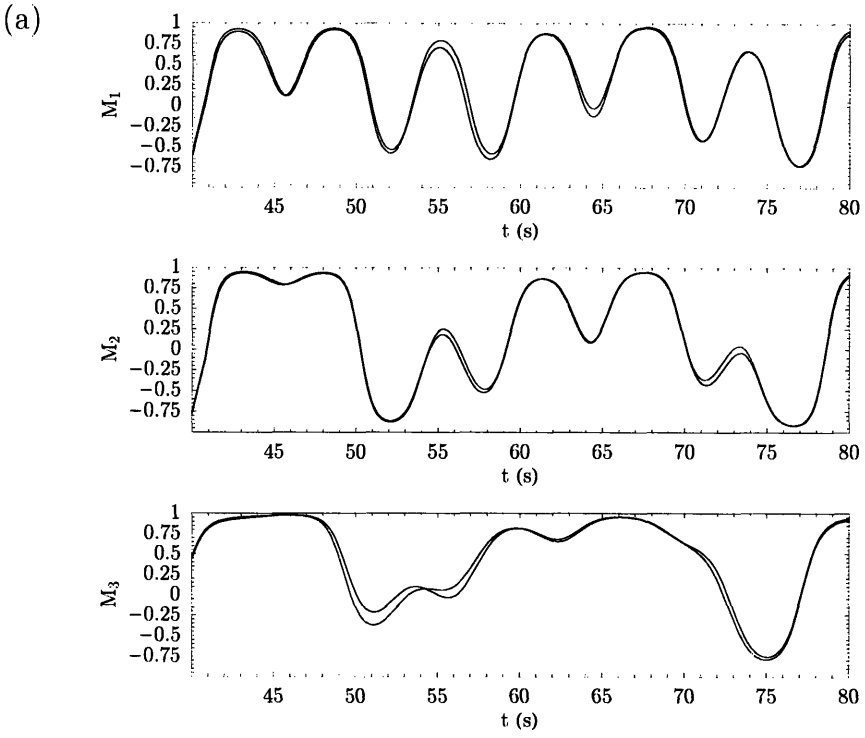



Figure 2-5: **Accurate Trajectory Prediction in the Absence of Feedback Confirms Effective Decoding.** A typical set ($i \in \{1, 2, 3\}$) of trajectory components $\tilde{M}_i(t)$ (plotted in black), estimated by the gradient-descent least-squares algorithm, are superimposed on their corresponding exact waveforms $M_i(t)$ (plotted in red). The time interval shown begins after the learning period has ended and extends for the same duration as the learning period. Feedback of $\mathbf{M}(t)$ into the system has therefore stopped and the filter parameters $W_{ij}^{(k)}$ have been fixed. The paired tracings show that the $\tilde{M}_i(t)$ continue to track the corresponding $\tilde{M}_i(t)$ in the absence of feedback, confirming the effectiveness of the decoding.

sionless figure of merit for decoding is the normalized mean squared error $\eta$, defined as

$$\eta \equiv \frac{1}{T} \int_{T}^{2T} dt \sum_{i=1}^{m} \left( \frac{M_i(t) - \tilde{M}_i(t)}{L_i} \right)^2 . \tag{2.27}$$

In the expression for $\eta$, $L_i$ denotes the length of the space available to the motor parameter trajectory in the $i$th dimension; that is, the maximum extent of excursions permitted to $M_i(t)$. The time $T$ denotes the length of the training interval. In the simulations presented here, hyperbolic tangent normalization scales $\mathbf{M}(t)$ to the unit $m$-cube, so $L_i = 2$. The quantity $\eta$ can be approximated by a sum of the form

$$\eta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{m} \left( \frac{M_i\left(T + \frac{n}{N}T\right) - \tilde{M}_i\left(T + \frac{n}{N}T\right)}{L_i} \right)^2 , \tag{2.28}$$

where a sum over $N$ time samples over the interval $[T, 2T]$ is used to approximate the integral in 2.27. Note that the average value of $\eta$ computed for any single $\tilde{M}_i(t)$, $i \in \{1, \ldots, m\}$, should be independent of $i$, so that $\eta = m\eta^{(1)}$, where the superscripted (1) indicates that $\eta$ has been computed for a single $\tilde{M}_i(t)$. These observations permit rough comparisons among performances reported in the literature for neural decoding systems operating on various scales and with different numbers of degrees of freedom, set by $L_i$ and $m$, respectively. It is possible to consider other figures of merit, including ones based on correlations between $\tilde{\mathbf{M}}(t)$ and $\mathbf{M}(t)$ rather than absolute error, but other authors have agreed that $\eta$-like figures of merit tend to reflect decoding system performance most reasonably [47].

Figure 2-6 presents the results of a set of computations of $\eta^{(1)}$ for the performance of the neural decoding system in simulations of the form described earlier in this section. The system was trained for intervals of varying length up to one minute, $T \in [0, 60\ s]$, and the value of $\eta^{(1)}$ was computed for each of 50 trials at each value of $T$. In a fraction $f$ of trials at each value of $T$, the decoding performance as reflected

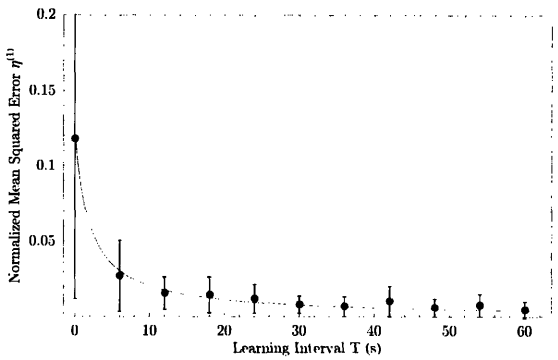



Figure 2-6: **Mean Squared Trajectory Prediction Error as a Function of Training Time for the Gradient-Descent Least-Squares Decoding Algorithm.** The gradient-descent least-squares decoding system was trained for intervals $T$ of varying length up to one minute, and the value of $\eta^{(1)}$, the normalized mean squared error, was computed for each of 50 trials at each value of $T$. The plot shows the mean value of $\eta^{(1)}$ as a function of the learning interval $T$, with error bars indicating one standard deviation. As expected, $\eta^{(1)}$ decreases rapidly as the training interval increases. (The data used to generate this plot exclude an outlying 10% of cases at each $T$ in which the system could be randomly reinitialized and retrained to generate markedly improved decoding results.)

by $\eta^{(1)}(T)$ was significantly worse than in the remaining fraction, $1-f$, of cases. In such cases markedly improved decoding, comparable to that achieved in the $(1-f)$-majority of cases, could be achieved simply by randomly reinitializing the parameters $W_{ij}^{(k)}$ and decoding again. The data presented in Figure 2-6 were obtained by setting $f = 0.1$. Figure 2-6 plots the mean value of $\eta^{(1)}$ as a function of the learning interval $T$, with error bars indicating one standard deviation. As expected, $\eta^{(1)}$ decreases rapidly as the training interval increases. At $T = 30$ s, for example, $\langle \eta^{(1)} \rangle = 0.0080 \pm 0.0077$, as compared with a baseline value of $\langle \eta^{(1)} \rangle = 0.118 \pm 0.106$ computed for an untrained system ($T = 0$) over 1000 trials.

These computations permit the simulated performance of the neural signal decoding system described here to be compared with the decoding ability of state-of-the-art decoding systems reported in the literature. One such system, described by Donoghue and colleagues, uses Kalman filtering and a Bayesian inference approach to neural signal decoding. This system has been tested using neural data previously recorded from the arm area of the primary motor cortices of macaque monkeys engaged in

| Decoding Method | $\eta^{(1)}$ |
|---|---|
| Population Vector | 0.011 |
| Linear Regression | 0.0038 |
| Kalman Filter | 0.0037 |
| Gradient-Descent Least-Squares | 0.0013 |

Table 2.1: **Comparing the Decoding Performance of the Gradient Descent Least Squares Technique to Results Obtained from Other Techniques.** The one-dimensional normalized mean squared prediction error, $\eta^{(1)}$, defined in Equation 2.27, can be used as a figure of merit to make rough comparisons among the decoding performances of various neural signal decoding systems. This table compares the results obtained using the gradient-descent least-squares algorithm in simulations described in this chapter, to three other decoding techniques described in [47].

two-dimensional arm movement tasks [47]. The experimental system described by Donoghue and colleagues has $m = 2$ and $L_x = L_y \approx 25$ cm [1]. The Kalman filter decoding technique obtains an unnormalized mean squared error of 4.66 cm$^2$, corresponding to $\eta^{(1)} \approx 0.0037$, after a training interval of $T = 150$ s. The same study also evaluated the performance of a linear-regression–based decoding technique and a population-vector–based technique in decoding the same experimental data and reported unnormalized mean squared errors of 4.74 cm$^2$ and 13.2 cm$^2$, respectively, for the two techniques, corresponding to $\eta^{(1)} \approx 0.0038$ for the linear regression method and $\eta^{(1)} \approx 0.011$ for the population vector method. By contrast, the neural decoding system described here yielded $\langle \eta^{(1)} \rangle \approx 0.0013$ for $T = 150$ in the simulations described in this section. These comparisons are summarized in Table 2.1.

Direct comparisons between the results reported by groups such as Donoghue and colleagues and those described here may not be entirely appropriate, as Donoghue and colleagues have tested their decoding methods in numerical experiments using real neural data, whereas the gradient-descent least-squares decoding technique described here has as yet only been used to track model trajectories encoded by simulated neural signals. However, the performance parameters listed in Table 2.1 can be thought of as

[1] While [47] states that the neural data used in the decoding experiments was obtained from macaques engaged in moving a manipulandum within a 25 cm ×25 cm workspace, the results presented in that study indicate that as much as 40−50% of the space available to the manipulandum in each direction is rarely if ever used during the trajectory-tracking trials. Normalizing $\eta$ to the smaller values that might appear appropriate on the basis of these reported results, however, could generate values of $\eta^{(1)}$ up to fourfold larger than the one reported here.

rough benchmark indicators of decoding performance, and in that sense they indicate that the gradient-descent least-squares decoding technique developed in this chapter compares favorably to state-of-the-art decoding techniques.

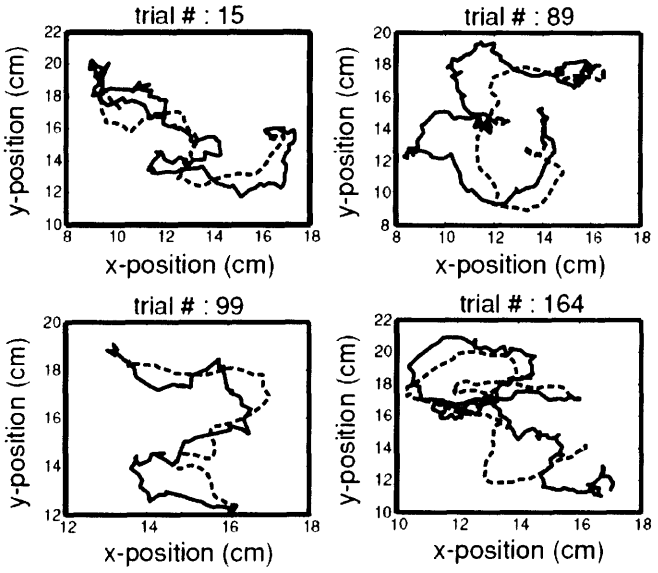
Despite the usefulness of figures of merit such as $\eta^{(1)}$ as performance summary, any single-valued performance parameter is likely to fall short of encapsulating the effectiveness of a particular decoding system. Figure 2-7 is therefore provided in order to facilitate a qualitative comparison of the gradient-descent least-squares technique against the state of the art as embodied by the Bayesian-inference–based Kalman-filter neural signal decoder of Donoghue and colleagues. Figure 2-7(a) reproduces the results published by that group [47]. In the terminology of this chapter, it superimposes $\mathbf{M}(t)$ (dashed lines) on $\tilde{\mathbf{M}}(t)$ (solid lines) as estimated by the Kalman filter over time intervals of 3.5 s. By contrast, Figure 2-7(b) shows a three-dimensional trajectory decoded by the gradient-descent least-squares neural decoding system described in this chapter, as modeled in the simulations discussed in this section. The gray trajectory indicates $\mathbf{M}(t)$ while the colored trajectory represents $\tilde{\mathbf{M}}(t)$. The shifting color of $\tilde{\mathbf{M}}(t)$ from red to green indicates time evolving from $T = 40$ s to $2T = 80$ s. The successful performance of the gradient-descent least-squares neural signal decoding system is apparent from the faithfulness with which $\tilde{\mathbf{M}}(t)$ reconstructs $\mathbf{M}(t)$.

## 2.5 Performance of Linear Decoding Algorithm on Data Obtained from Neural Recordings During Animal Behavior Trials

This section discusses a set of experimental tests of the neural signal decoder described and simulated in the previous sections, using data recorded from the parietal cortex of a live macaque monkey [2]. Arrays of recording electrodes had been surgically im-

[2]The experimental data used to perform the tests described in this section were very graciously provided by Professor Richard Andersen and Dr. Sam Musallam of the Division of Biology at the California Institute of Technology.

(a)




(b)

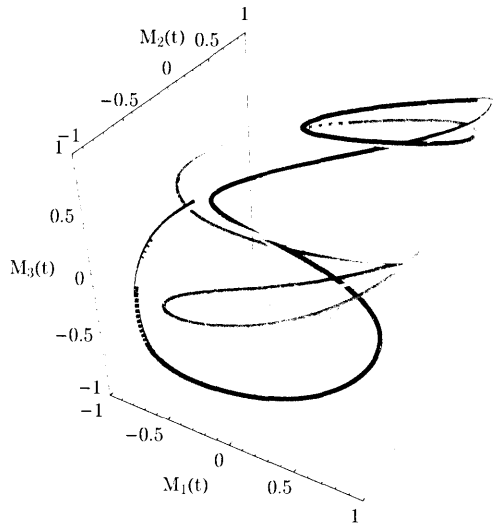



Figure 2-7: **Comparing the Simulated Performance of the Gradient-Descent Least-Squares Neural Decoding System to the Performance of a State-of-the-Art System.** (a) Results generated by the Bayesian-inference–based Kalman-filter neural signal decoder of Donoghue and colleagues. The plots superimpose $\mathbf{M}(t)$ (dashed lines) on $\tilde{\mathbf{M}}(t)$ (solid lines) as estimated by the Kalman filter over time intervals of 3.5 s. From [47]. (b) A simulated three-dimensional trajectory decoded by the gradient-descent least-squares neural decoding system described in this chapter. The gray trajectory indicates $\mathbf{M}(t)$ while the colored trajectory represents its accurate reconstruction by $\tilde{\mathbf{M}}(t)$. The shifting color of $\tilde{\mathbf{M}}(t)$ from red to green indicates time evolving after the learning period has ended, from $T = 40$ s to $2T = 80$ s.

planted at several cortical sites in this monkey, and the monkey had also been trained to perform a standard stimulus-response task involving arm-reaches to specific points in a plane. Neurons in the parietal cortex have been implicated in encoding motor function at the 'planning' or 'intention' stages [38], and so the purpose of these experiments was to explore the possibility of predicting arm movements before they were made, by learning to decode arm-movement intentions from the electrical activity of neurons in the parietal cortex. One can imagine many potential uses for a system capable of making such 'thought-reading' predictions, including 'neural prosthetic' devices that would enable paralyzed people to gain thought-based control of robotic artificial limbs. The results of these experiments have been published [25], and along with other related studies, some of which are cited in other sections, they indicate that accurate real-time decoding of neural signals is possible. However, all neural decoding systems reported to date, including the one used by the investigators in this experiment, have been implemented in software and are consequently unsuitable for practical use in a biologically implantable device. The results of this section, encapsulated in Figures 2-10, 2-12, and 2-13, indicate that the gradient-descent approach to adaptive-filter decoding can match the performance of state-of-the-art, software-based neural decoding systems; as discussed in Section 2.6, this approach has the advantage of being implementable in miniature, micropower electronic circuits suitable for long-term implantation in the brain. The present section is devoted to characterizing the performance of the neural signal decoding algorithm in experiments involving real neural data.

The behavioral task performed by the experimental monkey was structured as follows [25]. The monkey initiated each iteration of the task by touching a central cue point and looking at a nearby visual fixation point at $t = -800$ ms (its gaze was tracked using an eye-tracking device, and cues were presented on a touch-sensitive computer monitor). After a delay of 500 ms a peripheral cue was flashed from $t = -300$ to $t = 0$ ms at one of four target locations toward the top, right, bottom, or left edge of the screen. The monkey was rewarded if it reached to the indicated target at the end of a memory period of $1500 \pm 300$ ms (the reward consisted of

a calibrated sip of juice). The electrical activity of neurons in the parietal cortex of this monkey was monitored over the course of each trial, and action potentials ('spikes') were recorded from each of 54 isolated neurons. The resulting 'spike train' waveforms were converted to mean firing rates over the segment of the memory period from $t = 200$ to $t = 1100$ ms by counting the number of spikes produced by each neuron during that interval. These mean firing rates were used as the inputs to the neural decoder: $\mathbf{N}(t)$ in the notation of this chapter. (The beginning and end of each memory interval were omitted because these intervals sometimes contain residual neuronal activity corresponding to actual arm movement, whereas the object of the study was to determine how effectively intention-related neuronal activity could be used to predict future arm movements.) The motor output $\mathbf{M}(t)$ was defined as the two-dimensional position of the target to which the monkey reached at the end of a corresponding memory period.

The precise constructions used for $\mathbf{N}(t)$ and $\mathbf{M}(t)$ in the tests described in this section are as follow. Each iteration of the reach task has a single associated $\hat{\mathbf{N}}$, where $\hat{\mathbf{N}}$ is an $(n = 54 + 1)$-dimensional vector, each of whose first $n - 1$ components $\hat{N}_i$, $i \in \{1, \ldots, n - 1 = 54\}$ is the average firing rate of the $i$th neuron over the $t = 200$ to $t = 1100$ ms interval of that reach iteration. The final component, $\hat{N}_n = 1$, contains a constant offset provided by the system. The actual $\mathbf{N}$ used as the input to the decoder was a transformed version $\hat{\mathbf{N}}$, with each component rescaled according to

$$N_i = \frac{\hat{N}_i - \langle \hat{N}_i \rangle}{\max \hat{N}_i}, \tag{2.29}$$

where the mean indicated by the angled brackets and the maximum appearing in the denominator of Equation 2.29 are computed from the first $4s_0$ samples of $\hat{\mathbf{N}}$ used during the training period of the decoder ($s_0$ was typically fixed between 2 and 10). The means and maxima were computed in this way to facilitate simulations, but in principle they could be updated dynamically through the use of a moving window, and

such a scheme might be more convenient to implement in analog circuitry as it would not require long-term storage of any individual values of $\langle \hat{N}_i \rangle$ and $\max \hat{N}_i$. Neural recordings were made separately for each reach, and reaches in various directions were performed in a random sequence. So since the sample of $\mathbf{N}$ corresponding to a particular reach lasts for $w \equiv \Delta t = 1100 - 200 = 900$ ms, piecewise-constant input waveforms $\mathbf{N}(t)$ can be constructed by choosing a random subset $\{R_{j_1}, \ldots, R_{j_{4s_1}}\}$, $4s_0 < 4s_1 \leq r$, of reaches from the complete set $\{R_j\}$, $j \in \{1, \ldots, r\}$ of $r$ reaches made by the monkey over the entire course of the experiment. (The meaning of the double subscript indices is as follows: The first subscript $j \in \{1, \ldots, r\}$ indexes the full set of experimental reaches, while $j_1, \ldots, j_{s_1}$ indicate a subset of $s_1$ choices of the value of the index $j$). $\hat{\mathbf{N}}(R_j)$ then refers to the average firing rates of the observed neurons during the 200–1100 interval of the memory period corresponding to reach $R_j$. The waveform $\hat{\mathbf{N}}(t)$ is then defined to be constant on intervals of length $\Delta t = w$, with

$$\hat{N}_i(t) \equiv \hat{N}_i(R_{j_k}),\ (kw - 1) \leq t \leq kw,\ k \in \{1, \ldots, s_1\}, \tag{2.30}$$

and $N_i(t)$ is obtained from $\hat{N}_i(t)$ according to Equation 2.29. In other words, each component $N_i(t)$ of $\mathbf{N}(t)$ is piecewise-constant over time windows of length $w$, with the value of $N_i$ over each window defined by the rescaled average firing rate of the $i$th neuron during the memory period preceding a particular reach. The values of $\mathbf{M}(t)$ indicate the direction of the reach corresponding to $\mathbf{N}(t)$. In this experiment, $\mathbf{M}$ is a two-dimensional vector assuming one of four discrete values, encoded as follows:

| Direction | $M_1$ | $M_2$ |
|---|---|---|
| Up | +1 | +1 |
| Right | −1 | +1 |
| Down | +1 | −1 |
| Left | −1 | −1 |

where analog outputs generated by the decoder were thresholded according to

$$\tilde{M}_i \rightarrow \mathrm{sgn}\tilde{M}_i, \tag{2.31}$$

so that positive and negative outputs were interpreted as $+1$ and $-1$, respectively.

The piecewise-constant form of the input signals for the neural decoder reflects a qualitative difference between the decoding problem in this experiment, which requires the decoding system to make a series of decisions from among a finite set of options, and the decoding problem framed in Section 2.2 and simulated in Section 2.4, which requires the neural decoder to estimate a smooth trajectory as a function of time. While the gradient-descent least-squares approach is applicable to both kinds of problem, the convolution kernel chosen to implement the neural decoder, $W_{ij} = \frac{A_{ij}}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}}$, is designed to exploit the predictive value of past input signals. The degree to which past inputs $\mathbf{N}(t' < t)$ have predictive value is reflected by the value of the time constant $\tau_{ij}$, and as $\tau_{ij} \rightarrow 0$ the time interval over which $\mathbf{N}(t' < t)$ contribute significantly to the present-time output $\mathbf{M}(t)$ correspondingly vanishes. In this experiment the signal to be decoded corresponds to a time series of discrete, uncorrelated decisions made every $\Delta t = w$. Consequently, $\mathbf{N}(t')$ is completely uncorrelated from $\mathbf{N}(t)$ and $\mathbf{M}(t)$ for $|t - t'| \geq w$. (In concrete terms, since successive reaches are independent, neural activity preceding one reach contains no predictive information concerning the direction of the next reach.) As a result, effective decoding requires $\tau_{ij} \ll w$. More precisely, if the decoder is to predict an accurate value of $\tilde{\mathbf{M}}(t)$ by the end of the memory period of length $w$, that value must be independent of inputs from the preceding memory period, and so the convolution kernel must suppress inputs from the preceding memory period sufficiently to ensure that

$$W_{ij}(t+w) \ll W_{ij}(t) \tag{2.32}$$

$$\frac{A_{ij}}{\tau_{ij}} e^{-\frac{t+w}{\tau_{ij}}} \ll \frac{A_{ij}}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}} \tag{2.33}$$

$$e^{-\frac{w}{\tau_{ij}}} \ll 1. \tag{2.34}$$

In the limit as $\tau_{ij} \to 0$, the system becomes an 'instantaneous linear decoder' in the sense that the convolution performed on the input signals becomes a matrix multiplication:

$$\lim_{\tau_{ij}\to 0\,\forall j} \tilde{M}_i(t) = \lim_{\tau_{ij}\to 0\,\forall j} \sum_{j=1}^{n} W_{ij}(t) \circ N_j(t) \tag{2.35}$$

$$= \lim_{\tau_{ij}\to 0\,\forall j} \sum_{j=1}^{n} \int_{t-\delta}^{t} \frac{A_{ij}}{\tau_{ij}} e^{-\frac{t-u}{\tau_{ij}}} N_j(u) du \tag{2.36}$$

$$= \lim_{\tau_{ij}\to 0\,\forall j} \sum_{j=1}^{n} \frac{A_{ij}}{\tau_{ij}} N_j(t) \int_{t-\delta}^{t} e^{-\frac{t-u}{\tau_{ij}}} \tag{2.37}$$

$$= \lim_{\tau_{ij}\to 0\,\forall j} \sum_{j=1}^{n} \left\{ \frac{A_{ij}}{\tau_{ij}} N_j(t) \tau_{ij} \left[ e^{-\frac{x}{\tau_{ij}}} \right]_{x\equiv w-u=\delta}^{0} \right\} \tag{2.38}$$

$$= \lim_{\tau_{ij}\to 0\,\forall j} \sum_{j=1}^{n} \frac{A_{ij}}{\tau_{ij}} N_j(t) \tau_{ij} \left( 1 - e^{-\frac{\delta}{\tau_{ij}}} \right) \tag{2.39}$$

$$= \sum_{j=1}^{n} A_{ij} N_j(t). \tag{2.40}$$

Here the parameter $\delta$ is the time constant of a single decoder module and represents a characteristic timescale over which the filter parameters are updated, which must satisfy $\tau_{ij} \ll \delta < w$ so that the filter parameters can adapt within each time window. The physical significance of $\delta$ is indicated in Equation 2.54 of Section 2.6, which discusses the circuit implementation of the decoding system. Removal of $N_j(t)$ from the integral in Equation 2.37 is justified in this experiment because $N_j(t)$ is piecewise-constant and does not change on the time interval $[t-\delta, t]$ considered in the integral.

