KERNEL BASED MACHINE LEARNING FRAMEWORK FOR NEURAL DECODING

By

LIN LI

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2012

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

To my beloved parents and friends

ACKNOWLEDGMENTS

I would like to sincerely thank my Ph.D. advisor Prof. Principe for his invaluable guidance, understanding, patience, and most importantly, his continual faith and confidence in me during my Ph.D. studies at the University of Florida. I feel extremely fortunate to have him as my advisor, who is always willing to help me both academically and personally. I also owe my wholehearted gratitude to Prof. Justin Sanchez and Prof. Joseph Francis for enriching my knowledge on neuroscience and supporting my research. I would like to thank my Ph.D. committee members, Prof. Shea, Prof. Harris, and Prof. Banerjee, for their constructive suggestions and valuable comments on my Ph.D. research and dissertation. I also thank our DARPA team: Austin Brockmeier, Jihye Bae, John Choi, Matthew Emigh, and Kan Li. It has been a wonderful collaboration experience.

I have been fortunate to have many friends in CNEL, with whom I shared many great memory during my four year Ph.D. journey. I specially thank Songlin Zhao, Jihye Bae, Austin Brockmeier, Luis Sanchez Giraldo, Stefan Craciun, IL Park Memming, Sohan Seth, Alexander Alvarado, Dr. Badong Chen, Pingping Zhu, Erion Hasanbelliu, Bilal Fadlallah, Miguel Teixeira, Rakesh Chalasani, Kittipat Kampa, Rosha Pokharel, Goktug Cinar, Evan Kriminger, Matthew Emigh, Kan Li, Jongmin Lee, In Jun Park, Gavin Philips, and Gabriel Nallathambi, for many valuable discussions (and especially the English classes!) and their consistent support and wonderful friendship.

Last but not least, I am indebted to my parents for their support and love in everyday of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

KERNEL BASED MACHINE LEARNING FRAMEWORK FOR NEURAL DECODING

By

Lin Li

December 2012

Chair: Jose C. Principe
Major: Electrical and Computer Engineering

Brain machine interfaces (BMI) have attracted intensive attention as a promising technology to aid disabled humans. However, the neural system is highly distributed, dynamic and complex system containing millions of neurons that are functionally interconnected. How to best interface the neural system with human engineeried technology is a critical and challenging problem. These issues motivate our research in neural decoding that is a significant step to realize useful BMI. In this dissertation, we aim to design a kernel-based machine learning framework to address a set of challenges in characterizing neural activity, decoding the information of sensory or behavioral states and controlling neural spatiotemporal patterns.

Our contributions can be summarized as follows. First, we propose a nonlinear adaptive spike train decoder based on the kernel least mean square (KLMS) algorithm applied directly on the space of spike trains. Instead of using a binned representation of spike trains, we transform the vector of spike times into a function in reproducing kernel Hilbert space (RKHS), where the inner product of two sets of spike times is defined by the Schoenberg kernel, which encapsulates the statistical description of the point process that generates the spike trains, and bypasses the curse of dimensionality-resolution of the other spike representations. The simulation results indicate that our decoder has advantages in both computation time and accuracy, when the application requires fine time resolution.

Secondly, the precise control of the firing pattern in a population of neurons via applied electrical stimulation is a challenge due to the sparseness of spiking responses and neural system plasticity. In this work, we propose a multiple-input-multiple-output (MIMO) adaptive inverse control scheme that operates on spike trains in a RKHS. The control scheme uses an inverse controller to approximate the neural circuit's inverse. The proposed control system takes advantage of the precise timing of the neural events using the Schoenberg kernel based decoding methodology we proposed before. During operation, the adaptation of the controller minimizes a difference defined in the spike train RKHS between the system output and the target response and keeps the inverse controller close to the inverse of the current neural circuit, which enables adapting to neural perturbations. The results on a realistic synthetic neural circuit show that the inverse controller based on the Schoenberg kernel can successfully drive the elicited responses close to the original target responses even when significant perturbations occur.

Thirdly, the spike train variability causes fluctuations in the neural decoder. Local field potentials (LFPs) are an alternate manifestations of neural activity with a more common continuous amplitude representation and longer spatiotemporal scales that can be recorded simultaneously from the same electrode array and contain complementary information about stimuli or behavior. We propose a tensor product kernel based decoder for multiscale neural activity, which allows modeling the sample from different sources individually and mapping them onto the same RKHS defined by the tensor product of the individual kernels for each source. A single linear model is adapted as done before to identify the nonlinear mapping from the multiscale neural responses to the stimuli. It enables us decoding of more complete and accurate information from heterogeneous multiscale neural activity with only with an implicit assumption of independence on their relationship. The decoding results in the rat sensory stimulation experiment show that the decoder outperforms the decoders with either single-type

neural activities. In addition the multiscale decoding methodology is also used in the adaptive inverse control mode. Due to the accuracy and robustness of the decoder, the control diagram with open-loop mode is applied to control the spatiotemporal pattern of neural response in the rat somatosensory cortex with micro-stimulation in order to emulate the tactile sensation and obtained promising results.

Finally, we quantify and comparatively validate the temporal functional connectivity between neurons by measuring the statistical dependence between their firing patterns. Temporal functional connectivity provides a quantifiable representation of the transient joint information of multi-channel neural activity, which is important to completely characterize the neural state but is normally overlooked in the temporal decoding due to its complexity. The functional connectivity pattern is represented by a graph/matrix, which is again not a conventional input for machine learning algorithms. Therefore, we propose two approaches to decode the stimulation information from the neural assembly pattern. One is to use graph theory to extract topology feature vector as the model input, which makes conventional machine learning approach applicable but dismisses the information of structural details. Therefore, we also proposed a matrix kernel that is able to map the connectivity matrix into RKHS and enable kernel based machine learning approaches directly to operate on the connectivity matrix, which bypasses the information reduction induced by the feature extraction.

CHAPTER 1
INTRODUCTION

Brain machine interfaces (BMI) provide new means to communicate with the brain by directly accessing, interpreting and even controlling neural states. They have attracted huge attention as a promising technology to aid the disabled (i.e. spinal cord injury, movement disability, stroke, hearing loss, and blindness). Quantifying the information contained in neural activity, modeling the neural system and decoding the intention of movement or stimulation are the fundamental steps to design neural prosthetics and brain machine interfaces.

However, the neural system is highly distributed, dynamic and complex, creating challenges for the modeling task. First, neurons communicate through electrical pulses, called spikes. Thus, information is represented not in the usual signal amplitude but in spike timings. The sequence of spikes or spike train represents the activity of an individual neuron, for which the conventional signal processing approach based on signal amplitude variation is not well suited. In addition, besides the activity of individual neurons (spike trains), the development of recording technology allows us to access the brain activity also from other functional level including local field potential (LFP), electrocorticogram (ECG), and electroencephalogram (EEG), which contains complementary information of brain intentional states. Though it is clear there are relationships among those brain activity factors, it is unknown how these factors can be exploited to better decode neural-response-stimulus mappings. The challenge is how to handle the heterogenous signals with multiple spatiotemporal scales for neural decoding. Moreover, the neural mechanisms that select and coordinate the distributed cortical activity to produce movements have been the focus of system neuroscience [61]. The development of new tools to assess this activation is critical because neural assemblies provide a conceptual framework for the integration of distributed neural firing pattern. However, due to the spatially specific and transient

nature of the inter neuronal communication, quantifying the assembly and decoding information from its functional connectivity pattern is a difficult task. In our work, we address all the challenging issues in specific BMI tasks and develop new kernel-based machine learning approaches to overcome them, which are further discussed in the following sections.

1.1 Decoding from Spike Trains and Control

Intracranial stimulation has the possibility to deliver information directly to the brain, and the capacity of this information channel is directly linked to the ability to drive neural responses to match meaningful spiking responses created naturally sensory. Controlling neural response via stimulation has raised the prospect of mimicking the responses to natural sensory stimuli, thereby creating a sensory prothesis. This allows providing more naturalistic sensory feedback as part of a brain-machine interface and thus improving performance over the use of arbitrary spatiotemporal electrical stimulation [63]. Using electrode arrays for electrical microstimulation and others for recording allows simultaneous stimulation and recording from populations of hundreds of neurons. The temporal precision of the stimulation and recording is in the millisecond range, whereas the spatial precision is only limited by the microelectrode arrays. This enables the precise control of neural activity at the neuron and population levels.

From a control theory perspective, the neural circuit is treated as the “plant”, where the applied microstimulation is the control signal and the plant output is the elicited spike trains. There are several unique challenges that have to be considered before we address this problem. First, the transfer function between the microstimulation and the target neural response is unknown; thus, the traditional approach of a plant with known dynamics cannot be directly applied on a neural system. In addition, the perturbations induced by long-term learning and stimulation will cause changes to the neural system transfer function; this requires the controller parameters to automatically track neural plasticity. All these factors suggest that adaptive control is an appropriate approach to

address this problem. However, the controlled system output is a spike trains that is not conventional smooth amplitude signal, which is not compatible with the existing control framework,unless we use rate coding that destroys the time resolution.

In online adaptive control based on signal processing methodology, the controller model performs like a neural decoder: given the observed spike trains as inputs, the controller is used to estimate the stimulation such that the plant output approximates the target spike trains. However, all the exciting decoding techniques have been applied to discretized representations of spike trains, which has limited applicability for systems requiring a fine time resolution. For example, in the case of a somatosensory prosthesis, the time resolution of the micro-stimulation used to create tactile sensation is small (around 1 ms), which requires a small bin size. With such a small discretization step, the input space becomes sparse and the dimensionality of the input space from multi-electrode arrays also becomes a problem (curse of dimensionality). Instead, the set of spike occurrences provides a more effective and accurate description of spike trains, but the space of spike trains is not a conventional L_2 functional space enjoyed by typical discrete time continuous amplitude signals. Algorithms that rely on real or complex values cannot directly operate on the set of spike times. Indeed, the space of spike times does not possess an algebraic structure, i.e., operations such as addition and multiplication are not defined for spike trains.

In order to effectively apply machine learning algorithms for stimulation decoding, based on the mathematical theory of RKHS and functional representation of spike trains, a Schoenberg kernel between two spike trains enables kernel-based regression algorithms to be directly applied in the space of the neural event timings [44, 67, 68]. This decreases the computation time in contrast with the kernels based on the conventional rate representation and avoids sparseness and high-dimensionality of the binned spike trains. In Chapter 2, we propose a MIMO adaptive inverse control scheme with Schoenberg kernel-based regressor built in the spike train RKHS to control the

neural activity. The adaptation of the controller model also allows the control system to track the plasticity of the underlying neural system organization. The controller based on the nonlinear kernel-based neural system identification model has a linear structure in RKHS, which has a single global optimal solution.

1.2 Decoding from Multi-Scale Neural Activity

The problem of extracting information from spike train, local field potential (LFP), electrocorticogram (ECG), and electroencephalogram (EEG)is rooted in their unknown underlying relation, multiple spatial and temporal scale and heterogenous signal format. In our work, we mainly address the modeling task of spike trains and LFPs as an example of multiscale neural modeling, since it is able to cover all the challenges mentioned above. Another reason of selecting spike train and LFP is that they can be recorded from the same micro-electrode arrays, which guarantees the time alignment and thus simplify the data preprocessing procedure.

According to the literature, spike trains and local field potentials (LFPs) encode complementary information of the stimuli or behaviors [8, 37, 95]. In most recordings, spike trains are obtained by a high-pass filter with the cutoff frequency about $300-500$ Hz, while LFPs are obtained using a low-pass filter with the cutoff frequency about 300 Hz [77]. A spike train represents the single-unit neural activity with a fine temporal resolution. However, its stochastic properties induce a considerable trial-to-trial variability, especially when the stimulation amplitude is small. In contrast, LFPs reflect the sum of all local currents near the surface of the electrode, which limits specificity but provides robustness for characterizing the modulation induced by stimuli, even at low amplitudes. Therefore, an appropriate combination of LFPs and spike trains in decoding models make it possible to more accurately decode the stimuli or behaviors from the neural response. For example, the decoder can coordinate LFPs or spike patterns to tag particularly salient events or extract different stimulation features characterized by different type signals. However, the different signal properties between LFPs and

spike trains make integration in the same model difficult. First, the underlying stochastic properties are totally different between the two representations. A spike train that is a set of spike timings can be interpreted as a realization of a point process [41, 93], while LFPs are a continuous amplitude process. Moreover, the LFP time scale is significantly longer than spike trains. In chapter 3, we propose a tensor-product-kernel-based regressor to decode the stimulation information from multi-scale neural activities, which allows modeling heterogeneous signals individually and merge their information in the feature space defined by the tensor product of the individual kernels for each type signal [85].

1.3 Decoding from Temporal Functional Connectivity Pattern

In BMIs, the goal is to decode the intention of movement/stimulation, using multi electrode neural data from cortex synchronized with kinematic/stimulation variables measurements. Since neurons are selectively coordinated with each other, from a statistical modeling perspective, the complete information to solve this problem resides in the joint probability density function of the multivariate neural data and of the multivariate kinematic variables, but this is never done due to the high dimensionality of the joint distributions. Even when we develop models that estimate the conditional probability of the kinematics given the neural data, we do not use the joint information expressed in the multichannel neural data for the same reason. Instead, we model the conditional dependence only with respect to the history of a vector of neural spike trains. Since each neural channel is nothing but the marginal density of the joint probability of multivariate neural data, this procedure is only a reasonable solution when the neurons spike independently. In order to characterize this joint information among neurons, we quantify the temporal functional connectivity pattern among that comprises of pairwise dependencies between neurons involved in a given behavioral/stimulation task and investigate its association with behavioral/stimulation states in time in Chapter 4.

However, the functional connectivity patterns are represented by graphs or matrices instead of vectors, which carries difficulties when one wants to use graphs as an input feature for the neural decoder. Instead of directly addressing functional connectivity graphs or matrices, most research implements graph theory to quantify the topology property of a graph pattern including node degree, clustering coefficient, betweenness centrality, etc. [22, 82]. These topology metrics extract feature vectors from the functional connectivity graphs/matrices, which allows the implementation of the traditional modeling approaches with the vector-based input space. However, the extracting process reduces the details of the graph structure. Therefore, in chapter 4, we also design a kernel for functional connectivity matrices, which allows us to apply kernel based machine learning algorithms directly on the temporal functional connectivity pattern with less information reduction.

CHAPTER 2
A KERNEL-BASED FRAMEWORK FOR SPIKE TRAIN DECODING AND CONTROL

2.1 Background and Motivation

Applying low power current waveforms to intracranial microelectrodes (referred herein as electrical microstimulation)[1] is a valuable tool to learn the underlying structure of neural circuits and to control their activity. Modern electrophysiological techniques enable the implantation of multiple microelectrode arrays. Using arrays for electrical microstimulation and others for recording allows simultaneous stimulation and recording from populations of hundreds of neurons. The temporal precision of the stimulation and recording is in the millisecond range, whereas the spatial precision is only limited by the microelectrode arrays. This enables the precise control of neural activity at the neuron and population levels. Controlling the neural activity via stimulation has raised the prospect of generating specific spike patterns, even mimicking natural neural activity, which is central both for our basic understanding of neural information processing and for engineering "neural prosthetic" devices that can interact with the brain directly [30].

From a control theory perspective, the neural circuit is treated as the "plant", where the applied microstimulation is the control signal and the plant output is the elicited spike trains. There are several unique challenges that have to be considered before we address this problem. First, the transfer function between the microstimulation and the target neural response is unknown;thus, the traditional approach of a plant with known dynamics cannot be directly implemented on a neural system. In addition, the perturbations induced by long-term learning and stimulation itself will cause changes to the neural system transfer function; this requires the controller parameters to automatically track neural plasticity. The existing approaches applied to neural

[1] Although other forms of stimulation may be considered in the paradigm, we concentrate on the electrical microstimulation.

systems can be summarized into two major classes: model-free methods [12, 30, 47] and model-based methods [3, 14, 57]. For example global search optimization [30] is a model-free approach, which uses a genetic algorithm (GA) to search the stimulation space for the cost function minimum, but the cost function is not defined in terms of a target spiking pattern, and because the potential spatiotemporal space of stimuli is usually huge, the global search is slow. Another model-free method applied to the neural systems is the proportional–integral–derivative PID controller [47], which is a feedback control scheme that is able to reduce the system disturbance using the output error as the feedback signal. Since there is no easy way to automatically tune PID weights, such controllers can not successfully control a neural system with time-varying perturbations.

In model-based schemes, the controller performs like a decoder and is generally built in two ways. One is to train an inverse model of the transfer function from the stimulation to neural activity [14]. The target neural activity pattern is used as the initial controller input, then the elicited neural response is returned to the controller input as the feedback, which is capable of canceling system noise. Another model-based strategy relies on the Bayesian 'inversion' of the encoding model, such as Generalized Linear Model (GLM), to estimate the optimal stimulation with respect to the target neural response [3]. However, since the desired stimulation for the target firing pattern induced by the natural stimulation is unavailable for training, the controller may converge to the optimal inverse neural circuit model with respect to the training data but not necessarily to the target firing pattern; this strategy may cause limited accuracy of the control schemes unless the stimulation space is fully explored—a difficult problem in itself. Moreover, for model-based neural control, it is essential to adapt the model to track the system variation, which has not been formally addressed in any of the models mentioned above. Because of the unique challenges, adaptive control with system identification technology is an appropriate approach to address this problem, where the neural circuit is treated as the "time-variant plant". There are many classic

adaptive control methods [5] for a system with unknown dynamics, such as Stochastic Control, Model Reference Adaptive Controllers, etc., but how to appropriately fit these control scheme for spike trains exploiting the spike timing information has not been well developed. However, adaptive inverse control [105] is a input-output (system identification) based control scheme, and thus allows us to attack adaptive control problem using the methodology of adaptive signal processing. With neural signal processing methodologies, the adaptive inverse controller can be directly applied on spike trains. Therefore, multiple-input multiple-output (MIMO) adaptive inverse control is used in this work. The basic idea of adaptive inverse control is to learn an inverse model of the plant as the controller, such that the cascade of the controller and the plant will perform like an unitary transfer function. The target neural activity is used as the controller's command input. The controller parameters are updated to minimize the dissimilarity between the target neural activity and the cascade output during the whole control process, which enables the controller to track any variation in the plant time-variation and cancel any noise.

The controller model performs like a neural decoder: given the observed spike trains as inputs, the controller is used to estimate the stimulation such that the plant output approximates the target spike trains. For the development of the decoding algorithm, most the decoding techniques have been applied to discretized representations of spike trains because of its simplicity and applicability, which can be summarized into two categories: regression scheme [11, 54, 55, 89, 104] and Bayesian scheme [71, 101]. However, the decoder based on the discretized representation of spike trains has limited applicability for systems requiring a fine time resolution. For example, in the case of a somatosensory prosthesis, the time resolution of the micro-stimulation used to create tactile sensation is small (around 1 ms), which requires a small bin size. With such a small discretization size, the input space becomes sparse and the dimensionality of the input space from multi-electrode arrays also becomes a problem

(curse of dimensionality). Instead, the set of the spike time occurrences provides a more effective and accurate description of spike trains, but the space of spike trains is not a conventional L_2 functional space enjoyed by typical discrete time continuous amplitude signals. Algorithms that rely on real or complex values cannot directly operate on the set of spike times. Indeed, the space of spike times does not possess an algebraic structure, i.e., operations such as addition and multiplication are not defined for to spike trains. In order to effectively apply machine learning algorithms for stimulation optimization, it is necessary to define an appropriate functional structure for the spike train space where optimization algorithms can be developed. Positive definite functions in the spike train space are defined in [67]. Based on the mathematical theory of RKHS and functional representation of spike trains, a Schoenberg kernel between two spike trains enables kernel-based regression algorithms to be directly applied in the space of the neural event timings [44]. This decreases the computation time in contrast with the kernels based on the conventional rate representation and avoids sparse high-dimensional vectors corresponding to binned spike trains.

In this chapter, we propose a MIMO adaptive inverse control scheme with Schoenberg kernel-based regressor built in the spike train RKHS to control the neural activity. The goal of this work is to elicit neural responses that mimic those induced by the natural stimulation. The adaptation of the controller model also allows the control system to track the plasticity of the underlying neural system organization. The controller based on the nonlinear kernel-based neural system identification model has a linear structure in RKHS, which has a single global optimal solution.A realistic synthetic neural circuit is built to test the control system.