More generally, however, that algebraic step is justified because values of $N_j(u \neq t)$ are exponentially suppressed by the convolution kernel in the $\tau_{ij} \rightarrow 0$ limit.

Because the decoding task in this experiment involved a time series of uncorrelated inputs, the time constants $\tau_{ij}$ were initialized to zero and not modified during the training interval. Therefore, in the sense of Equation 2.40, the decoding scheme as applied to the reach-intention neural data reduces to an instantaneous linear decoder, analogous to a single-layer perceptron implemented with continuous-time feedback.

The neural data used in the experiments reported here were obtained by Andersen and colleagues, who have described the performance of their own method of decoding the recorded signals [25]. This technique, implemented entirely in software, involved an analysis of variance to preselect a subset of the identified neurons having optimal directional tuning, followed by Bayesian inference on the mean firing rates and first several Haar wavelet coefficients of the signals obtained from the selected neurons. The opportunity to test the gradient-descent least-squares approach to adaptive-filter decoding on the same neural data used by Andersen and colleagues legitimizes a more direct comparison with state-of-the-art neural decoding systems than was possible in the discussion of Section 2.4, in which simulations of continuous-trajectory decoding were compared with related results obtained by Donoghue and colleagues. The principal performance measure reported by Andersen and colleagues is a 64.4% success rate in predicting the correct one of four allowed reach directions. Under corresponding training conditions the neural decoding system described in the present work generated accurate predictions in $65 \pm 9\%$ of trials (the uncertainty preceded by the $\pm$ indicates the magnitude of one standard deviation). As discussed in the remainder of this section, decoding performance depends on a number of modifiable parameters and can be improved over the initially quoted success rates. These benchmark figures are provided at the outset in support of the idea that circuit-based adaptive-filter decoding can match the performance of even elaborate, software-based neural decoding algorithms.

Figures 2-8 (a) and (b) display the performance of the decoder during and after training, respectively. Figure 2-8 (a) plots the direct output of the decoder as a

black line for each component $\tilde{M}_i(t)$, $i \in \{1, 2\}$, and the thresholded value of $\tilde{M}_i(t)$ indicated in Equation 2.31 is plotted as a filled blue square at time intervals equal to the system update constant $\delta$ (the zero-crossing threshold is indicated by a horizontal line across each plot). The correct value, $M_i(t)$, is plotted at each time point as an open red square, so that correct predictions appear at each time point as blue boxes with red borders, whereas empty red boxes indicate incorrect predictions. A correct prediction of $\mathbf{M}(t)$ requires $\tilde{M}_1(t) = M_1(t)$ and $\tilde{M}_2(t) = M_2(t)$, a condition indicated graphically by vertically aligned pairs of red-ringed blue boxes; the ratio of ringed to unringed blue boxes increases as the training period progresses, indicating (along with the changing shape of the black waveform) the adaptation occurring during that interval. The jagged shape of the waveform is a consequence of $\delta < w$, as required for feedback to enable adaptation during the memory interval preceding each reach movement (in the simulation shown $w = 4\delta$). Figure 2-8 (b) consists of a similar pair of plots showing $\tilde{M}_i(t)$ over an interval of equal length immediately following another training period. In the absence of feedback, the waveforms $\tilde{M}_i(t)$ are piecewise-constant because the $N_i(t)$ signals they decode are piecewise constant.

In the $\tau_{ij} \to 0$ limit of the instantaneous linear decoder, the two-valued predictions ($\pm 1$ in each direction after thresholding) for each $\tilde{M}_i$ facilitate the use of a geometric construction to visualize the function and performance of the decoding system. Suppose $\mathbf{N}(t)$ is interpreted as the position of a point in the $(n - 1 = 54)$-dimensional space of transformed mean firing rates. Then the decoding system can be understood in geometric terms as adaptively learning to draw one $(n - 1 = 54)$-dimensional hyperplane in $N_j$-space for each $M_i$ that partitions the set of $\mathbf{N}(t)$ into a set of spatial regions, each of which contains points corresponding to a single decision. In this experiment there are four such regions, corresponding to arm movement in each of the allowed directions ('Up,' 'Down,' 'Right,' or 'Left'). The information presented to the decoder during training can be interpreted as consisting of a sequence of points $\mathbf{N}(t)$ and corresponding statements for each component $i$ as to whether $\mathbf{N}(t)$ lies above or below the $i$th hyperplane. Learning can be understood geometrically as a process of adjusting the position of the hyperplanes to accommodate this information as it

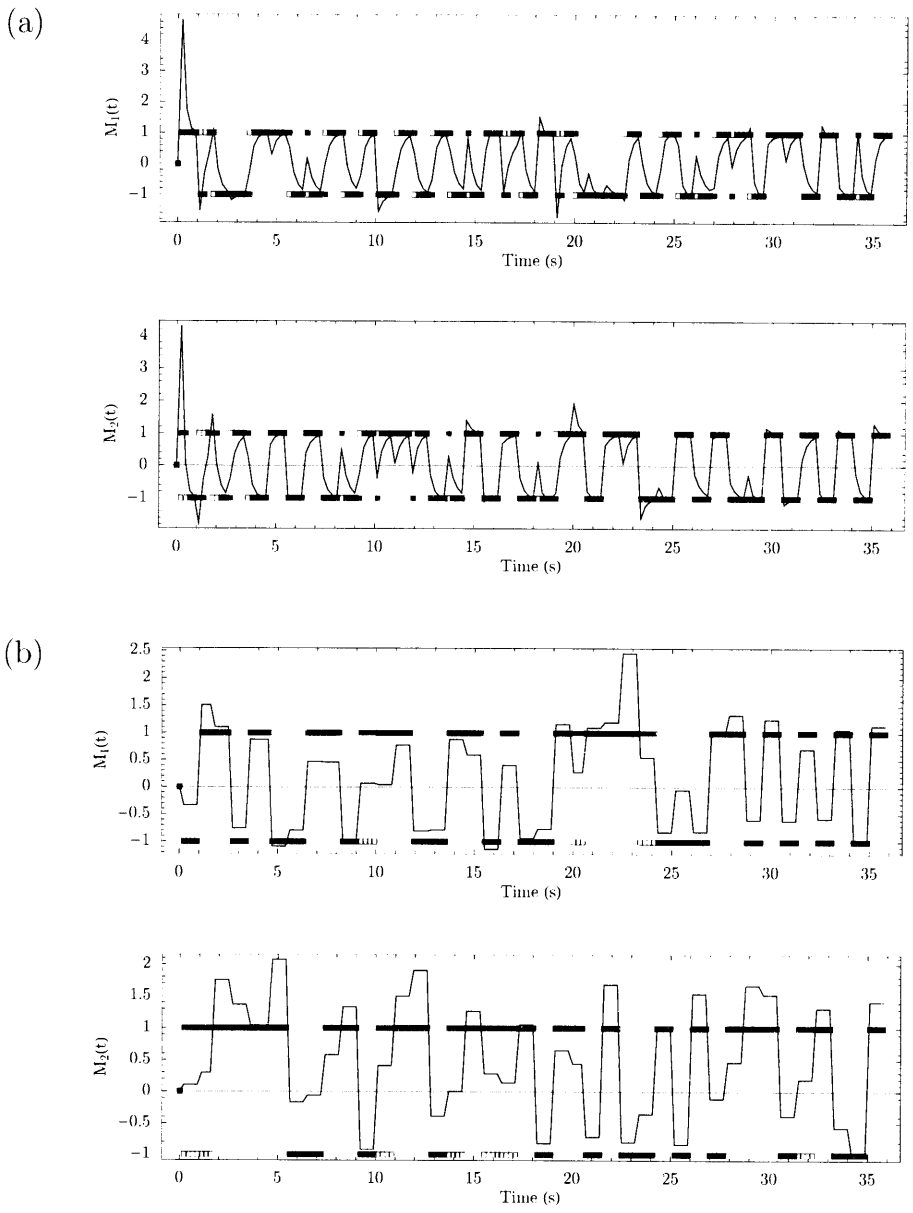



Figure 2-8: **Output Waveforms from the Neural Decoder During and After Training on Experimental Neural Recordings from the Parietal Cortex of a Macaque Engaged in an Arm-Movement Task.** The direct output of the decoder is plotted as a black line for each component $\tilde{M}_i(t)$, $i \in \{1, 2\}$, and the thresholded value is plotted as filled blue squares (the zero-crossing threshold is indicated by a horizontal line across each plot). The correct value, $M_i(t)$, is plotted at each time point as an open red square, so that correct predictions appear at each time point as blue boxes with red borders, whereas empty red boxes indicate incorrect predictions. A correct prediction of $\mathbf{M}(t)$ requires $\tilde{M}_1(t) = M_1(t)$ and $\tilde{M}_2(t) = M_2(t)$, indicated graphically by vertically aligned pairs of red-ringed blue boxes. (a) Output During Training. The ratio of ringed to unringed blue boxes increases as the training period progresses, indicating (along with the changing shape of the black waveform) the adaptation occurring during that interval. (b) Output After Training.

is presented. Since the decision threshold for the decoder output $\tilde{\mathbf{M}}(t) = A_{ij}\mathbf{N}(t)$ is zero, the $i$th hyperplane is defined by the equation

$$0 = \sum_{j=1}^{n-1} A_{ij}x_j + A_{in}, \tag{2.41}$$

where the $x_j$ correspond to coordinates in the firing-rate space occupied by the values of $\mathbf{N}(t)$, and $j \in \{1, \ldots, n-1\}$ indexes the observed neurons. In this geometric interpretation, the role of the extra degree of freedom, the constant offset having index $j = n$, is to free the hyperplane decision boundaries from having to pass through the origin. Figure 2-9 illustrates the performance of the decoding system using this geometric construction. The figure shows a three-dimensional subspace of the full 54-dimensional $\mathbf{N}(t)$-space, defined by the firing rates of the three neurons ($j = 15$, $j = 34$, and $j = 40$) with the largest values of $\sum_i A_{ij}^2$, the $A_{ij}$ having been set through the standard learning period used in the tests reported here and by Andersen and colleagues, which contained 30 reaches in each direction. Values of $\mathbf{N}(t)$ are plotted in this subspace and color-coded according to their corresponding reach directions. The sections of the $i = 1$ and $i = 2$ hyperplanes corresponding to the $(j = 15, 34, 40)$ subspace is shown as well, and the figure demonstrates the manner in which these hyperplanes function as decision boundaries for the decoder. The observed values of $\mathbf{N}(t)$ form diffuse clouds of points, with each cloud predominantly containing points associated with arm movement in one of the allowed directions. The $i = 1$ and $i = 2$ hyperplanes partition these clouds into different regions of space. A remarkable property of the decoding algorithm is its ability to learn effective placement for the decision boundaries when the data are presented as time series in on-line learning experiments. That is, the system only receives information about one point at a time, and must reposition randomly initialized hyperplanes by feedback-guided iterative perturbation.

Interestingly, in almost any chosen low-dimensional subspace, the $i = 2$ hyperplane separating 'Up' and 'Right' (plotted as red and blue points, respectively) from 'Down'

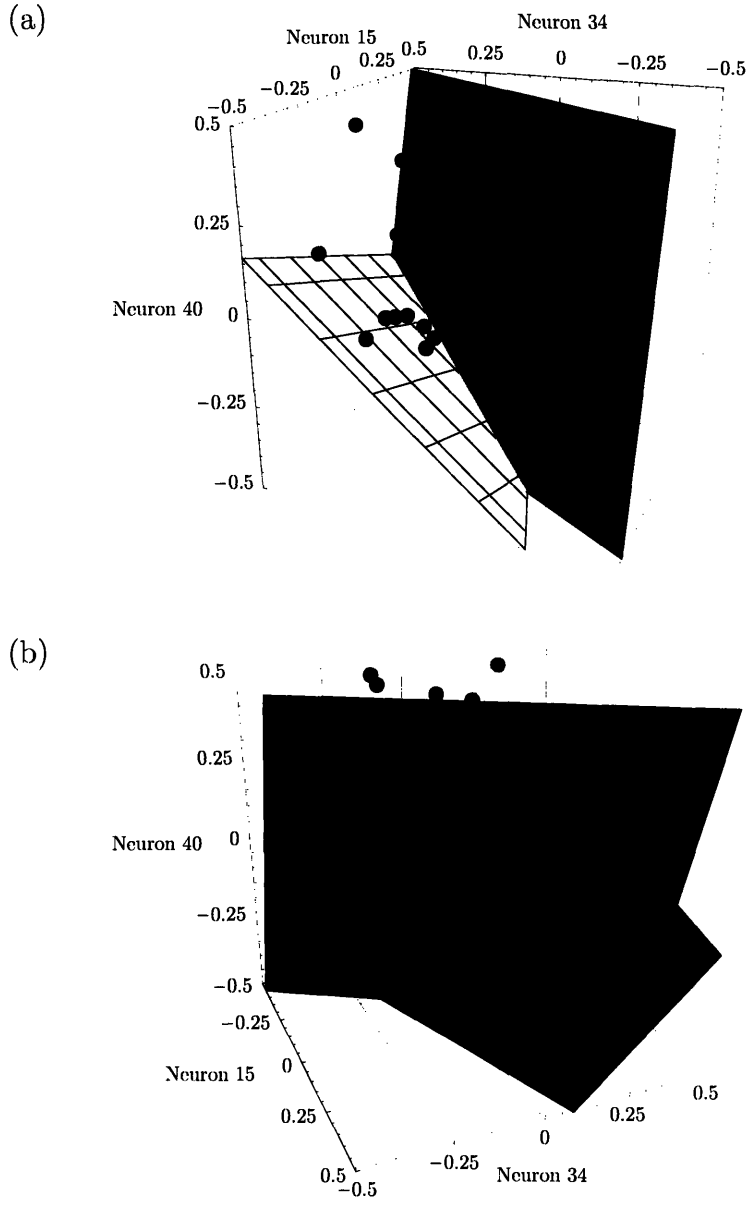



Figure 2-9: **Hyperplane Decision Boundaries Optimized by the Neural Decoding System.** This figure illustrates the geometric interpretation of the performance of the decoding system discussed in the text. It shows a three-dimensional subspace of the full 54-dimensional $\mathbf{N}(t)$-space, defined by the firing rates of the three neurons ($j = 15$, $j = 34$, and $j = 40$) with the largest values of $\sum_i A_{ij}^2$ at the end of a learning period. Values of $\mathbf{N}(t)$ are plotted in this subspace and color-coded according to their corresponding reach directions: Up (red), Right (blue), Down (black), Left (green). The sections of the $i = 1$ and $i = 2$ hyperplanes corresponding to the $(j = 15, 34, 40)$ subspace is shown as well, illustrating the manner in which these hyperplanes function as decision boundaries for the decoder, partitioning clouds of points corresponding to different reach directions into different regions of space.

and 'Left' decisions (plotted as black and green points, respectively) provides a more robust classification than does the hyperplane for $i = 2$ (which distinguishes). This phenomenon is illustrated in Figure 2-9, of which Part (a) shows the $\tilde{M}_2 = +1$ ('Up' and 'Right') side and Part (b) shows the $\tilde{M}_2 = -1$ ('Down' and 'Left') side of the $i = 2$ hyperplane. It is difficult to find a low-dimensional subspace in which the decoder robustly classifies the $M_1$-components. If this is a meaningful observation, its significance is more likely to be biological than algorithmic, perhaps reflecting an asymmetry in neural coding, or alternatively suggesting a bias (possibly due to neuroanatomic location) within the population of neurons isolated by the implanted recording electrodes. Apart from illustrating the greater accuracy with which $M_2$-values can be predicted by depicting the learned positions of the $M_1$ and $M_2$ decision boundaries, Figure 2-9 suggests that the clouds of $\mathbf{N}(t)$-points, which are not plane-separable in the subspace shown, may also not be hyperplane-separable. While some of the ambiguity indicated by the cluster overlap shown in Figure 2-9 is resolved by separation along axes not shown in the figure, perfect hyperplane-separation of the data does not generally seem possible. These two observations can be illustrated more quantitatively by examining the confusion matrix, shown in Table 2.2, for the training set of $\mathbf{N}(t)$-points plotted in 2-9 along with the decision boundaries fixed by the $A_{ij}$ learned during their corresponding training period. The diagonal elements in the confusion matrix of Table 2.2 indicate correct predictions. An examination of the off-diagonal elements, corresponding to incorrect predictions, reveals increased uncertainty in $M_1$ relative to $M_2$. In the experiment illustrated in Figure 2-9 and Table 2.2, the increase is 9.2-fold. Averaging over randomized initial conditions and different selections of the training set revealed that after learning, errors in $M_1$ (confusing Up with Right or Down with Left) are 5.3 times more likely than errors in $M_2$: this ratio is less than 1 at a significance level of $p < 10^{-7}$, and the 99% confidence interval for $\left\langle \frac{p(\tilde{M}_1(t) \neq M_1(t))}{p(\tilde{M}_2(t) \neq M_2(t))} \right\rangle$ is $(3.8, 7.3)$.

Figure 2-10 shows the dependence of decoding performance on the size of the training set used by the adaptive filters to optimize their $A_{ij}$. Each data point represents an average over random initializations of $A_{ij}$ and different subsets $\{R_{j_1}, \ldots, R_{j_{4s_1}}\}$

| | $\mathbf{M}$ = Up | $\mathbf{M}$ = Right | $\mathbf{M}$ = Down | $\mathbf{M}$ = Left |
|---|---|---|---|---|
| $\tilde{\mathbf{M}} = (+1, +1)$ Up | 112 | 3 | 1 | 3 |
| $\tilde{\mathbf{M}} = (-1, +1)$ Right | 5 | 117 | 0 | 0 |
| $\tilde{\mathbf{M}} = (+1, -1)$ Down | 1 | 0 | 113 | 8 |
| $\tilde{\mathbf{M}} = (-1, -1)$ Left | 1 | 0 | 7 | 109 |

Table 2.2: **Confusion Matrix.** This confusion matrix permits a more quantitative assessment of the accuracy of the classification shown graphically in Figure 2-9. Diagonal elements correspond to correct decisions, while off-diagonal elements correspond to incorrect decisions by the decoder. Statistical analysis of the elements of such matrices obtained from repeated trials permits a comparison of the different kinds of errors made by the decoder, such as the increased tendency to confuse err in $\tilde{M}_1$ relative to $\tilde{M}_2$.

with indices $j \in \{1, \ldots, r\}$ and constructed so that each training subset contained $s_1$ trials, and each trial consisted of a reach in one of the four possible directions, with the directions ordered randomly. Signals from all 54 isolated neurons were decoded to make the corresponding predictions of reach direction. As expected, decoding performance improves with increased training. The plot also illustrates several subtler points concerning decoding system performance. First, only a small amount of training is required to generate predictions significantly more reliable than chance. (The horizontal line across the plot indicates the 0.25 threshold corresponding to unbiased guessing.) In particular, after only five trials (18 seconds of training), the mean prediction accuracy is $37 \pm 11\%$, which is greater than chance at a statistical significance level of $p < 0.14$. The statistical significance of greater-than-chance decoding performance improves rapidly with increased training, falling to $p < 0.06$ after 10 trials (36 seconds of training) and $p < 0.04$ after 20 trials, as decoding performance increases to $58 \pm 21\%$ and $59 \pm 19\%$, respectively. Second, the marginal benefit of additional training declines rapidly, and the mean prediction accuracy after 30 trials is within 6% of the accuracy achieved after 60 trials. Third, prediction accuracy is highly variable, particularly for short learning intervals. One source of this variability is the random initialization of the parameters $A_{ij}$. When the random initialization places the $A_{ij}$ far from their collective optimum in parameter space, the convergence rate of the learning algorithm may not be sufficiently fast to ensure complete convergence

in less than the time set by the learning interval. A second source of variability may be intrinsic to the neural system. Andersen and colleagues also conducted smaller sets of trials in which their experimental monkeys were permitted to reach to one of six or eight targets. In these experiments, just as in the four-target trials, their typical decoding success rate was $65 \pm 2\%$. These equivalent performances shown in three decoding problems of graded difficulty suggest that a factor other than the decoding algorithm itself is limiting the accuracy of decoding. That two different decoding approaches, the one used by Andersen and colleagues and the one describe here, yield similarly imperfect results further supports this possibility. In particular, if the decoding algorithm is not the principal factor limiting decoding performance, the neural signal itself might be. It seems reasonable to expect that even neural signals associated with low-frequency events (including the discrete arm movements studied in this experiment) encode information in signal attributes of higher order than the mean firing rates used to construct $\mathbf{N}(t)$ in this set of tests. This expectation is supported by the ability of Andersen and colleagues improve decoding performance by considering Haar wavelet coefficients of order greater than zero (the zeroth-order coefficient corresponds to the mean firing rate for the analyzed time interval). In particular, they reported a success rate of 87% in off-line decoding experiments using large training sets [26]; the ability to implement such signal analysis in real time using analog circuitry may therefore represent an important challenge. As might be expected, Andersen and colleagues found that the zeroth-order wavelet coefficient contained the most predictive information, with higher-order coefficients having decreasing predictive value. A further point to consider is that the parietal cortex may encode incomplete information concerning intended limb movement, so that optimal prediction might require neural signals from additional regions of the cortex.

The neural decoding scheme used by Andersen and colleagues is based in part on the known tendency of direction-sensitive neurons to 'tune' to a single preferred direction in the sense that only movement in a narrow range of directions centered on the preferred direction induce the neuron to modulate its firing rate away from a baseline rate [7, 2]. Figure 2-11 illustrates this phenomenon for a single neuron

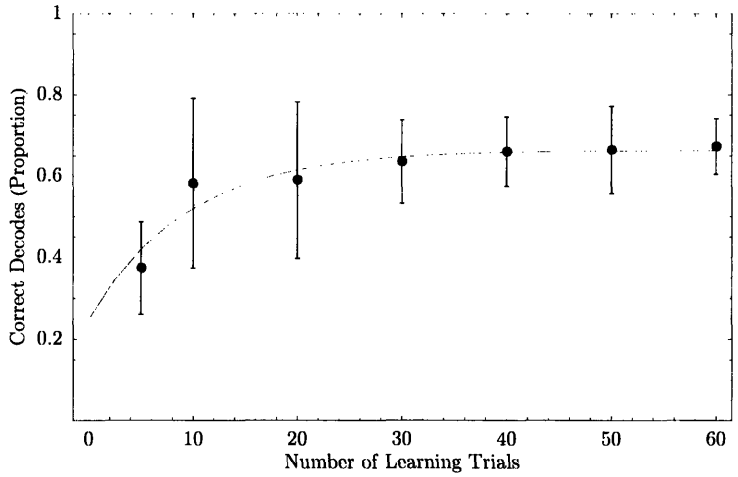



Figure 2-10: **Decoding Performance as a Function of Training Set Size (Training Time).** This plot shows the dependence of decoding performance on the size of the training set used by the adaptive filters to optimize their parameters $A_{ij}$ (5 training trials entailed 18 seconds of training time). Each data point represents an average over random initializations of $A_{ij}$ and different training subsets. Signals from all 54 isolated neurons were decoded to make the corresponding predictions of reach direction. Decoding performance is better than chance even for short training intervals, and improves with increased training, while the marginal benefit of additional training declines rapidly.

($j = 10$) by plotting neuronal spike rate waveforms over a memory period of 900 ms for a trial involving one reach in each direction; the $j = 10$ neuron is evidently tuned to direction $(-1, +1)$ ('Down'). After observing a training set of reach trials, Andersen and colleagues performed an off-line analysis of variance on the observed set of spike trains (corresponding to arm reaches in each direction for each isolated neuron) to rank the isolated neurons by degree of directional sensitivity. The computational intensity of their decoding scheme was sufficiently high that inputs from only a subset of isolated neurons were used in decoding after the learning period ended, and this ranking provided a means of prioritizing neurons for use as decoder inputs. By contrast, the gradient-descent adaptive-filter decoder described here easily handles all 54 neuronal inputs in computer simulations of real-time decoding. Moreover, and the analog-circuit–based implementation of that decoder described in 2.6 processes all neuronal inputs in parallel, so the computational intensity of the decoding task does not constrain the number of neuronal inputs the system can handle.

In the interest of considering the relative computational intensities of Bayesian versus adaptive-filter decoding in a somewhat more quantitative fashion, the following observations can be made. The number of operations required to handle a single input to the decoder after training scales as $n \times m \times p$ for an adaptive filter, where $n$, $m$, and $p$ denote the number of input channels, the number of output channels, and the number of parameters per kernel, respectively. (The number of operations per input during training scales in the same way.) The total memory requirements of the system also scale as $n \times m \times p$ because this is the total number of filter parameters in the system. In spite of this scaling, however, the parallel architecture of the adaptive filter ensures that computation time does not scale as a function of $n$, $m$, or $p$ since all of the convolution kernels operate in parallel and all filter parameters are updated in parallel. By contrast, the number of operations required by a Bayesian decoder to handle a single input after training scales approximately as $n \times m \times d \times \log q$, where $d$ denotes the number of possible discrete output values per output channel, and $q$ denotes the number of quantization levels into which the input signal is divided when the prior probability distributions are computed during training; the logarithm

reflects the requirement that the decoder look up the prior probability associated with a given input in a stored table containing $q$ elements. The series architecture of a typical digital processor ensures that the computation time per input scales in the same way, so that the maximum sampling frequency $f$ for inputs to the decoder is limited approximately by

$$f < \frac{s}{nmd \log q}, \tag{2.42}$$

where $s$ denotes the speed of the digital processor and has dimensions number of operations per time. Furthermore, when the desired decoder output is continuous, as in continuous-motion decoding rather than tasks involving movement in one of a discrete number of allowed directions, the value of $d$ required to approximate continuous performance can grow large. Finally, the total memory requirements of a Bayesian decoder scale as $n \times m \times d \times q$.

The foregoing analysis indicates that the gradient-descent adaptive-filter approach to decoding scales favorably with the number of neuronal inputs to the system as compared with a Bayesian method. This is an important virtue of adaptive-filter decoding, as there is consensus among investigators that decoding accuracy improves and more complex decoding tasks can be performed without sacrificing accuracy as more neuronal inputs are used [5], and improvements in multielectrode neural recording methodologies and technologies continue to facilitate recording from increasing numbers of neurons [27, 39, 46, 12].

The differences between the Andersen and adaptive-filter decoders with regard to choosing neuronal inputs motivate the question of whether gradient-descent optimization of the adaptive-filter implicitly organizes neuronal inputs according to relative degree of direction selectivity, effectively accomplishing in an on-line fashion the input selection task explicitly performed off-line in the Andersen decoding system. This question can be partially addressed by comparing the direction sensitivity of neuronal firing rates with the magnitude of the coefficients $A_{ij}$. The magnitudes of the $A_{ij}$ re-

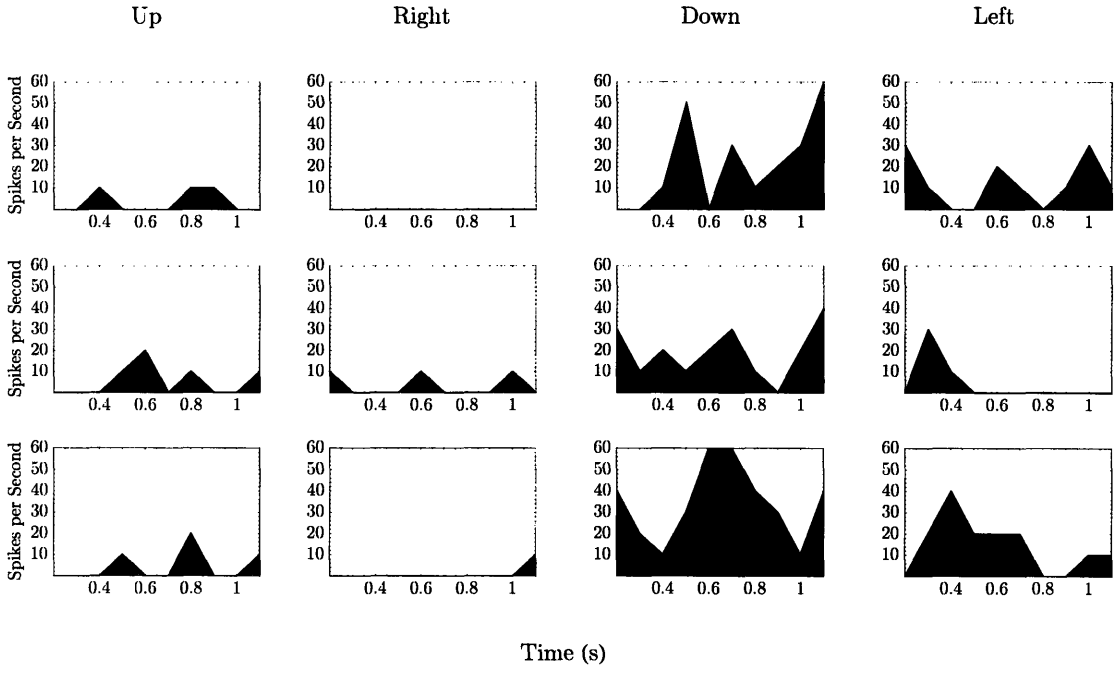



Figure 2-11: **Directional Tuning in a Single Neuron Persists Over Many Trials.** Each row of plots displays neural recording data from a single set of reach trials in which the macaque monkey being studied made one arm reach in each of the allowed directions. The intervals shown correspond to the memory period after the monkey was cued regarding the direction to which it should reach, but before it was permitted to perform the reach. Average neuronal spike rates were estimated over these intervals by counting the number of spikes in each 100 ms subinterval and dividing by interval length. The plots indicate that the neuron under study is tuned to the 'Down' direction, and its spike rate averaged over the entire memory interval is consistently highest in association with reaches in that direction.

flect the weights assigned to the corresponding inputs $N_j$ in predicting the output $\tilde{M}_i$, and might therefore be expected to correlate with the directional information content of the $N_j$ signals. An indicator of the directional information content of the $N_j$ can be obtained from the function

$$D(\hat{N}_j) = \sum_d \frac{\left(\hat{N}_j^{(d)} - \left\langle \hat{N}_j^{(d)} \right\rangle\right)^2}{\left\langle \hat{N}_j^{(d)} \right\rangle^2}, \tag{2.43}$$

where the index $d$ specifies a reach direction for which the raw firing rates $\hat{N}_j$ of neuron $j$ are considered, and $\left\langle \hat{N}_j^{(d)} \right\rangle$ denotes the mean value of the raw firing rate of neuron $j$ associated with a reach in direction $d$. Figure 2-12 shows a set of plots that compare the parameters $A_{ij}$ with the directional sensitivity of the corresponding neuronal inputs $\hat{N}_j$ as measured by $D(\hat{N}_j)$. Figure 2-12 (a) shows the values of $D(\hat{N}_j)$ for all $j$, with the indices $j$ sorted in order of decreasing $D(\hat{N}_j)$. The plot suggests that a subset of neurons contain markedly higher amounts of direction-specific information than the remaining neurons. Similarly, Figure 2-12 (b) shows the values of $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$ for all neurons $j$, with the indices sorted in order of decreasing $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$ (the average values of $A_{ij}$ were obtained by averaging over values of $A_{ij}$ learned under different sets of randomized initial conditions and different sets of 30 training trials). As is the case with the degree of directional tuning, the largest magnitudes of the coefficients $A_{ij}$ are associated with a subset of all isolated neurons, although the threshold between high and low values of $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$ is less clearly demarcated than in the case of the $D(\hat{N}_j)$. Figure 2-12 (c) is a scatterplot of the set of points $\left(D(\hat{N}_j), \sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}\right)$, $j \in \{1, \ldots, 54\}$. The threshold separating high and low values of $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$, corresponding to the vertical line across Figure 2-12, was drawn to correspond with the sharp drop in that parameter between its ninth- and tenth-highest observed values. The threshold distinguishing high from low values of $D(\hat{N}_j)$ was drawn so that half of all observed values would fall above and half would fall below the threshold. The scatterplot can be used to compare the numbers of points

in the four regions distinguished by the thresholds just defined, and it indicates that neurons assigned large regression coefficients $A_{ij}$ are on average approximately twice as likely to exhibit strong directional tuning as are neurons assigned small regression coefficients. Together these three plots suggest that gradient-descent optimization of adaptive-filter parameters can function as an on-line method of prioritizing neuronal inputs according to degree of directional tuning, implicitly and at low computational cost performing a function analogous to the computationally intensive off-line analysis used by Andersen and colleagues.

The ability of the adaptive-filter decoder to handle large numbers of neurons provides an opportunity to explore a decoding regime not accessible to Andersen and colleagues. Figure 2-13 plots decoder performance as a function of the number of neuronal inputs used by the decoder. While Andersen and colleagues only compute these performance curves for up to sixteen neurons [25], the computational efficiency of the adaptive filter enables performance to be evaluated for much larger numbers of neuronal inputs. The computation illustrated in Figure 2-13 is therefore limited only by the total number of neuronal inputs available. The computation was performed under the standard training condition of 30 trials, and the error bars indicate the magnitude of a standard deviation after averaging over sets of randomized initial conditions and training inputs. As expected, decoder performance increases from just above the 25% chance threshold to the maximum of approximately 65% reported earlier in this section. The lower curve corresponds to random selections of the input neurons, while the upper curve corresponds to preselection of neurons in order of decreasing $D(\hat{N}_j)$. The latter curve suggests, as might be anticipated from the analysis presented in Figure 2-12, that decoding input signals from the subset of neurons transmitting the greatest amount of directional information results in performance nearly equivalent to that obtained from using the full set of available signals.

The experiments discussed in this section demonstrate that the gradient-descent least-squares adaptive-filter approach can be used to decode neural signals predicting intended limb movement with a degree of accuracy comparable to that achieved by state-of-the art systems. A major advantage to the decoding approach presented here

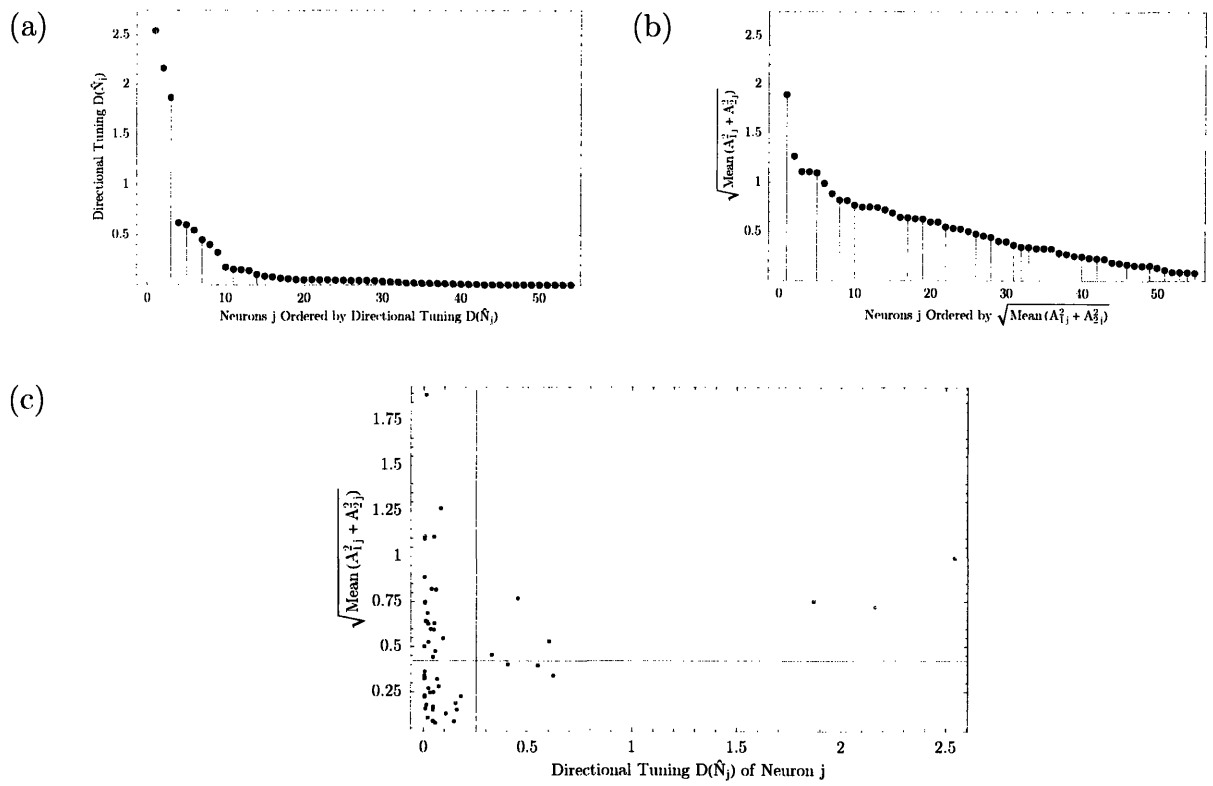



Figure 2-12: **The Adaptive Filter Implicitly Screens for Neurons with Strong Directional Tuning in an On-Line Fashion.** (a) Degree of directional tuning $D(\hat{N}_j)$ for all neurons $j$, with the indices $j$ sorted in order of decreasing $D(\hat{N}_j)$. The plot suggests that a subset of neurons contain markedly higher amounts of direction-specific information than the remaining neurons. (b) Root-mean-squared regression coefficients $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$ for all neurons $j$, with the indices sorted in order of decreasing $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$. The largest magnitudes of the coefficients $A_{ij}$ are associated with a subset of all isolated neurons, although the threshold between high and low values of $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$ is less clearly demarcated than in the case of the $D(\hat{N}_j)$. (c) Scatterplot of the set of points $\left( D(\hat{N}_j), \sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle} \right)$, $j \in \{1, \ldots, 54\}$. The vertical line corresponds to the threshold separating high and low values of $\sqrt{\sum_{i=1,2} \langle A_{ij}^2 \rangle}$, while the horizontal line corresponds to the median value of $D(\hat{N}_j)$. Relative numbers of points in the four regions distinguished by the indicated thresholds suggest that neurons assigned large regression coefficients $A_{ij}$ are on average approximately twice as likely to exhibit strong directional tuning as are neurons assigned small regression coefficients. Evidently the adaptive filter implicitly learns to screen for neurons with relatively strong directional tuning, a function implemented at high computational cost in other neural decoding systems.

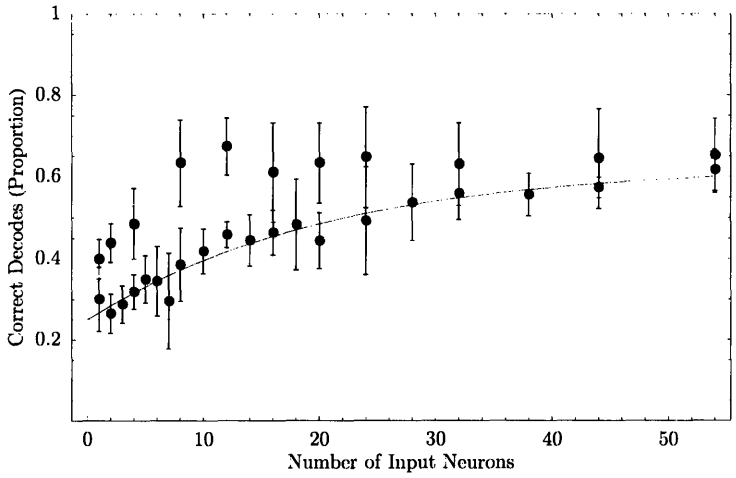



Figure 2-13: **Decoding Performance as a Function of Neuron Number for Randomly Selected Neurons and Neurons with Strong Directional Tuning.** Decoding performance is plotted as a function of the number of neuronal inputs used by the decoder. Blue points correspond to experiments in which neurons were selected at random for use by the decoder, while red points correspond to experiments in which neurons were selected for use in decreasing order of their degree of directional tuning (as measured by $D(\hat{N}_j)$). While both curves tend toward the same maximal performance when all neurons are used, decoding input signals from the subset of neurons transmitting the greatest amount of directional information results in performance nearly equivalent to that obtained from using the full set of available signals.

is its low computational complexity and speed of operation relative to the state of the art. As discussed in Section 2.6, these features enable the adaptive-filter decoding system to be implemented in a micropower device suitable for long-term implantation in the brain. The ability to decode intended limb movement from neural signals suggests that such a device could be used as a neural prosthesis for patients suffering from paralysis, enabling them to regain natural, thought-based control of artificial limbs—or perhaps eventually their own paralyzed limbs. By contrast, state-of-the-art neural decoding systems depend on computationally intensive techniques and algorithms implemented off-line using software. Such systems are necessarily power-hungry and therefore unsuitable for safe implantation in delicate neural tissue. A further advantage to the decoding system described here is that its learning occurs on-line, so that the decoder is operational immediately following training, and training can be continued until a preestablished error threshold is met. By contrast, the Bayesian decoding technique used by Andersen and colleagues requires off-line analysis of data obtained and stored during the training session. A delay between training and operability is inevitable in such systems. Furthermore, it is not possible to determine the error rate of such a system until the off-line analysis is complete, so the possibility of having to iterate training and analysis in the event that performance is not sufficiently good represents a certain inherent inefficiency relative to on-line learning.

The 65% success rate of the decoding system, while comparable to that achieved by more computationally intensive systems including those of Nicolelis and colleagues, Schwartz and colleagues [41], and Andersen and colleagues [25], indicates that there is considerable room for improvement. An outstanding question in the field of neural prosthetics concerns the degree to which intelligent users can compensate for imperfect decoding through biofeedback. Marked improvements in performance along these lines have been observed over time in both monkeys and humans [40, 4, 25, 14], but such contributions from biological learning are evidently insufficient. As indicated earlier in this section, improved performance can likely be achieved by considering the temporal structure of the input neural signals at higher than zeroth order, which

might be achieved by changing the form of the filter kernels $W_{ij}$. The results achieved by Andersen and colleagues using Haar wavelets suggest that adaptive-filter implementation of wavelet-like decoding might be possible by incorporating variable delay parameters into the form of $W_{ij}$ to translate rescaled versions of the filter kernel. The question of whether a particular form of $W_{ij}$ is generally optimal for neural decoding is an interesting subject for further experimentation, especially since other forms of $W_{ij}$ can be tested using the techniques and procedures described in this chapter.

## 2.6 A Low-Power Analog Electronic Architecture to Implement Linear Decoding of Neural Signals

This section outlines a method for implementing the gradient-descent least-squares neural signal decoder in a system of low-power analog electronic circuits, using a custom 0.18 $\mu$m CMOS process. The circuit design work described here represents an ongoing collaboration with Woradorn Wattanapanitch, Graduate Student in the MIT Department of Electrical Engineering and Computer Science. Several components of the system design discussed in this section represent innovations, and the designs of those components are described in detail. Other system components rely on standard analog circuit building blocks or on the previous work of members of the Analog VLSI and Biological Systems Group in the MIT Research Laboratory of Electronics; such components are discussed in less detail and references to the relevant literature are provided. Many of the ideas contained in this section are also included in an unpublished manuscript written jointly with Woradorn Wattanapanitch and other members of the Analog VLSI and Biological Systems Group [42], as well as a recently accepted conference paper that discusses the implementation of the analog-circuit–based neural signal decoder in the context of a full neural prosthetic system [35].

This section has several subdivisions. Section 2.6.1 considers the two primary classes of input signals to be used by the decoder and how to preprocess them in an

analog context before passing them to the system used to implement the gradient-descent least-squares algorithm. Decoding based on action potentials ('spikes') would require a means of converting spike waveforms into time-averaged mean firing rates. Decoding based on local field potentials (LFPs) would require band-pass filtering and preliminary analysis of the spectral bands known to convey information relevant to motor intentions. Analog signal preprocessing appropriate to each mode of operation is addressed in Section 2.6.1. Section 2.6.2 describes the circuit building blocks required to implement the gradient-descent least-squares neural decoding algorithm. As described in Section 2.3 and summarized in Figure 2-1, implementation of this algorithm requires five main block types: (1) Adaptive filters corresponding to the $W_{ij}$; (2) Parameter-learning filters to adapt the $\{W_{ij}^{(k)}\} = \{A_{ij}, \tau_{ij}\}$; (3) Biasing circuits to support the operation of the parameter-learning filters; (4) Multipliers; and (5) Adders and subtracters [3]. Figure 2-14 shows a single module from the mathematical block diagram of Figure 2-1 alongside a block diagram indicating the circuit building blocks required to implement the functions of that module. The design of each block is outlined in Section 2.6.2.

### 2.6.1 Input Signals for the Neural Decoder

#### Spike-Based Decoding

Neuronal action potential voltage spikes typically have a width on the order of $10^{-3}$ s, corresponding to frequencies in the kilohertz range. Spikes are therefore intrinsically high-frequency events, and consequently unsuitable for direct use as input signals for a low-power signal decoder. Spike-based inputs can be used, however, if they are first transformed to lower-frequency signals (of order 1 Hz) through time-averaging. Such

[3] A complete implementation would also include memory blocks for storing parameter values after the learning phase ends. Such blocks could consist of analog memory elements operating in a switched sample-and-hold scheme to permit memoryless adaptation during the learning phase and parameter storage as soon as learning terminates. Output from the memory and biasing circuits could be multiplexed onto the adaptive filter nodes whose voltages correspond to the adaptive filter parameters using a CMOS transmission gate. This scheme is indicated in Figure 2-14. But even using digital memory elements might not increase total power consumption significantly, since the termination of a learning phase is a rare event and so writing to memory, with its associated power cost, occurs only infrequently.

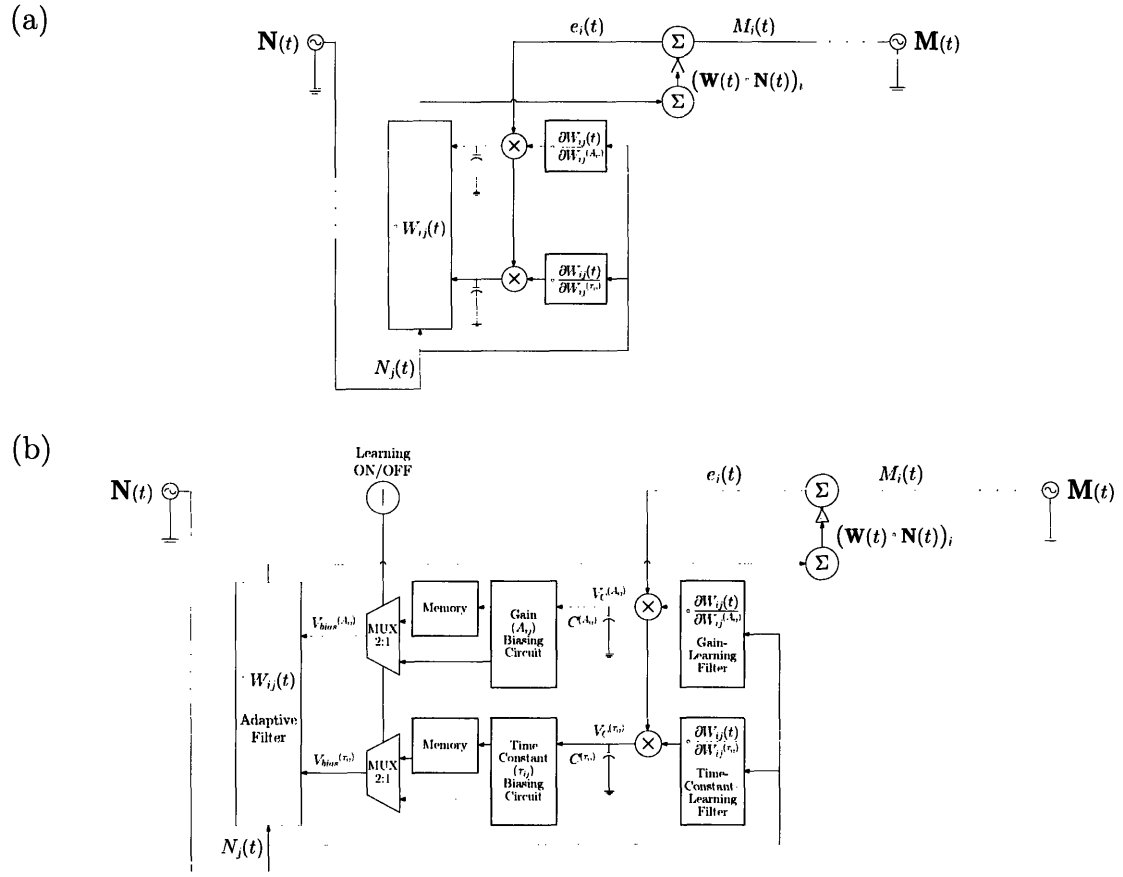



Figure 2-14: **Mathematical and Circuit Building Blocks for a Single Module of the Gradient-Descent Least-Squares Neural Signal Decoder.** (a) A single module from Figure 2-1 showing (in block-diagram form) the mathematical operations required to implement a single gradient-descent least-squares decoding module. (b) A block diagram indicating the circuit building blocks required to implement the functions of a single decoding module.