2.2 Neural Decoding

In neurophysiology it is widely accepted that neurons communicate through a discrete pulse-like wave, called an action potential. Action potentials record the voltage differential to a distant ”ground” point, typically the skull. Despite the inevitable noise in

these recordings, it is easily observed that action potentials have a very stereotypical shape, with a characteristic fixed amplitude and width associated with a given neuron. From the neurophysiological perspective, the actual shape and magnitude of an action potential is redundant because it contains no information, but the moment it occurs is important. Under this perspective, the action potentials are simplified to spike train, a sequence of spikes ordered in time, which neglect the stereotypical shape of an action potential, but preserves only the time instants at which they occur (i.e., spike times) [24]. A spike train **s** is a sequence of ordered spike times i.e $\mathbf{s} = \{t_n \in \mathcal{T} : n = 1, \dots, N\}$, in the interval $\mathcal{T} = [0, T]$. which can be defined by

$$s(t) = \sum_{m=1}^{N} \delta(t - t_m), \{t_m \in \mathcal{T} : n = 1, \dots, N\}, \tag{2–1}$$

where δ denotes the Dirac delta. Since the spike train is nothing but a set of event times, the information is coded in the event times. However, there is no algebraic structure in the space of spike trains. No conventional machine learning algorithm can be directly applied to it. Binning is a predominant process in neural signal processing, which transform the spike train into amplitude signal, and then allows the application of conventional machine learning learning to for neural decoding. In this section, we first review the existing neural decoding approaches on binned data. Then we proposes a nonlinear adaptive decoder for somatosensory micro-stimulation based on the kernel least mean square (KLMS) algorithm applied directly on the space of spike trains. Instead of using a binned representation of spike trains, we transform the vector of spike times into a function in reproducing kernel Hilbert space (RKHS), where the inner product of two spike time vectors is defined by intensity driven kernel. This representation encapsulates the statistical description of the point process that generates the spike trains, and bypasses the curse of dimensionality-resolution of the binned spike representations.

2.2.1 Neural Decoder with Discretized Spike Train Representation

Binning is a predominant approach in current spike train analysis and processing methods [24]. Statistically, it is motivated by the counting process representation of a point process, which is obtained by counting the number of spikes that appear during a unit interval and normalized by the duration of the interval. The binned data is denoted by r, where

$$r(n) = \frac{1}{\Delta} \int_{(n-1)\Delta}^{n\Delta} s(t)dt \tag{2–2}$$

The binned data can be easily determined from a single trial and transform the point processes into discrete amplitude time series, which allows the analysis using the conventional signal processing method.

Because of its simplicity, a variety of machine learning techniques have been applied on the discrete representations of spike train (binned data), which can be roughly divided into two categories: regression models and bayesian models. For the development of the regression model, the influential optimal linear decoder is posed as a version of the Wiener-Hopf filter [11]. Elaborations on this idea include linear [55, 89] and nonlinear regression models, such as neural network [104], Volterra kernel [54], and kernel regression techniques [78, 91], where the kernel-based model with spikernel allows estimation of the nonlinear functional relationship between stimuli and neural response and further improve the decoding performance. Bayesian modeling is another promising scheme [71, 101], in which the prior distribution of the signal to be decoded is combined with an encoding model describing the probability of the observed spike train. In the following sections, a Spikernel-based model and GLM as representative models of two different categories will be further introduced.

2.2.1.1 Generalized-linear-model-based decoder

The generalized linear model (GLM) [71] is a decoding algorithm based on Bayes' rule, in which the prior distribution of the signal is combined with a encoding model

describing the probability of the observed spike train given the input signal. The resulting Bayes estimate is optimal in principle, assuming that the prior distribution and encoding model are correct. To be specific, the GLM predicts the current number of spikes using the recent spiking history and the preceding stimulus. In GLM, a spike train as a point process is generated by

$$\lambda_i(t) = f(\mathbf{k}_i \cdot \mathbf{x}(t) + \sum_{j,\alpha} \mathbf{h}_{ij}(t - t_{ja}) + b_i), \tag{2–3}$$

where $\mathbf{x}(t)$ is the stimulation input, $\lambda_i(t)$ denotes the instantaneous firing rate of the ith neuron, $\mathbf{k}_i$ is the ith neuron linear receptive field, b_i is additive constant determining the baseline firing rate, and $\mathbf{h}_{ij}$ is a post-spike effect from the jth neuron to the ith neuron. In general, the concave nonlinear function $f(\cdot)$ is the exponential function, because of Poisson assumption and also s the application of computation efficiency. The model parameters $\theta = \{\mathbf{k}_i, \mathbf{h}_{ij}, b_i\}$ are optimized by maximizing the log-likelihood function $\log p(\mathbf{r}|\mathbf{x}, \theta) = \sum_{i,n} r_i(n) \log \lambda_i(n) - \Delta \sum_{i,n} \lambda_i(n) + const$. To estimate the optimal $\mathbf{x}_{map}$ the posterior log-likelihood is maximized over the stimulus given the target firing pattern: $\log p(\mathbf{x}|\mathbf{r}, \theta) = \log p(\mathbf{r}|\mathbf{x}, \theta) + \log p(\mathbf{x}) + const$, where $p(\mathbf{x})$ is computed based on a pre-knowledge of its distribution.

2.2.1.2 Spikernel-based decoder

The Spikernel proposed in [90] is a promising decoder due to its capability to model the nonlinear functional relationship between the neural response and stimulus. It applies the kernel-based machine learning framework and maps spike count sequences into an high dimensional abstract vector space in which we can perform linear optimization to obtain a nonlinear mapping with no local minima. It has been shown that the Spikernel outperforms all the other standard vector based kernels [90], because it is designed according to the following important properties of spike patterns.

1. Similar firing patterns may have small differences in a bin-by-bin comparison, which is due to not only the inherent noise of the neural system but also responses

to external factors that are not recorded and are not directly related the to the task performed. Therefore, different length subsequences of spike count sample are compared in the kernel definition which is weighted according to the subsequence length.

2. Two patterns may be quite different in a simple bin-wise comparison but if they are aligned by some non-linear time distortion or shifting, the similarity becomes apparent. Therefore, time alignment scheme is induced in the definition of kernel.

The kernel between two spike count sequences is defined by

$$\kappa(\mathbf{s}_l, \mathbf{s}_m) = \sum_{n=1}^{N} p^n \kappa_n(\mathbf{s}_l, \mathbf{s}_m), \tag{2–4}$$

which is a sum of kernels for different pattern length weighted with p^n, where $\mathbf{s}_l$ and $\mathbf{s}_m$ are the lth and mth firing rate vector obtained by sliding window. The feature map of $\mathbf{s}$ is defined by

$$\phi_u(s) = \sum_{i \in I_n} \mu^d(\mathbf{s}_i, \mathbf{u}) \lambda^{|\mathbf{s}-i_1|}, \tag{2–5}$$

which represents the similarity of $\mathbf{s}$ to the prototype $\mathbf{u}$. $\mathbf{u}$ represents a possible spike train prototype of fixed length $n < |\mathbf{s}|$. μ and λ are free parameters. Since the value of the $\kappa(\mathbf{s}, \mathbf{t})$ is positive related to $N = max_{l,m}(\kappa(\mathbf{s}_l, \mathbf{s}_m))$, therefore the kernel matrix for all the sample is normalized by N.

2.2.2 Kernel based Decoder with Spike Timing Representation

The binning process can easily transform the point processes into discrete amplitude time series, which allows the analysis using the conventional signal processing method, but at the expense of losing all temporal resolution about variations in neural response. This means that any temporal information in the spikes within and between bins is disregarded. This is especially alarming when neurons spike timing precision is on the millisecond range, and the actual spike times encode additional information. Moreover, a tradeoff is at what time scale to analyze the data. The small bin

size is required to preserve the fine time resolution, but sparsify the signal, increasing the artifact variability and causing the model dimensionality , which require large data sets for proper generalization in signal processing to be a problem.

We note that the most accurate and efficient description of a firing pattern is just the set of spike times. However, there is no algebraic structure in the space of spike trains, which means that no conventional regression algorithm can be directly applied to spike trains. Our idea is whether we can define a positive definite kernel that is an operator on the set of spike times. Then the set of spike times can be projected into a reproducing Hilbert space (RKHS) defined by this kernel, then we can build a decoder by applying regression algorithm in the RKHS. In this way, any kernel-based regression method can be directly applied to spike trains.

2.2.2.1 Kernel design for spike trains

Reproducing kernel Hilbert spaces (RKHS) are special Hilbert spaces of functions where all evaluations are finite. RKHS are completely defined by a positive definite function that is called a kernel [4], which specifies the inner product metric. The advantage of defining a RKHS for spike trains is to obtain a continuous functional space, where the conventional linear signal processing algorithms can be applied. The issue is how to define a kernel that will be able to preserve the statistical law that defines the point process. According to the literature, since spike trains always correspond to an observation or measurement associated with some underlying point process, it is appropriate to consider the spike train to be a realization of point process, from which only the number of spikes and the time instances they occur are relevant. In general, for a point process p_i the conditional intensity function $\lambda(t|H_t^i)$ is used to completely characterize a point process, where $t \in \tau = [0, T]$ denotes the time coordinate and H_t^i is the history of the process up to t. A recent area of research [66, 68] is to define an injective mapping from spike trains to RKHS based on the kernel between their

conditional intensity functions of two point processes [66]. There are a family of point process kernels defined as follows:

1. **Cross-intensity (CI) kernel** is defined by the inner product (correlation) of the conditional intensity functions

$$\begin{aligned}\kappa_{CI}(p_i, p_j) &= \langle \lambda(t|H_t^i), \lambda(t|H_t^j) \rangle_{L_2(\tau)} \\ &= \int \lambda(t|H_t^i)\lambda(t|H_t^j)dt. \end{aligned} \tag{2–6}$$

   which is a linear operator in the space spanned by the intensity function and it is positive semi-definite [66].

2. **Nonlinear cross-intensity (NCI) kernel** is defined by the nonlinear product of the conditional intensity functions

$$\kappa_{NCI}(p_i, p_j) = \int exp(\frac{-(\lambda(t|H_t^i) - \lambda(t|H_t^j))^2}{\sigma^2})dt, \tag{2–7}$$

   which is a nonlinear and positive definite kernel of conditional intensity functions. In fact it is the correntropy between intensity functions and thus preserves more information beyond correlation [49]. σ is a free parameter, which controls the metric space between the instantaneous values of two intensity functions. large σ creates a metric space similar to the L_2 norm, whereas small σ make the metric space approaching to L_0 norm.

3. **Schoenberg kernel** is defined by a nonlinear transformation of the difference square of the conditional intensity functions

$$\begin{aligned}\kappa(\lambda(t|H_t^i), \lambda(t|H_t^j)) &= \exp(-\frac{\|\lambda(t|H_t^i) - \lambda(t|H_t^j)\|^2}{\sigma^2}) \\ &= \exp(-\frac{\int_\tau (\lambda(t|H_t^i) - \lambda(t|H_t^j))^2 dt}{\sigma^2}), \end{aligned} \tag{2–8}$$

   where σ is the kernel size. Schoenberg kernel is a Gaussian-like kernel defined on intensity functions, which is strictly positive definite kernel and sensitive to the nonlinear coupling of two intensity function [66].

The last two kernels are strictly positive definite kernels, thus the mapping between the spike train space and the RKHS is injective [4]. Notice that this kernel is very different from the ones used in kernel based machine learning, e.g. for the support vector machine. In fact here a windowed the spike train is mapped into a function in the RKHS. Different spike trains will then be mapped to different locations in the RKHS, as shown in Figure 2-1. The RKHS of windowed spike trains provide a simply way to

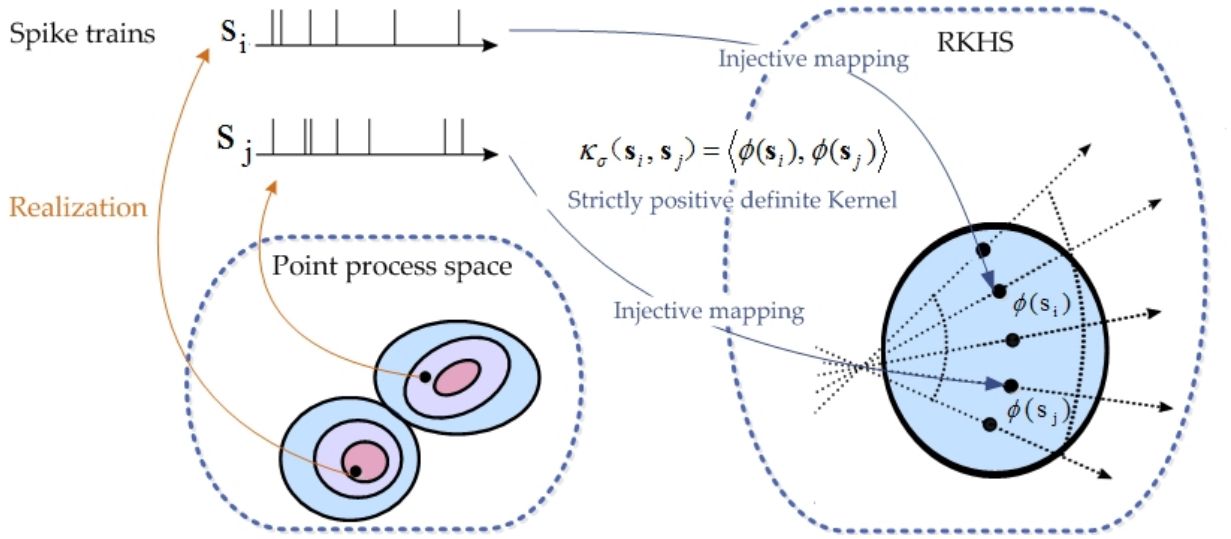


Figure 2-1. Relation between the original space of spike trains and the RKHS defined by the strictly positive defined kernel.

estimate the conditional intensity function nonparametrically, but induce a free parameter – window length,which affects the structure of the RKHS. In order to investigate the property of each kernel and thus select the best candidate for neural modeling, we analyze and compare their decoding performance in section 2.2.2.4.

In order to be useful, the methodology must lead to a simple estimation of the quantities of interest (e.g. the kernel) from experimental data. A practical choice used in our work estimates the conditional intensity function using a kernel smoothing approach [66, 76], which allows estimating the intensity function from a single realization. The estimated intensity function is obtained by simply convolving $\mathrm{s}(t)$ with the smoothing

kernel $\mathrm{g}(t)$, yielding

$$\hat{\lambda}(t) = \sum_{m=1}^{M} \mathrm{g}(t - t_m), \{t_m \in \mathcal{T} : m = 1, \dots, M\}, \tag{2–9}$$

where the smoothing function $\mathrm{g}(t)$ integrates to 1. Here $\hat{\lambda}(t)$ can be interpreted as an estimation of the intensity function. The rectangular and exponential function [24, 66] are both popular smoothing kernels, which guarantee injective mappings from the spike train to the estimated intensity function. In order to decrease the kernel computation complexity, the rectangular function $\mathrm{g}(t) = \frac{1}{\mathcal{T}}(\mathrm{U}(t) - \mathrm{U}(t - \mathcal{T}))$ ($\mathcal{T} \gg$ the inter-spike interval) is used in our work, where $\mathrm{U}(t)$ is a Heaviside function. This rectangular function with low computation complexity compromises the locality of the spike trains, i.e. the mapping gives more emphasis on the early spikes than the later spikes in the estimated intensity function. However, our experiments show that this compromise only causes a minimal impact on the kernel-based regression performance.

The main advantage of this functional representation compared to binning is that the precision in the event location is better preserved and the limitations of the sparseness and high-dimensionality is also avoided since kernel smoothing maps the spike train into a continuous function. The added advantage is that the RKHS has a linear structure, so any linear signal processing algorithm can be easily implemented with the so called “kernel trick” which decreases significantly the computation complexity (inner products of infinitely long vectors can be computed by an evaluation of the kernel).

2.2.2.2 Kernel based adaptive regressor

For neural decoding, our goal is to build a regression model in the spike train kernel space to continuous stimulation waveforms. The great appeal of kernel-based filters is the usage of the linear structure of RKHS to implement well-established linear adaptive algorithms and to obtain a nonlinear filter in the input space that leads to universal approximation capability without the problem of local minima. There are several candidate kernel-based regression methods [20], such as support vector

regression (SVR) [92], radial basis function (RBF), kernel recursive least squares (KRLS), kernel least mean square (KLMS) [50] and etc.. Considering the dynamics of neural system, KLMS is selected in our work due to its online adaptability and low computation cost.

The basic idea of KLMS is to transform the data $\mathbf{s}_i$ given by $\mathbf{s}_i = \{t_m - (i-1)\tau, t_m \in [(i-1)\tau, (i-1)\tau + T] : m = 1, \ldots, M\}$ (the ith window of the spike time sequence obtained by sliding the T-length window with step τ) from the input space to a high dimensional feature space of vectors $\phi(\mathbf{s}_i)$, where the inner products can be computed using a positive definite kernel function satisfying Mercer's condition: $\kappa(\mathbf{s}_i, \mathbf{s}_j) = \langle \phi(\mathbf{s}_i), \phi(\mathbf{s}_j) \rangle$ [50]. For our work the RKHS of spike trains is defined by the Schoenberg kernel, unlike the work in adaptive filtering that almost exclusively uses the Gaussian kernel. However the algorithm structure is exactly preserved, i.e., the linear least-mean-square (LMS) algorithm is directly applied in the RKHS as a special case of the general methodology to formulate linear filters and gradient descent algorithms in terms of the kernel $\kappa(\mathbf{s}_i, \mathbf{s}_j)$. Let Ω be the filter weight function (which can be considered an infinite dimensional vector) in RKHS and $\mathbf{s}_i$ be the input spike train and $\phi(\mathbf{s}_i)$ the transformed input function in the RKHS, then the regressor output in the input space is $y = \langle \Omega, \phi(\mathbf{s}_i) \rangle$. During online adaptation Ω becomes $\Omega(n)$ at time n. Following the stochastic gradient LMS update, the KLMS in the kernel feature space using a stochastic instantaneous estimate of the gradient vector, yields

$$\begin{aligned} \Omega(0) &= \mathbf{0} \\ \Omega(n+1) &= \Omega(n) + \eta e_n \phi(\mathbf{s}_n) \\ \Omega(n) &= \eta \sum_{i=0}^{n-1} e_i \phi(\mathbf{s}_i) \end{aligned} \tag{2–10}$$

where $e_i = d_i - y_i$ and d_i is the desired signal. Given Ω and the input $\phi(\mathbf{s}_n)$, the output is given by

$$\begin{aligned} y_n &= \langle \Omega(n), \phi(\mathbf{s}_n) \rangle \\ &= \eta \sum_{i=1}^{n-1} e_i \langle \phi(\mathbf{s}_i), \phi(\mathbf{s}_n) \rangle \\ &= \eta \sum_{i=1}^{n-1} e_i \kappa(\mathbf{s}_i, \mathbf{s}_n) \end{aligned} \tag{2–11}$$

KLMS topology can be regarded as a growing radial basis function (RBF) network, which allocates a new unit at each sample with $\mathbf{s}_n$ as the center and the scaled prediction error ηe_n as the weight coefficient. The coefficients $a(n) = [\eta e_1, \cdots, \eta e_n]$ and the centers $\mathcal{C}(n) = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\}$ are stored in memory during training. KLMS on spike trains computes adaptively nonlinear regression without converging to local minima, which enables better adaption to neural system perturbations. The KLMS algorithm is intrinsically regularized by the step size, therefore this parameter should be carefully determined because it also affects the convergence rate [50].

The main bottleneck of kernel adaptive filtering is the linearly growing filter structure with each new sample, which poses both computational as well as memory issues especially for continuous adaptation scenarios. To address this problem, a variety of techniques [21, 29, 48, 73] have been proposed to curb the growth of the networks, where only non redundant input data are accepted as new centers. In this work, we adopt the quantization approach [19] to constrain the network growth, which discretizes the input space (and hence curbs the network size) by quantizing the input vector in the weight update equation. The algorithm is summarized below (Algorithm 1). Algorithm 1 shows the pseudocode for the KLMS with a simple online vector quantization (VQ) method, where the quantization is performed based on the distance between the new input $\mathbf{s}_n$ and each existing center $\mathbf{s}_i$. In this work, the distance between two spike timing sequences is defined by $\|\lambda_{\mathbf{s}_n} - \lambda_{\mathbf{s}_i}\|^2$. If

the smallest distance is smaller than a pre-specified quantization size ε, the new coefficient ηe_n adjusts the weight of the closest center, otherwise a new center is added. The simple online VQ method is not optimal but is very efficient. Since in our case the spike trains are of limited length, the algorithm must be applied several times to the same data, to allow convergence. Therefore, we choose $\varepsilon = 0$, which merges the repeated centers and enjoys the same performance as KLMS.