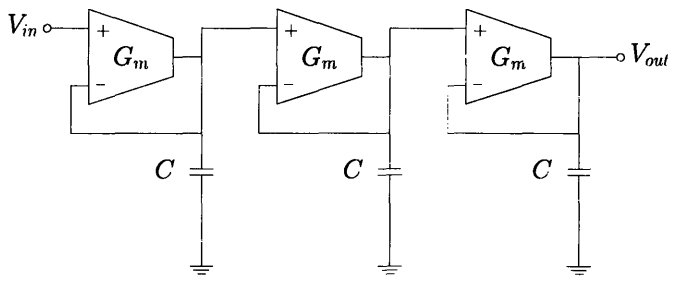



Figure 2-15: **Interpolation Filter to Extract Mean Firing Rate Inputs for Spike-Based Neural Signal Decoding.** A third-order $G_mC$ interpolation filter formed by cascading three first-order filters. The filter extracts a low-frequency mean spiking rate from intrinsically high-frequency neuronal action potentials. This signal processing step is necessary for the decoding algorithm to operate at the low frequencies required for a small power budget.

time-averaging can be implemented by low-pass 'interpolation' filters. The simplest such filter is a first-order low-pass filter with transfer function $H_1(s \equiv i) = \frac{1}{1+\tau_1 s}$, and cutoff frequency $f_c = \frac{1}{2\pi\tau_1}$ designed to 1 Hz. Smoother interpolation can be obtained from a higher-order filter such as the third-order filter with transfer function $H_3(s) = \frac{1}{(1+\tau_1 s)^3}$, obtained by cascading three first-order filters as indicated in Figure 2-15, each having $f_c = 1$ Hz. The analog implementation of such filters can be achieved using a $G_mC$ design: $G_m$ refers to the transconductance of the operational transconductance amplifier (OTA) component, while $C$ denotes the filter capacitance. In such a filter $f_c = \frac{1}{2\pi}\frac{G_m}{C}$, so a low cutoff frequency requires $C$ to be large or $G_m$ to be small. Circuit layout area restrictions will constrain the maximum value of $C$ to approximately 4 pF, so a low $f_c$ requires $G_m$ to be small. Since the transconductance $G_m$ is proportional to the bias current of its associated OTA, small values of $G_m$ require small bias currents. At $f_c = 1$ Hz, transconductance amplifier bias currents on the order of $10^{-14}$–$10^{-13}$ A. Such small bias currents are not easily controlled due to noise effects and transistor leakage currents, but suitable wide-linear-range transconductance amplifiers have been developed and described in the analog circuits literature [33].

### Local-Field-Potential–Based Decoding

As discussed in Section 2.4, local field potentials recorded from the parietal cortex encode information concerning limb trajectories in the $\gamma$-band of the power spectrum (25–90) Hz. As modeled in Section 2.4, an envelope waveform proportional to the gamma-band content of the local field potential is therefore a suitable input to the neural signal decoder. Such an input waveform can be generated by passing the raw local field potential signals through a band-pass amplifier tuned to the $\gamma$-band, rectifying the output of the band-pass filter, and passing the result through a peak-detector circuit to generate the envelope. Combined band-pass amplifier and envelope detector circuits suitable for these purposes have previously been developed by members of the Analog VLSI and Biological Systems Group in the context of a bionic ear processor [34]. While the circuit architecture for extracting a $\gamma$-band envelope waveform will be analogous to that used in the bionic ear processor, lower-frequency operation due to the low frequency range of local field potentials would reduce power consumption to approximately 1 $\mu$W in the case of a neural signal amplifier.

## 2.6.2 Analog Circuit Building Blocks for Implementing Gradient-Descent Least Squares Neural Signal Decoding

### Adaptive Filters

Each of the $m \times n$ adaptive filters used to implement the gradient-descent least-squares algorithm must be designed to have a transfer function of the form

$$H_{W_{ij}}(s) = \frac{A_{ij}}{1 + \tau_{ij}}, \tag{2.44}$$

as discussed in previous sections. Such a transfer function can be obtained from a filter having the topology shown in Figure 2-16, which contains three standard, nine-transistor wide-range operational transconductance amplifiers of the form described by Mead [22]; the associated transconductances are denoted $G_M^{(A_{ij})}$, $G_M^{(\tau_{ij})}$, and $G_M^{(R_{ij})}$.

Low-power performance can be assured by operating the adaptive filter circuit with every transistor in its subthreshold regime. The transconductance $G_M^{(A_{ij})}$ is $\frac{\kappa I_{(A_{ij})}}{V_T}$, where $\kappa$ denotes the gate-coupling coefficient of the MOS transistors in the filter, which in this analysis are assumed to match well and have $\kappa \approx 0.7$; and $V_T = \frac{kT}{q}$, where $k$ denotes the Boltzmann constant, $q$ denotes the electron charge, and $T$ denotes the Kelvin temperature. The transconductance $G_M^{(R_{ij})}$ is $\frac{\kappa I_{(R_{ij})}}{V_T}$, and the associated OTA is connected in a unity-gain–follower feedback configuration, with capacitor $C_d$ providing dominant-pole compensation; this circuit element provides reference values for the filter parameters. Finally, the transconductance $G_M^{(\tau_{ij})}$ is $\frac{\kappa I_{(\tau_{ij})}}{V_T}$, and the associated OTA is also connected in a unity-gain–follower feedback configuration. This configuration functions to set the time constant of the adaptive filter, $\tau_{ij}$ of the transfer function in Equation 2.6.2. The approximate transfer function of the adaptive filter is therefore

$$\frac{V_{out}(s)}{V_{in}(s)} = \frac{I_{(A_{ij})}}{I_{(R_{ij})}} \frac{1}{1 + s\frac{C_{\tau_{ij}}}{G_M^{(R_{ij})}}}, \tag{2.45}$$

which has the form required by Equation , with $A_{ij} = \frac{I_{(A_{ij})}}{I_{(R_{ij})}}$ and $\tau_{ij} = \frac{C_{\tau_{ij}}}{G_M^{(R_{ij})}}$. The gain $A_{ij}$ of each adaptive filter can therefore be tuned by adjusting the current ratio $\frac{I_{(A_{ij})}}{I_{(R_{ij})}}$, while each time constant $\tau_{ij}$ can be tuned by modifying the ratio $\frac{C_{\tau_{ij}}}{G_M^{(\tau_{ij})}}$ through adjusting the bias current $I_{(\tau_{ij})}$.

### Parameter-Learning Filters

As discussed mathematically in Section 2.3, the gradient-descent least-squares decoder realized here implements the convolution kernels $W_{ij}$ with adaptive filters having impulse response functions $W_{ij}(t) = \frac{A_{ij}}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}}$, corresponding to frequency-domain transfer functions of the form given in Equation 2.6.2. The central idea behind gradient-descent decoding is that optimization of the convolution kernels $W_{ij}$ can be achieved through tuning of the filter parameters $\{W_{ij}^{(k)}\} = \{A_{ij}, \tau_{ij}\}$ in proportion to $-\vec{\nabla}_{ij}^{(k)} E$.

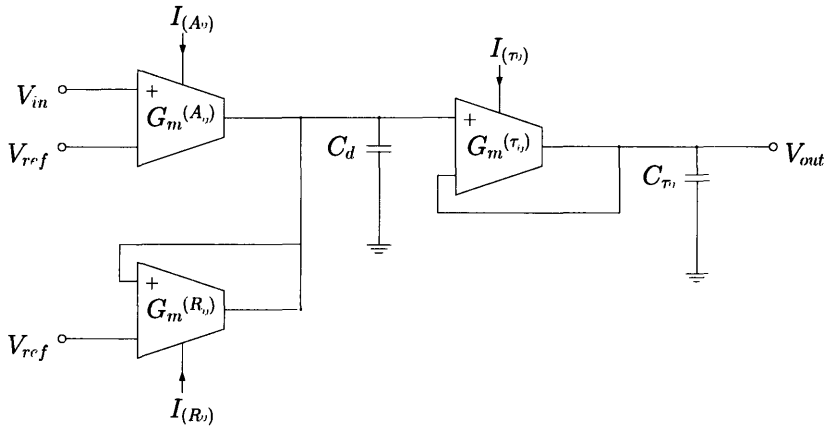



Figure 2-16: **Adaptive Filter with Tunable Parameters for Learning the Optimal Convolution Kernels for Neural Signal Decoding.** This adaptive filter features three standard nine-transistor wide-range operational transconductance amplifiers operated in their subthreshold regimes in order to ensure low power consumption. This filter implements the transfer function $H_{W_{ij}}(s) = \frac{A_{ij}}{1+\tau_{ij}}$ as $\frac{V_{out}(s)}{V_{in}(s)} = \frac{I_{(A_{ij})}}{I_{(R_{ij})}} \frac{1}{1+s\frac{C_{\tau_{ij}}}{G_M^{(R_{ij})}}}$, so the gain $A_{ij}$ of each adaptive filter can be tuned by adjusting the current $I_{(A_{ij})}$ ( $I_{(R_{ij})}$ is a reference current), while each time constant $\tau_{ij}$ can be tuned by modifying the ratio $\frac{C_{\tau_{ij}}}{G_M^{(\tau_{ij})}}$ through adjusting the bias current $I_{(\tau_{ij})}$.

As indicated in 2.22, such tuning can be accomplished in real time using signals proportional to $-\vec{\nabla}_{ij}^{(k)}E$, whose construction requires convolution kernels proportional to $\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k)}}$. These convolution kernels can be implemented by 'parameter-learning filters' with corresponding frequency-domain transfer functions of the following forms:

$$\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k)}} \rightarrow \frac{\partial W_{ij}(s)}{\partial W_{ij}^{(k)}} \tag{2.46}$$

$$= \frac{\partial}{\partial W_{ij}^{(k)}} \frac{A_{ij}}{1+s\tau_{ij}} \tag{2.47}$$

$$\frac{\partial W_{ij}(s)}{\partial A_{ij}} = \frac{1}{1+s\tau_{ij}} \tag{2.48}$$

$$\frac{\partial W_{ij}(s)}{\partial \tau_{ij}} = -\frac{A_{ij}s}{(1+s\tau_{ij})^2}. \tag{2.49}$$

Figure 2-17 shows the designs for a pair of paramter-learning filters. Figure 2-17 (a) shows a first-order low-pass $G_mC$ filter with transfer function $W_{ij}^{A_{ij}}(s) = \frac{1}{1+s\tau_{ij}}$ to be used as a 'gain-learning filter.' Figure 2-17 (b) shows a second-order band-pass $G_mC$ filter with transfer function $W_{ij}^{\tau_{ij}}(s) = \frac{s\tau_{ij}}{(1+s\tau_{ij})^2}$ to be used as a 'time-constant–learning filter.' The time constant $\tau_{ij} = \frac{C_{\tau_{ij}}}{G_M^{(\tau_{ij})}}$ for the two parameter-learning filters is identical to that of the adaptive filter, as described in Section 2.6.2. Correspondingly, the bias currents and therefore the transconductances $G_M^{(\tau_{ij})}$ in all three types of filter are identical, so the time constants of all filters in the learning architecture are updated simultaneously. Note that the actual transfer function of the time-constant–learning filter need only be proportional to $W_{ij}^{\tau_{ij}}(s)$, so the factor-of- $\frac{\tau_{ij}}{A_{ij}}$ difference between the transfer function of Equation 2.49 and the filter shown in Figure 2-17 (b) is acceptable. Implementation of the negation required by Equation 2.49 will be addressed in Section 2.6.2.

### Multipliers

The multipliers that perform the operations $e_i(t) \times \frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k)}}$ required by the gradient descent algorithm, denoted by the symbol $\times$ in Figures 2-1 and 2-14, can be imple-

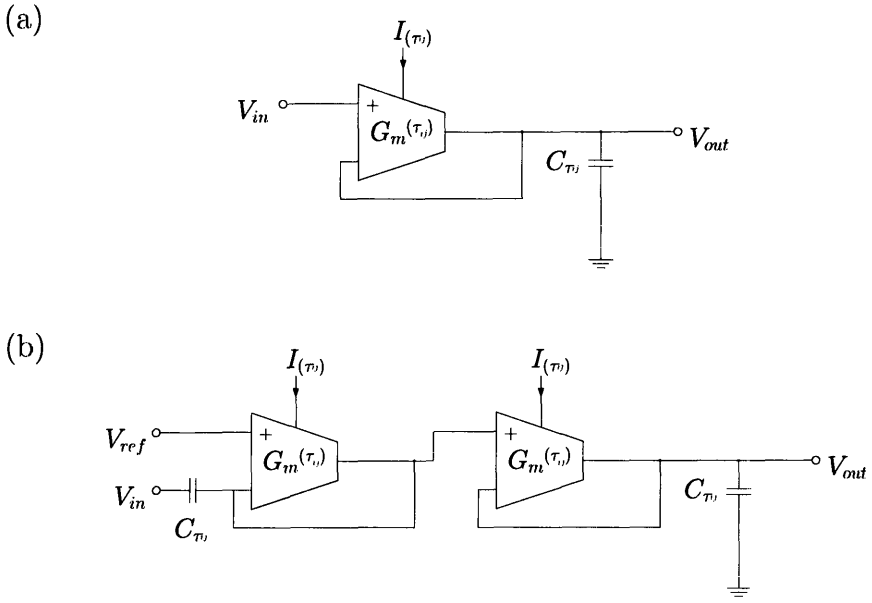



Figure 2-17: **Parameter-Learning Filters for Tuning Adaptive Filter Parameters Based on Error-Signal Feedback.** The parameters of the adaptive filters can be tuned in real time through error-signal feedback, using signals proportional to $-\vec{\nabla}_{ij}^{(k)}E$. Construction of such signals requires convolution kernels proportional to $\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(k)}}$. These convolution kernels can be implemented by 'parameter-learning filters' with corresponding frequency-domain transfer functions; appropriate transfer functions are implemented by the filters shown. (a) A first-order low-pass $G_mC$ filter with transfer function $W_{ij}^{A_{ij}}(s) = \frac{1}{1+s\tau_{ij}}$ to be used as a 'gain-learning filter.' (b) A second-order band-pass $G_mC$ filter with transfer function $W_{ij}^{\tau_{ij}}(s) = \frac{s\tau_{ij}}{(1+s\tau_{ij})^2}$ to be used as a 'time-constant–learning filter.'

mented using wide-range four-quadrant Gilbert multipliers of the kind described by Mead, which take four voltage inputs $V_i$, $i \in \{1,2,3,4\}$ and a bias current $I_b$, and generate an output current [22]. Each multiplication operation required by the system must be implemented by a distinct Gilbert multiplier, but in the interest of notational clarity multiplier circuit parameters and state variables are not indexed; circuit quantities referenced will be understood to correspond to the particular multiplier under discussion. As in previous sections, $W_{ij}^{(k=1)}$ stands for the 'gain' parameter $A_{ij}$ and $W_{ij}^{(k=2)}$ stands for the 'time constant' parameter $\tau_{ij}$. Low-power performance can be obtained by operating the multiplier circuit with all transistors in the subthreshold regime. The input-output characteristic for the Gilbert multiplier is

$$I_{out} = I_b \tanh \frac{\kappa(V_1 - V_2)}{2V_T} \tanh \frac{\kappa(V_3 - V_4)}{2V_T} \tag{2.50}$$

$$\approx I_b \left(\frac{\kappa}{2V_T}\right)^2 (V_1 - V_2)(V_3 - V_4), \tag{2.51}$$

where the approximation of Equation 2.51 is valid in the intended operating region, where $V_1 \approx V_2$ and $V_3 \approx V_4$. Noninverting multiplication, as required in Equation 2.48 for adapting the $A_{ij}$, can be implemented by feeding $e_i$ into $V_1$, $\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(A_{ij})}}$ into $V_3$, and setting $V_2$ and $V_4$ to a constant reference voltage. On the other hand, inverting multiplication, as required by the negation in Equation 2.49 for adapting the $\tau_{ij}$, can be implemented by interchanging the role of $V_3$ and $V_4$, feeding $\frac{\partial W_{ij}(t)}{\partial W_{ij}^{(\tau_{ij})}}$ into $V_4$, and setting $V_3$ to a constant reference voltage.

The output of each multiplier is a current, $I_{out}$, so the integrations required in Equation 2.22 for updating the parameters $A_{ij}$ and $\tau_{ij}$ are conveniently implemented by linear capacitors, as indicated in Figure 2-14. The voltages $V_C^{(A_{ij})}$ and $V_C^{(\tau_{ij})}$ on the capacitors $C^{(A_{ij})}$ and $C^{(\tau_{ij})}$ used in adapting the parameter values $A_{ij}$ and $\tau_{ij}$, respectively, are therefore given by

$$V_C^{(A_{ij})} = \frac{I_b}{C^{(A_{ij})}} \left(\frac{\kappa}{2V_T}\right)^2 \int_{u=0}^{t} e_i(u) \left(\frac{\partial W_{ij}(u)}{\partial A_{ij}} \circ N_j(u)\right) du \tag{2.52}$$

$$V_C^{(\tau_{ij})} = \frac{I_b}{C^{(\tau_{ij})}} \left(\frac{\kappa}{2V_T}\right)^2 \int_{u=0}^{t} e_i(u) \left(\frac{\partial W_{ij}(u)}{\partial \tau_{ij}} \circ N_j(u)\right) du, \tag{2.53}$$

which have the form required by Equation 2.22. The filter parameters therefore vary continuously in time according to the following equations:

$$\Delta V_C^{(A_{ij})} = \lim_{\delta \to 0} \frac{I_b}{C^{(A_{ij})}} \left(\frac{\kappa}{2V_T}\right)^2 \int_{u=t-\delta}^{t} e_i(u) \left(\frac{\partial W_{ij}(u)}{\partial A_{ij}} \circ N_j(u)\right) du \tag{2.54}$$

$$\Delta V_C^{(\tau_{ij})} = \frac{I_b}{C^{(\tau_{ij})}} \left(\frac{\kappa}{2V_T}\right)^2 \int_{u=t-\delta}^{t} e_i(u) \left(\frac{\partial W_{ij}(u)}{\partial \tau_{ij}} \circ N_j(u)\right) du, \tag{2.55}$$

where $\delta$ is the time constant of a single decoder module and represents a characteristic timescale over which the filter parameters are updated.

### Biasing Circuits

As discussed in Section 2.6.2, the filter parameters $A_{ij}$ and $\tau_{ij}$ defining the transfer function of adaptive filter $W_{ij}$ are stored on the capacitors $C^{(A_{ij}}$ and $C^{(\tau_{ij}}$, respectively. Furthermore, as indicated in Section 2.6.2, the values of the filter parameters can be tuned by adjusting the bias currents that determine $G_M^{(A_{ij})}$ and $G_M^{(\tau_{ij})}$. Since the gain $A_{ij}$ depends on $\frac{I_{(A_{ij})}}{I_{(R_{ij})}} \propto I_{(A_{ij})}$, while the time constant $\tau_{ij}$ depends on $\frac{C_{\tau_{ij}}}{G_M^{(\tau_{ij})}} \propto \frac{1}{I_{(\tau_{ij})}}$, real-time adaptive parameter tuning requires a scheme for modifying $I_{(A_{ij})}$ in proportion to $V_C^{(A_{ij})}$ and $I_{(\tau_{ij})}$ in inverse proportion to $V_C^{(\tau_{ij})}$.

Tuning $I_{(A_{ij})}$ in proportion to variations in the capacitor voltage $V_C^{(A_{ij})}$ can be accomplished by converting $V_C^{(A_{ij})}$ into a current proportional to $V_C^{(A_{ij})}$ and then using a current mirror to generate a copy of that current that is in turn used to set the transconductance $G_M^{(A_{ij})}$ of the adaptive filter. The conversion of $V_C^{(A_{ij})}$ into a current proportional to $V_C^{(A_{ij})}$ can be performed by a wide-linear-range transconductance amplifier (WLR) of the form described by Sarpeshkar and colleagues [33]. To ensure

that the current flowing into the NMOS side of the current mirror is always positive, an offset current $I_0$ is added to the output current of the WLR. Figure 2-18 (a) shows a schematic of the gain-biasing circuit used to generate $I_{(A_{ij})} \propto V_C^{(A_{ij})}$.

Tuning $I_{(\tau_{ij})}$ in inverse proportion to variations in the capacitor voltage $V_C^{(\tau_{ij})}$ can be accomplished using the circuit shown in Figure 2-18 (b), which operates as follows. First, $V_C^{(\tau_{ij})}$ is converted into a proportional current $I^p_{(\tau_{ij})}$ as in the gain-biasing circuit. A translinear circuit, formed by the four well matched MOS transistors $M_1$–$M_4$, is then used to invert $I^p_{(\tau_{ij})}$, producing $I_{(\tau_{ij})} = \frac{I_s^2}{I^p_{(\tau_{ij})}}$, where $I_s$ is a current reference that 'scales' the inversion. A mirror copy of $I_{(\tau_{ij})}$ is then used as the bias current that sets transconductance $G_M^{(\tau_{ij})}$.

### Adders and Subtracters

Each adder, denoted by the symbol $\Sigma$ in Figures 2-1 and 2-14, sums the $n$ outputs of each set $\{W_{ij}\}$, $j \in \{1, \ldots, n\}$ of adaptive filters contributing to $(\mathbf{W}(t) \circ \mathbf{N}(t))_i = \tilde{M}_i(t)$. The adders can be implemented using a follower-aggregation circuit of the kind described by Mead [22]. The corresponding error signal, $e_i(t)$, is generated by performing the subtraction $M_i(t) - \tilde{M}_i(t)$. This operation can be implemented by another adder, with a unity-gain inverting amplifier negating the adder input from $\tilde{M}_i(t)$.