Algorithm 1. Quantized kernel least mean square (QKLMS) algorithm

---

Input: $\{\mathbf{s}_n, d_n\}$, $n = 1, 2, \ldots, N$

Initialization: Choose step-size parameter $\eta > 0$, kernel size $\sigma > 0$,and quantization size $\varepsilon \geq 0$ and initialize the weight vector $\Omega(1)$: codebook (set of centers) $\mathcal{C}(0) = \{\}$ and coefficient vector $a(0) = []$

Computation:

for $n = 1, 2, \ldots, N$

1) compute the output

$y_n = \langle \Omega(n), \phi(\mathbf{s}_n) \rangle = \sum_{j=1}^{\text{size}(\mathcal{C}(n-1))} a_j(n-1)\kappa(\mathbf{s}_n, \mathcal{C}_j(n-1))$

2) compute the error $e_n = d_n - y_n$

3) compute the distance between $\mathbf{s}_n$ and each $\mathbf{s}_i \in \mathcal{C}(n-1)$ $d(\mathbf{s}_n, \mathcal{C}(n-1)) = \min_j \|\mathbf{s}_n - \mathcal{C}_j(n-1)\|$

4) If $d(\mathbf{s}_n, \mathcal{C}(n-1)) \leq \varepsilon$, then keep the codebook unchanged: $\mathcal{C}(n) = \mathcal{C}(n-1)$, and update the coefficient of the center closest to $\mathbf{s}_n$: $a_k(n) = a_k(n-1) + \eta e(n)$, where $k = \text{argmin}_j \|\mathbf{s}(n) - \mathcal{C}_j(n-1)\|$

5) Otherwise, store the new center: $\mathcal{C}(n) = \{\mathcal{C}(n-1), \mathbf{s}_n\}$, $a(n) = [a(n-1), \eta e_n]$

end

2.2.2.3 Multiple-input-multiple-output (MIMO) decoding model

The rapid advance of microelectrode arrays and electrophysiological recording techniques has enabled the simultaneous stimulation and recording of spatiotemporal activity of hundreds of neurons. The MIMO model is utilized to decode multiple-channel microstimulation with multiple-channel neural activity. Although the transfer function from the spike train to the microstimulation may be a complex nonlinear relationship, in RKHS it is modeled as a linear dynamic filter $\mathbf{Y} = \langle \mathbf{W}, \phi(\mathbf{S}) \rangle$ that corresponds to a universal mapper in the input space. Assuming the decoder has K-channel inputs **S** (spike trains) and M-channel outputs **y** (stimulation), it is easy to extend the previous single input single output (SISO) model into a MIMO linear dynamic filter represented by a weight array defined by

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{11}\ \mathbf{W}_{12} \cdots \mathbf{W}_{1K} \\ \mathbf{W}_{21}\ \mathbf{W}_{22} \cdots \mathbf{W}_{2K} \\ \vdots \\ \mathbf{W}_{M1}\mathbf{W}_{M1} \cdots \mathbf{W}_{MK} \end{bmatrix} \begin{bmatrix} \phi(\mathbf{s}_1) \\ \phi(\mathbf{s}_2) \\ \vdots \\ \phi(\mathbf{s}_K) \end{bmatrix} \tag{2–12}$$

where each $\mathbf{W}_{ij}$ is a weight vector in the RKHS (Ω in (2–10)) in the is optimized by applying the quantized KLMS regression on each pair of $\mathbf{s}_j$ and $\mathbf{y}_i$. For illustration, the ith output of the controller $\hat{C}(z)$ can be calculated by

$$\begin{bmatrix} y_{i1} \\ y_{i2} \\ \vdots \\ y_{iM} \end{bmatrix} = \begin{bmatrix} a_{11}\ a_{12} \cdots a_{1N} \\ a_{12}\ a_{22} \cdots a_{2N} \\ \vdots\ \ \vdots \\ a_{M1}a_{M2} \cdots a_{MN} \end{bmatrix} \begin{bmatrix} \phi(\mathbf{c}_{11})\,\phi(\mathbf{c}_{12}) \cdots \phi(\mathbf{c}_{1K}) \\ \phi(\mathbf{c}_{21})\,\phi(\mathbf{c}_{22}) \cdots \phi(\mathbf{c}_{2K}) \\ \vdots\ \ \vdots \\ \phi(\mathbf{c}_{N1})\phi(\mathbf{c}_{N2}) \cdots \phi(\mathbf{c}_{NK}) \end{bmatrix} \begin{bmatrix} \phi(\mathbf{s}_{i1}) \\ \phi(\mathbf{s}_{i2}) \\ \vdots \\ \phi(\mathbf{s}_{iK}) \end{bmatrix} \tag{2–13}$$

where $\mathbf{c}_{nk}$ is the kth channel of the nth center and a_{mn} is the coefficient assigned to the nth kernel center for the m channel of the output. The way that the SISO decoder is extended to the MIMO decoder treats all the input channels independently without using the joint information among channels.

2.2.2.4 Comparison over kernels

In order to select an appropriate kernel for neural decoder from three candidates: CI kernel, NCI kernel, and Schoenberg kernel, we compare their decoding performance in synthetic data as a function of stimulation duration and signal-noise-ratio (SNR). The stimulation durations provide a way to control the transient stationarity of the neural response. More peaky the stimuli are, More dynamic the neural response are. The synthetic data is generated from a leaky integrate-and-fire (LIF) neuron stimulated by the continuous sin wave signal added with white noise. The stimulation patterns have four different durations (150ms 100ms 50ms and 20ms)as shown in the Figure 2-2, and two SNR levels (0dB and -2dB) that is tuned with noise power.

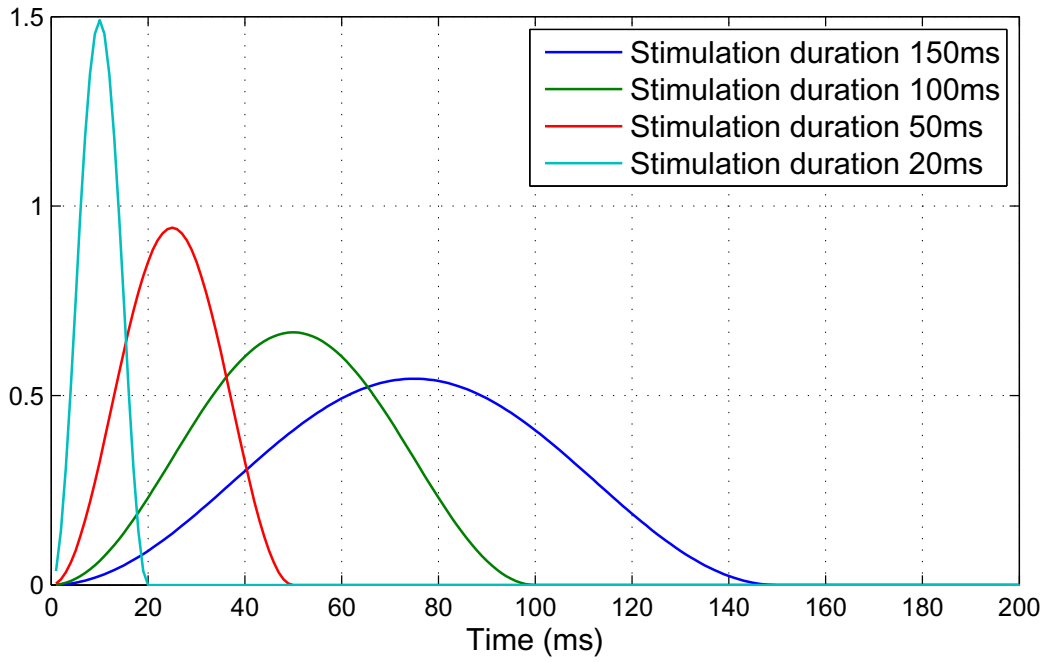


Figure 2-2. Synthetic stimulation

The kernel performance of 50 Monte Carlo runs is evaluated with respective to the stimulation duration and SNR, which is shown in the Figure 2-3. The results illustrate that the decrease of stimulation SNR is inversely proportional to the performance for all kernels. Since CI kernel-based decoder can only capture the linear relationship between

spike train and stimulation, CI kernel-based decoder performs worst. For the other two nonlinear kernels, Schoenberg kernel has a better performance than NCI kernel in small stimulation duration 0dB case. The reason is that NCI kernel more relies on the stationarity assume of the temporal data, since NCI kernel is similar to quantify the correntropy between two spike trains that is based on the average over time. Moreover, in contrast with NCI kernel, Schoenberg kernel is less influenced by the decrease of SNR, because the NCI pay more attention to the instantaneous value difference of two intensity function that is easily influence by the noise, whereas Schoenberg kernel emphasizes the overall temporal pattern of intensity function, which make Schoenberg kernel more reasonable for neural decoding. In order to further investigate the property of the kernels, We draw the performance surface with respect to the free parameters: window size and kernel width, as shown in Figure 2-4. The performance surface is estimated for the setting (stimulation duration 1ms with SNR 0dB). In this case, all three kernels are able to successfully decode the stimulation sequence. For NCI kernel, the kernel width is not dependent on the window size, which only related to the instantaneous value of the intensity function. In contrast, the performance of the Scheonberg kernel is more flat, which means it less sensitive to the selection of the free parameters. The decoding accuracy and robustness of the free parameters conclude that Schoenberg kernel is better candidate for our neural decoding task. Therefore, it is selected to build the the kernel-based neural decoder in the following work.

2.2.3 Synthetic Experiment

2.2.3.1 Neural circuit

We build a biological plausible plant model of an ensemble of neurons, which is implemented in the neural Circuit SIMulator (CSIM) [38]. We model the projection of a time-varying three dimensional electric field at the neurons from bipolar and monopolar stimulations on a 2×8 electrode configuration, as shown in Figure 2-6. In this manner

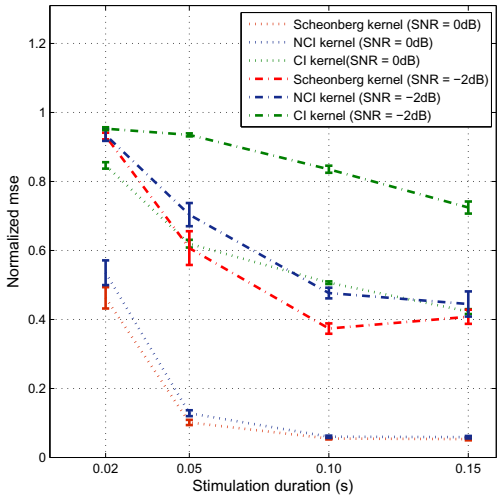


Figure 2-3. Comparison among the kernels defined in the space of spike trains: CI, NCI, and Schoenberg kernel

the current to each neuron is the projection between the three dimensional electric field onto a neuron-specific unit vector, representing neuronal elements such as axons which carry the current to neural circuit and are preferentially excited by one of the projections of the electric field.

The neural ensemble has a two layer structure, as shown in Figure 2-5. The left layer is the input layer, where each input neuron is colored by its projection direction. The ensemble of neurons consists of 135 leaky integrate-and-fire neurons, 20% of which are randomly chosen to be inhibitory (orange). Random circuits are constructed with sparse, primarily local connectivity. Parameters of neurons and synapses are based on [51], which are chosen to fit data from microcircuits in rat somatosensory cortex. Random circuits are constructed with sparse, primarily local connectivity, both to fit anatomical data and to avoid chaotic effects.

The stimulation is generated by filtering the electrical field with a low pass filter $\exp(-\frac{x}{\tau})$, where $\tau = 10$ ms, which is in the normal time constant range of neuron cell body and dendrites [39]. For each stimulation configuration,which includes a discrete set

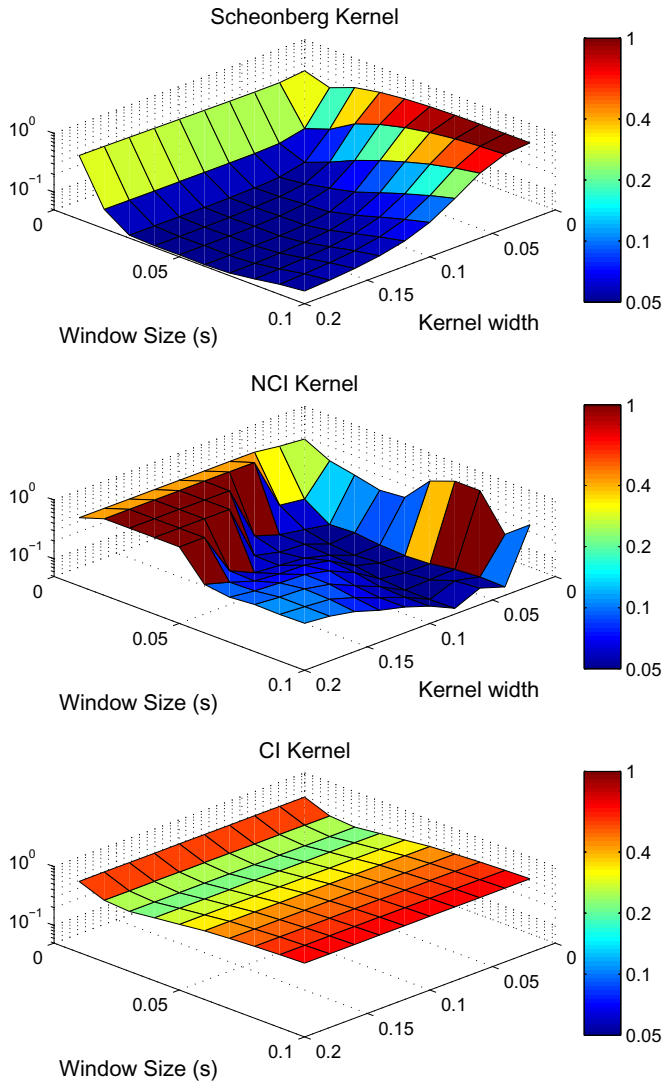


Figure 2-4. Decoder performance surface with respect to the kernel free parameters

of amplitudes, the single neural firing rate is estimated in the 50 ms interval right after the stimulation. Its mean and standard deviation over 10 repeated trails are shown in Figure 2-6, where the first subfigure shows the bipolar and monopolar stimulations on a 2×8 electrode configurations which is used to model the 3D electric field as shown in the second subfigure. The continuous stimulation is generated by passing these 3D electrical field configuration patterns through a low pass filter $exp(-\frac{x}{\tau})$, where $\tau = 10$

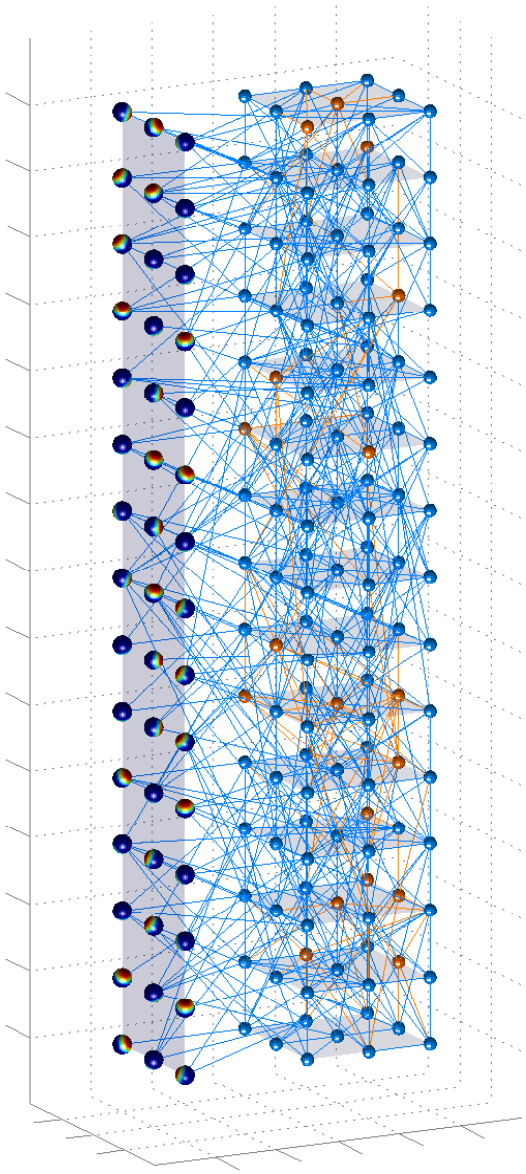


Figure 2-5. A biological plausible synthetic neural circuit.

ms and imprinted into the synthetic neural model (Figure 2-5). For each stimulation configuration, the single neural firing rate is estimated in the 50 ms interval right after the stimulation. Its mean and standard derivation over 10 repeated trails are shown in the third and fourth subfigures, respectively.

The continuous stimulation sequence is formed by uniformly drawing the stimulation configuration with i.i.d. intervals between stimulations. The distribution of the intervals is

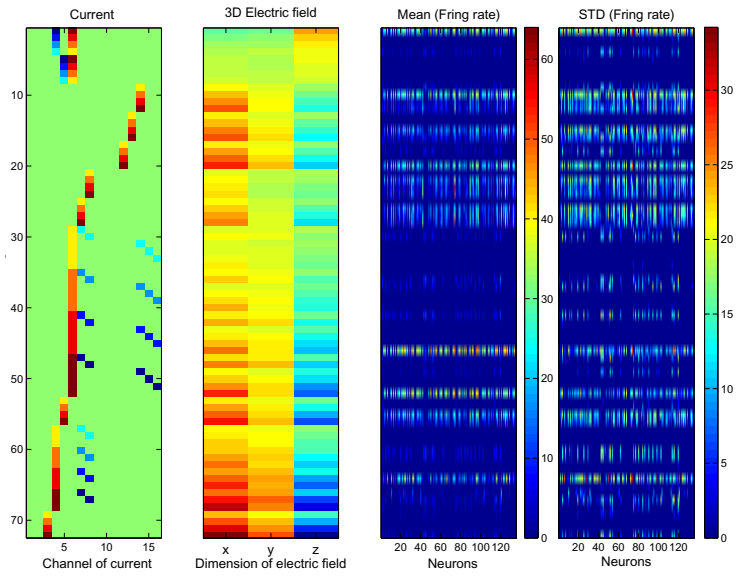


Figure 2-6. The stimulation configuration and the statistics of the neuron response.

a truncated normal distribution $T \sim N(0.05, 0.0025)$, $T \in [0, 0.1]$ (s). An example of the stimulation and the corresponding neural response are shown in Figure 2-7.

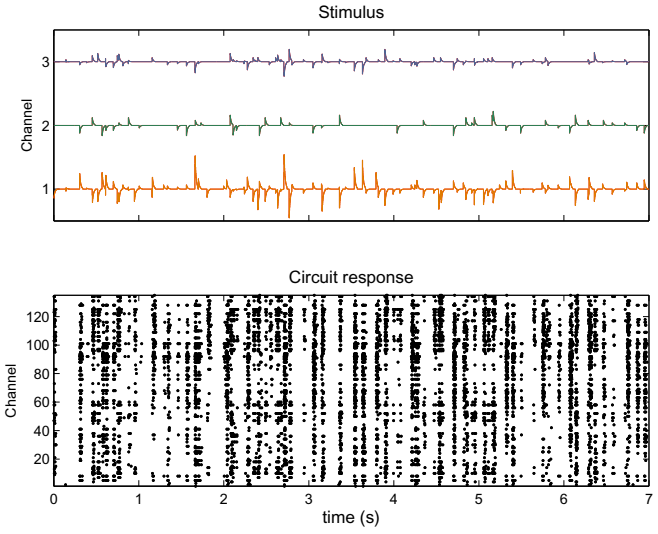


Figure 2-7. The pattern of the 3D stimulation defined by the electrical field (the upper plot) and the resulting neural response (the bottom plot).

2.2.3.2 Comparison over decoders

The decoder inputs are neural event timing vectors, which are obtained by sliding a 200 ms window every 2 ms (window size: [-100 ms 100 ms]). Each window includes the previous -100 ms spike train to help cancel the ambiguity induced by the lasting influence of previous stimulus on the current neural firing pattern. The corresponding output is the estimated amplitude of the stimulation at the time t.

A decoding model based on the Schoenberg kernel induced RKHS is updated via KLMS. For comparison, we also apply the other two neural decoders based on the discretization representation of spike train: GLM [71][2] and spikernel [91] [3] (KLMS is also used to update the model coefficients). Time discretization is 2 ms, which is determined by the required time resolution of the micro-stimulation. The models are determined by the best results after scanning the parameters, (for the Schoenberg KLMS the kernel width $\sigma^2 = 0.3$ and the learning rate $\eta = 0.001$ are chosen). The normalized mean square error (NMSE) between the reconstructed stimulation (**y**) and the desired stimulation (**d**) is utilized as an accuracy criteria defined by

$$NMSE = \frac{\frac{1}{N}\sum(d(n) - y(n))^2}{\mathrm{var}(d(n))} \tag{2–14}$$

NMSEs are obtained by performing ten fold cross-validation for each decoder, which are produced by randomly splitting the data into 10 groups, 400 samples per group. We use nine out of the ten folds of the data to train decoders and compute an independent test error on the remaining fold. The results are shown in Table 1. Schoenberg-kernel-based decoder out-performs the spikernel-based decoder

[2] $p(\mathbf{x})$ is computed based on a Gaussian distribution $\sim N(\mu, \sigma^2)$ (μ and σ^2 are estimated from the data).

[3] Free parameters μ and λ are 0.7 and 0.99, which are determined by the best results after scanning the parameters.

Table 2-1. Comparison among neuron decoders. N_i and M_i represent the spike number and the window length, respectively.

Property \ Method	Schoenberg kernel	Spikernel	GLM
NMSE (mean/STD)	0.23/0.15	0.29/0.12	0.75/0.23
Kernel computation time	$O(N_i N_j)$	$O(M_i M_j)$	X
Kernel free parameter	1	3	X

and the GLM-based decoder. Although the performance of spikernel is close to that of Schoenberg kernel, its computation time $O(M_i M_j)$ is much greater than that of Schoenberg kernel $O(N_i N_j)$, where $M_i = 100$ and $N_i \approx 4$ with the average firing rate of 20 Hz.

2.2.4 Animal Experiment

2.2.4.1 Rat data

Animal procedures were approved by SUNY Downstate Medical Center IACUC and conformed to National Institutes of Health guidelines. A single female Long-Evans rat (Hilltop, Scottsdale, PA) was implanted with two microarrays (16 contacts each in a 2×8 grid Neuronexus). The electrodes covered somatosensory areas of the cortex S1, and the VPL nucleus of the thalamus [32]. Neural recordings were made using a multichannel acquisition system (Tucker Davis). The rat was placed into a small chamber with a mesh floor which was suspended above a table. The apparatus helped keep it calm and stationary even though they remained awake. Spike and field potential data was preamplified 1000x (filter cutoffs at 0.7 and 8.8kHz) and digitized at 25kHz. LFPs were further filtered from 1 to 300Hz using a 3rd order Butterworth filter. Spike sorting was achieved using k-means clustering of the first 2 principal components of the detected waveforms. The experiment involves delivering microstimulation to VPL and tactile stimulation to rat's fingers in separate sections.

Microstimulation was administered on adjacent pairs (dipoles) of the thalamic array. The stimulation waveforms were single symmetric biphasic rectangular current pulses; Each rectangular pulse was 200μs long and had an amplitude from the set 10μA; 20μA;

and 30μA. Inter-stimulus intervals were exponentially distributed with mean interval of 100ms. Stimulus isolation used a custom built switching headstage. The bipolar micro-stimulation being imprinted in Thalamus, which contains 24 patterns: 8 different locations, 3 different amplitude levels for each location, and 125 events for each pattern, as shown in Figure 2-8.

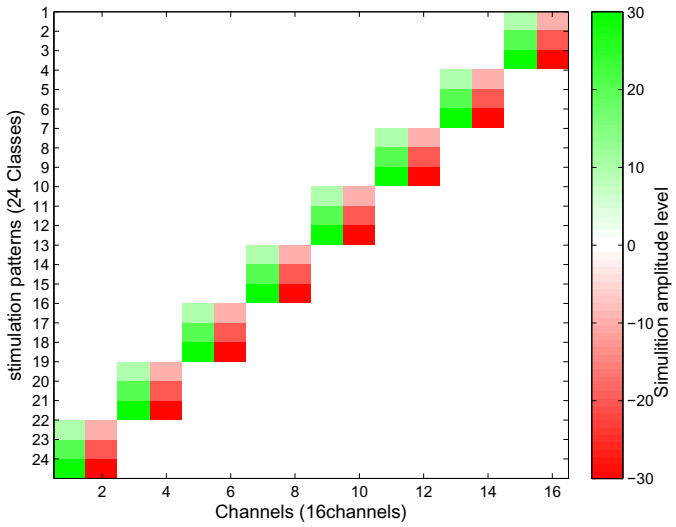


Figure 2-8. Micro-stimulation patterns

The experimental procedure also involved delivering 30-40 short 100ms tactile touches to the rat's fingers (repeated for digit pads 1-4) using a hand-held probe. The rat remained still for the recording durations, and trials were canceled if the rat changed orientation. The applied force was measured using a lever attached to the probe that pressed against a thin-film resistive force sensor (Trossen Robotics) when the probe tip contacted the rat's body. The resistive changes were converted to voltage using a bridge circuit and were filtered and digitized in the same way as described above. The digitized waveforms were then de-meaned and filtered at 1 to 60Hz using a 3rd order Butterworth filter. The first derivative of this signal is used as the desired stimulation signal, which are shown in Figure 2-9.

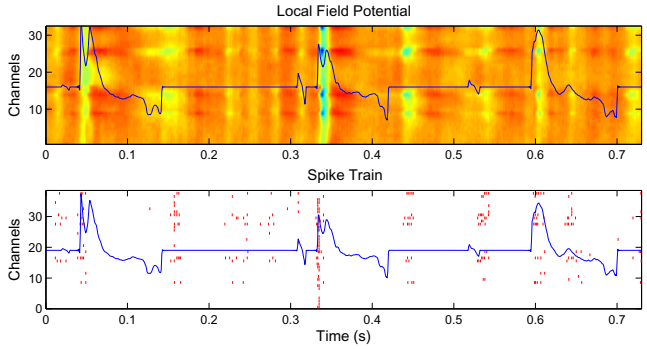


Figure 2-9. Data of the rat sensory stimulation experiment

2.2.4.2 Results of tactile stimulation

We now present the results of decoding the tactile stimulus waveform and micro-stimulation using KLMS operating on the Schoenberg kernel. Time discretization is 5 ms. The learning rate and kernel width are determined by the best results of test data after scanning the parameters. The normalized mean square error (NMSE) between the estimated stimulation (**y**) and the desired stimulation (**d**) is utilized as an accuracy criteria.

NMSEs of tactile are obtained across 8 trials data sets. For each trial, we use 20s data to train the decoders and compute an independent test error on the remaining 2.5s data. The mean and standard derivation of normalized mean square error is 0.63/0.11 (mean/standard derivation). In order to illustrate the details of the decoding performance, the test results of the first trial are illustrated in Figure 3-3. The decoder output is able to capture main structure of the desired tactile stimulation. It is observed that the output of the spike decoder fluctuates enormously and misses some pulses, i.e., around 0.65 s, because of the sparsity and variability of the spike train.

2.2.4.3 Results of micro-stimulation stimulation

We also implement the decoder to reconstruct the micro-stimulation. First, we map the 8 different stimulation locations to 8 channels. We dismiss the shape of each stimulus, since the time scale the stimulus width is two short (200μs). The desired stimulation pattern of each channel is represented by a sparse time series of the

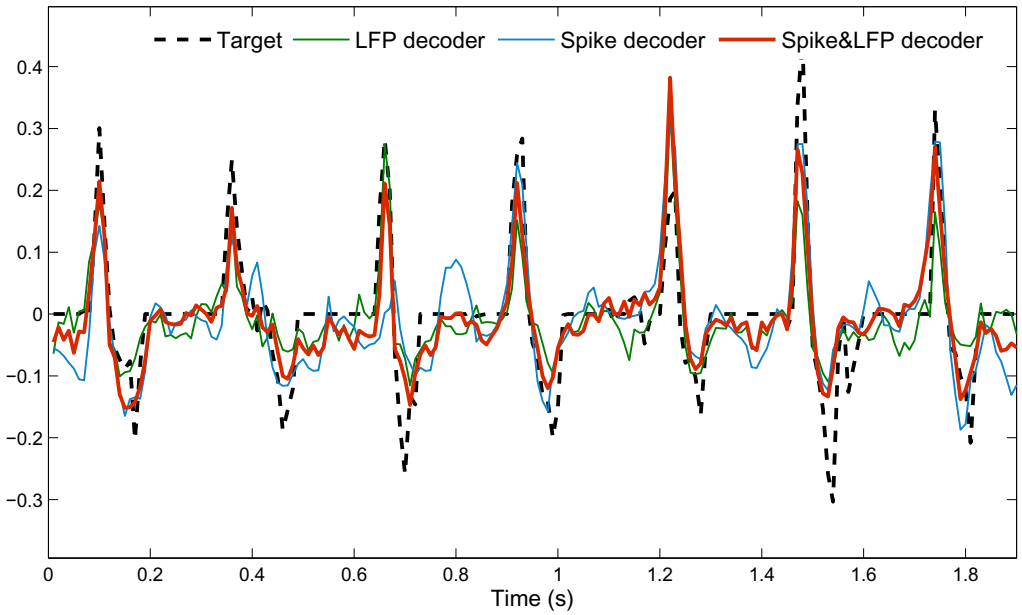
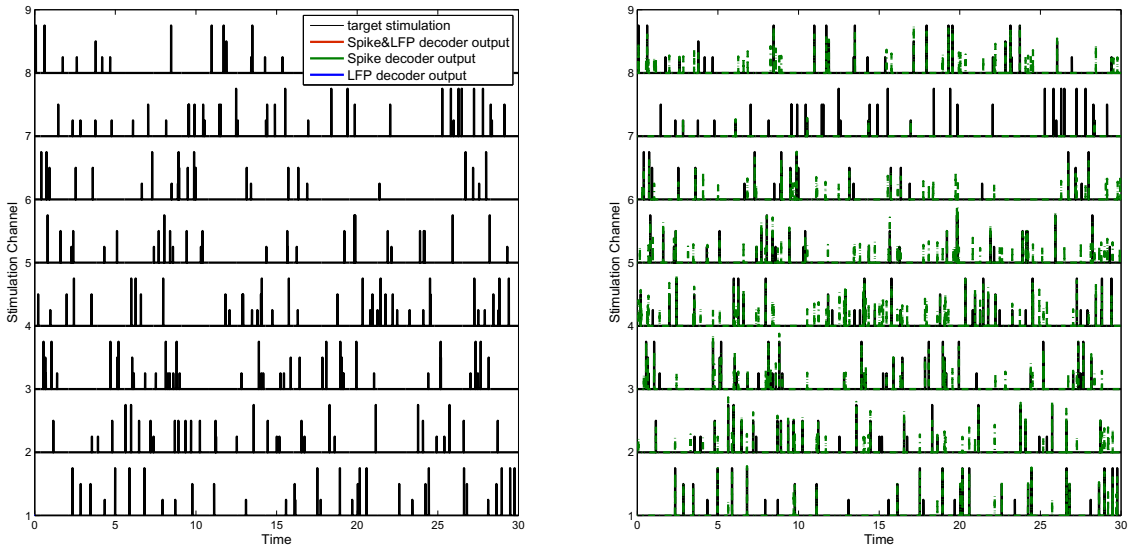


Figure 2-10. Results of the Schoenberg kernel based decoder of the first stimulation trial.

stimulation amplitude with sampling rate 200Hz. We implement Shoenberg kernel with KLMS for MIMO system on the micro stimulation data. NMSEs are obtained with ten subsequence decoding results, where we use 120s data to train the decoders and compute an independent test error on the remaining 20s data. The spike train decoder successfully captures the stimulation pattern in Channel 1 2 3 4 5 6 and 8. The stimulation is not well discriminated from different output channels, which causes the false-stimuli present in spike decoder output.



A Desired stimulation pattern

B Spike decoder output

Figure 2-11. Results of reconstructing micro-stimulation.

2.3 Adaptive Inverse Control of Neural Spatiotemporal Spike Patterns

In the MIMO control system scheme, the plant represents the neural circuit driven by the stimulation control signal and produces spike activity outputs. The basic idea of adaptive inverse control is to optimize the controller to be the inverse of the plant 2-12. The adaptive regression algorithm attempts to make the cascade of the controller and the plant behave like a unit transfer function (i.e. a perfect wire with some delay). Therefore, the target spike train is used as the command input of the inverse controller to generate the control signal output, which is able to drive the plant such that the output spike train follows the target firing pattern.

2.3.1 Filtered-ϵ LMS algorithm

The filtered-ϵ LMS adaptive inverse control diagram [105] shown in Figure 2-12 represents the filtered-ϵ approach to find $\hat{C}(z)$. If the ideal inverse controller $C(z)$ is the actual inverse controller, the mean square of the overall system error ϵ_k would be minimized. The objective is to make $\hat{C}(z)$ as close as possible to the ideal $C(z)$. The

difference between the outputs of $C(z)$ and $\hat{C}(z)$, both driven by the command input, is therefore an error signal ϵ'. Since the target stimulation is unknown, instead of ϵ', a filtered error ϵ, obtained by filtering the overall system error ϵ_k through the inverse plant model $\hat{P}^{-1}(z)$, is used for adaptation in place of ϵ'.

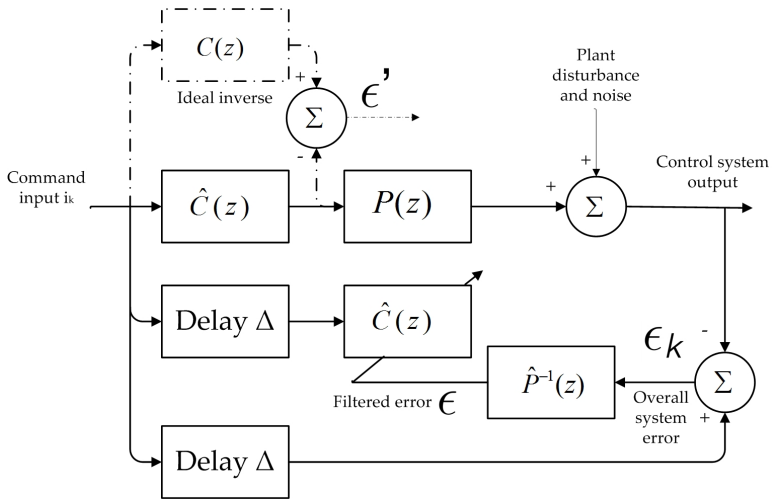


Figure 2-12. A Filtered-ϵ LMS adaptive inverse control diagram.

During the adaption of the inverse controller $\hat{C}(z)$, because of the neural system response time, a modeling delay is advantageous, which is determined by the sliding window length that is used to obtain the inverse controller input. There is no performance penalty from the delay Δ as long as the input to $\hat{C}(z)$ undergoes the same delay. The parameters of the inverse model $\hat{P}^{-1}(z)$ are initially modeled offline and updated during the whole system operation, which allows $\hat{P}^{-1}(z)$ to incrementally identify the inverse system and thus make ϵ approach ϵ'. Moreover, the adaptation enables $\hat{P}^{-1}(z)$ to track neural plasticity. Thus minimizing the filter error obtained from $\hat{P}^{-1}(z)$ makes the controller to follow the system variation.

If $\hat{P}^{-1}(z)$ is the perfect inverse plant and the plant is free of disturbance, then the filtered error is exactly equivalent to $\epsilon\prime$. In spite of these limitations, the filtered error can still perform well for $\hat{C}(z)$ adaptation, because first, the plant disturbances are normally

uncorrelated with the plant input and the command input and second, errors in $\hat{P}^{-1}(z)$ may not be critical to the correct convergence of $\hat{C}(z)$ [105].

2.3.2 Control design in RKHS

In this control scheme, both the command input—the target spike trains—and the controlled neural system output are spike trains. The spike train kernels map the spike trains into the RKHS; this avoids the sparseness and dimensionality problem of using binned data. In addition, in the RKHS, both of the models (the plant inverse, $\hat{P}^{-1}(z)$ and the controller $\hat{C}(z)$) are linear so there is no danger of converging to local minima. In our work, KLMS is used to model the controller, which keeps the adaptation computation cost at $\mathrm{O}(n)$, where n represents the sample size. A kernel-based adaptive inverse control has been proposed in [102], which also allows a linear control in feature space. However, the support vector regression was used to model the controller, which makes the computation cost become large ($\mathrm{O}(n^2)$).

Specifically, the controller $\hat{C}(z)$ and the plant inverse $\hat{P}^{-1}(z)$ are separately modeled with the Schoenberg kernel, and the model coefficients are updated with KLMS. This structure is shown in Figure 2-13. The model coefficients $\mathbf{W}_{\hat{C}}$ and $\mathbf{W}_{\hat{P}^{-1}}$ represent the weight matrix of $\hat{C}(z)$ and $\hat{P}^{-1}(z)$ obtained by KLMS, respectively. $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are the windowed target spike train (command input) of the controller, the estimated stimulation, and the windowed plant output spike train, respectively. $\mathbf{x}_\Delta$ is delayed target signal, which is aligned with the plant output $\mathbf{z}$. $\phi(\cdot)$ represents the mapping function from input space to RKHS.

For the controller $\hat{C}(z)$, the filter error ϵ is used as a substitute of $\epsilon\prime$ to estimate kernel center coefficients. The filtered error is obtained by filtering the overall system error ϵ_k through an inverse plant model $\hat{P}^{-1}(z)$. For the neural system, the overall system error ϵ_k should be the difference between two spike trains. Fortunately, in the RKHS for ith channel $\phi(\mathbf{x}_{\Delta i}) - \phi(\mathbf{z}_i)$ is well defined. Therefore, the overall system error for channel i can be defined by $\epsilon_k(i) = \phi(\mathbf{x}_{\Delta i}) - \phi(\mathbf{z}_i)$, which mean the adaptation of the

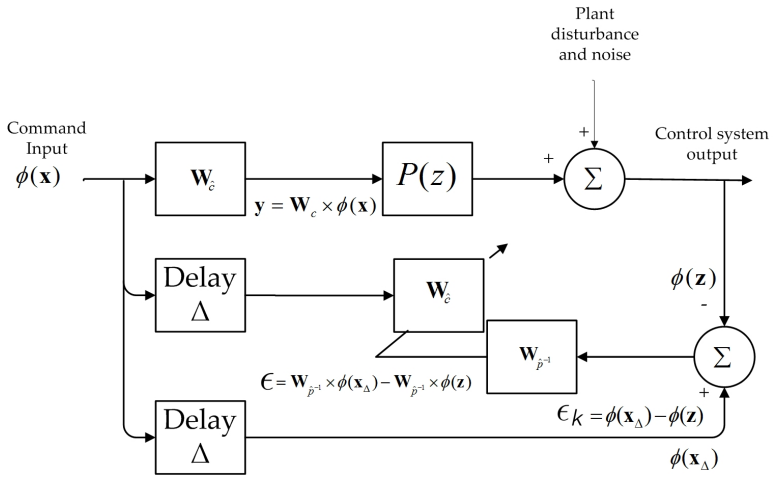


Figure 2-13. An adaptive inverse control diagram in RKHS of spike train.

controller parameters seeks to minimize the distance between the target spike train and the plant output in the RKHS of the plant inverse $\hat{P}^{-1}(z)$. In this way, since the inverse model $\hat{P}^{-1}(z)$ has a linear structure in RKHS, the filtered error for stimulation channel $j \in 1, \ldots, M$ is

$$\begin{aligned}\epsilon(j) = &\left[\mathbf{W}_{j1}^{\hat{P}^{-1}}\mathbf{W}_{j2}^{\hat{P}^{-1}}\cdots\mathbf{W}_{jK}^{\hat{P}^{-1}}\right]\left[\phi(\mathbf{x}_{\Delta 1})\phi(\mathbf{x}_{\Delta 2})\cdots\phi(\mathbf{x}_{\Delta K})\right]' \\ &- \left[\mathbf{W}_{j1}^{\hat{P}^{-1}}\mathbf{W}_{j2}^{\hat{P}^{-1}}\cdots\mathbf{W}_{jK}^{\hat{P}^{-1}}\right]\left[\phi(\mathbf{z}_1)\phi(\mathbf{z}_2)\cdots\phi(\mathbf{z}_K)\right]'\end{aligned} \tag{2–15}$$

We assume $\hat{C}(z)$ has K-channel inputs **x** (spike trains) and M-channel outputs **y** (stimulation). KLMS with quantization is used to model $\hat{C}(z)$ with N input samples. Since among different trials, the target spike train is repeated, which means the repeated samples will be merged to the same kernel center by the quantization and thus the network size of the inverse controller is fixed (N centers), only the coefficient matrix **a** is updated with the filtered error ϵ during the whole control operation. The ith output of $\hat{C}(z)$ can be calculated by

$$\begin{bmatrix} y_{i1} \\ y_{i2} \\ \vdots \\ y_{iM} \end{bmatrix} = \begin{bmatrix} a_{11}\ a_{12}\cdots a_{1N} \\ a_{12}\ a_{22}\cdots a_{2N} \\ \vdots\quad\vdots \\ a_{M1}a_{M2}\cdots a_{MN} \end{bmatrix} \begin{bmatrix} \phi(\mathbf{c}_{11}^{\hat{c}})\,\phi(\mathbf{c}_{12}^{\hat{c}})\cdots\phi(\mathbf{c}_{1K}^{\hat{c}}) \\ \phi(\mathbf{c}_{21}^{\hat{c}})\,\phi(\mathbf{c}_{22}^{\hat{c}})\cdots\phi(\mathbf{c}_{2K}^{\hat{c}}) \\ \vdots\qquad\vdots \\ \phi(\mathbf{c}_{N1}^{\hat{c}})\phi(\mathbf{c}_{N2}^{\hat{c}})\cdots\phi(\mathbf{c}_{NK}^{\hat{c}}) \end{bmatrix} \begin{bmatrix} \phi(\mathbf{x}_{i1}) \\ \phi(\mathbf{x}_{i2}) \\ \vdots \\ \phi(\mathbf{x}_{iK}) \end{bmatrix} \tag{2–16}$$

Where $\mathbf{c}_{nk}^{\hat{c}}$ is the kth channel of the nth center and a_{mn} is the coefficient assigned to the nth kernel center for the m channel of the output.