## 2.6.3 Estimated Power Consumed by the Gradient-Descent Least-Squares Decoder Implemented in Analog Circuitry

Using the circuit designs presented in the preceding sections, a single decoding module (corresponding to a single $W_{ij}$ and the circuitry required to optimize its parameter values) can be implemented in approximately 3000 $\mu$m$^2$ and should consume approximately 0.3 $\mu$W of power from a 1V supply in a 0.18 $\mu$m technology. This low power consumption is due to the use of subthreshold bias currents for the transistors in the analog filters and other components described in Section 2.6.2. A full-scale sys-

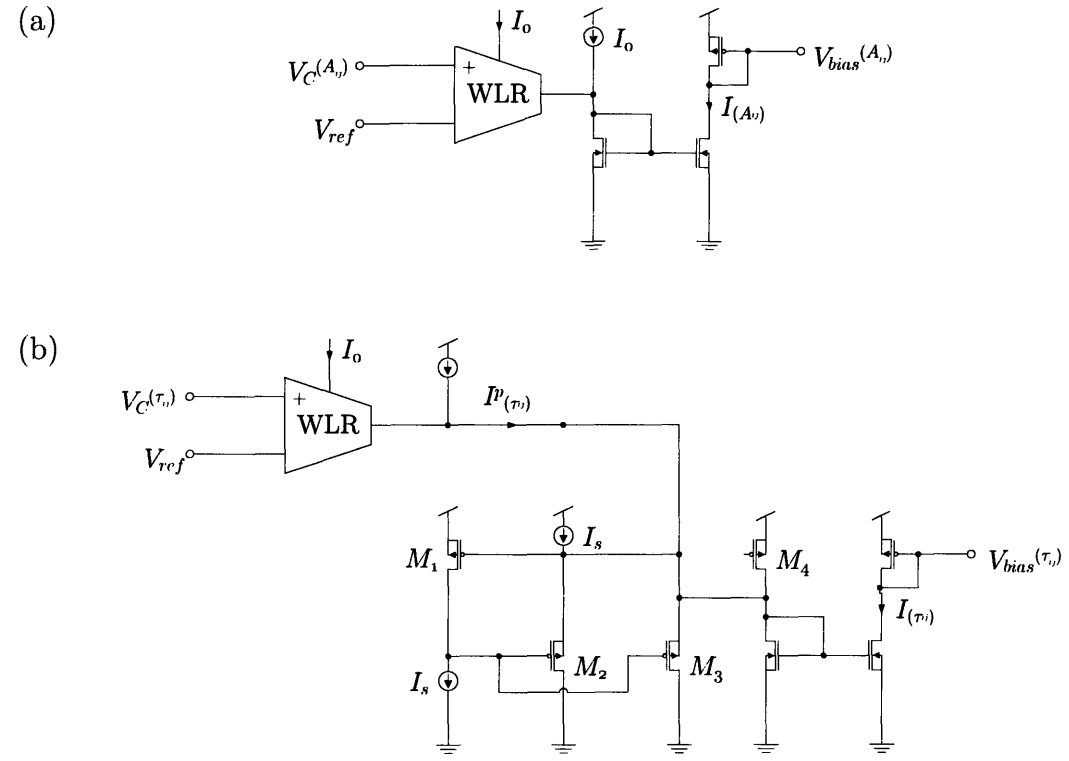



Figure 2-18: **Circuits for Setting the Bias Currents and Transconductances that Determine the Adaptive Filter Parameters.** The filter parameters $A_{ij}$ and $\tau_{ij}$ defining the transfer function of adaptive filter $W_{ij}$ are stored on the capacitors $C^{(A_{ij})}$ and $C^{(\tau_{ij})}$, respectively. Since the gain $A_{ij}$ depends on $\frac{I_{(A_{ij})}}{I_{(R_{ij})}} \propto I_{(A_{ij})}$, while the time constant $\tau_{ij}$ depends on $\frac{C_{\tau_{ij}}}{G_M^{(\tau_{ij})}} \propto \frac{1}{I_{(\tau_{ij})}}$, real-time adaptive parameter tuning requires a scheme for modifying $I_{(A_{ij})}$ in proportion to $V_C^{(A_{ij})}$ and $I_{(\tau_{ij})}$ in inverse proportion to $V_C^{(\tau_{ij})}$. (a) The gain-biasing circuit, which uses a wide-linear-range transconductance amplifier (WLR) to generate $I_{(A_{ij})} \propto V_C^{(A_{ij})}$. The offset current $I_0$ added to the WLR output keeps the NMOS input current positive. (b) The time-constant biasing circuit, which tunes $I_{(\tau_{ij})}$ in inverse proportion to variations in the capacitor voltage $V_C^{(\tau_{ij})}$ using a WLR, a translinear circuit, and a current mirror, as explained in the text.

tem with $n = 100$ (100 neuronal inputs $N_j(t)$ comprising $\mathbf{N}(t)$) and $m = 3$ (three motor control parameters $M_i(t)$ comprising $\mathbf{M}(t)$) would require $m \times n = 300$ decoding modules, occupying a total area of approximately 1 $mm^2$ and consuming only approximately 90 $\mu$W of total power.

# Chapter 3

# Decoding and Modeling Neural Parameter Space Trajectories During Thinking and Dreaming

## 3.1 Overview

The present chapter explores the use of adaptive-filter–based neural signal decoding to analyze the activity of neuronal populations in contexts other than those intended for the control of limb movement or external devices. In particular, Section 3.3 demonstrates how the decoding method developed in Chapter 2 can be used to predict the head direction of a laboratory rat in real time on the basis of neuronal spike train data recorded from a small population of thalamic neurons. One way of viewing this result is as a new approach to real-time decoding of the thoughts of a live behaving experimental animal. The successful decoding demonstrated in 3.3 then motivates the development of a model intended to explain how networks of neurons learn to encode information about the structure of previously unfamiliar parameter spaces (such as the manifold of head orientations as experienced in a new environment) by exploring those spaces.

## 3.2 Broadening the Definition of a Receptive Field to Decode and Model Cognitive Maps of General Parameter Spaces

In analyzing the ability of an adaptive-filter to decode arm movement intentions from the activity patterns of parietal neurons, Section 2.5 alluded to the concept of a neuronal receptive field. The statement that a particular neuron demonstrates 'receptive field' behavior refers to the tendency of that neuron to exhibit an activity pattern that differs from baseline activity when certain stimuli are present. In Section 2.5 and as illustrated in Figure 2-11, parietal neurons were observed to exhibit direction-dependent activity. Similar phenomena are observed in the context of continuous parameter spaces such as those involved in selecting from a continuous range of possible directions as opposed to the discrete set available in the experiments discussed in Section 2.5. Receptive field behavior, and in particular the tendency of individual neurons to exhibit a sharp increase or decrease in activity specifically in the presence of a well defined set of input conditions, is observed in a wide variety of neural systems [16, 2]. One classic example of such a system involves the hippocampal place cell, which exhibits location-specific tuning. The decoding presented in Section 3.3 is based on recordings from a similar population of thalamic cells, whose receptive fields tune to head direction. Place cell receptive fields form the basis of the cognitive map hypothesis of O'Keefe and Nadel, which contends that place cells form the basis of neuronal encoding of physical environments [28]. Section 3.4 proposes a more general conjecture: Receptive-field behavior in populations of neurons can form the basis of neuronal representations of more general parameter spaces. It has been established that the stimuli to which place cells respond, rather than physical locations themselves, are in fact a variety of sensory stimuli associated with those physical locations [2]. Learned associations between combinations of those stimuli and physical locations are responsible for the observation of location-specific receptive fields. If such associations form the basis of the cognitive ability to form representations of

experienced interactions with the physical environment, one wonders how the associations can be learned by populations of neurons. Section 3.4 explores this question. Moreover, regarding sensory stimuli as input parameters in a model neuronal system raises the possibility that not only physical place but also coordinates in more general parameter spaces can be learned and encoded by populations of neurons. The model developed in Section 3.4 gives results that are in qualitative agreement with experimental observations discussed in Section 3.3, and suggests possible means by which biological neural networks form representations of the parameter space trajectories that define their interactions with the environment. It is possible that such representations form the basis of neuronal activity when environmental feedback is absent, such as during thinking and dreaming.

## 3.3 Continuous Real-Time Decoding of Head Direction from Thalamic Neuronal Activity

'Head direction' cells of the rat thalamus are neurons known to exhibit receptive fields tuned to specific orientations of the head relative to the environment [18]. The tuning properties and temporal firing patterns of these and other place-cell–like neurons are typically determined off-line after recording from populations of such cells during behavioral experiments, through statistical analysis of recorded spike trains [20, 19, 21]. This section demonstrates a new technique for on-line interpretation of the tuning properties and temporal firing patterns of head direction cells, based on the neural signal decoding technique developed in Chapter 2.

The experiment used to generate the neuronal firing data whose decoding is described in this section involved a laboratory rat roaming freely in a circular maze for a 30-minute period, during which its position, head direction, and neuronal activity were monitored and recorded [1]. The position and head direction of the animal were

[1]The experimental data used to perform the tests described in this section were very graciously provided by Hector Penagos of the Wilson Laboratory in the Department of Brain and Cognitive Sciences at the Massachusetts Institute of Technology.

tracked using a pair of light-emitting diode arrays mounted on the head, which was imaged at a sampling frequency of 30 Hz using a $300 \times 300$-pixel charge-coupled device (CCD) array. Data from the imager were time-stamped for subsequent synchronization with neural recording data. The neural recordings whose analysis is presented in this section were made by a single tetrode placed in the thalamus of the experimental animal. Spike sorting analysis of the recorded data isolated six neurons, and spike trains from these neurons were used as inputs to the neural signal decoder.

Figures 3-2 and 3-1 illustrate the receptive field tuning properties of the thalamic neurons used as inputs to the decoder. In Figure 3-2, spiking activity in the isolated neurons is plotted as a function of position and head direction. Each of the six plots shown in Subfigures 3-2 (a)–(f) represents spike activity in one of the six cells, and each point represents a single spike. The position of each point corresponds to the location of the rat when the spike was detected, while point color indicates the direction in which the head of the rat was oriented (relative to the positive horizontal position axis) when the corresponding spike was detected. The spatial distribution of points is similar in all six plots, indicating that the thalamic neurons under study have relatively little positional selectivity. By contrast, the points of each subfigure are dominated by a different hue or set of hues, illustrating the relatively high sensitivity of the corresponding cells to particular orientations of the head in absolute space. Figure 3-1 displays plots of spiking activity and head direction as functions of time, both in order to illustrate the receptive field behavior of the isolated neurons and to show that the receptive fields of those cells are distributed over the encoded parameter space (the 0–360° range of possible head direction angles). Figure 3-1 (a) shows the normalized spike rate of each isolated thalamic neuron as a function of time over a 40-second interval during which the rat moved through its maze. A peak in the spike rate of each neuron reflects the location in head-direction space of the receptive field of that neuron. Figure 3-1 (b) plots the head direction of the experimental animal as a function of time over the same interval shown in (a), showing that individual neurons are consistently activated and deactivated as head direction enters or exits the corresponding receptive fields.

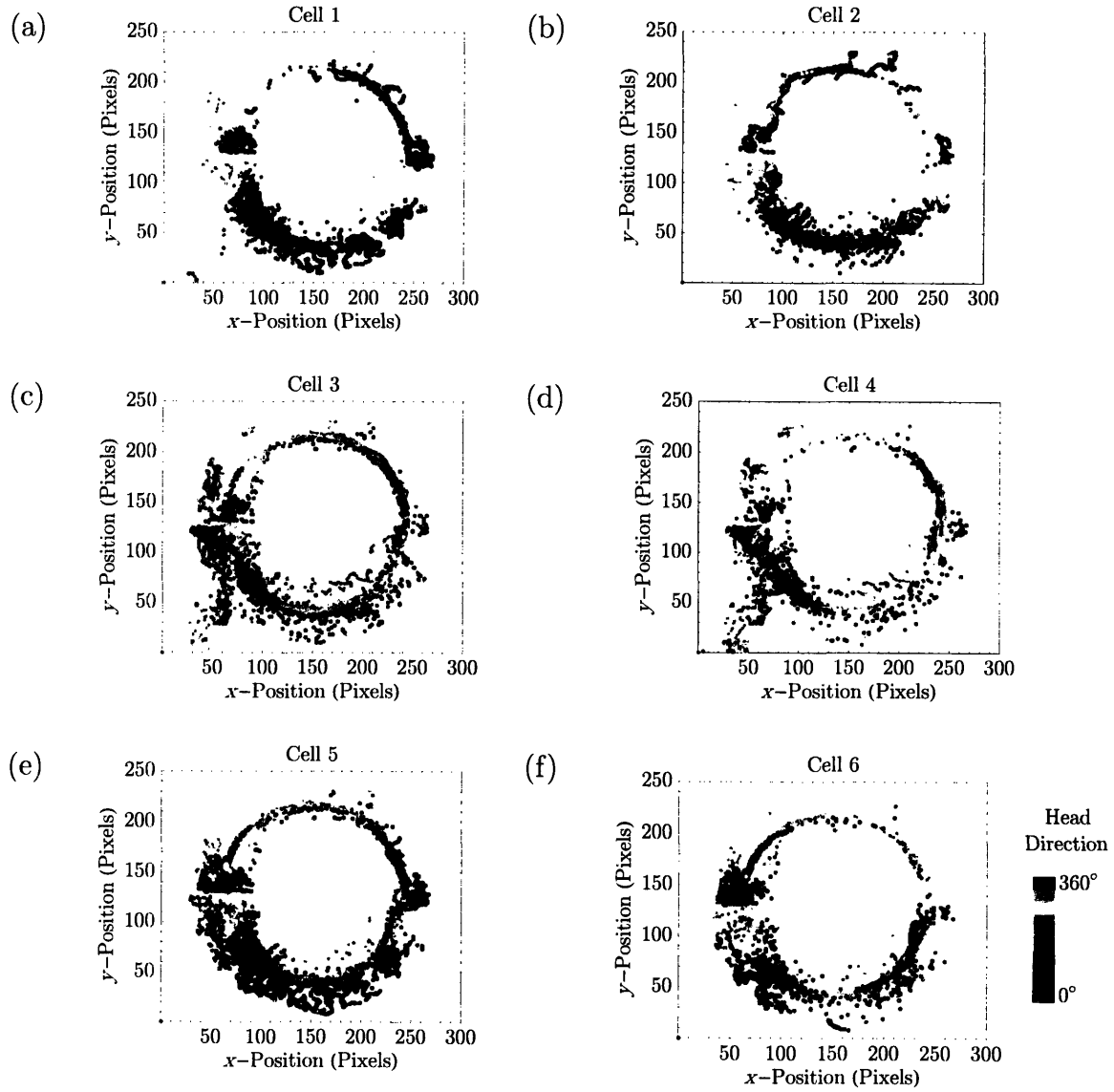



Figure 3-1: **Spike Activity Plotted as a Function of Position and Head Direction Illustrates the Directional Tuning of Six Neurons in the Rat Thalamus.** Images (a) through (e) plot spikes recorded by a single tetrode from six thalamic neurons in a laboratory rat. The rat roamed freely in a circular maze for a 30-minute period, during which its position, head direction, and neuronal activity. Each plot represents spike activity in one of the six cells, and each point represents a single spike. The position of each point corresponds to the location of the rat when the spike was detected, while point color indicates the direction in which the head of the rat was oriented (relative to the positive horizontal position axis) when the corresponding spike was detected. The correspondence between head direction and point color is indicated by the legend beside (f). The spatial distribution of points is similar in all six plots, indicating that the thalamic neurons under study have relatively little positional selectivity. By contrast, the points of each plot are dominated by a different hue, illustrating the relatively high sensitivity of the corresponding cells to particular orientations of the head in absolute space.

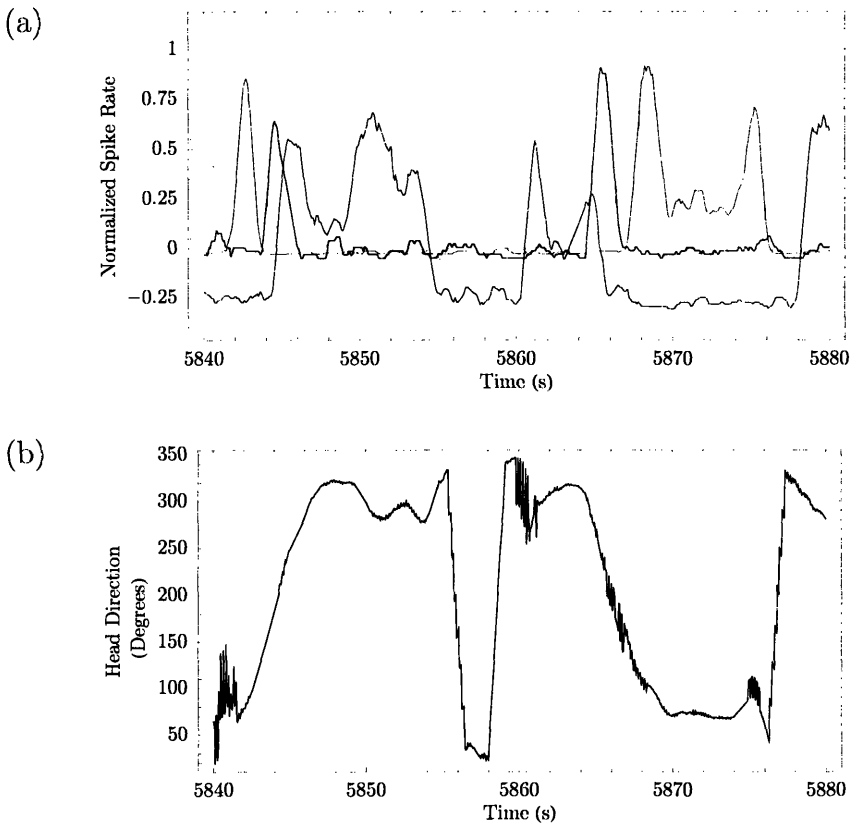



Figure 3-2: **Spiking Activity and Head Direction Plotted as Functions of Time Illustrate Neuronal Receptive Fields and the Distribution of their Peaks Over All Possible Angles.** The two plots shown in this figure correspond to data obtained from a single maze-roaming laboratory rat over the same 40-second time interval. In (a) the normalized spike rate of each isolated thalamic neuron is plotted as a function of time. The tracing for each of cells 1 through 6 has been plotted in a different color, and the correspondence is yellow (1), green (2), light blue (3), dark blue (4), magenta (5), red (6). Plot (b) displays head direction angle as a function of time over the same temporal interval, and its time axis is vertically aligned with that of (a). The pair of plots illustrates the receptive field of each of the isolated neurons, showing that the spiking activity of each cell increases sharply as the rat turns its head toward a particular direction. Moreover, the receptive fields of the six isolated neurons are distributed over the range of possible angles, with preferred orientations (as reflected by peak spiking rates) displaced from one another over the range of possible head direction angles.

Head direction was decoded from the activity of the $n = 6$ isolated thalamic neurons according to the method described in Chapter 2. Using the notation developed in that chapter, the recorded spike train for each of the six neurons was used to construct a single component $\hat{N}_j(t),\ j \in \{1, \ldots, 6\}$ of the raw input waveform $\hat{\mathbf{N}}(t)$. The signal $\hat{N}_i(t)$ was defined as the number of spikes detected from neuron $i$ in the interval $[t - \Delta t, t],\ \Delta t = 1\text{s}$. The actual waveform used as input to the decoder, $\mathbf{N}(t)$, was a transformed version of $\hat{\mathbf{N}}(t)$, normalized by component according to Equation 2.29, which is reproduced as Equation 3.1:

$$N_j = \frac{\hat{N}_j - \langle \hat{N}_j \rangle}{\max \hat{N}_j}. \tag{3.1}$$

The mean indicated by the angled brackets and the maximum appearing in the denominator of Equation 3.1 were computed over an initial recording period of 200s, although as indicated in Section 2.5 it may be convenient in future applications to use continuously updated values for $\langle \hat{N}_j \rangle$ and $\max \hat{N}_j$, computed over moving time windows. The convolution kernels $W_{ij}(t),\ i = 1,\ j \in \{1, \ldots, 6\}$ were again given the form $\frac{A_{ij}}{\tau_{ij}} e^{-\frac{t}{\tau_{ij}}}$ described in Chapter 2. The decoder output was defined as

$$\tilde{M}_{i=1}(t) = \sum_{j=1}^{n=6} W_{ij}(t) \circ N_j(t) \mod 360, \tag{3.2}$$

where $\tilde{M}_1(t)$ is intended to estimate the head direction $M_1(t) \in [0, 360)$ computed from the time-stamped CCD data. The adaptive filter parameters $W_{ij}^{(p)} \in \{A_{ij}, \tau_{ij}\}$ were optimized through gradient descent over training intervals during which the decoder error,

$$e_{i=1}(t) = \left( M_i(t) - \tilde{M}_i(t) \mod 360 \right) - 180 \tag{3.3}$$

was made available to the adaptive filter in the feedback configuration described in

Chapter 2. Following these training intervals feedback was discontinued and the performance of the decoder was assessed by comparing the decoder output $\tilde{M}_i(t)$ as defined in Equation 3.2 with $M_1(t)$ for $t$ outside the training interval.

Figure 3-3 compares the output $\tilde{M}_1(t)$ of the decoder to the measured head direction $M_1(t)$ over a pair of consecutive 200s intervals. During the first interval, corresponding to Figure 3-3 (a), the filter parameters $W_{ij}^{(p)}$ were adapted through gradient descent on the square of the error function given in Equation 3.3. Figure 3-3 (a) shows $\tilde{M}_1(t)$ (black) tracking $M_1(t)$ (blue) with increasing accuracy as training progresses, illustrating that while initial predictions are poor, they improve with feedback over the course of the training interval. Figure 3-3 (b) shows $\tilde{M}_1(t)$ (red) and $M_1(t)$ (blue) over the 200s time interval immediately following training, after feedback has ceased. Qualitatively, the figure illustrates that the output of the neural decoder closely reproduces the shape of the correct waveform, predicting head direction on the basis of neuronal spike rates to within a variable offset and with a slight time delay.

The performance of the decoder in predicting head direction was assessed quantitatively using the normalized mean-squared error measure $\eta^{(1)}$ derived in Section 2.4. In the present system, $\eta^{(1)}$ is defined as

$$\eta^{(1)} = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} dt \left( \frac{e_1(t)}{L_i} \right)^2 \tag{3.4}$$

$$\approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{e_1\left(t_1 + n\frac{t_2 - t_1}{N}\right)}{L_i} \right)^2, \tag{3.5}$$

where $L_i = 180°$, $t_1$ denotes the end of the training interval, and $t_2$ denotes the end of the testing interval over which the performance of the decoder was evaluated. In the error computations described here, decoder error was assessed over an interval of constant length $t_2 - t_1 = 200$s, with decoder error sampled at a rate of $\frac{t_2 - t_1}{N=200} =$ 1 Hz. In order to quantify the accuracy of head direction decoding as a function of training time, $\eta^{(1)}$ was computed for a set of training and decoding trials with

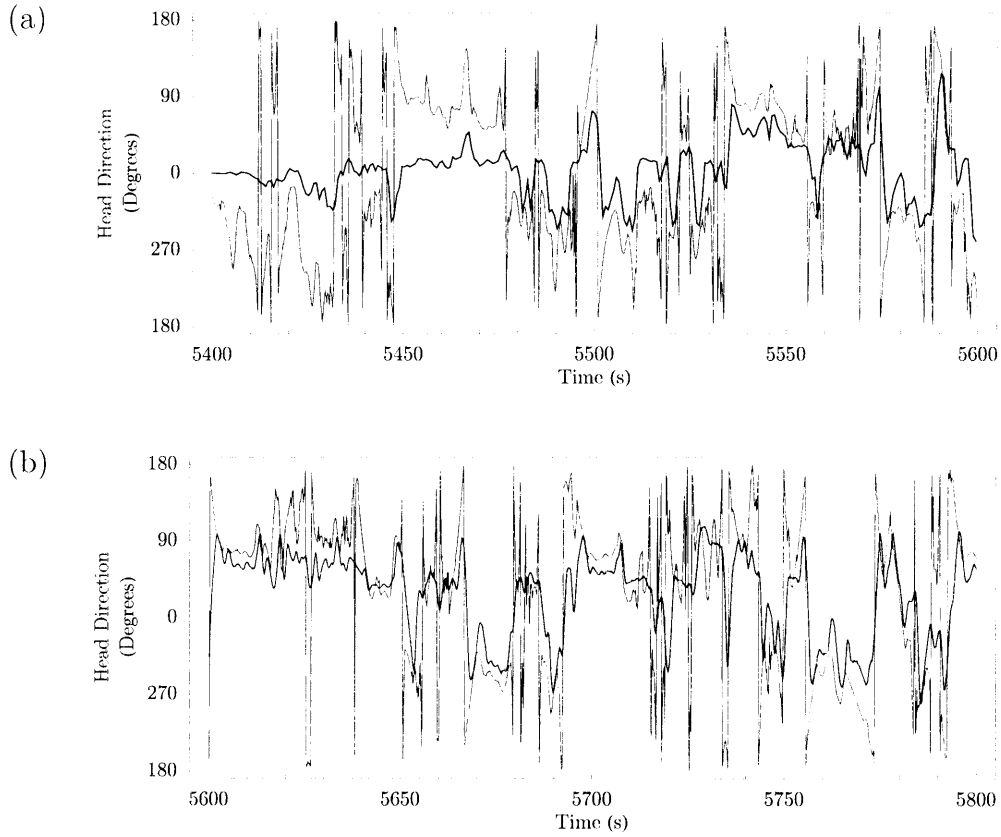



Figure 3-3: **Continuous Decoding of Head Direction from Neuronal Spiking Activity.** The gradient-descent least-squares method of Chapter 2 was used to train an adaptive filter to decode the head direction of a maze-roaming laboratory rat from spike trains recorded from six isolated thalamic neurons. Both training and testing of the adaptive filter were conducted in simulated real-time. Subfigure (a) shows filter output (black, predicted head direction angle) and measured head direction (blue) over the 200s time interval during which the filter parameters were adapted through gradient descent as described in Chapter 2. The black tracing tracks the blue one with increasing accuracy as training progresses, illustrating that while initial predictions are poor, they improve with feedback over the course of the training interval. Subfigure (b) shows filter output (red) and measured head direction (blue) over the 200s time interval immediately following training, after feedback has ceased. The output of the neural decoder predicts head direction on the basis of neuronal spike rates to within a variable offset and with a slight time delay, closely reproducing the shape of the correct waveform.

increasing lengths of the training period, at each training period length averaging over randomized initial settings of the filter parameters and different choices of training interval. The results of this computation are displayed in Figure 3-4, which shows improving accuracy of head direction decoding with increased training.