2.3.3 Synthetic experiment

2.3.3.1 Controlling neural firing pattern without perturbations

We conduct an online control of the synthetic neural circuit with the adaptive inverse control framework, where the inverse controller $\hat{C}(z)$ and the inverse model $\hat{P}^{-1}(z)$ are modeled by Schoenberg-kernel-based decoders. The inverse model $\hat{P}^{-1}(z)$ is initially trained offline with 6 s of training data: the 135-channel windowed spike timing sequences as the input and the corresponding stimulation as the desired signal. After initialization, a novel 2 s spike trains (135 channels) are used as the target firing pattern and repeatedly concatenated as the input of the inverse controller. The resulting stimulation is feed to the neural system plant. The parameters of the inverse controller $\hat{C}(z)$ are adjusted sample-by-sample with the learning rate $\eta = 0.001$ by minimizing the current filtered error $\epsilon(i)$, which is obtained by passing the output of the plant ($\mathbf{z}(i)$) and target spike trains with delay $\Delta = 100$ ms ($\mathbf{x}_\Delta(i)$) through the inverse model $\hat{P}^{-1}(z)$. To maintain the system stability, we update the inverse model $\hat{P}^{-1}(z)$ every 2 s with the plant output $\mathbf{y}$ recorded in the last 2 s as input and the corresponding plant stimulation as the desire signal (learning rate $\eta = 0.001$). We implement the same adaptation procedure in the following parts of this chapter. Figure 2-14 illustrates the comparison between the neural response driven by the control signal and the target spike trains.

The main goal of control is to minimize the dissimilarity between the system output and the target firing pattern. Therefore, the control performance is evaluated using the

similarity measure proposed in [65, 88] between two spike trains. The two spike train $\mathbf{s}_1(t)$ and $\mathbf{s}_2(t)$ are first convolved with the smoothing kernel $h(t) = exp(-t/\tau)U(t)$, yielding the time series of $\hat{\lambda}_{\mathbf{s}_1}(t)$ and $\hat{\lambda}_{\mathbf{s}_2}(t)$, respectively, where the time constant $\tau = 100$ ms that overlaps 2 spikes on average (mean firing rate of all neurons: 20 Hz). Then the pairwise correlation coefficient between $\hat{\lambda}_{\mathbf{s}_1}(t)$ and $\hat{\lambda}_{\mathbf{s}_2}(t)$ is computed:

$$Similarity = \frac{\int_{\mathcal{T}} \hat{\lambda}_{\mathbf{s}_1}(t)\hat{\lambda}_{\mathbf{s}_2}(t)dt}{\sqrt{\int_{\mathcal{T}} \hat{\lambda}^2_{\mathbf{s}_1}(t)dt}\sqrt{\int_{\mathcal{T}} \hat{\lambda}^2_{\mathbf{s}_2}(t)dt}} \tag{2–17}$$

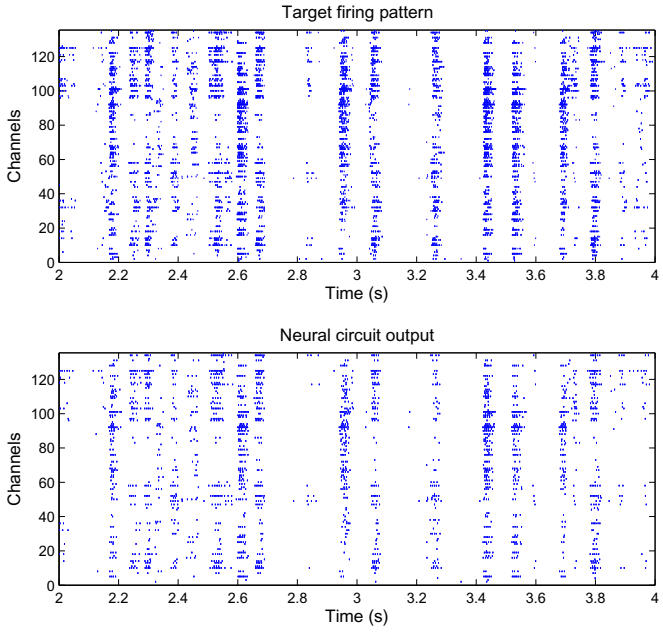
where $Similarity \in [0, 1]$. If and only if $\mathbf{s}_1(t)$ and $\mathbf{s}_2(t)$ are identical, $Similarity = 1$. If two spike trains are both empty, the similarity is defined by 0.

The similarity between the system output and target spike trains over each channels is calculated to evaluate the controller performance to drive the neural circuit to produce the target spatiotemporal firing pattern, as shown in Figure 2-14B. The similarity per channel is sorted by neurons' firing rate. The boxplot of the similarity over the channels are estimated, which demonstrates that the similarity of 50% neurons are over 0.83. Because of the sparse, primarily local connectivity and the existing inhibitory synaptic connection, neurons with low average firing rate are unable to be modulated by the stimulation, which explains the low similarity of neurons with low firing rates.

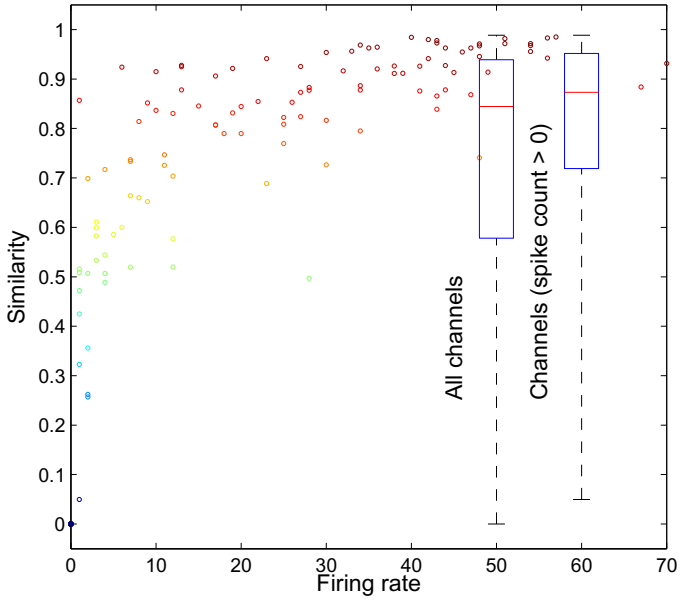
In this case, since the target neural pattern is generated from the current neural circuit, the plant model is able to reproduce the target spatiotemporal firing pattern well, when the controller models the inverse of the neural circuit with high accuracy, so this is a best case scenario. In any case, our results show that under these conditions, this control scheme is able to find the hidden desired microstimulation and force the output neurons to fire in a similar pattern.

2.3.3.2 Controlling neural firing pattern with perturbations

Considering neuronal plasticity induced by injury, environmental changes, or micro-stimulation, it is necessary to test the ability of the inverse controller to track



A Comparison between the neural system output and the target spatiotemporal firing pattern.



B Single channel similarity between the neural system output and the target spatiotemporal firing pattern. The similarity boxplot is for the population of neurons.

Figure 2-14. Performance of stimulation optimization using adaptive inverse control.

underlying changes of the functional relationship between stimulation and neural responses. The perturbation induced by randomly resetting the synaptic connectivity and the localization of the inhibitory neurons in neural circuit is used to test the adapting capability of our control system. In order to quantify the difference between the original neural circuit and the perturbed one, the spike-triggered average (STA) of each neuron for three dimension stimulation is estimated to demonstrate the difference of the neural response property in Figure 2-15, where each subfigure shows the 100 ms STAs for one dimension stimulation, where STAs of 135 neurons are demonstrated in a 15(x-axis)$\times$9(y-axis) array.

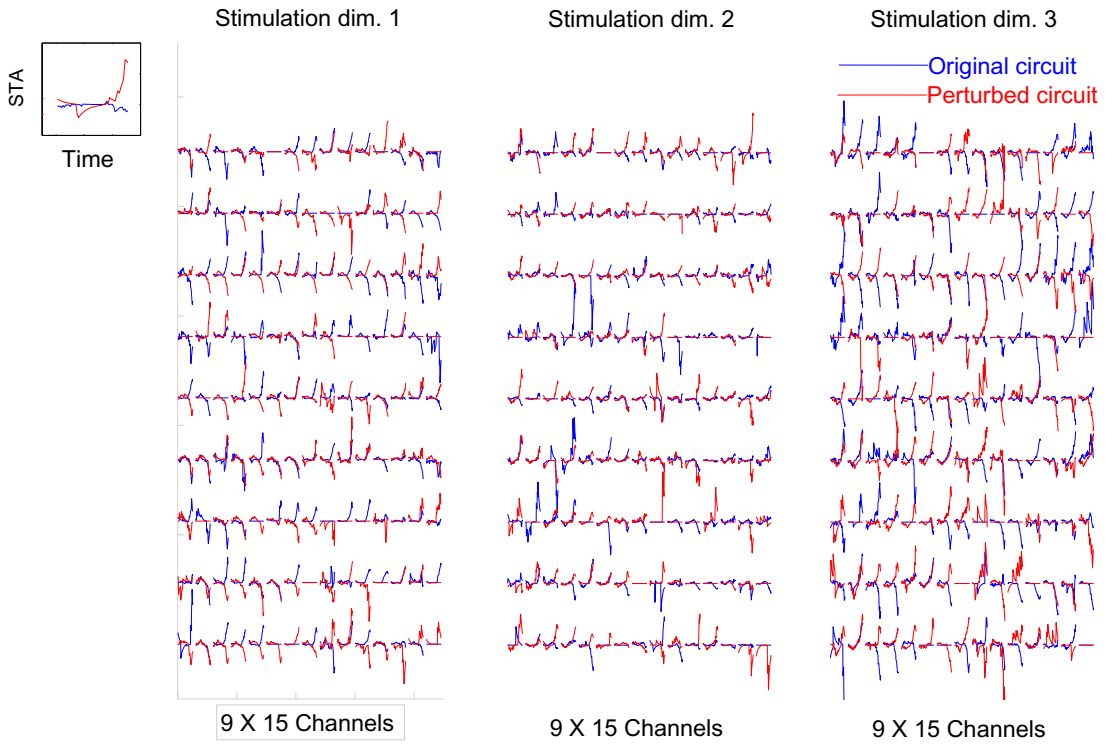
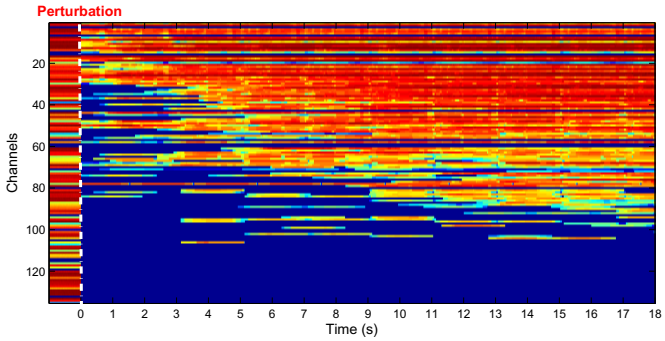


Figure 2-15. Comparison of spike-triggered averages (STA) of the three dimensional filtered electric field between the original neural circuit and the perturbed neural circuit.

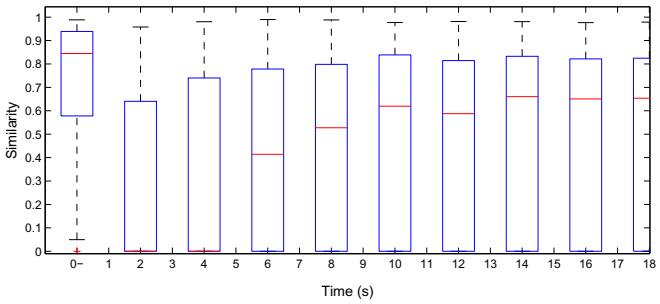
We use the inverse controller $\hat{C}(z)$ and the inverse model $\hat{P}^{-1}(z)$ that converged to the inverse of the original neural circuit with the previous data as the initial condition.

The same adaptation process is implemented on the control system to control the perturbed neural circuit. The neural perturbation adapting procedure is illustrated by the sample-by-sample evolution of the similarity between the neural system output and the target spike trains in Figure 2-16A, which is estimated by sliding a 2 s window by 2 ms steps and sorted by the average firing rate of each channel. We also draw the similarity boxplot for the population of neurons every 2 s as shown in Figure 2-16B.

Variations in the underlying neural organization initially cause dissimilarity between the system output and target spike trains (left of the Figure 2-16A. The adaptation allows the inverse controller to converge to the inverse of the current neural circuit and thus progressively increases the similarity between the system output and target.



A Performance evolution during adaptation.



B Boxplot of similarity over channels during adaptation.

Figure 2-16. The performance evolution during adaptation of the inverse controller to follow the perturbed neuron circuit.

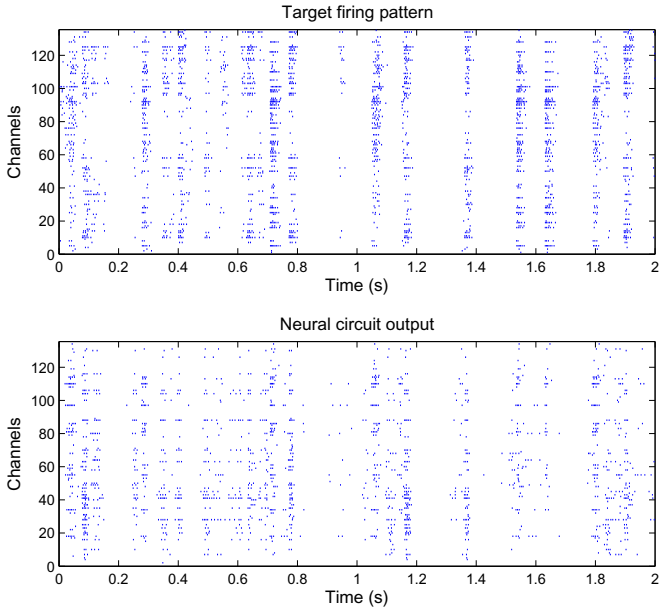
The controller adaptation converges after 9 trials with the following stopping criterion: the difference of the mean similarity of all neurons between two adjacent adaptations is less than 0.015. At this point, the similarity boxplot in Figure 2-17B shows that 50% of neurons have a similarity of at least 0.64. The similarity per neuron sorted by their firing rate is shown in Figure 2-17B, which reveals that the control scheme is capable of tracking neural perturbations by adapting the controller to the inverse of the perturbed neural circuit. In Figure 2-17B, the perturbation of neuron circuit happens at 0 s. Figure 2-16A shows the single channel similarity between the neuron system output and the target spatiotemporal firing pattern. Figure 2-16B shows the similarity boxplot over all channels, which is estimated every 2 s. The first boxplot is estimated immediately before the perturbation. The first boxplot in Figure 2-16B is equal to the one in Figure 2-14B (before perturbation) and the last boxplot in Figure 2-16B is equal to the one in Figure 2-17B (after convergence).

However, the results of the perturbed neural circuit become poorer than the ones with the original neural circuit. This is expected, because the target firing pattern is not generated by the perturbed neural circuit. Therefore, there is no guarantee that the perturbed neural circuit is capable of producing the target neural pattern, i.e. there may not exist a stimulation that can drive the perturbed neural circuit to reproduce the target firing pattern. In any case, this is actually a more realistic situation, when we use the elicited neural response to mimic the firing pattern induced by the natural stimulation, since the micro-stimulation may change the original underlying neural circuit structure that generates the target firing pattern with natural stimulation. Nevertheless, the inverse controller will always converge to minimum of the cost, because the whole control scheme has a linear structure in RKHS and it is trained with a convex cost function.

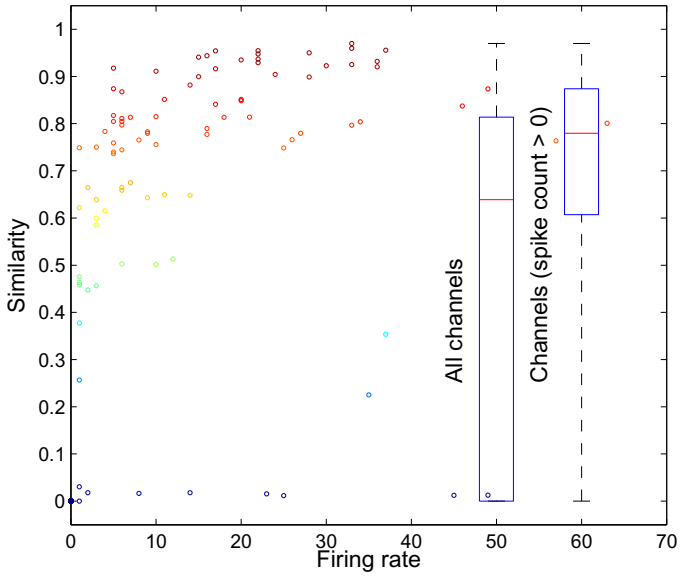
2.4 Discussion

The sparseness of neural signals and the neural system plasticity induced by habituation to stimulation, injury, environment changes, etc. present a challenge when

attempting to precisely control the neural response to mimic the target spatiotemporal firing pattern induced by natural stimulation. This chapter proposes a neural decoding methodology with high time resolution based on a Schoenberg kernel regressor defined directly in the space of the neural event timings instead of the more conventional intensity rate representation. This methodology bypasses the sparseness of conventional intensity rate representation, efficiently decreases the computation time by using only the precise timing of the neural events instead of high model orders, and the results on the synthetic data show that this decoding methodology outperforms the methods based on the discretized representation of spike train: Spikernel and GLM. With this novel decoding methodology, a MIMO adaptive inverse control scheme is built on spike train RKHS to control neural response to match the target spatiotemporal firing pattern. The experimental results are presented in a synthetic neural system built from LIF neurons but the excitation is diverse and broad to mimic more realistic conditions. They show that the controller adaptation allows the control system to provide the desired firing patterns with high accuracy. Moreover, the controller is also able to adapt to perturbation of the underlying neural system organization, which is an essential quality for the control system, otherwise the control system will fail to track the target signal when perturbations happen. In this more realistic case, the desired stimulation pattern may not exist to reproduce the target signal but the linear structure of the controller and its convex cost function guarantee that this control scheme is able to find the global optimal spatiotemporal pattern of stimulation that can minimize the dissimilarity between the system output and the target signal. These preliminary results show that this RKHS based methodology for somatosensoty stimulation may be applicable in real stimulations. Our major concern is the requirement to have sufficient neural spiking to drive the adaptation and the influence of the spike train variability and this will be addressed in real neural system.



A Comparison between the neural system output and the target spatiotemporal firing pattern after the inverse controller converged to the inverse of the perturbed neuron circuit.



B Single channel similarity between the neural system output and the target spatiotemporal firing pattern after the inverse controller converged to the inverse of the perturbed neuron circuit. The similarity boxplot is for the population of neurons.

Figure 2-17. Performance of stimulation optimization after a perturbation using adaptive inverse control.

CHAPTER 3
NEURAL DECODING FROM MULTI-SCALE NEURAL ACTIVITY

The problem of extracting information from spike trains and local field potentials (LFPs), electrocorticogram (ECG), and electroencephalogram (EEG) is rooted in their unknown underlying relation, multiple spatial and temporal scale and heterogenous signal format. In our work, we mainly address the modeling task of spike trains and LFPs as example of multiscale neural modeling, since it is able to cover most challenges in multiscale neural decoding. Another reason of selecting spike trains and LFPs is that they are recorded form the same micro-electrode arrays, which guarantees the time alignment and thus simplify the data preprocessing procedure.

Spike trains and LFPs encode complementary information of the stimuli or behaviors [8, 37]. In most recordings, spike trains are obtained by using a high-pass filter with the cutoff frequency about $300-500$ Hz, while LFPs are obtained by using a low-pass filter with the cutoff frequency about 300 Hz [77]. Spike train represent the single-unit neural activity with a fine temporal resolution. However, its stochastic properties induce considerable variability, especially when the stimulation amplitude is small, that is, same stimuli can not elicit the same firing pattern in repeated trails. The experimenters typically address this problem through multiple trial repetitions and averaging the neural response in order to dimmish the variability. However, in BMI decoding the kinematic/stimulation pattern has to be instantaneous from single trial. Therefore, trail averaging is impossible in this context.

In contrast, LFPs reflect the average synaptic input to a region near the electrode [16], which limits specification but provides robustness for characterizing the modulation induced by stimuli, even at low amplitudes. Furthermore, it is known that LFPs provide the information of population neural activity, which can not be completed measured only with spike trains from a limited subset of the total neurons in a region. Moreover,

the population activity state has been shown to be associated with fluctuations in spike trains [16].

Therefore, an appropriate combination of LFPs and spike trains in decoding models make it possible to more accurately decode the stimuli or behaviors from the neural response. For example, the decoder can coordinate LFP or spike patterns to tag particularly salient events or extract different stimulus features characterized by different type signals.

However, different signal properties and formats between LFPs and spike trains make merging in the same model difficult. First, the signal formats are totally different between the two representations. A spike train that is a set of ordered spike timings, which can be interpreted as a realization of a point process [41, 93], while LFP is a continuous amplitude process. Moreover, the time scale of LFPs is significant longer than spike trains. Therefore, only small number of works with simply assumption of the relationship between two signals have been done for modeling with both spike trains and LFPs as input features [42]. However, the full relationship between LFPs and spike trains is still a subject of some controversy [9, 46, 106]. The model assumptions about the relationship between spike train and LFP limits the its ability.

In this chapter, we propose a tensor-product-kernel-based regressor to decode the stimulus information from both spike train and LFP as a example of multiscale neural decoding. The tensor product kernel allows modeling different types of signals individually and merge their information in the feature space defined by the tensor product of the individual kernels for each type signal [87] with no assumption between two data source.

Spike trains characterize stimulus information with a fine time resolution but a large variability. We models spike trains by using the Schoenberg kernel defined in the spike timing space introduction in Chapter 2, which decreases the computation time in contrast with the kernels based on the conventional rate representation and avoids

sparse high-dimensional vectors corresponding to binned spike trains. In contrast, LFPs, as continuous processes, are modeled by the Schoenberg kernel defined in the continuous space on the time structure (nonlinear correlation), which is able to enhance the robustness of stimulus estimation. The tensor product of the two kernels map data into a joint kernel feature space, where the kernel least mean square (KLMS) algorithm estimates the nonlinearly mapping from the neural response to the stimuli as necessary in somatosensory stimulation studies.

In addition, controlling neural activity with adaptive inverse control is also extend to the multiscale neural activity using the tensor product kernel based decoding methodology. Multiscale neural activity provide more complete information of the optimal stimulation pattern with a more robust way, which will also enhance the robustness and accuracy of the control performance.

3.1 Tensor Product Kernel for Multi-Scale Neural Activity

We utilize distinctive kernels to map the spike train $\mathbf{s}_i$ and LFP $\mathbf{x}_i$ into feature functions $\varphi(\mathbf{s}_i)$ and $\psi(\mathbf{x}_i)$ in two RKHSs defined by $\kappa_x(\mathbf{x}_i, \mathbf{x}_j)$ and $\kappa_s(\mathbf{s}_i, \mathbf{s}_j)$, respectively. In order to merge the information from LFPs and spike trains, the tensor product kernels κ_s and κ_x incorporate two individual RHKSs $\chi_s \times \chi_s$ and $\chi_x \times \chi_x$ into a joint kernel defined feature space $(\chi_s \times \chi_s) \times (\chi_x \times \chi_x)$. The tensor product kernel is defined by

$$\begin{aligned} \kappa(\mathbf{x}_i, \mathbf{s}_i, \mathbf{x}_j, \mathbf{s}_j) &= (\kappa_s \otimes \kappa_x)(\mathbf{x}_i, \mathbf{s}_i, \mathbf{x}_j, \mathbf{s}_j) \\ &= \kappa_x(\mathbf{x}_i, \mathbf{x}_j)\kappa_s(\mathbf{s}_i, \mathbf{s}_j) \end{aligned} \tag{3–1}$$

This construction has several advantages that are listed here

1. if both κ_s and κ_x are positive definite kernels, then their tensor product $\kappa = \kappa_s \otimes \kappa_x$ is also a positive definite kernel [87].
2. Since subsequent processing and estimation are conducted in the kernel space, there are no constrains on the format of the input signals and their corresponding kernel functions.

3. The kernel imposes no assumption of the relationship between two inputs.

Those advantages solve the issue that two signal have different signal formats, time scales, and spatial scales. The kernels for spike trains and LFPs can be selected individually based on their own properties. For spike trains that can be interpreted as observations of a point process, the Schoenberg kernel defined in the spike timing space is applied, which we investigated in Chapter 2. LFPs are a continuous amplitude stochastic process. Therefore, the standard vector-based kernels are applicable for LFP, which will be discussed in the following.

1. **Kernel for spike trains**

   This section utilizes the Schoenberg kernel on spike trains introduction in Chapter 2, which interprets a spike train as a realization of an underlying point process and defines an injective mapping based on a strictly positive definite kernel between the conditional intensity functions of two point processes defined by [66].

$$\kappa_s(s_i, s_j) = \exp(-\frac{\int_{\mathcal{T}_s}(\hat{\lambda}_{s_i}(t) - \hat{\lambda}_{s_j}(t))^2 \mathrm{d}t}{\sigma_s^2}), \tag{3–2}$$

   where σ is the kernel size. The estimated intensity function is obtained by simply convolving $\mathrm{s}(t)$ with the smoothing kernel $\mathrm{g}(t)$, yielding

$$\hat{\lambda}_s(t) = \sum_{m=1}^{M} \mathrm{g}(t - t_m), \{t_m \in \mathcal{T}_s : m = 1, \dots, M\}, \tag{3–3}$$

   In order to decrease the kernel computation complexity, the rectangular function $\mathrm{g}(t) = \frac{1}{T_s}(\mathrm{U}(t) - \mathrm{U}(t - T_s))$ ($T_s \gg \text{the interspike interval}$) is used here, where $\mathrm{U}(t)$ is a Heaviside function.

   Individual LFP or spike train channels are a relatively poor decoders of stimulus or behavior on a single-trial basis. Therefore, we defined the kernel for multi-channel

spike trains as follows [85]

$$\kappa_s(\mathbf{s}_i(t), \mathbf{s}_j(t)) = \sum_{n=1}^{N} \kappa_s(s_i^n(t), s_j^n(t)) \tag{3–4}$$

Where N is the number of channels of spike trains.

2. **Kernel for LFP**

In contrast with spike trains, the feature selectivity of the LFP is broader [46]. In time domain, the feature of spike train can be obtained by sliding the window, which describe the temporal shape information of LFP. In frequency domain, spectral power and phase in different frequency band are also known as the informative feature for decoding. In our work, we only select the standard feature of LFP in time domain, as an example, since our work is more focused on providing the machine learning framework for multiscale neural decoding instead of the feature selection. The feature selection of LFP has also attracted massive attention in the BMI area, but it is not the main point of our work.

In the time domain, we can simply treat LFPs as a time series, and apply the Schoenberg kernel to continuous time signals. The Schoenberg kernel defined in continuous space t maps the correlation time structure of the LFP $x(t)$ into a function in RKHS,

$$\begin{aligned}
\kappa_x(x_i(t), x_j(t)) &= \exp(-\frac{\|x_i(t) - x_j(t)\|^2}{\sigma_x^2}) \\
&= \exp(-\frac{\int_{\mathcal{T}_x}(x_i(t) - x_j(t))^2 \mathrm{d}t}{\sigma_x^2}) \\
&= \exp(-\frac{\int_{\mathcal{T}_x} x_i(t)x_i(t) + x_j(t)x_j(t) - 2x_i(t)x_j(t)\mathrm{d}t}{\sigma_x^2})
\end{aligned} \tag{3–5}$$

where $\mathcal{T}_x = [0\ T_x]$.

Similarly to spike trains, the kernel for multi-channel LFPs is defined by

$$\kappa_x(\mathbf{x}_i(t), \mathbf{x}_j(t)) = \sum_{n=1}^{N} \kappa_x(x_i^n(t), x_j^n(t)) \tag{3–6}$$

Where N is the number of channels of LFPs. The time scale of the analysis for LFPs and spikes is very important and needs to be defined by the characteristic of each signal as will be explained below.

3.2 Sensory Stimulation Experiment

3.2.1 Experiment Motivation and Framework

Most Brain Machine Interface (BMI) systems that performs complex limb movements rely on visual feedback alone to provide subjects the position of the device in the external world [17, 18, 26, 89, 97, 100]. However, somatosensory feedback remains underdeveloped in BMI, which is important for motor and sensory integration during movement execution, such as proprioceptive and tactile feedback about limb state during interaction with external objects [40, 58]. A number of early experiments have shown that spatiotemporally patterned micro-stimulation delivered to somatosensory cortex can be used to guide the direction of reaching movements [31, 62, 64]. In order to effectively apply the artificial sensory feedback in BMI, it is essential to find out how to use the micro-stimulation to replicate the spatiotemporal pattern in somatosensory cortex that are necessary to emulate sensory stimulation. Somatosensory information originates in the periphery and ascends through VPL thalamus on its way to the somatosensory cortices. Since this information is transmitted via neuron responses in VPL, we expect that a suitably fine electrode array could be used to selectively stimulate a local group of VPL neurons and transmit similar information into somatosensory cortex. Electrophysiological experiments [32] suggest that the rostral portion of the rat VPL nucleus carries a large amount of proprioceptive information. Caudal to this region is a zone where the cutaneous receptive fields are focal with a fine topographic map of the fore and hind limbs. Since the body map in the VPL is arranged with such remarkable somatotopic precision that neurons responsible for relaying sensory information up to the S1 cortex can be easily located and stimulated with implanted

electrodes. Thus, suitably precise spatiotemporal microstimulus pattern of these local areas are able to reproduce S1 cortical responses induced by the natural stimulus.

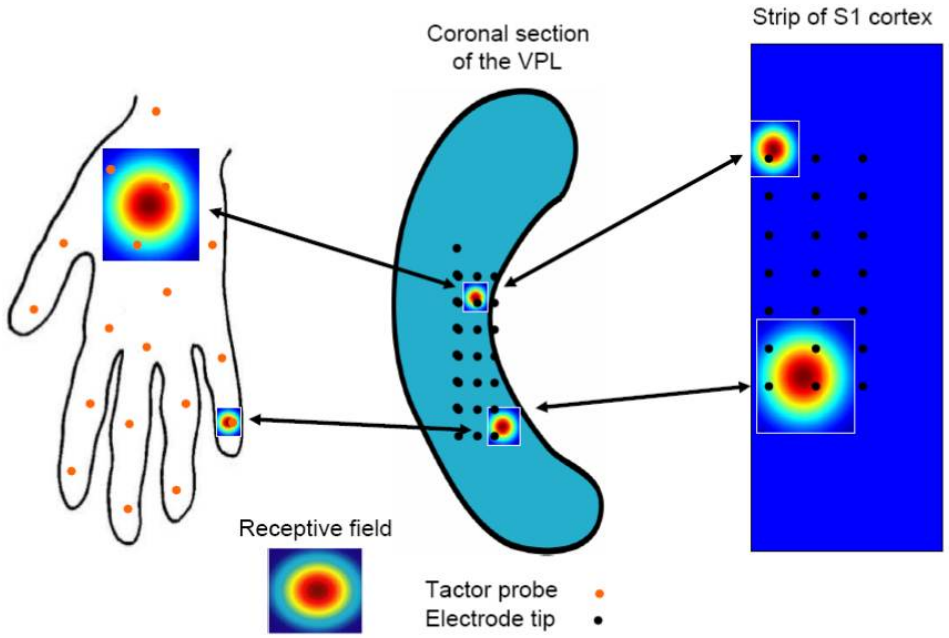


Figure 3-1. Neural elements in tactile stimulation experiments

Figure 3-1 depicts the neural elements that will be involved in our experiments. To the left is our rat's hand with some representative cutaneous receptive fields shown as well as positions where our tactor will touch the hand. When the tactor touches a particular "receptive field" location on the hand, VPL thalamus will receive this information and send it on to the S1 cortex. In order to emulate "natural touch" with microcirculation, we address in this section the following computational problem: how to generate optimal spatiotemporal microstimulation patterns in ventral posterolateral nucleus (VPL) of thalamus so that the elicited spike trains from somatosensory regions (S1) is able convey the natural sensation to the animal. We hypothesize that if the firing patterns in S1 produced by tactile (tactor) and the artificial (electrical) stimulation are similar the animal should behave in the same way.

In our work, we use kernel-based adaptive inverse control diagram to solve this problem, which is propose in Chapter 2. Before implementing the whole diagram, the decoding method should be tested first, in order to guarantee the it is able to reconstruct both tactile stimulation and micro-stimulation with a reasonable good accuracy. In the Chapter 2, we test the spike decoder on both tactile data set and micro-stimulation data set. Because of the variability of spike trains, fluctuation in decoder output occur. Since our spike train processing extracted only partial information, the stimulation output channel can not be discriminated well. Therefore, we'd like to find out whether the multi-scale neural decoder is able to enhance the decoding performance. In the following part we first test the multi-scale decoder in both tactile and micro-stimulation data sets, which we mentioned in Chapter 2.

3.2.2 Time Scale Estimation

The tensor product kernel (3–1) allows modeling the sample from different sources individually. Therefore, the time scales of LFPs and spike trains can be specified based on their own properties. In order to find reasonable time scales, we estimate the autocorrelation coefficients of LFPs and spike trains, which indicate the response duration induced by the stimulation. For this purpose, spike trains are binned with binsize 1ms. The local field potential are also resampled with sampling rate 1000Hz. The autocorrelation coefficients of each signal average over channels are calculated by

$$\hat{\rho}_h = \frac{\sum_{t=h+1}^{T}(y_t - \bar{y})(y_{t-h} - \bar{y})}{\sum_{t=1}^{T}(y_t - \bar{y})^2}. \tag{3–7}$$

Its 90% confidence bounds of the hypothesis that the autocorrelation coefficient is effectively zero are approximately estimated by $\pm 2SE\rho$, where

$$SE\rho = \sqrt{(1 + 2\sum_{i=1}^{h-1}\rho_i^2)/N}. \tag{3–8}$$

The average confidence bounds for LFP and spike trains are $[-0.032\ 0.032]$ and $[-0.031\ 0.031]$, respectively. The autocorrelation coefficients of LFP fall into the

confidence interval after 20ms, while the autocorrelation coefficients of spike trains die out after 9 ms, as shown in Figure 3-2. Therefore, the decoder inputs built from spike train and LFP are obtained by sliding the window with window size T_s =9ms for spike train and T_x =20 ms for LFP.

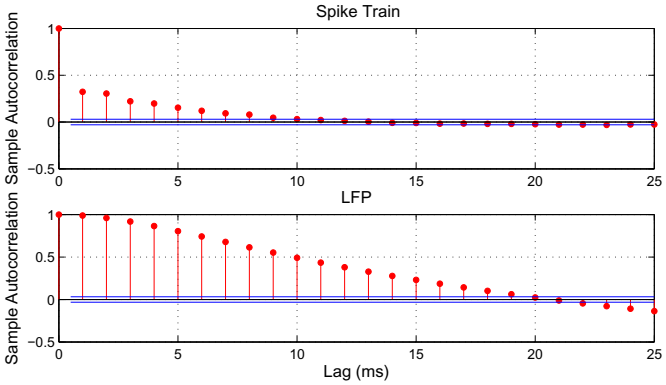


Figure 3-2. Autocorrelation of LPFs and spike trains

3.2.3 Decoding Results

We now present the results of decoding the tactile stimulus waveform and micro-stimulation using KLMS operating on the tensor product kernel. For comparison, we also apply the kernel-based decoder on only one type of signals to find out what is the performance enhancement gained by using multi-scale signals. Time discretization is 5 ms. The learning rates for each decoder are determined by the best results of test data after scanning the parameters. The kernel size σ_s and σ_x are determined by the average distance in RKHS of each pair of training samples. The normalized mean square error (NMSE) between the estimated stimulation (**y**) and the desired stimulation (**d**) is utilized as an accuracy criteria.

3.2.3.1 Results of tactile stimulation

NMSEs of tactile stimulation are obtained across 8 trials data sets. For each trial, we use 20s data to train the decoders and compute an independent test error on the

Table 3-1. Comparison among neural decoders.

Property \ input	LFP&spike	LFP	spike
NMSE (mean/STD)	0.48/0.05	0.55/0.03	0.63/0.11

remaining 2.5s data. The results are shown in Table 1, where we can observe that the LFP&spike decoder out-performs both the LFP decoder and the spike decoder.

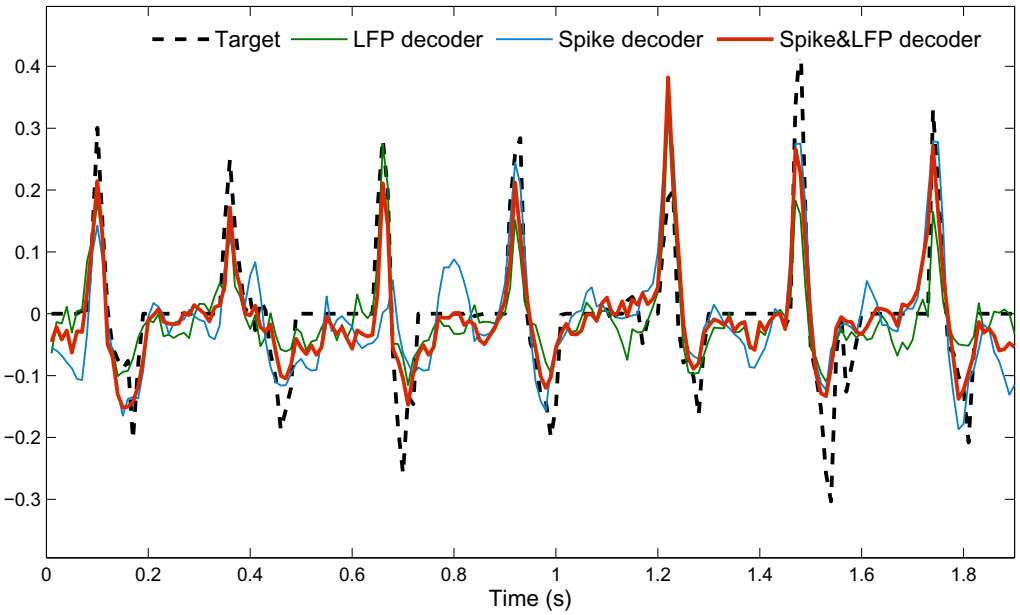


Figure 3-3. Results of the LFP&spike decoder of the first stimulation trial.