The values of $\eta^{(1)}$ obtained in head direction decoding are considerably larger than those observed in the systems discussed in Section 2.4. Two principal limitations on decoder accuracy are likely responsible for a considerable portion of this difference in performance. First, the small number of neurons used to provide inputs to the decoder limits decoding resolution due to the size and distribution of neuronal receptive fields over head-direction space. Only $n = 6$ neurons were used in this study, whereas $n \geq 50$ neurons were available to the decoder in Section 2.5. The larger number is much more typical of systems reported in the literature on neural prosthetics [40, 4, 25, 14]. Furthermore, as discussed in Section 2.5 and verified through simulations reported in that section, decoder performance improves as a function of the number of neuronal inputs. A second limitation on head-direction decoding performance is the intrinsic noisiness and grainy quantization of the head direction tracking system, data from which were used both to generate the feedback signal used to train the decoder and to compute $\eta^{(1)}$. The neural signal decoder smoothens some of this noisiness in a way that is consistent with typical speeds of rat head movement, but at the expense of increasing values of the normalized mean-squared error. Insofar as the adaptive filter was required to optimize its parameters on the basis of a noisy feedback signal, it might be regarded as having to solve a problem analogous to that of 'learning from a noisy teacher,' which has been investigated in the machine learning literature and is known to be more difficult than the case of noiseless feedback [10]. The slow rate of improvement in decoder accuracy with increasing training may also be due in part to the noisiness of the available head direction waveform, though the speed of effective training is certainly limited by the physical speed with which the experimental animal moves its head, exposing the decoder to different correct associations of input and output signals.

An approximate physical meaning can be attributed to the parameters $\tau_{1j}$ and $A_{1j}$

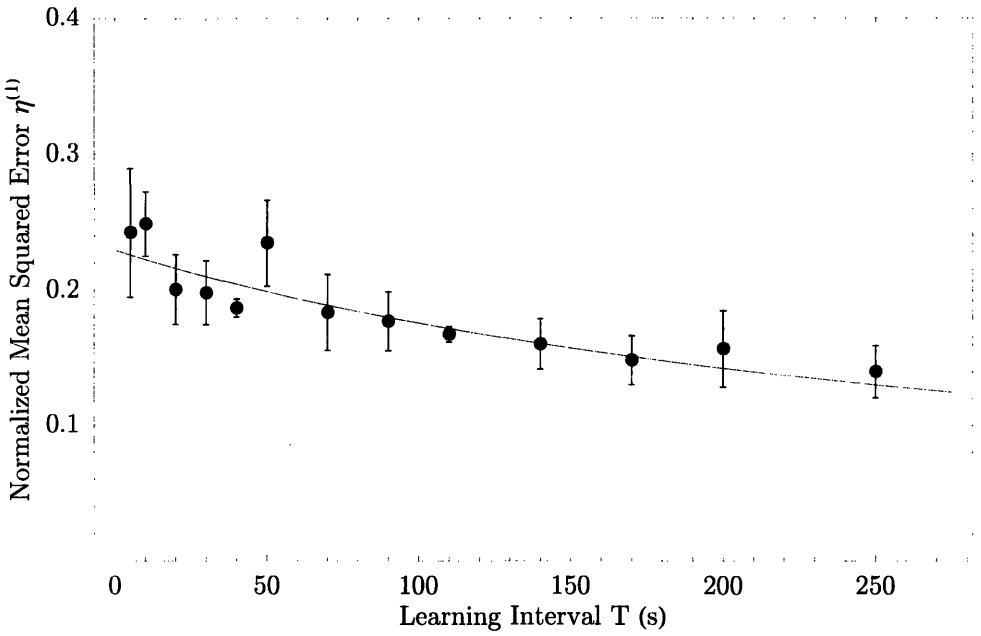



Figure 3-4: **Accuracy of Head Direction Decoding Improves with Increasing Training.** This plot shows the improving accuracy of head direction decoding with increased training as indicated by decreasing values of the normalized mean-squared error, $\eta^{(1)}$ associated with longer training periods. The mean squared error was computed as described in Section 2.4 (but the prediction error was estimated over a 200s interval following training rather than an interval equal in length to the training interval) for a set of training and decoding trials with increasing lengths of the training period, averaging over randomized initial settings of the filter parameters and different choices of training interval. It seems likely that higher values obtained for $\eta^{(1)}$ relative to those observed in the systems discussed in Section 2.4 are due at least in part to the intrinsic noisiness of the head direction tracking system, data from which were used both to generate the feedback signal used to train the decoder and to compute $\eta^{(1)}$. The neural signal decoder smoothens some of this noisiness in a way that is consistent with typical speeds of rat head movement, but at the expense of increasing values of the normalized mean-squared error. The slow rate of improvement in decoder accuracy with increasing training may also be due in part to the noisiness of the available head direction waveform, though the speed of effective training is also limited by the physical speed with which the rat moves its head, exposing the decoder to different correct associations of input and output signals.

learned by the adaptive filter. Since head movement describes a continuous trajectory in the periodic one-dimensional parameter space of head direction angles, neuronal inputs from times $t - \Delta t$ will convey information regarding head direction at time $t$, where $\Delta t \ll \frac{360°}{\nu}$ and $\nu$ denotes a typical angular velocity of head movement. For the $j$th neuron, $\tau_{1j}$ reflects an approximate value for $\Delta t$ over which past firing rates of neuron $j$ contain information about present head direction. Since $N_i(t)$ can only contribute to $\tilde{M}_1$ when $N_i(t) \neq 0$, the time constant $\tau_{1j}$ reflects the width $\Delta\theta_j$ of the receptive field of neuron $j$ in head-direction space and should scale approximately as $\frac{\Delta\theta_j}{\nu}$, where $\theta_j$ denotes the head direction angle at which neuron $j$ is maximally active (the center of its receptive field) and $\Delta\theta_j$ denotes the full width at half maximum of its spike rate as a function of head direction angle, $N_j(\theta)$. The regression coefficients $A_{1j}$ can be used to estimate the values of $\theta_j$ in the approximation of nonoverlapping receptive fields and $\tau_{1j} \ll \frac{\Delta\theta_j}{\nu}$,

$$\theta_j = A_{1j} \frac{\max \hat{N}_j - \langle \hat{N}_j \rangle}{\max \hat{N}_j} \tag{3.6}$$

$$= A_{1j} \max_{\theta} \left( N_j(\theta) \right), \tag{3.7}$$

where $\langle \hat{N}_j \rangle$ and $\max \hat{N}_j$ are defined as in Equation 3.1. Equation 3.7 states that in the indicated approximation, the central tuning angle $\theta_j$ of the receptive field of neuron $j$ can be estimated as the product of $A_{1j}$ and the maximum normalized firing rate of that neuron. This approximation holds because when $\tau_{1j}$ are small the system approximates an instantaneous linear decoder as discussed in Section 2.5, and when receptive fields do not overlap, maximal activity of neuron $j$ indicates $M_1(t) = \theta_j$. Figure 3-5 illustrates the viability of this interpretation of $A_{1j}$. The histograms in Subfigures 3-5 (a)–(f) show the probability densities for neurons 1 through 6, respectively, to spike as a function of the head direction angle of the experimental animal. They were computed from neuronal recording data obtained over a 30-minute period during which the rat roamed freely through its maze. The grey vertical bar in the histogram for the $j$th cell is centered on the head direction angle computed

using Equation 3.7 from a value of $A_{1j}$ obtained after averaging over randomized initial settings of the filter parameters and different choices of training interval. The width of the $j$th grey bar spans one standard deviation in either direction from the predicted value of preferred head direction. These estimates identify the preferred head direction of three of the six observed neurons, and the standard deviation of the estimation error

$$A_{1j} \max_{\theta} (N_j(\theta)) - \theta_j \tag{3.8}$$

for the set of six observed neurons is 51°.

## 3.4 A Model for Learning and Neural Encoding of Parameter Space Structures

The ability of the adaptive filter decoding technique to learn to interpret the activity patterns of a population of place-like–neurons in real time, as demonstrated in Section 3.3, suggests that this method of neural signal decoding might be generally applicable to studying the ways in which experience-related parameters are encoded in biological neural networks. Section 2.5 demonstrated that this decoding method can interpret motor parameters encoding the directions of intended arm movements, while Section 3.3 demonstrated that the decoder can interpret the activity of head direction cells, whose activity reflects the presence of combinations of sensory parameters. Analysis of the adaptive filter parameters learned by the decoder applied to both systems revealed that those parameters reflect the tuning properties of the receptive fields of the neurons whose activity patterns were used as input signals for the decoder. The derivation of the decoding system, presented in Section 2.2, indicates that successful decoding would be possible for input signals encoding information in ways other than through receptive-field–style tuning. However, the prevalence of receptive fields as an encoding scheme in biological neural networks suggests that a model of receptive-

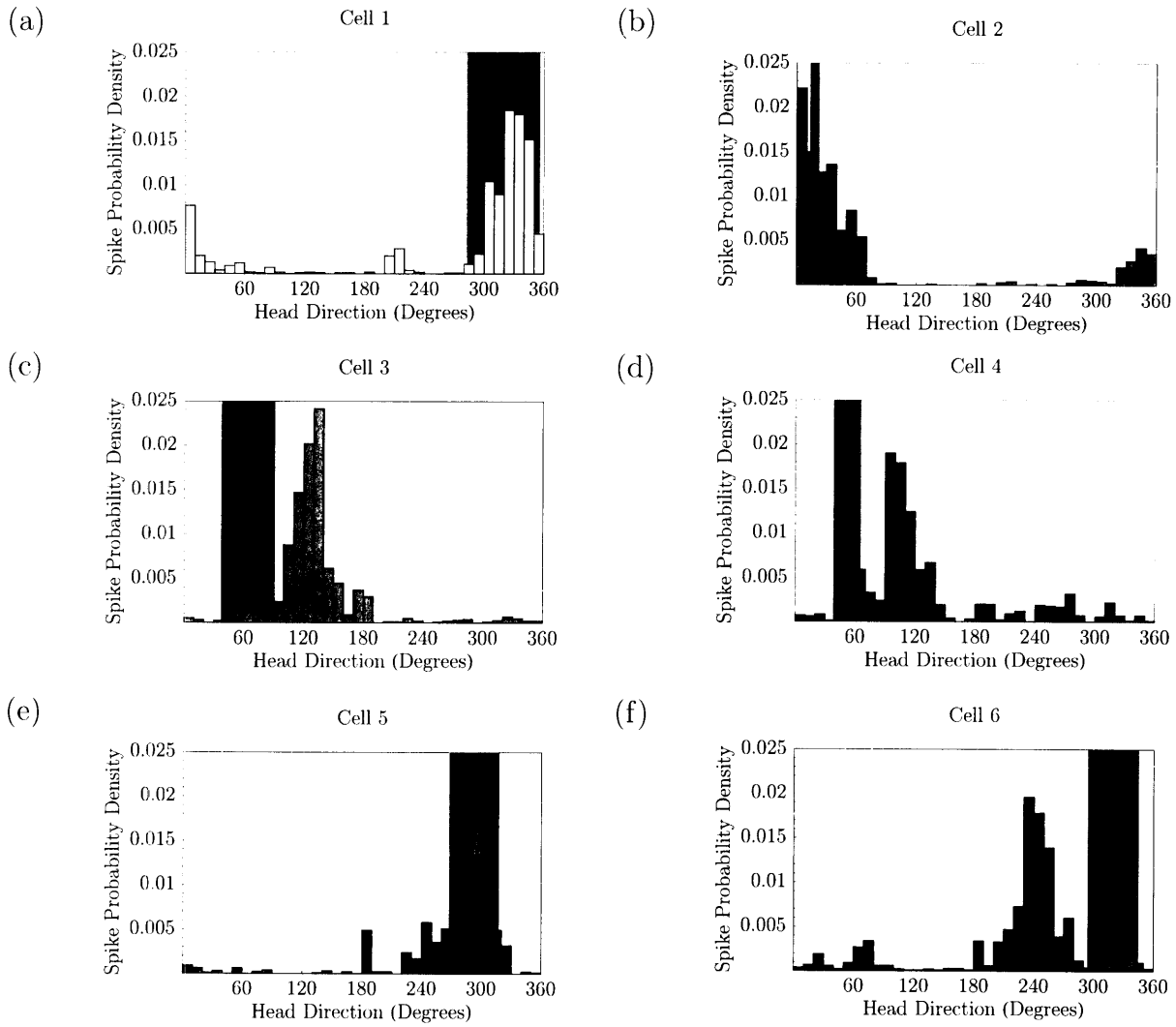



Figure 3-5: **Learned Filter Parameter Values Correlate with Receptive Field Tuning of Neurons Used for Decoder Input.** The adaptive filter parameters $A_{1j}$, $j \in \{1, \ldots, 6\}$ learned by the decoder can be interpreted as approximations of the preferred head direction angle of the corresponding neurons. The accuracy of this approximation is limited by the widths and degree of overlap among the receptive fields of the neurons providing the decoder inputs, as well as by the degree to which decoder output depends on past inputs (reflected by the nonzero values of $\tau_{1j}$, $j \in \{1, \ldots, 6\}$). The histograms in (a) through (f) show the probability densities for neurons 1 through 6, respectively, to spike as a function of the head direction angle of the rat. They were computed from neuronal recording data obtained over a 30-minute period during which the rat roamed freely through its maze. The grey vertical bar in the histogram for the $j$th cell is centered on the head direction angle computed from a value of $A_{1j}$ obtained after averaging over randomized initial settings of the filter parameters and different choices of training interval. The width of the $i$th grey bar spans one standard deviation in either direction from the predicted value of preferred head direction. These estimates identify the preferred head direction of three of the six observed neurons.

field–based encoding of general parameter spaces by neural networks could be a useful counterpart to the decoding system capable of interpreting the activity patterns of such systems. This section begins the development of such a model.

How do biological neural networks make sense out of new experiences in unfamiliar environments? Regarding unfamiliar environments as regions of parameter space coordinatized by values of sensory stimuli, and modeling experiences as trajectories in those parameter spaces, the question just posed can be reinterpreted as asking how neural networks learn the topological structure of new parameter spaces. Parameter space trajectories experienced through exploration of a new environment convey topological information about parameter space structure by indicating continuous paths through the space. The model presented in the following subsections seeks to address two questions. First, Subsection 3.4.1 addresses the question of how neural networks can learn to represent position in unfamiliar parameter spaces by adapting the distribution of neuronal receptive fields in response to new stimuli. Second, Subsection 3.4.2 addresses the question of how a neural network can learn allowed parameter space trajectories. Subsection 3.4.3 indicates investigational lines along which this model could be refined and extended.

### 3.4.1 Adaptation of Receptive Fields to a New Parameter Space

In order to motivate the model of the present subsection through continuity with the decoding problem of Section 3.3, consider the problem faced by a population of head direction cells in the rat thalamus when the rat is introduced into an unfamiliar maze. The range of sensory parameter values corresponding to what the animal sees, hears, and feels with its head oriented in various directions in this new environment might be entirely different from the range of sensory stimuli it has experienced in other environments. How does the population learn the range of values and combinations of sensory stimuli that correspond to the possible head orientations in this new environment? A plausible hypothesis, grounded in experimental observations concerning

receptive field encoding, is that the neuronal population adapts the configuration of its receptive fields, in response to experience, toward a distribution whose activity as a function of sensory-parameter–space position conveys a maximal amount of information regarding head direction. The model presented in this subsection proposes one scheme for such an adaptive reconfiguration of receptive fields.

Let a network of $n$ neurons sensitive to stimuli $\{s_1, \ldots, s_d\}$ defining a $d$-dimensional parameter space exhibit parameter-space-position–dependent activity (analogous to normalized spike rate) functions $\{f_1, \ldots, f_n\}$. Each functions $f_i$ is characterized by a preferred point $\vec{x}_i$ in parameter space and a set of $d$ distribution half-widths $\{w_i^{(j)}\}$, $j \in \{1, \ldots, d\}$ in each parameter-space direction. The preferred point $\vec{x}_i$ represents the center of the receptive field of neuron $i$ and is coordinatized by $(s_1(\vec{x}_i), \ldots, s_d(\vec{x}_i))$. A convenient family of bell-shaped functions with these properties can be obtained from quartic polynomials restricted to domains defined by the half-widths $w_i^{(j)}$:

$$f_i(s_1, \ldots, s_d) = \prod_{k=1}^{k=d} \phi_k(s_k) \tag{3.9}$$

$$\phi_k(s_k) = \begin{cases} 0 & |s_k - s_k(\vec{x}_i)| \geq w_i^{(k)} \\ \frac{1}{\left(w_i^{(k)}\right)^4} \left((s_k - s_k(\vec{x}_i)) - w_i^{(k)}\right)^2 \left((s_k - s_k(\vec{x}_i)) + w_i^{(k)}\right)^2 & |s_k - s_k(\vec{x}_i)| < w_i^{(k)}. \end{cases} \tag{3.10}$$

The definitions of Equations 3.9 and 3.10 construct the $f_i$ so that each $f_i$ is maximized at $x_i$, with $f_i(x_i) = 1$; $f_i(\vec{x}) = 0$ for $\vec{x}$ outside the $d$ dimensional hyperprism centered at $\vec{x}_i$ and having edges of length $2w_i^{(j)}$ in each of the $d$ parameter space directions $j \in \{1, \ldots, d\}$; and $f_i(\vec{x})$ decreasing monotonically with distance from $\vec{x}_i$ to the boundaries of the indicated hyperprism. Suppose further that the neurons are mutually connected so that each neuron receives a signal $\mathbf{F}(t) = (f_1(t), \ldots, f_n(t))$ describing the activity of all other neurons as a function of time.

When the network is first exposed to an unfamiliar region $X$ of parameter space corresponding to a new environment, the receptive fields modeled by the $f_i$ are

assumed to be randomly tuned, with the $\vec{x}_i$ and $w_i^{(j)}$ uncorrelated with one another and with the structure of $X$. Criteria for an optimal configuration of the receptive fields might require that there be minimal overlap among the receptive fields $f_i$ but that every point $\vec{x} \in X$ correspond to a nonzero activity pattern $\mathbf{F}(\vec{x}) = (f_1(t), \ldots, f_n(t)) \neq (0, \ldots, 0)$. One approach to optimizing the configuration of receptive fields in this sense is to adapt the parameters $\vec{x}_i$ and $w_i^{(j)}$ in response to the activity patterns $\mathbf{F}(\vec{x}(t))$ experienced as $X$ is explored along parameter space trajectories $\vec{x}(t)$, as follows. At every time step $t$ all pairs of neurons are considered in turn. The half-width parameters $w_i^{(j)}$ are modified according to

$$w_i^{(j)}(t + \Delta t) = w_i^{(j)}(t) \prod_{k \neq i} (1 + \epsilon)\, e^{-(f_i(\vec{x}(t)) f_k(\vec{x}(t)))}, \tag{3.11}$$

where $\epsilon$ is a small parameter that sets the threshold for receptive field overlap. The iterative modification of half-widths defined in Equation 3.11 narrows the widths of receptive fields that are coactive at a given point $\vec{x}(t)$ in parameter space, sharpening receptive fields around their maxima. But in order to ensure nonzero activity in the population at all $\vec{x} \in X$, receptive field half-widths are incremented if the level of coactivity of neurons $i$ and $k$, as measured by the product $f_i(\vec{x}(t))\, f_k(\vec{x}(t))$ of their activity functions, is less than the overlap threshold $-\ln\left(\frac{1}{1+\epsilon}\right) \approx \epsilon$. Receptive field centers are modified at $t$ in a repulsive or attractive manner depending on the magnitude of receptive field coactivity:

$$\vec{x}_i(t + \Delta t) = \vec{x}_i(t) + \sum_{k \neq i} (\vec{x}_i - \vec{x}_k)\, (f_i(\vec{x}(t))\, f_k(\vec{x}(t)) - \epsilon), \tag{3.12}$$

so that receptive field centers interact repulsively or attractively when the coactivity of their fields is greater or less than $\epsilon$, respectively.

Figure 3-6 presents the results of a simulation of the adaptive scheme outlined in this subsection for modifying the receptive field configuration of a neural network

with $n = 6$ exploring a $(d = 1)$-dimensional periodic parameter space, for the purpose of comparison with experimental data observed in Section 3.3 in the context of head direction cell receptive fields. Subfigures 3-6 (a)–(f) illustrate the adaptation of the receptive fields as the system explores the 360 degrees of periodic parameter space along a trajectory of the form $\vec{x}(t) = t \mod 360$. Initially the receptive fields are broadly and randomly tuned to inputs from the new parameter space, and the series of subfigures illustrates slow convergence toward a stable equilibrium configuration in which receptive fields distribute over the new parameter space in a way that optimizes the information content conveyed by their joint firing functions with respect to parameter space position. The displayed results also bear a qualitative similarity to the experimental observations illustrated in Figure 3-2, which shows real activity in a population of $n = 6$ thalamic neurons as a function of the head direction of a laboratory rat exploring a circular maze. This resemblance supports the possibility that a similar adaptive mechanism may be at work in biological neural networks for learning the structure of unfamiliar parameter spaces.

### 3.4.2 Neural Network Learning of Parameter Space Trajectories

After the receptive fields of a neural network have assumed a suitable distribution over the parameter space region of interest, the network faces the problem of having to learn the structure of that region. When external stimuli are present, the temporal sequence of sensory parameter stimuli experienced by the network defines a trajectory in parameter space, $\vec{x}(t)$, as described in Subsection 3.4.1, that determines the state $\mathbf{F}(\vec{x}(t)) = (f_1(\vec{x}(t)), \ldots, f_n(\vec{x}(t)))$ of the network as a function of time. Periods when stimuli are present can be understood as training intervals for the network. Importantly, however, place cells also exhibit activity in the absence of external stimuli. In particular, a fascinating set of experiments has shown that firing patterns observed in populations of place cells observed during exploratory behavior in awake animals are recapitulated during sleep, in the absence of the original environmental stimuli

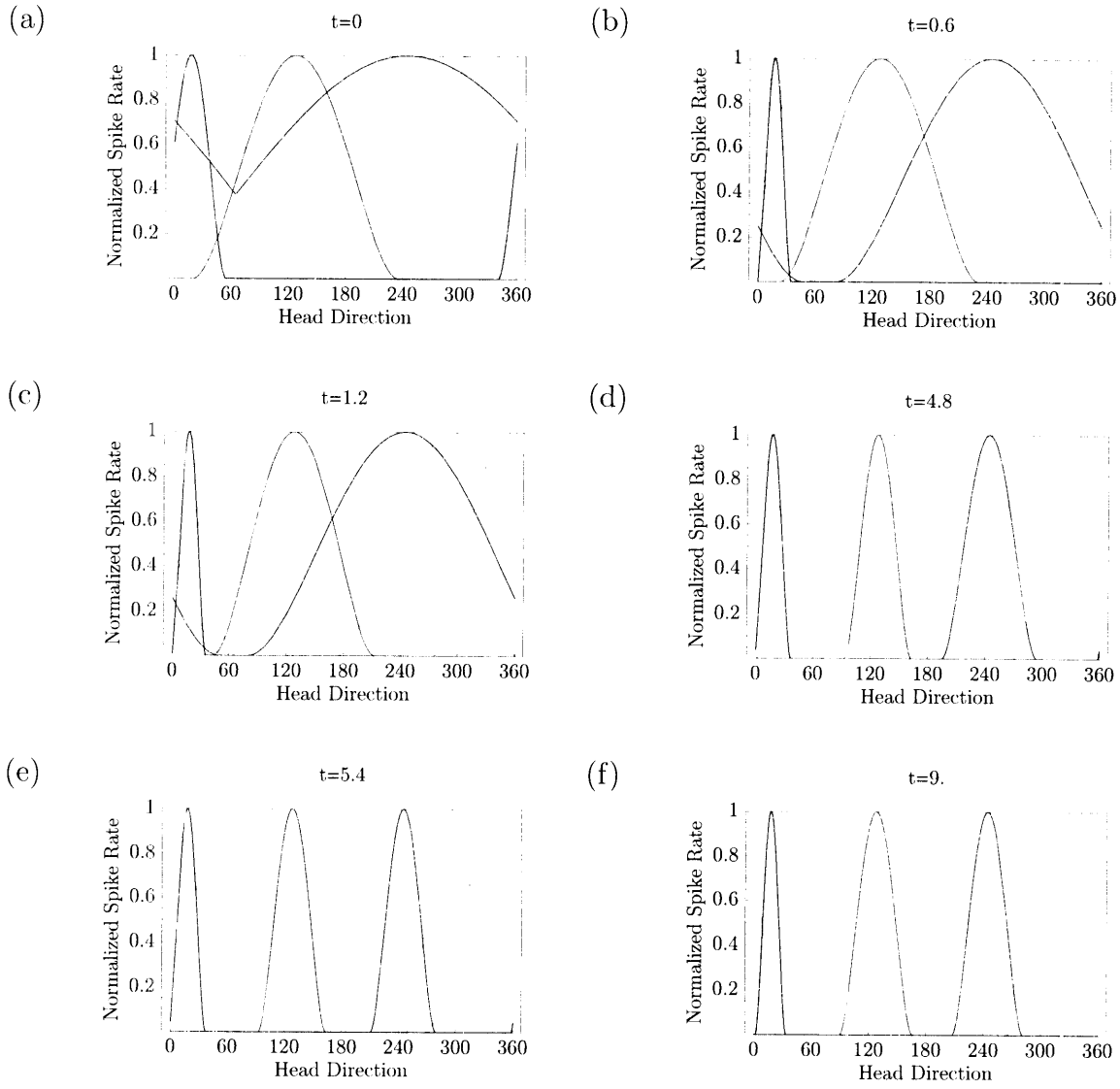



Figure 3-6: **Adaptive Sharpening of Receptive Field Tuning in Model Neurons Exposed to Inputs from a New Parameter Space.** Subfigures (a) through (f) show the timecourse of adaptive sharpening of the receptive fields of six model neurons exposed to inputs from a new parameter space. For concreteness, the parameter space coordinates are associated with head direction. Initially the receptive fields are broadly and randomly tuned to inputs from the new parameter space. In the simulation that generated these plots, the system explored the new parameter space along a trajectory corresponding to constant-direction cycles through head direction angles. Adaptive sharpening involved coactive neurons reducing their receptive field widths and shifting their field centers in an activity-dependent fashion. The series of figures illustrates that this scheme for modifying neuronal receptive fields enables a population of neurons to spread over a new parameter space in a way that raises the information content conveyed by their joint firing functions with respect to parameter space position. These results are also qualitatively similar to the experimental observations illustrated in Figure 3-2.