In order to illustrate the details of the decoding performance, the test results of the first trial are illustrated in Figure 3-3. It is observed that the output of the spike decoder fluctuates a lot and misses some pulses, i.e., around 0.65 s, because of the sparsity and variability of the spike train. In contrast, the output estimated by LFP is smooth, because the LFP reflects the sum of all local currents on the surface of the electrodes, which causes the robustness but is a imitation to the specificity of the technique. The LFP&spike decoder preforms better than the LFP decoder by gaining the precise pulse timing information from spike trains. The learning curves are also estimated by calculating the testing NMSE after the model parameter updated after each new sample

entered. Three learning curves of spike decoder, LFP decoder and LFP&spike decoder are shown in Figure 3-4, respectively.

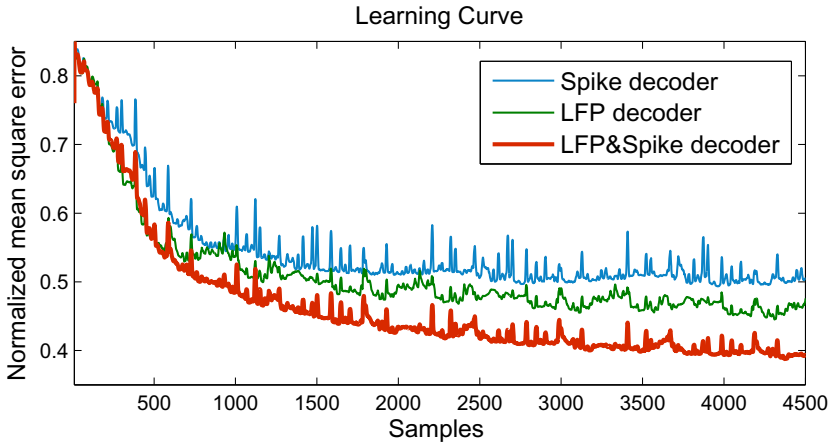


Figure 3-4. Learning curves (NMSE of the test set) of the LFP&spike decoder of the first stimulation trial.

3.2.3.2 Results of micro-stimulation

We also implement the decoder to reconstruct the micro-stimulation pattern. First, we map the 8 different stimulation locations to 8 channels. We dismiss the shape of each stimulus, since the time scale the stimulus width is only 200μs. The desired stimulation pattern of each channel is represented by a sparse time series of the stimulation amplitude with sampling rate of 200Hz. We implement tensor product kernel with KLMS for MIMO system that is proposed in Chapter 2 on the micro stimulation data. NMSEs are obtained with ten subsequence decoding results. We use 120s data to train the decoders and compute an independent test error on the remaining 20s data. The Spike&LFP decoder out-performs both the LFP decoder and the spike decoder. The comparison of results are shown in Figure 3-5, which indicates that for Spike&LFP decoder is able to obtain the best performance of all the stimulation channels. Especially for channel 2, 4, 6, and 7, Spike&LFP decoder performs significantly better than both the LFP decoder and the spike decoder. A qualitative comparison of an excerpt is shown in Figure 3-6, which explains some details about the decoder performance. First, only channel, 4, 5, 6, 7, and 8 stimulations can not be decoded from LFPs only

models at all, since the fine time information is average out in LFPs. For spike trains only models, the stimulation is not well discriminated from different output channels, which cause the false-stimuli present in spike decoder output. However, the combination of spike trains and LFPs models provides more complete stimulation information, which contributes to a better discrimination of stimulation pattern cross channels and allows the model to capture the precise stimulation timing.

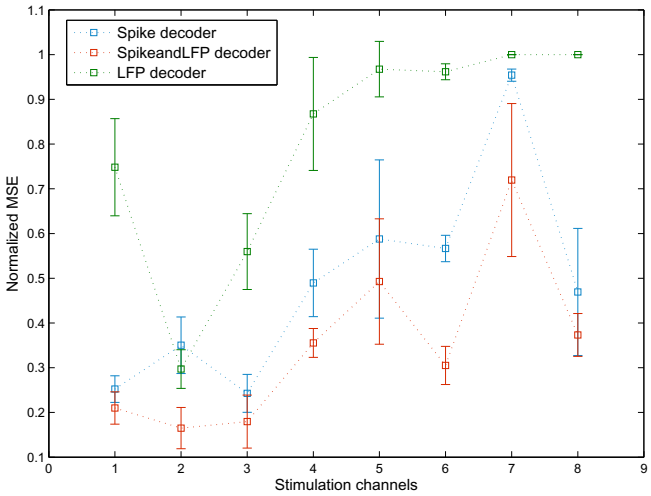
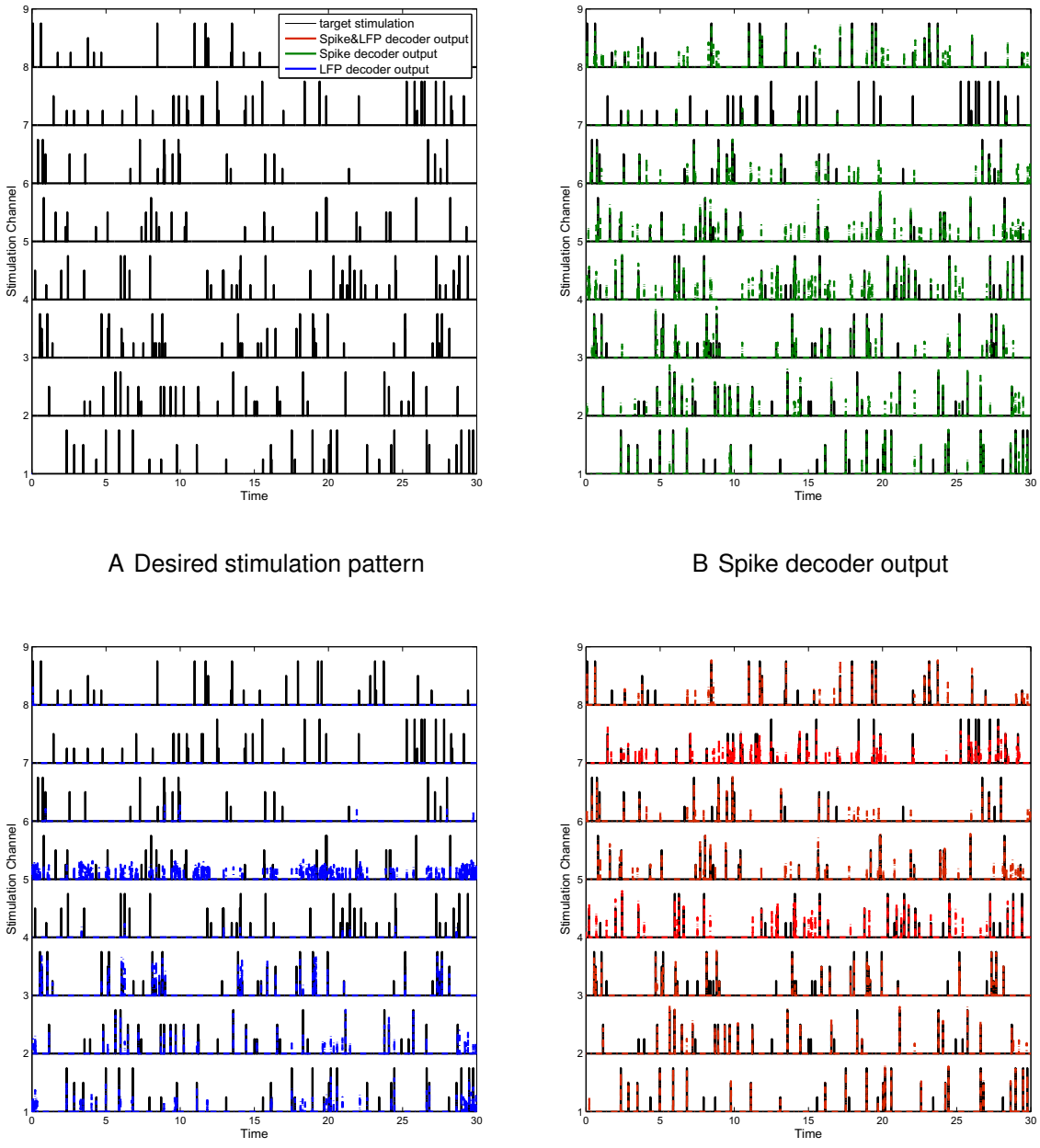


Figure 3-5. Comparison among spike decoder, LFP decoder and spikeandLFP decoder.

3.3 Open Loop Adaptive Inverse Control

In order to emulate ”natural touch” with micro-stimulation, we used micro-stimulation imprinted in VPL to control the neural activity in S1. However, the variability of neural signals and the neural system plasticity induced by injury, environment changes, etc present a challenge of precisely controlling the neural response to mimic the target spatiotemporal firing pattern induced by the natural stimulation. In Chapter 2, we illustrate the ability of adaptive inverse control diagram to track the neural plasticity and the promising results of multi-scale neural decoder indicates that the decoder is robust for interfacing with the real animal brain. Considering these unique challenges,



A Desired stimulation pattern

B Spike decoder output

C LFP decoder output

D SpikeandLFP decoder output

Figure 3-6. Results of reconstructing micro-stimulation.

the kernel based adaptive inverse control diagram and tensor product kernel based multi-scale neural decoding methodology is combined to address this problem.

In our current step, considering the experimental setting, only open loop adaptive inverse control is implemented in the rat experiment. First, the inverse controller $C(z)$ is pre-trained with 300s data. In each trial, 60s data recorded during the tactile stimulation is used as the target pattern. Given the target neural response as the input, the trained controlled generate a sequence of multiple channel stimulation. Since the generated stimulation sequence is sparse but continuous signal, it does not meet the restrictions of bipolar stimulation that can be used as micro-stimulation. Therefore, some post-processing of stimulation is implemented before interfacing with the microstimulator as follows:

1. The minimal interval between two stimuli 10ms is suggested by the experimental setting. The mean shift algorithm is used to locate the local maxima of a subsequence of stimulation (10ms) for each single channel. The time location and corresponding maximum amplitude is used to set the time and amplitude of stimuli.
2. At one time point, only one pulse a cross channels can be stimulated. Therefore, at each time point, only the maximum value cross channels is select for stimulation. The value at other channels are set to zero.
3. The constrain of the maximum/minimum stimulation amplitude is [8μA 30μA], which has been suggested as the effective and safe amplitude range in previous experiments.

To assess the performance of the open-loop control, the generated multi-channel microstimulation sequence is applied immediately following computation. The microstimulation response set is recorded and then compared with the actual natural response set. A graphical comparison of an excerpt is shown in Figure 3-7. It is observed that the multi-unit neural response in the target set and the one generate by the micro-stimulation have the similar spatiotemporal firing patterns. Due to the short response time of the micro-stimulation, a sequence of micro-stimuli is used to emulate

long term bursting of spike train in the target firing pattern, whose sensory influence is unclear at this moment.

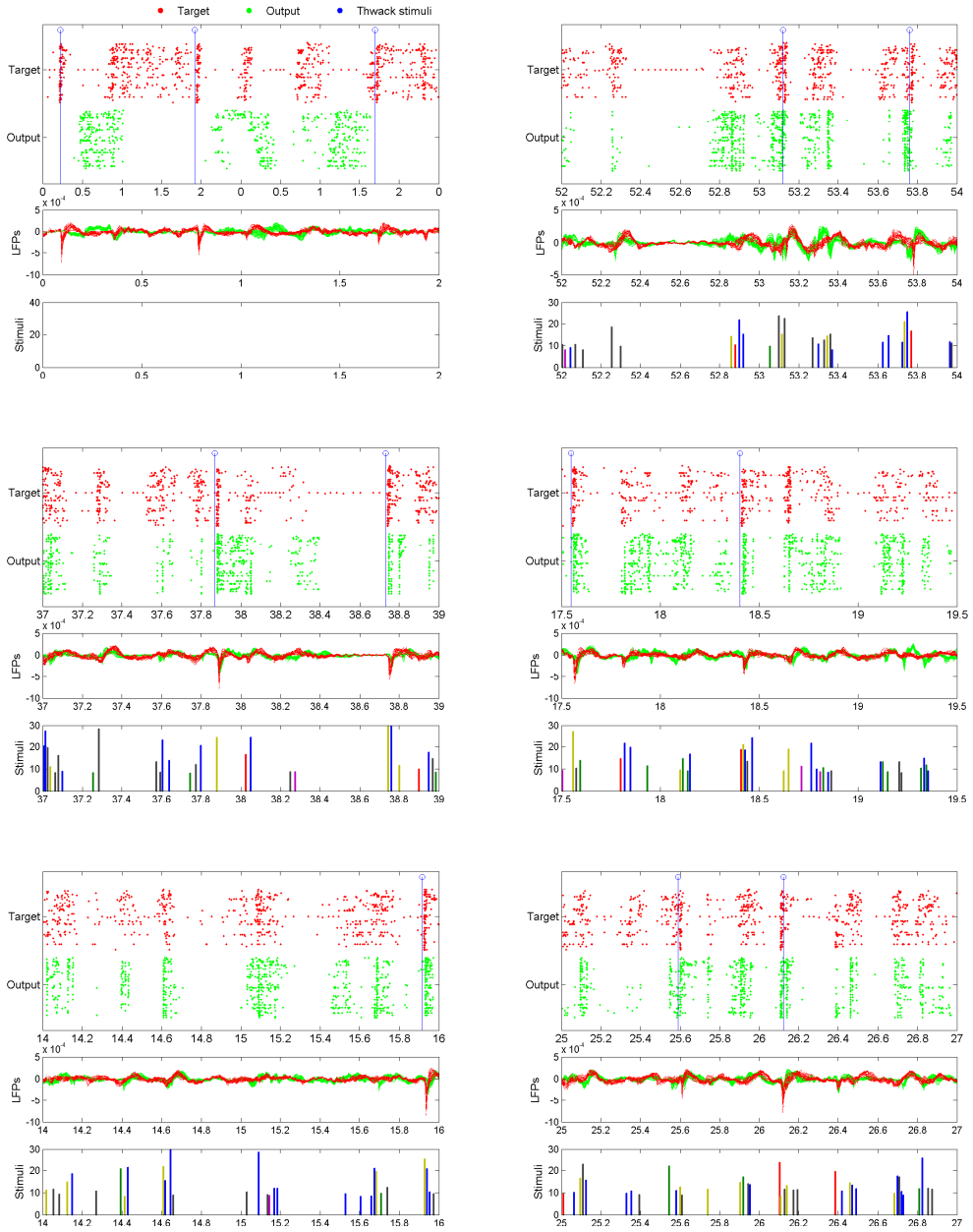


Figure 3-7. Qualitative comparison of excerpts between target and trained system output.

For quantitative analysis, the correlation coefficient for each channel is calculated as shown in Figure 3-8. To simplify the estimation of correlation of spike train, we bin

the data with bin size 10 ms. It is observed that the maximum correlation coefficient lies around 0 lag for most channels. Both the model inaccuracy and the jitters in neural signal cause the blur over time.

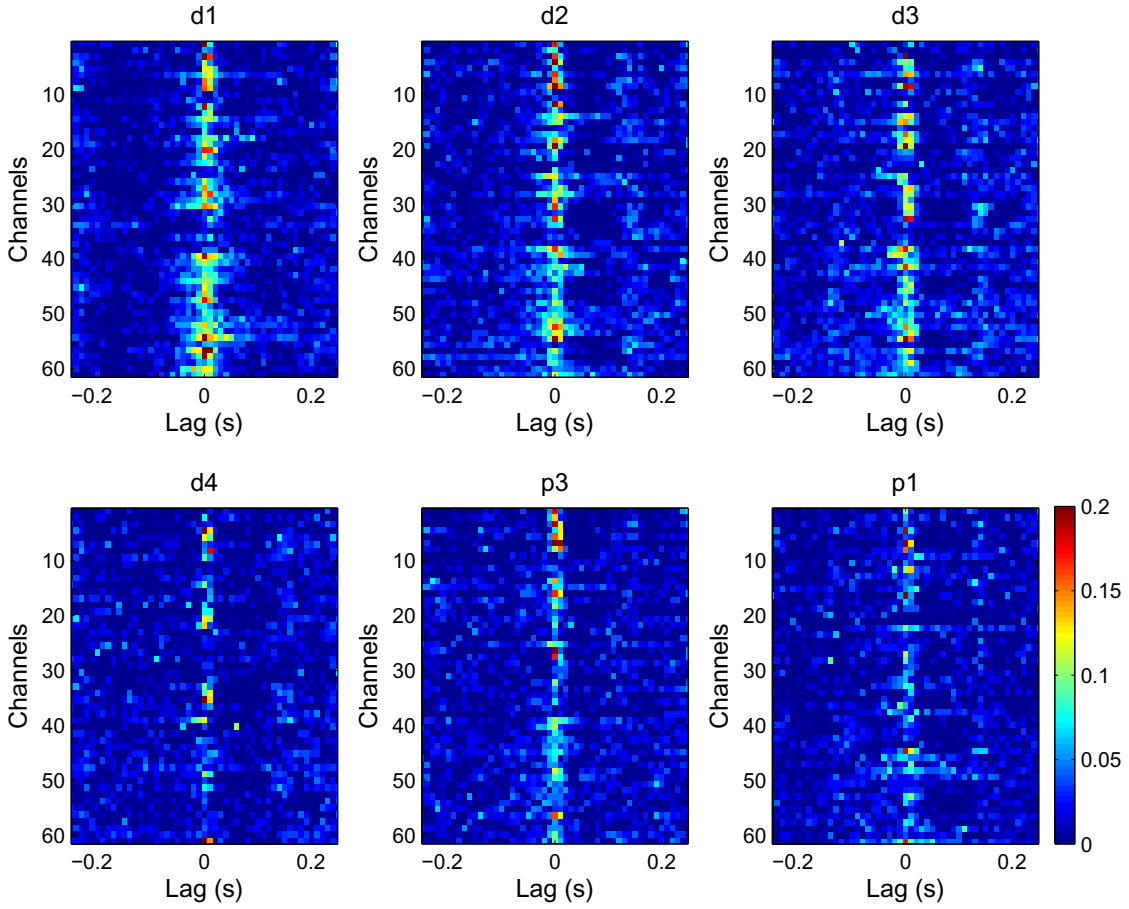


Figure 3-8. Correlations between target spike trains and neural system output of each channel for each tactile stimulation location (6 digits: d1 d2 d3 d4 p3 and p1).

The promising results of open loop control show the effectiveness of controller modeled by the multi-scale neural decoding methodology. In future work, the whole control diagram will be implemented to solve this emulation of natural touch. We expect that the online feedback is able to enhance the control performance.

CHAPTER 4
FUNCTIONAL CONNECTIVITY ANALYSIS AND MODELING

Neurons selectively coordinate with other neurons distributed in the neighboring cortical area, which contributes to the movement [61]. However, it is very challenging to quantify the nature of this local functional coordination. According to the literature, neural assemblies provide a conceptual framework for the integration of distributed and individual neural firings, which are defined as distributed local networks of neurons transiently linked by active dynamic couplings [56]. The general hypothesis is that the emergence of specific neuronal assemblies underly cognitive actions [61, 99]. There are some difficult issues to characterize such neural assemblies induced by the spatially specific and transient nature of the inter neuron communication.

In our work, we focus on decoding the intention of movement or the sensory stimulation, using multi electrode neural data from cortical cortex synchronized with kinematic/stimulation variables measurements. From a statistical modeling perspective, the complete information to solve this problem resides in the joint probability density function of the multivariate neural data and of the multivariate kinematic variables, but this is never done due to the high dimension of the joint distributions. Even when we develop models that estimate the conditional probability of the kinematics/stimulation given the neural data, we do not use the joint information expressed in the multichannel neural data for the same reason. Instead, we model the conditional dependence only with respect to the history of a vector of neural spike trains. Since each neural channel is nothing but the marginal density of the joint probability of multivariate neural data, this procedure is only a reasonable solution when the neurons spike independently. In our work, we aim to quantify and analyze pairwise dependencies between cortical neurons involved in a given behavioral task using the information from neural data conditioned on the kinematic/stimulaiton information. Pairwise dependence does not fully quantify the joint distribution, but the information contained in the combined pairs better describes

the dynamics in an assembly. In this work the dependencies that occur transiently in time between pairs of neurons will be named interactions, and their spatiotemporal patterns are interpreted as signatures of functional connectivity. Our hypothesis is that given a kinematic state or stimulation pattern defined by the experiment, specific neural assemblies are transiently activated, which can be quantified by a significant enhancement of the dependencies between pairs of neurons, i.e. their functional connectivity.