[20, 19]. These observations suggest that networks of place-like cells are capable of learning parameter space trajectories and in that sense forming representations of parameter space structure internal to the network. This subsection proposes a model for how neural networks can learn paramter space trajectories and reproduce them in the absence of feedback.

Suppose that after an initial period of adaptation the receptive field distribution of an $n$-neuron network has stabilized so that every point $\vec{x}$ in a new region $X$ of parameter space is associated with a unique network state $\mathbf{F}(\vec{x}) = (f_1(t), \ldots, f_n(t))$. Suppose further, as in Subsection 3.4.1, that the $n$ neurons are mutually interconnected. In the presence of a parameter space input $\vec{x}(t)$ at time $t$, the state of the network is determined by $\vec{x}$ and is $\mathbf{F}(\vec{x}(t)) = (f_1(\vec{x}(t)), \ldots, f_n(\vec{x}(t)))$. In the absence of an input $\vec{x}(t) \in X$, however, the state of the network at time $t$ is a function of its state $\mathbf{F}(t' < t)$ at earlier times, and of the strengths of the mutual synaptic connections among the networked neurons. As a spatially and temporally discretized analogy to the neural decoder used in previous sections, consider time to proceed in steps of size $\Delta t$ and let the neural network state at $\mathbf{F}(t+\Delta t)$ at $t+\Delta t$ be a function of its state at the two previous time steps, $\mathbf{F}(t)$ and $\mathbf{F}(t-\Delta t)$. The ability to distinguish among several trajectories passing through a point $\vec{x} \in X$ requires more information than simply $\mathbf{F}(t)$, and so allowing $\mathbf{F}(t + \Delta t)$ to depend on $\mathbf{F}(t - \Delta t)$ is analogous to providing velocity information at time $t$ and as an additional initial condition, enabling the network to resolve such ambiguities. Further let the parameter space region $X$ consist of a lattice of $\ell$ discrete points $\{\vec{x}_{(0)}, \ldots, \vec{x}_{(\ell-1)}\}$. Then each point $\vec{x}_{(i)}$ corresponds to a unique network state $|i\rangle$, defined as

$$|i\rangle \equiv \mathbf{F}\left(\vec{x}_{(i)}\right). \tag{3.13}$$

Using this notation, the network state at time $t + \Delta t$ can be expressed as

$$|s(t+\Delta t)\rangle = \mathbf{T}|s(t-\Delta t)\rangle \otimes |s(t)\rangle, \tag{3.14}$$

where $s(t') \in \{0, \ldots, \ell-1\}$ and the transition operator $\mathbf{T}$ reflects the synaptic weights defining the directed, asymmetric connections among the $n$ neurons. Since $|s(t+\Delta t)\rangle$ is a function of $|s(t)\rangle$ and $|s(t-\Delta t)\rangle$, it is convenient to represent $\mathbf{T}$ as an $\ell^2 \times \ell^2$ matrix operating on the product space having elements of the form $|s(t-\Delta t)\rangle \otimes |s(t)\rangle$ and generating time-translated product states $|s(t)\rangle \otimes |s(t+\Delta t)\rangle$:

$$\mathbf{T}|s(t-\Delta t)\rangle \otimes |s(t)\rangle = |s(t)\rangle \otimes |s(t+\Delta t)\rangle. \tag{3.15}$$

In this description, the problem of enabling the neural network to learn parameter space trajectories consistent with the structure of $X$ on the basis of trajectories $\vec{x}(t)$ presented through sensory experience can be framed as a problem of adapting the transition matrix $\mathbf{T}$. Trajectories can be embedded in $\mathbf{T}$ through learning according to a modified Hebbian adaptation scheme. At each time step $t$ along a presented trajectory $\vec{x}(t)$, the matrix element $\langle s(\vec{x}(t))| \otimes \langle s(\vec{x}(t+\Delta t))|\mathbf{T}|s(\vec{x}(t-\Delta t))\rangle \otimes |s(\vec{x}(t))\rangle$ corresponding to the correct forward-time transition along the trajectory, is incremented; the matrix element $\langle s(\vec{x}(t-\Delta t))| \otimes \langle s(\vec{x}(t))|\mathbf{T}|s(\vec{x}(t))\rangle \otimes |s(\vec{x}(t+\Delta t))\rangle$, corresponding to the backward-time transition along the trajectory, is decremented; and matrix elements corresponding to all other possible transitions are decayed by a small amount. This learning scheme can be implemented by restricting all elements of $\mathbf{T}$ to $[0, 1]$ and choosing a small parameter $\epsilon < 1$ to set the rate of matrix element adaptation. Matrix element incrementing can be accomplished by the transformation

$$\begin{aligned} &\langle s(\vec{x}(t))| \otimes \langle s(\vec{x}(t+\Delta t))|\mathbf{T}|s(\vec{x}(t-\Delta t))\rangle \otimes |s(\vec{x}(t))\rangle \\ \rightarrow\ &\left(\langle s(\vec{x}(t))| \otimes \langle s(\vec{x}(t+\Delta t))|\mathbf{T}|s(\vec{x}(t-\Delta t))\rangle \otimes |s(\vec{x}(t))\rangle\right)^{\epsilon}, \end{aligned} \tag{3.16}$$

while matrix element decrementing can be accomplished using the inverse transformation

$$\begin{aligned} &\langle s\left(\vec{x}(t-\Delta t)\right)| \otimes \langle s\left(\vec{x}(t)\right)|\mathbf{T}|s\left(\vec{x}(t)\right)\rangle \otimes |s\left(\vec{x}(t+\Delta t)\right)\rangle \\ &\rightarrow \left(\langle s\left(\vec{x}(t-\Delta t)\right)| \otimes \langle s\left(\vec{x}(t)\right)|\mathbf{T}|s\left(\vec{x}(t)\right)\rangle \otimes |s\left(\vec{x}(t+\Delta t)\right)\rangle\right)^{\frac{1}{\epsilon}}. \end{aligned} \tag{3.17}$$

The matrix element decay transformation must be chosen so that if the mean recurrence time for a trajectory is $r\Delta t$, $r-1$ decays followed by one increment result in an overall increment. If the decay transformation is to have a form $T_{ij} \rightarrow T_{ij}^{d}$ similar to that of the incrementing and decrementing functions, then the decay exponent is therefore bounded from above by the condition

$$\left(T_{ij}^{(n-1)d}\right)^{\epsilon} \geq T_{ij} \tag{3.18}$$

$$(n-1)d\epsilon \leq 1 \tag{3.19}$$

$$d \leq \frac{1}{(n-1)\epsilon}. \tag{3.20}$$

The decay exponent is also bounded from below by $1 < d$, since $0 \leq T_{ij} \leq 1$ and decay implies $T_{ij}^{d} \leq T_{ij}$. Choosing $1 < d = \frac{1}{2} + \frac{1}{2(n-1)\epsilon} < \frac{1}{(n-1)\epsilon}$ therefore completes the learning rule.

Figure 3-7 graphically depicts the process of learning the matrix $\mathbf{T}$ of state transitions corresponding to the set of allowed parameter space trajectories when $X$ is a one-dimensional periodic lattice containing $\ell = 6$ sites. This parameter space structure only admits state transitions of the form $|i\rangle \rightarrow |i \pm 1 \mod \ell\rangle$ between neighboring lattice sites, so learning the structure of this parameter space amounts to learning to transform product states of the form $|s(t-\Delta t)\rangle \otimes |s(t)\rangle$ into corresponding states $|s(t)\rangle \otimes |s(t+\Delta t)\rangle$ such that $s(t) = s(t-\Delta t) \pm 1 \mod \ell$ and $s(t+\Delta t) = s(t) \pm 1 \mod \ell$, where the $\pm$ designates the sense of the trajectory around the lattice. For $\ell = 6$, $\mathbf{T}$ is an $(\ell^2 \times \ell^2 = 36)$-dimensional matrix, represented in each of Subfigures

3-7 (a)–(f) by an array of shaded squares. Each matrix element is represented by a square whose shade is a function of matrix element value, with 0 and 1 represented by black and white, respectively, and intermediate values represented by interpolated shades of gray.

In the example presented here, the neural network learned the allowed transitions by exploring the parameter space in both possible directions along a trajectory that periodically reversed direction, with synaptic weights evolving according to the modified Hebbian scheme described earlier. The result of this learning scheme is that a randomly initialized synaptic weight matrix $\mathbf{T}(t = 0)$, depicted in Subfigure 3-7 (a), evolves over time as depicted in Subfigures 3-7 (b)–(e) into one that encodes the trajectories permitted by the structure of the parameter space, as the system explores that space. The perfect transition matrix, shown in Subfigure 3-7 (f), is nearly identical to the matrix obtained at the end of the learning interval, shown in 3-7 (e).

Figure 3-8 illustrates the ability of the trained neural network to reproduce allowed parameter space trajectories after learning, in the absence of external stimuli. Matrix elements $T_{ij}$ can be interpreted as transition probability amplitudes, and when $\mathbf{T}$ has been learned perfectly its matrix values assume only values of 1 or 0 so that a coherent state $|s(t-1)\rangle \otimes |s(t)\rangle$ is transformed into the next state in the trajectory as a coherent state, $|s(t)\rangle \otimes |s(t+1)\rangle$. A sequence of such transformations, constituting replication of a learned parameter space trajectory, is illustrated in Subfigure 3-8 (a), which shows the time evolution of the neural network encoding the one-dimensional periodic lattice with $\ell = 6$ sites described earlier. The transition matrix was learned almost perfectly, so that after presentation with an initial condition the system cycles through the lattice sites, occupying each in turn with nearly unit probability. Subfigure 3-8 (b), by contrast, illustrates the result of learning and subsequent performance under noisy conditions in which the system occasionally experienced disallowed transitions during learning, so that the learned matrix $\mathbf{T}$ permits transitions of the form $|i\rangle \rightarrow |j \neq i \pm 1 \mod \ell\rangle$ with small probability. The result following learning is decaying occupation probability of the correct-trajectory state over time, with leakage of probability flux to multiple states so that coherent initial states are observed to decohere over time.

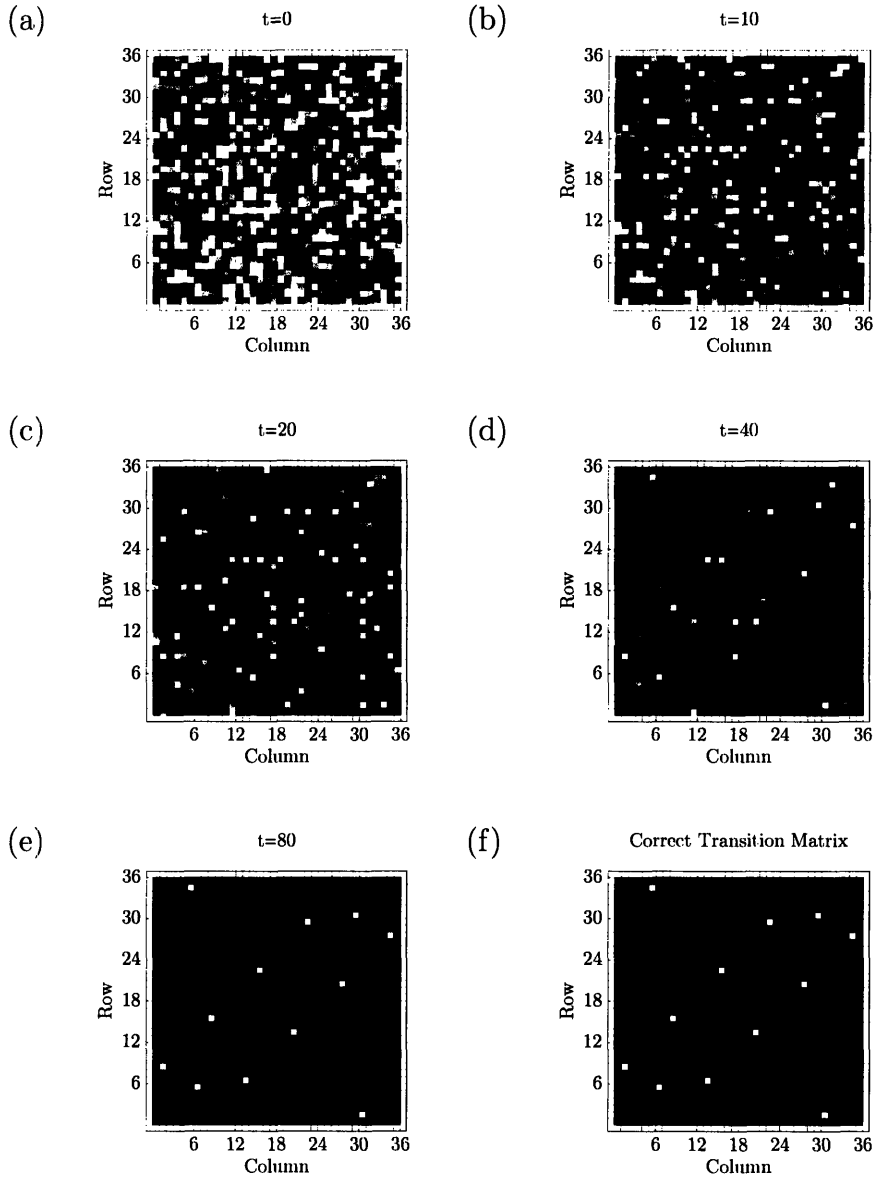



Figure 3-7: **Learning the Matrix of State Transitions Corresponding to a Set of Parameter Space Trajectories.** The synaptic weight matrix $\mathbf{T}$ that translates neural network states forward in time, defining the parameter space trajectories the network can store and reproduce. The parameter space considered is a 1-dimensional periodic lattice of $\ell = 6$ sites, so $\mathbf{T}$ is $(36 \times 36)$. The only permissible state transitions are of the form $|i\rangle \rightarrow |i \pm 1 \mod \ell\rangle$, so the correct form (f) of $\mathbf{T}$ has 12 elements equal to 1 and all others zero. The network learns allowed transitions by exploring parameter space in both directions, with $T_{ij}$ evolving according to an asymmetric modified Hebbian scheme augmented by a decay rule. Learning enables a randomly initialized $\mathbf{T}(t = 0)$, depicted in (a), to evolve through exploration of parameter space as depicted in (b)–(e), to encode the trajectories permitted by the structure of the space. Matrix elements are represented by squares shaded according to element value. Black and white represent 0 and 1, respectively, and interpolated shades of gray represent intermediate values.

As the noise is also present after learning, it can present new initial conditions to the network at random, resulting in apparent reversals of trajectory direction (compare the trajectory from $t = 10$ to $t = 20$ with that from $t = 30$ to $t = 40$) as well as superposition states of both the forward and reverse trajectories (observed here starting at $t = 0$ and again at $t = 70$).

If dreaming is identified with post-learning (feedback-free or external-stimulus–free) activity in biological neural networks, the system properties illustrated in Subfigure 3-8 (b) suggest a model for some phenomena alluded to earlier in this section. In particular, the observations presented in Subfigure 3-8 (b) suggest an explanation for the observation that rat place cells exhibit activity patterns during dreaming similar to those observed during maze-roaming (parameter space exploration). The model presented here suggests that transitions between learned parameter space trajectories (analogous to 'trains of thought') that occur during dreaming may be due to the decay of one recurrent neural activity pattern followed by the noise-induced initiation of another, where noise sources could include sensory stimuli present during sleep. The trajectory superposition phenomenon illustrated in Subfigure 3-8 (b) might explain in part the appearance of never-experienced associations common in human dreams.

### 3.4.3 Refining and Extending the Model of Neural Network Learning of Parameter Space Trajectories

The model for neural network learning of parameter space structure presented in Section 3.4 raises a number of questions that indicate interesting directions for continued investigation. In particular, the model was illustrated for a simple, one-dimensional topology, explored in discretized space and time. It is logical to ask whether the model can be extended to more complicated topologies, higher-dimensional parameter spaces, and continuous space and time. Furthermore, the notion of learning parameter space structure through exploring allowed paths and then reproducing them in the absence of input parameters suggests that the principal structural information derived from exploratory learning is topological. However, since movement

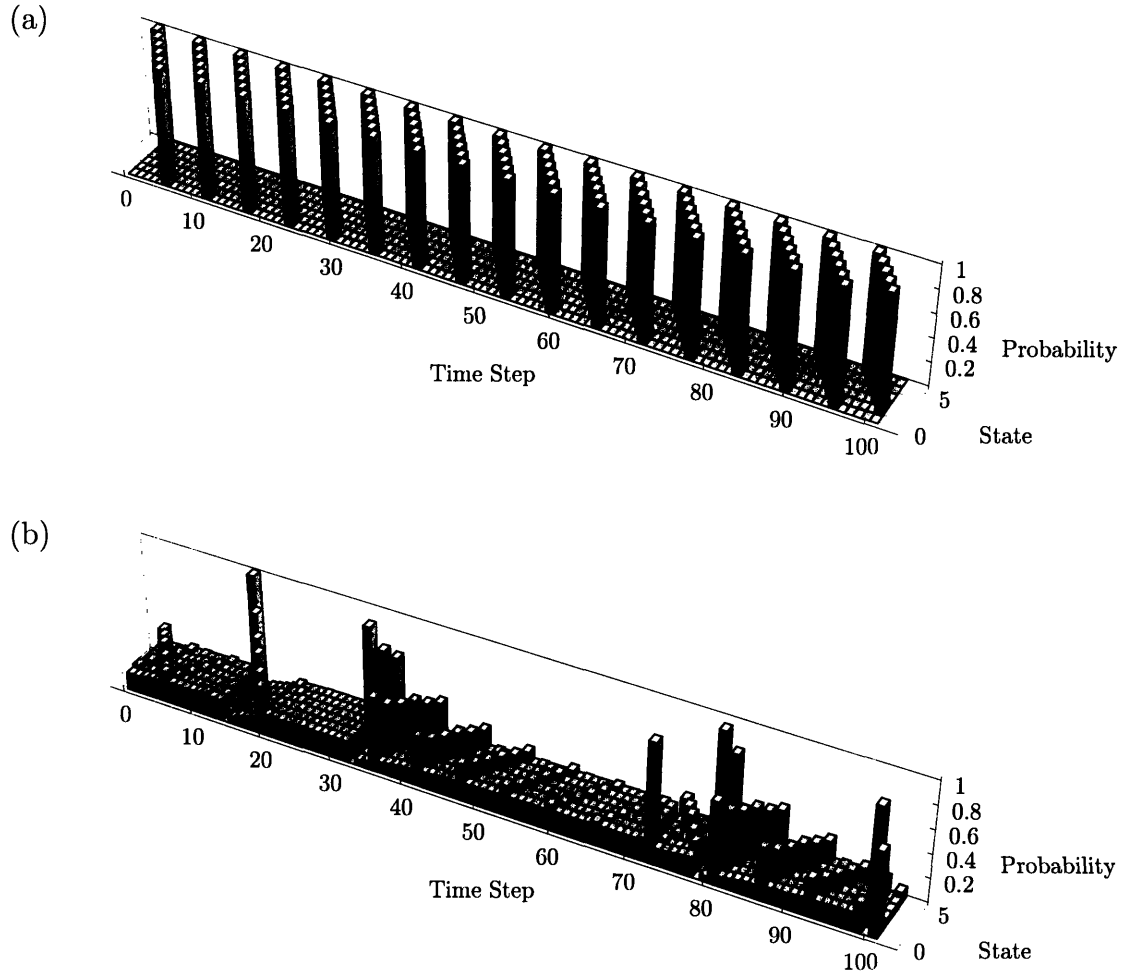



Figure 3-8: **Presence or Absence of Noise During and After Learning Determines Whether Trajectories are Followed Indefinitely and Whether Trajectory Switching Can Occur.** Subfigures (a) and (b) illustrate how state occupation probabilities evolve over time after the synaptic weight matrix $\mathbf{T}$ has been learned and the system initialized in state $|0\rangle \otimes |\ell - 1\rangle$. Subfigure (a) illustrates the result of learning under noiseless conditions. Following learning and the presentation of the initial condition the system evolves indefinitely according to $|i\rangle \rightarrow |i - 1 \mod \ell\rangle$. By contrast, (b) illustrates the result of learning under noisy conditions in which the system occasionally experienced disallowed transitions during learning, so that the learned matrix $\mathbf{T}$ permitted transitions of the form $|i\rangle \rightarrow |j \neq i \pm 1 \mod \ell\rangle$ with small probability. The result following learning is decaying occupation probability of the correct-trajectory state over time, with leakage of probability flux to incorrect states. As the noise is also present after learning, it can present new initial conditions to the network at random, resulting in apparent reversals of trajectory direction (compare the trajectory from $t = 10$ to $t = 20$ with that from $t = 30$ to $t = 40$) and possible superposition states of the forward and reverse trajectories (observed starting at $t = 0$ and $t = 70$). As discussed in the text, these properties suggest a model for some phenomena associated with dreaming.

through parameter space has a temporal component, it seems possible that a refined model, particularly one capable of operating in continuous space and time, might be able to inquire whether neural networks can encode metric information about parameter spaces in addition to topological information. This question has experimental correlates, as the place cell activity patterns recapitulated in the rat hippocampus during sleep are observed at varying speeds relative to their occurrence during awake exploration [20, 19].

Considering the results of this chapter with a view toward technical applications leads to two sets of observations. First, the ability of the neural signal decoder to interpret neural signals other than those intended for motor control indicates that in contrast with current practice [6, 40, 4, 25, 14], algorithms and electronic circuits intended for future use in neural prosthetic devices might be tested effectively and extensively in rodents rather than highly trained primates or humans. Second, and perhaps more interestingly, the head direction cell decoding results of Section 3.3 suggest that the neural decoding system described and simulated in Chapter 2 may be broadly applicable to interpreting information encoded in neuronal population activity patterns across a variety of brain regions. A microchip-based implementation of the decoder, based on the circuit designs presented in Section 2.6, could therefore be a useful investigational tool for experimental neuroscience. In particular, the results of this chapter suggest that such a system could be used as an implantable interpreter of simple thoughts and dreams in laboratory animals. The author hopes that through continued collaborations with the Wilson Laboratory of the Department of Brain and Cognitive Sciences at the Massachusetts Institute of Technology this proposal will come to fruition.

# Chapter 4

# Future Directions and Conclusions

This thesis develops a system for adaptively and automatically learning to interpret patterns of electrical activity in neuronal populations in a real-time, on-line fashion. The system is primarily intended to enable the long-term implantation of low-power, microchip-based recording and decoding hardware in the brains of human subjects in order to treat debilitating neurologic disorders. In particular, the decoding system developed in the present work is shown to be capable of interpreting neural signals encoding arm movement intention, suggesting that the system could function as the decoder in a neural prosthetic limb, potentially enabling a paralyzed human subject to control an artificial limb just as the natural one was controlled, through thought alone. The same neural signal decoder is also used successfully to interpret the activity patterns of a population of neurons in the thalamus of a laboratory rat that encode head orientation in absolute space. The success of the decoder in this context motivates the development of a model to explain how networks of neurons adapt the configurations of their receptive fields in response to new stimuli, subsequently learn to encode the structure of new parameter spaces, and ultimately retrace trajectories through such spaces in the absence of the original stimuli. This combination of results suggests that the neural signal decoder is applicable to a broader scope of neural systems than those involved in the control of neural prosthetic devices, and that a microchip-based implementation of the decoder based on the designs presented in this thesis could function as a useful investigational tool for experimental neuroscience.

In particular, results presented in this work in the context of head direction activity decoding suggest that such a system could be used as an implantable interpreter of simple thoughts and dreams—at least in laboratory animals.

The present chapter summarizes the principal results presented in this thesis and discusses avenues for further research that seem potentially fruitful on the basis of the findings described in this work.

## 4.1 Future Directions for Low-Power Decoding Architecture Development

Chapter Two begins by developing the mathematical structure of an adaptive filter to be used to decode electrical signals derived from populations of neurons into observed behaviors or intended control signals. The filter consists of banks of convolution kernels, each defined by a set of tunable parameters, which transform signals from many input channels into a set of output waveforms. An algorithm is then presented for tuning the filter parameters in response to a feedback signal derived from the difference between filter outputs and a correct reference signal; tuning is optimized through gradient descent on the square of the error signal. A particular functional form is selected for the convolution kernels, and the resulting system is shown to be capable of decoding simulated local field potentials into three-dimensional limb trajectories in real time, with an accuracy rivaling the state-of-the-art. A modified version of the same decoder is then used to interpret real neural data recorded from the parietal cortex of a trained macaque monkey, which the system uses to predict intended arm movements by the experimental animal with an accuracy equivalent to that of a state-of-the-art system but at a dramatically lower computational cost. The final section of Chapter Two describes a collaboratively designed set of circuits capable of implementing the decoding system in the context of a micropower analog electronic chip suitable for long-term implantation in the human brain.

The results presented in Chapter Two indicate two principal directions for further

research. One direction involves developing techniques to improve decoding accuracy. A promising approach in this regard, as suggested in Section 2.5, involves the use of increasingly complex input signals. In particular, each normalized mean spiking rate constituting a single input channel to the decoder could be replaced with a set of sub-channels, each transmitting the value of a wavelet coefficient of a particular order in the decomposition of the spike train input to the original channel. The architecture of the decoder could accommodate such an expansion in the number of input channels; the primary challenge amounts to selecting or designing a wavelet basis suitable for real-time implementation in the context of low-power analog circuitry. The second and most important direction for further research into the neural decoding system presented in Chapter Two is to construct and test the analog implementation described in the final section of that chapter. The actual design process is likely to raise further questions about details of implementation and optimization, but more importantly, as the results from Chapter Three indicate, once a prototype is built the decoder may find a variety of applications in unforeseen contexts.

## 4.2 Future Directions in Decoding and Modeling Neural Parameter Space Trajectories During Thinking and Dreaming

Chapter Three begins by demonstrating that the adaptive-filter approach to decoding neural signals can be used to predict the head direction of a laboratory rat in real time on the basis of neuronal spike train data recorded from a small population of thalamic neurons. Such on-line, real-time analysis of place-like cell activity appears to represent an innovation in the analysis of activity in such neuronal populations, and one way of viewing the result is as a new approach to real-time decoding of the thoughts of a live, behaving experimental animal. The successful extension of adaptive-filter decoding to a population of place-like neurons motivates the development of a model intended to investigate the way in which such populations learn to encode information.

The model first broadens the notion of a receptive field to apply to inputs from generalized parameters, reconceptualizing receptive fields more abstractly as functions of parameter space position. A subsequent section then demonstrates a mechanism by which generalized 'parameter-place' cells can adaptively reconfigure and optimize the distribution of their receptive fields in response to experience while exploring an unfamiliar region of parameter space. Finally, the model demonstrates how a neural network of 'parameter-place' cells can encode the structure of an unfamiliar region of parameter space by learning allowed trajectories through parameter space exploration. Once such trajectories have been learned by the network they can be regenerated in the absence of external stimuli, a phenomenon reminiscent of the ability to think and dream, and consistent with experimental observations of populations of hippocampal place cells.

A final section of Chapter Three suggests potentially interesting ways of extending and refining the generalized place cell model developed in that chapter. It also proposes that a microchip-based implementation of the neural signal decoder could not only be broadly applicable to interpreting information encoded in neuronal population activity patterns across a variety of brain regions, but might also be used as an implantable interpreter of simple thoughts and dreams in laboratory animals.

The model developed in Chapter Three raises an additional question for future investigation not addressed in that chapter, a problem that is in some ways a natural extension of the question asked in Chapter Three—that of how networks of biological neurons learn to encode the structure of unfamiliar parameter places—but which also has implications for fields other than neuroscience. The parameter space cell construct developed in Chapter Three models the activity functions of networked neurons and provides insight into how populations of neurons can learn to model generic parameter spaces and, by learning allowed trajectories in those spaces, also model relationships among pieces of information embedded in such spaces. An inverse question might be posed along the following lines: Given a set of observed signals produced by a set of networked elements, what can be inferred about the structures of the relationships among those elements, and about the parameter space in which

they are embedded? In the case of a population of neurons or an artificial neural network, these questions correspond to asking what structural information concerning the synaptic connections among neurons within the population (and possibly also postulated neurons outside the population and hence not observable except through their effects on neurons within the population whose signals can be directly recorded) can be inferred from observing the electrical activity patterns of those neurons, and with what degree of certainty? Since the microscopic structures of neuronal interconnectivity in three dimensions are extremely intricate, such structures are difficult to determine by direct observation [1]. As elusive as these structures are, their topology is of great interest to theorists and experimentalists alike, as many models of memory and neural computation attribute paramount functional significance to the topology of synaptic connections in a population of neurons. The case of synaptic connections in a population of neurons, however, is only one example of a more general problem applicable to a variety of systems across several intellectual disciplines, particularly those in which large sets of data-producing elements can be directly observed but the relationships among those elements, while of interest, cannot.

A specific example serves to illustrate the kind of system in which the inverse problem of inferring topology from observed activity patterns is of particular interest. Consider the tree graph shown in Figure 4-1. Termed a 'dendrogram,' it was constructed to represent postulated functional relationships among genes expressed in cultured human fibroblasts exposed to serum for time intervals of varying duration. In the terminology used in the preceding paragraph, the data-producing elements in

[1] In a conversation in September 2006 with Professor Sebastian Seung of the Massachusetts Institute of Technology Departments of Physics and Brain and Cognitive Science, the author learned that the Seung Laboratory is preparing experiments in which it will be possible to serially section samples of brain tissue and image them using high-resolution electron microscopy. Subsequent computer analysis of the resulting images will then reconstruct neuronal interconnectivity. Professor Seung was of the opinion that such experiments would yield more reliable reconstructions of in vivo neural network topologies than the approach proposed here. Nevertheless, it might be informative to compare the predictions of an indirect inferential approach with the direct one being developed in the Seung Laboratory. In addition, an inferential analysis might demonstrate how certain knowledge of the network topology may enable other information, such as the rules governing communication among network elements, to be inferred with quantifiable levels of uncertainty. Furthermore, while direct observation of neural network topology may become possible using the Seung approach, the inferential methods proposed here might reasonably be applied to systems outside the field of neuroscience.

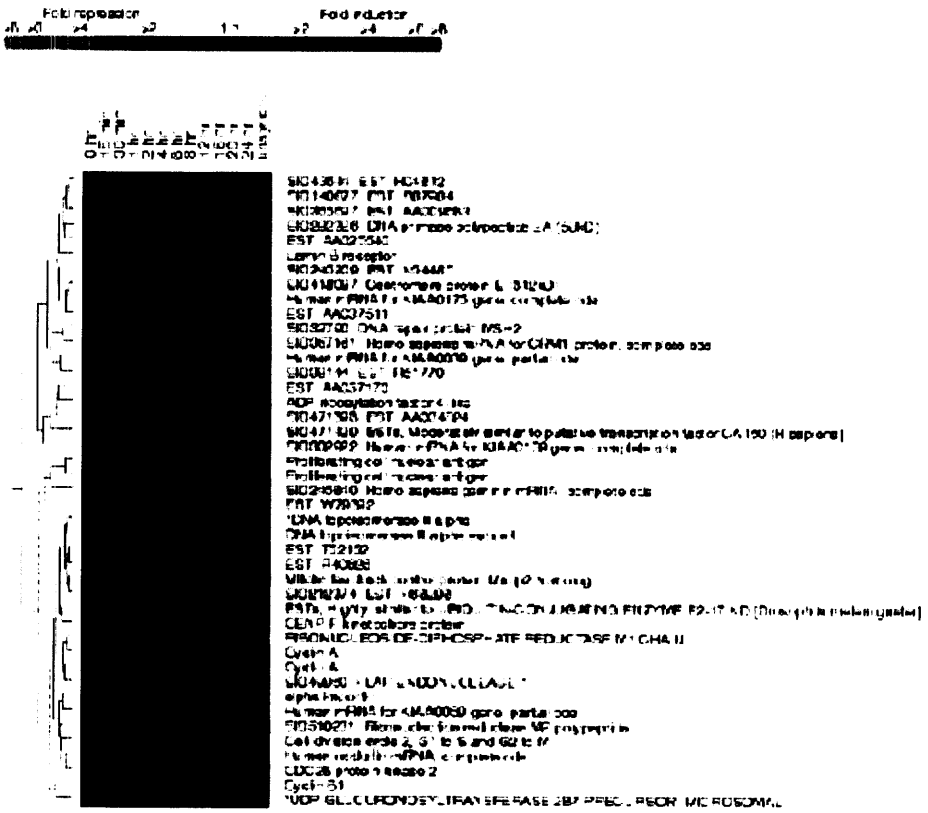


Figure 4-1: **Dendrogram Displaying Relationships Among Human Fibroblast Gene Expression Profiles.** The graph stretched vertically along the left side of the image is a dendrogram displaying relationships among gene expression profiles measured in human fibroblasts exposed to serum for time intervals of varying duration. The expression profile associated with a particular gene is displayed as a row of boxes colored according to the extent of the increase or decrease in its expression level relative to control (serum-free) conditions. Each row is labeled with the name of its corresponding gene. Columns of boxes correspond to different exposure time intervals. The dendrogram groups genes into classes, visualized as groups of branches sharing common high-order nodes, and permits informed speculation concerning the functions of those genes whose roles are not known, on the basis of their positions in the dendrogram relative to genes of known function. Excerpted from [15], Supplemental Information.

this genetic system are the genes themselves, while the observable signals they generate are their activity levels under a variety of experimental conditions, as measured by levels of mRNA detected by hybridization of radiolabeled cDNA in a microarray assay. The existence of correlations in 'expression profiles' of various genes permits the construction of a dendrogram tree graph. An 'expression profile' refers to a set $\{a_{gi}\}$ of activity levels $a_{gi}$, $g \in \{1, \ldots, G\}$, $i \in \{1, \ldots, N\}$ of one of $G$ genes across a given set of $N$ experimental conditions, often represented as a vector of $N$ real components, with $a_{gi}$ equal to the ratio of the activity level of gene $g$ under the $i$th experimental condition to its activity level under the control condition. Terminal nodes of the graph correspond to observed genes, higher-order nodes connect lower-order nodes with the greatest degree of correlation (where the correlation between two expression profiles $\{a_{g_1 i}\}$ and $\{a_{g_2 i}\}$ is typically defined as a form of correlation coefficient: $C\left(\{a_{g_1 i}\}, \{a_{g_2 i}\}\right) = \frac{1}{N} \frac{\sum_{i=1}^{N} \left(a_{g_1 i} - \langle a_{g_1 i} \rangle\right)\left(a_{g_2 i} - \langle a_{g_2 i} \rangle\right)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \left(a_{g_1 i} - \langle a_{g_1 i} \rangle\right)^2} \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left(a_{g_2 i} - \langle a_{g_2 i} \rangle\right)^2}}$; angled brackets denote the mean of the enclosed quantities taken over their free index), and each such higher-order node is subsequently treated as a 'gene' with an 'expression profile' given by a weighted average of expression profiles of genes below it in the dendrogram hierarchy [9]. The resulting dendrogram groups genes into classes, visualized as groups of branches sharing common high-order nodes. Such an analysis, in the case of a set of genes, permits informed speculation concerning the functions of those genes whose roles are not known, on the basis of their positions in the dendrogram relative to genes of known function.

The problem of constructing a dendrogram from a set of gene expression profiles raises four key questions relevant to analyzing analogous problems of inferring topological structures relating data-generating elements. First, what is the appropriate correlation function for the observed signals generated by different elements? Second, what is the general form of the graph (corresponding to the tree graph in the dendrogram example) anticipated to describe the relationships among the signal-generating elements? Selection of a graph type is in some ways analogous to the selection of the functional form for a regression model in applying classical methods of descriptive statistics to numerical data sets. Third, what is an appropriate algorithm to use to

infer structural information concerning the relationships among the data-generating elements from their observed signals, and does that topological structure change in time or as a function of other system parameters? Finally, how can the topological model constructed be assigned a level of confidence to quantify its probable accuracy in a well defined way? This approach to structuring the analysis casts the general problem in terms reminiscent of statistical regression analysis, and so the overall problem, suggested as an inverse problem relative to the question of how individual neurons form networks capable of encoding structured information, might be regarded as a problem of developing methods and models for a kind of 'topological regression analysis.' The availability of increasing volumes of data in a variety of fields suggests that these questions might find applications outside neuroscience and genetics.

## 4.3 Conclusion

The principal goal of the work described in this thesis was to develop a decoding system for neural signals that would be suitable for implementation in micropower analog circuitry and long-term implantation in the human brain for use in neural prostheses and other therapeutic systems for patients with neurologic disorders. As summarized in this chapter, the research presented here expanded beyond its intended scope. In addition to developing the sought-after decoder design and demonstrating its viability, the work presented here demonstrates the applicability of such a decoder to more general neural signal decoding problems and indicates ways in which it could generate both experimental and theoretical insights into how information is encoded in biological neural networks.

# References


[1] R. A. Andersen, S. Musallam, and B. Pesaran. Selecting the signals for a brain-machine interface. *Current Opinion in Neurobiology*, 14:1–7, 2004.

[2] P. J. Best, A. M. White, and A. Minai. Spatial processing in the brain: The activity of hippocampal place cells. *Annual Review of Neuroscience*, 24:459–86, 2001.

[3] E. N. Brown, R. E. Kass, and P. P. Mitra. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature Neuroscience*, 7(5):456–461, May 2004.

[4] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. ODoherty, D. M. Santucci, D. F. Dimitrov, P. G. Patil, C. S. Henriquez, and M. A. L. Nicolelis. Learning to control a brain-machine interface for reaching and grasping by primates. *Public Library of Science Biology*, 1(2):1–16, October 2003.

[5] J. K. Chapin. Using multi-neuron population recordings for neural prosthetics. *Nature Neuroscience*, 7(5):452–455, May 2004.

[6] J. K. Chapin, K. A. Moxon, R. S. Markowitz, and M. L. Nicolelis. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. *Nature Neuroscience*, 2:664–670, 1999.

[7] Y. E. Cohen and R. A. Andersen. A Common Reference Frame for Movement Plans in the Posterior Parietal Cortex. *Nature Reviews Neuroscience*, 3:553–562, July 2002.

[8] J. P. Donoghue. Mind over movement: Development of the braingate neuromotor prosthesis, June 2005. Public Presentation at the Veterans Affairs Medical Center.

[9] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, 95:14863–14868, December 1998.

[10] A. Engel and C. van den Broeck. *Statistical Mechanics of Learning.* Cambridge University Press, Cambridge, 2001.

[11] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner. Neuronal population coding of movement direction. *Reviews in the Neurosciences*, 233:1416–1419, September 1986.

[12] R. R. Harrison, P. T. Watkins, R. J. Kier, R. O. Lovejoy, D. J. Black, B. Greger, and F. Solzbacher. A Low-Power Integrated Circuit for a Wireless 100-Electrode Neural Recording System. *IEEE Journal of Solid-State Circuits*, 42(1):123–133, January 2007.

[13] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prendice Hall, Upper Saddle River, New Jersey, 1999.

[14] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature*, 442:164–171, July 2006.

[15] V. R. Iyer, M. B. Eisen, D. T. Ross, G. Schuler, T. Moore, J. C. Lee, J. M. Trent, L. M. Staudt, J. Hudson, M. S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P. O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999. On-Line Supplemental Data Accessed at http://genome-www.stanford.edu/serum/fig2cluster.html.

[16] E. R. Kandel, J. H. Schwartz, and T. M. Jessell. *Principles of Neural Science.* McGraw-Hill Medical, New York, fourth edition, 2000.

[17] P. R. Kennedy, M. T. Kirby, M. M. Moore, B. King, and A. Mallory. Computer control using human intracortical local field potentials. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 12(3):339–334, September 2004.

[18] J. J. Knierim. Neural representations of location outside the hippocampus. *Learning and Memory*, 13(4):405–415, July-August 2006.

[19] A. K. Lee and M. A. Wilson. Memory of Sequential Experience in the Hippocampus during Slow Wave Sleep. *Neuron*, 36:1183–1194, December 2002.

[20] K. Louie and M. A. Wilson. Temporally Structured Replay of Awake Hippocampal Ensemble Activity during Rapid Eye Movement Sleep. *Neuron*, 29:145–156, January 2001.

[21] B. L. McNaughton, F. P. Battaglia, O. Jensen, F. I. Moser, and M. B. Moser. Path integration and the neural basis of the 'cognitive map'. *Nature Reviews Neuroscience*, 7(8):663–678, August 2006.

[22] C. Mead. *Analog VLSI and Neural Signals.* Addison Wesley, 1989.

[23] Pacemaker. MedlinePlus, U.S. National Library of Medicine, Illustration by A.D.A.M. Accessed On-Line 26 December 2006: http://www.nlm.nih.gov/medlineplus/ency/imagepages/19566.htm.

[24] Medtronic. Activa Parkinson's Control Therapy. Accessed On-Line 25 December 2006: http://www.medtronic.com/physician/activa/parkinsons.html.

[25] S. Musallam, B. D. Corneil, B. Greger, H. Scherberger, and R. A. Andersen. Cognitive control signals for neural prosthetics. *Science*, 305:258–262, July 2004.

[26] S. Musallam, B. D. Corneil, B. Greger, H. Scherberger, and R. A. Andersen. Supplementary Material. *Science*, 305, July 2004. Published on-line with 'Cognitive Control Signals for Neural Prosthetics'.

[27] M. A. L. Nicolelis, D. Dimitrov, J. M. Carmena, R. Crist, G. Lehew, J. D. Kralik, and S. P. Wise. Chronic, multisite, multielectrode recordings in macaque monkeys. *Proceedings of the National Academy of Sciences of the United States of America*, 100(19):11041–11046, September 2003.

[28] J. O'Keefe and L. Nadel. *The Hippocampus as a Cognitive Map.* Clarendon Press, Oxford, England, 1978.

[29] B. pesaran, J. S. Pezaris, M. Sahani, P. P. Mitra, and R. A. Andersen. Temporal structure in neuronal activity during working memory in macaque parietal cortex. *Nature Neuroscience*, 5(8):805–811, August 2002.

[30] K. L. Priddy and P. E. Keller. *Artificial Neural Networks: An Introduction.* Tutorial Texts in Optical Engineering. SPIE Press, Bellingham, Washington, 2005.

[31] G. A. Reina, D. W. Moran, and A. B. Schwartz. On the relationship between joint angular velocity and motor cortical discharge during reaching. *Journal of Neurophysiology*, 85(6):2576–2589, June 2001.

[32] G. Santhanam, S. I. Ryu, B. M. Yu, A. Afshar, and K. V. Shenoy. A high-performance brain-computer interface. *Nature*, 442:195–198, July 2006.

[33] R. Sarpeshkar, R. F. Lyon, and C. Mead. A low-power wide-linear-range transconductance amplifier. *Analog Integrated Circuits and Signal Processing*, 13(1-2):123–151, 1997.

[34] R. Sarpeshkar, C. Salthouse, J.-J. Sit, M. W. Baker, S. M. Zhak, T. K.-T. Lu, L. Turicchia, and S. Balster. An ultra-low-power programmable analog bionic ear processor. *IEEE Transactions on Biomedical Engineering*, 52(4):711–727, April 2005.

[35] R. Sarpeshkar, W. Wattanapanitch, B. I. Rapoport, S. K. Arfin, M. W. Baker, S. Mandal, M. S. Fee, S. Musallam, and R. A. Andersen. Low-Power Circuits for Brain-Machine Interfaces. *Proceedings of the IEEE International Symposium on Circuits and Systems*, May 2007.

[36] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue. Instant neural control of a movement signal. *Nature*, 416:141–142, March 2002.

[37] K. V. Shenoy, D. Meeker, S. Y. Cao, S. A. Kureshi, B. Pesaran, C. A. Buneo, A. R. Batista, P. P. Mitra, J. W. Burdick, and R. A. Andersen. Neural prosthetic control signals from plan activity. *Neuroreport*, 14(4):591–596, March 2003.

[38] L. H. Snyder, A. P. Batista, and R. A. Andersen. Coding of intention in the posterior parietal cortex. *Nature*, 386:167–170, March 1997.

[39] S. Suner, M. R. Fellows, C. Vargas-Irwin, G. K. Nakata, and J. P. Donoghue. Reliability of signals from a chronically implanted, silicon-based electrode array in non-human primate primary motor cortex. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13(4):524–541, December 2005.

[40] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz. Direct cortical control of 3d neuroprosthetic devices. *Science*, 296:1829–1832, June 2002.

[41] S. I. H. Tillery, D. M. Taylor, and A. B. Schwartz. Training in cortical control of neuroprosthetic devices improves signal extraction from small neuronal ensembles. *Reviews in the Neurosciences*, 14(1-2):107–119, 2003.

[42] W. Wattanapanitch, B. Rapoport, S. Arfin, S. Musallam, R. Andersen, and R. Sarpeshkar. An Analog Architecture and Circuits for Linear Decoding and Learning in Neuromotor Prosthetics. Unpublished manuscript, July 2006.

[43] J. Wessberg and M. A. L. Nicolelis. Optimizing a linear algorithm for real-time robotic control using chronic cortical ensemble recordings in monkeys. *Journal of Cognitive Neuroscience*, 16(6):1022–1035, 2004.

[44] J. Wessberg, C. R. Stambaugh, J. D. Kralik, P. D. Beck, M. Laubach, J. K. Chapin, J. Kim, S. J. Biggs, M. A. Srinivasan, and M. A. L. Nicolelis. Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature*, 408:361–365, November 2000.

[45] T. Wichmann and M. R. DeLong. Deep brain stimulation for neurologic and neuropsychiatric disorders. *Neuron*, 52:197–204, October 2006.

[46] K. D. Wise, D. J. Anderson, J. F. Hetke, D. R. Kipke, and K. Najafi. Wireless implantable microsystems: High-density electronic interfaces to the nervous system. *Proceedings of the IEEE*, 92(1):76–97, January 2004.

[47] W. Wu, Y. Gao, E. Bienenstock, J. P. Donoghue, and M. J. Black. Bayesian population decoding of motor cortical activity using a kalman filter. *Neural Computation*, 18:80–118, 2006.

[48] J. Wyatt. *Analog VLSI and Neural Systems*, chapter Least-Squares Methods and Gradient-Descent Solutions. Addison Wesley, 1989.

[49] K. Zhang, I. Ginzburg, B. L. McNaughton, and T. J. Sejnowski. Interpreting neuronal population activity by reconstruction: Unified framework with application to hippocampal place cells. *Journal of Neurophysiology*, 79:1017–1044, February 1998.