The functional connectivity is commonly quantified by correlation, coherence, and synchrony [1, 13, 35, 36, 72]. Correlation and coherence capture linear relations in time and frequency domain respectively, while (statistical) synchrony measures the simultaneous firing of neurons in excess of coincidental firings. However, in general, the interaction between neurons can be highly non-linear and not exclusively in the form of synchrony [27]. The general form of interaction can be quantified by statistical means, namely by dependence measures, which estimate arbitrary relationships between two random variables. For examples see [10, 45, 75].

Neural assemblies have a transient, dynamic existence that spans the time scale of behavior. The activation of a neural assembly is long enough for neural activity to propagate through the assembly. A propagation that necessarily involves cycles of reciprocal spike exchanges with transmission delays lasts tens of milliseconds. In order to quantify and study this interaction during movements, we need to focus on the temporal dynamics of neural networks in the hundreds to thousands of millisecond range. Due to the sparse firing of cortical neurons, in this chapter we first concentrate on comparing statistical methods that support small sample estimators of dependence. Four methods, *mean square contingence* (MSC), *mutual information* (MI), *phase synchronization* (PhS), and *cross correlation* (CC) were tested on synthetic spike trains generated by a network of leaky integrate-and-fire neurons [38]. We select two baseline methods with different properties: CC measures the linear relation between two neuron

activities in time, while PhS measures the pairwise synchrony of neurons. In contrast, MI and MSC quantify non-linear dependence of two nominal variables based on two major independence tests: Chi-square test and G-test of independence. MI and MSC calculate the Kullback-Leibler divergence and χ^2 divergence of the joint probability distribution with respect to the product of the two marginal probability distributions, respectively. Although both MI and MSC are widely used in applied statistics measurements, their capability in estimating functional connectivity with small sample size has not been fully investigated. See [2] for the use of MI in neural assemblies. We use statistical power and dispersion criteria to evaluate the performance of these four methods as a function of the sample size.

Furthermore, in Section 4.2 the dependence measure is applied to monkey cortical neural activity recorded during a food reaching task with three well defined kinematic states. The estimated functional connectivity describes the assemblies associated with the time varying kinematic states. The nodes (neurons) and edges (the functional connections between pairs of neurons) in an assembly graph represent different aspects of the time spatial neuronal interactions. Specifically, we visualize the local assembly graph within four cortical areas during reaching tasks.Moreover, we estimate the activation degree of a specific assembly based on the number of activated edges in the graph. The activation degree of a state-related assembly reaches a peak when the corresponding kinematic state begins, which also reveals that the network of interactions among the neurons is the key factor for the operation of a specific behavior. The evolution of this complex local functional connectivity over the cortical space and over time can be readily visualized in a movie (http://cnel.ufl.edu/research/linli.php) or in static graphs as we demonstrate in the results section.

In addition, the inference of the functional connectivity dynamics with cognitive behavior, disease, stimulation has also been explored in [7, 15, 22, 25, 34, 82], which all indicates the temporal functional connectivity pattern is a good way to describe the

spatiotemporal neural activity pattern. After investigating how to quantify the temporal functional connectivity pattern, our idea is whether we can extract more information of the stimuli or kinematic states from the temporal connectivity pattern comparing with the temporal pattern of neural signal with channel independence assumption. However, the functional connectivity patterns are represented by graphs or matrices instead of vectors, which make it challenging to use as input features for neural system modeling. Instead of directly addressing the functional connectivity graph or matrix, most research implements graph theory to quantify the topology of graph properties including node degree, clustering coefficient, betweenness centrality, and etc. [22, 82]. These topology metrics are not complete but abstract feature vector from the functional connectivity graph/matrix, which allows the implementation of the traditional modeling with the vector-based input space [22]. However, the abstract topologies only characterize the importance of a neuron in a network commutation from one certain perspective, but reduces the details of the graph structure, for example how the neurons are functional distributed in the network or what is their direct neighbors. Therefore, in Section 4.3 we not only use the topology vector as the input vector, but also design a kernel for the functional connectivity matrix, which allow us to implement kernel based machine learning algorithm directly on the temporal functional connectivity pattern for decoding with no further pre-feature extraction. The decoding performance using graph topology or functional connectivity matrix with matrix kernel are compared in Section 4.3.

4.1 Functional Connectivity Measures

The existence of massive anatomical connections among cortical neurons does not make them functionally connected all the time. Even in the absence of neural code knowledge, exchange of information across the neuropil can be spotted and quantified externally by the co-occurrence in time of action potentials produced by neurons. In this section, we quantify functional connectivity by the statistical dependence between neural firing rates. First we briefly review different neural functional connectivity measures.

Consider two discrete firing rate time series x_n and y_n, $n = 1, \dots, N$, recorded simultaneously and produced by two neurons. Each dependence measure has particular characteristics and we divide them into two groups: statistical-based and phase-based.

4.1.1 Statistical Measures

Second order measures

The most widely used technique to measure the similarity between two time series of firing rate x and y is the cross-correlation, defined in the time domain as a function of the time lag $\tau = -(N-1), \dots, 0, \dots, N-1$:

$$C_{YX}(\tau) \begin{cases} = \frac{1}{N-\tau} \sum_{n=1}^{N-\tau} \left(\frac{x_{n+\tau}-\bar{x}}{\sigma_x}\right)\left(\frac{y_n-\bar{y}}{\sigma_y}\right) & \tau \geq 0 \\ = C_{YX}(-\tau) & \tau \leq 0 \end{cases} \tag{4–1}$$

The cross correlation is obtained by normalizing the cross covariance and thus ranges from minus one (anti-phase correlation) to one (in-phase correlation), while values close to zero are attained for uncorrelated time series. In time series analysis using a linear generative model with Gaussian inputs, the cross correlation coefficient quantifies totally the dependence between the time series.

As a measure of functional connection between two neurons, the cross correlation coefficient is calculated between two firing rate time series with a certain bin size. In this study we selected the bin size at 100 ms, not because it is the best choice but because it is a reasonable compromise for the application. From multiscale studies on spike trains for BMIs [83] and also from the analysis of function connectivity in neural ensembles [28] it is known that the best approach to functional connectivity is to use multiple time scales to analyze spike trains. However, this methodology is rather expensive computationally and it is incompatible with online studies as the ones we envisage in this work. Therefore, we have to select a single bin size that is a good compromise between specificity of neural firing and sufficient similarity with

behavior. From [83] 100 ms is the time scale that bridges the gap between reasonable correlations with the movements and a reasonable specificity regarding neural firings. Moreover, since the average firing rate of cortical neurons is around 10 Hz, with 100 ms bin size, the average spike number per bin is 1, which yields a suitable sparseness for estimating dependency. Because the neural data we used is collected from neighboring cortical regions and we quantify firing rates with bin size of 100 ms, we can expect that the dependence induced by the functional connection between a pair of neurons usually happens within the 100 ms lag. Therefore, only the correlation with lag zero is calculated.

The correlation of the firing pattern can also be directly calculated in the space of spike train with no binning process. The quantification of correlation defined in the space of spike train call Schreiberg et al. Similarity Measure, which is also based on the functional representation of a spike train $\hat{\lambda}(t)$, which is defined by

$$S_s = \frac{\int \hat{\lambda}(t)\hat{\lambda}\prime(t)dt}{\sqrt{\int \hat{\lambda}(t)\hat{\lambda}\prime(t)dt}\sqrt{\int \hat{\lambda}(t)\hat{\lambda}\prime(t)dt}} \tag{4–2}$$

where the smoothing filter $h(t)$ may for example exponential or Gaussian. Considering the computation efficiency, exponential smooth function is used in this work. This Similarity measure provides an estimation of the reflective correlation coefficient between two point processes. The value S_S is scaled into $[0\ 1]$. if and only if two spike trains $s(t)$ and $s\prime(t)$ are identical, then $S_s = 1$. Through the correlation bypass the limits induced by the binning process, it also introduce the free parameters to the calculation, which need to be specified. Therefore, this similarity measure is suggested to be used in the context requesting fine time resolution.

However, cross correlation only indicates the strength of a linear dependence between two variables, but its value cannot completely characterize their relationship that may appear in higher order statistical moments (skewness, kurtosis, etc.).

Dependence measures

In contrast to cross-correlation, both MI and MSC quantify the statistical dependencies between two random variables using the full statistical information in their joint space, with no assumption about the form of their respective densities and implicitly their generating processes, which can be linear or nonlinear. It is therefore more appropriate when the data generation mechanism is not fully understood, but it is much more difficult to estimate in practice.

Mean square contingency Mean Square Contingency (MSC) is a measure of dependence based on an independence test, and it was first defined by Pearson for two discrete random variables [69]. MSC calculates the normalized χ^2 divergence between the joint probability distribution and the product of the two marginal probability distributions of two firing-rate time series [45].

For a test of independence, an *observation* consists of a pair of values and the null hypothesis is that the occurrences of these values are statistically independent. Each observed probability is allocated to one cell of a two-dimensional array (known as contingency table) according to its values. If there are r rows and c columns in the table, the theoretical probability for a cell given the independent assumption is $E_{i,j} = P_{i+}P_{+j}$, $P_{i+} = \sum_{k=1}^{c} P_{i,k}$ and $P_{+j} = \sum_{k=1}^{r} P_{k,j}$ are the marginal probability, and $P_{i,j}$ is the observed (joint) probability of the cell in row i and column j in contingency table.

The MSC value, also called the Pearson contingency coefficient, denoted by P, is given by

$$P = \left(\frac{\Phi^2}{1 + \Phi^2} \right)^{1/2}, \tag{4–3}$$

where

$$\Phi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(P_{i,j}-E_{i,j})^2}{E_{i,j}} = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{P_{i,j}^2}{P_{i+}P_{+j}} - 1 \tag{4–4}$$

Note that this statistic basically scales the χ^2 statistic to a value between 0 (independence) and 1 (maximum dependence). It has the desirable property of scale invariance. That is, the value of Pearson's contingency coefficient does not change as long as the probabilities remain constant. Unfortunately, the estimation is biased when the sample size is small. William's correction is used to decrease the inaccuracy [94] and the bias corrected expression P^{∞} is

$$P^{\infty} = \left(\frac{\Phi^2}{q+\Phi^2}\right)^{1/2}, \tag{4–5}$$

where

$$q = 1 + \frac{\sum_{i=1}^{r}\frac{1}{P_{i+}}\sum_{j=1}^{c}\frac{1}{P_{+j}}}{6N(r-1)(c-1)}, \tag{4–6}$$

where N is the total sample size.

Mutual information In contrast with MSC, MI quantifies the KL-divergence between the joint probability distribution and the product of the two marginal probability distributions, which is used to determine the synaptic connection structure of neuronal networks [107]. According to the contingency table, the Shannon entropy of the row (X) and column (Y) vector are defined respectively as

$$\hat{H}(X) = -\sum_{i=1}^{r} P_{i+}\ln P_{i+}, \quad \hat{H}(Y) = -\sum_{i=1}^{c} P_{+j}\ln P_{+j}, \tag{4–7}$$

where P_{i+} and P_{+j} are the marginal probability and P_{ij} is the observed (joint) probability of the cell in row i and column j in contingency table. The estimation of $H(row)$ and $H(column)$ is heavily biased negatively when the contingency table contains zero entries ($P_{+j} = 0$ or $P_{i+} = 0$) although the true probability is non-zero, which is likely to occur with a small sample size compared to the number of entries of the table. The first order bias corrected entropy H^{∞} [81], is then defined as:

$$H^{\infty}(X) = \hat{H}(X) + \frac{b_X - 1}{2N}, \tag{4–8}$$

$$H^{\infty}(Y) = \hat{H}(Y) + \frac{b_Y - 1}{2N}, \tag{4–9}$$

where b_X and b_Y are the number of states for which $P_{i+} \neq 0$ and $P_{+j} \neq 0$, respectively. The joint entropy between X and Y is defined as

$$H(XY) = -\sum_{i,j} P_{i,j} \ln P_{i,j}, \tag{4–10}$$

The mutual information between X and Y is defined as

$$\hat{MI}(X, Y) = \hat{H}(X) + \hat{H}(Y) - \hat{H}(X, Y) \tag{4–11}$$

and its bias corrected expression MI^{∞} is

$$MI^{\infty}(X, Y) = \hat{MI}(X, Y) + \frac{b_X + b_Y - b_{XY} - 1}{2N}, \tag{4–12}$$

where b_{XY} are the number of cells in contingency table greater than zero. In the following, the mutual information is bias corrected and also normalized by dividing it by $ln(N)$.

4.1.2 Phase Synchronization Measures

Phase synchronization (PhS) is an ubiquitous phenomenon in many physical oscillatory systems, which also occurs in neural structures. However, the methodology is based on deterministic dynamical system principles that may not apply to neural structures due to their intrinsic stochastic nature. Nevertheless, phase synchronization has been widely applied to neural systems [70]. In this work, we apply PhS to the time series of firing rate. We view the time series of firing rate as a continuous signal $x(t) = x_n, n\Delta \leq t < (n+1)\Delta$, where $\Delta = 100ms$ represents the bin size. This long bin width is sufficiently long to smooth short term fluctuations produced by neural stochasticity. Phase synchronization indicates the following phase locking condition applied for any time t, $\varphi(t) = |n\phi_x(t) - m\phi_y(t)| \leq constant$, where $\phi_x(t)$ and $\phi_y(t)$ represent the phase of the signal recorded from the neuron x and y, respectively [70]. In the following computations we take the terms with $|m| = |n| = 1$. Basically, phase synchronization analysis proceeds into two steps: (i) estimation of the instantaneous phases and (ii) quantification of the phase locking.

Estimation of instantaneous phases In this study, the phase is extracted via Hilbert transform, although there are many other methods [53] (for a survey of methods, we refer to [70]). The analytic signal $\xi(t)$ of the univariate measurement is a complex function of continuous time $x(t)$ defined as

$$\xi(t) = x(t) + ix_h(t) = a_\xi(t)e^{j\phi_\xi(t)}, \tag{4–13}$$

where the function $x_h(t)$ is the Hilbert transform of $x(t)$.

$$x_h(t) = \frac{1}{\pi}\mathbf{P}.\mathbf{V}.\int_{-\infty}^{+\infty}\frac{x(t)}{t-\tau}d\tau. \tag{4–14}$$

P.V. indicates that the integral is taken in the sense of Cauchy principal value. $a_\xi(t)$ and $\phi_\xi(t)$ are the instantaneous amplitude and phase of the analytic signal $\xi(t)$ of $x(t)$. The instantaneous phase $\phi_x(t)$ is taken equal to $\phi_\xi(t)$, while the phase $\phi_y(t)$ is estimated from $y(t)$ following the same procedure for phi(x).

Phase-locking quantification For neural signals, the phase locking condition can be better understood in a statistical sense, by estimating in the distribution of cyclic relative phase $\varphi\prime(t) = \varphi(t)\ mod\ 2\pi$ [79]. For independent time series $x(t)$ and $y(t)$, the distribution of the relative phase $\varphi\prime(t)$ is uniform within a given time window. The detection of phase locking involves quantifying how far from uniform is the relative phase distribution. Several measures have been proposed for this purpose [43, 96]. The one we use is [43]

$$\rho = \| \frac{1}{N} \sum_{j=1}^{N} e^{i\varphi\prime(t_j)} \|, \tag{4–15}$$

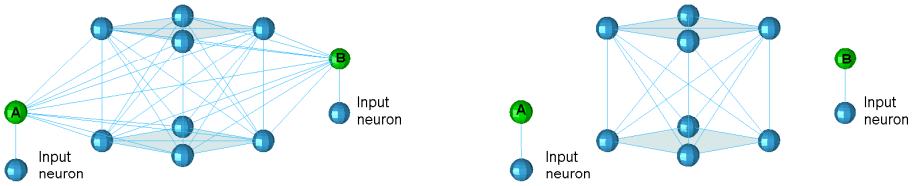
where N is the number of sample in a window and ρ is the mean phase coherence of the angular distribution [59]. By construction, it is bounded by 0 (no synchronization) and 1 (perfect synchronization).

4.1.3 Performance Comparison over Measures

4.1.3.1 Simulated data

We test the capability of the four methods in estimating functional connectivity as a function of the window size of observations. The experiment is performed in a neural Circuit SIMulator (CSIM) [38], using a network of 10 neurons organized as a cell assembly. Given the parameters of the simulation, this number is sufficient to generalize the conclusions of their performance for larger network sizes [80]. Two neurons, A and B from the cell assembly are selected for measurement. Neuron A and neuron B are excited by independent spike trains. The spike trains are recorded in two states: the cell assembly is either fully connected (CON state as shown in Figure 4-1A) or disconnected

(direct and indirect) between neuron A and B (DIS state as shown in Figure 4-1B). The interaction between neuron A and B is estimated based on dependence measures.



A CON state (the cell assembly is fully connected)

B DIS state (the cell assembly is disconnected (direct and indirect) between neuron A and B)

Figure 4-1. The stimulated network of 10 neurons.

All neurons are modeled as Leaky Integrate-and-Fire (LIF) units, since the integrate-and-fire neuron model is one of the most widely used spiking neuron models for analyzing the behavior of neural systems. It provides practical and relatively realistic descriptions of the neuron membrane potential (spike) in terms of the synaptic inputs and the injected current that it receives. Neuron parameters are [52]: membrane time constant 30 ms, absolute refractory period 3 ms (excitatory neurons), threshold 15 mV (for a resting membrane potential assumed to be 0), reset voltage 14.3 mV, constant nonspecific background current 13.5 nA, input resistance 1 MΩ, and input noise 9 nA. The postsynaptic current is modeled as an exponential decay $Wexp(-t/\tau_s)$ with $\tau_s = 3$ ms. The average synaptic weight W is 1. All neurons fire at a reasonable firing rate range (1 Hz to 10 Hz) as shown in Figure 4-2. For analysis, neuronal spike events are binned in non-overlapping windows of 100 ms.

4.1.3.2 Criterion for comparing different measures

To compare the different measures of functional connectivity in terms of their time resolution, the statistical power analysis and the variance analysis are utilized

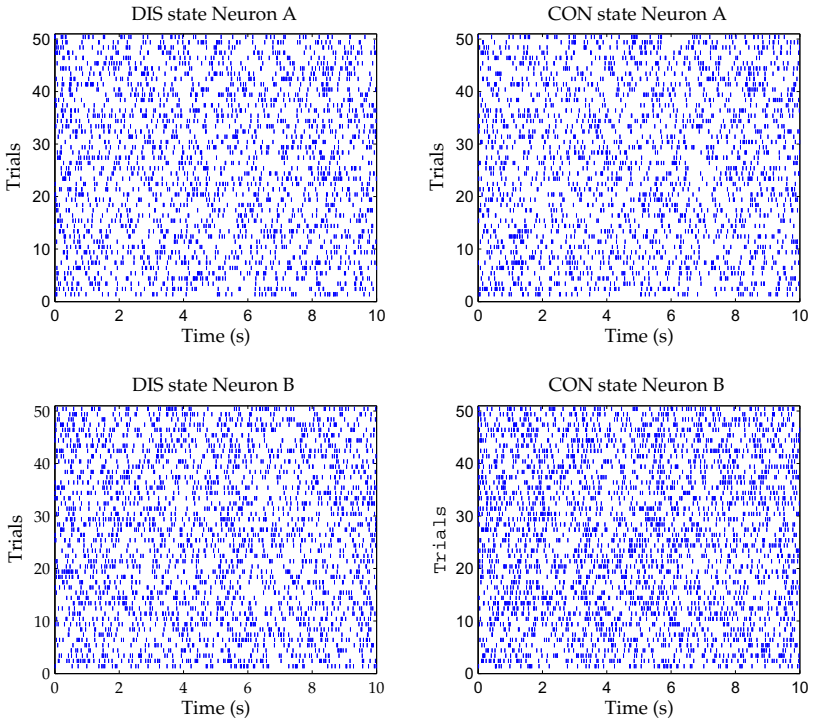


Figure 4-2. The raster plots of neuron A and B under DIS and CON state. Under each state, neurons A and B fire at a reasonable firing rate range (1 Hz to 10 Hz).

to evaluate their capability to detect connectivity and the precision of the connection strength estimation, respectively.

Statistical power estimates the probability of detecting a functional connection given that a functional connection exists with a fixed type-I error [108]. We assume DIS state and CON state to be the *null-hypothesis* and the *alternative-hypothesis*, respectively. Since the mean value of dependence in CON state is larger than that in DIS state, we perform a one-tail hypothesis test. The threshold is defined by the dependence of DIS state with the significance of 0.05. The statistical power is the probability that the dependence of CON state is larger than the threshold.

We compare the performance of four methods as a function of window sizes by computing the statistical powers. With the same window size, the larger statistical power indicates the better capability of detecting functional connectivity.

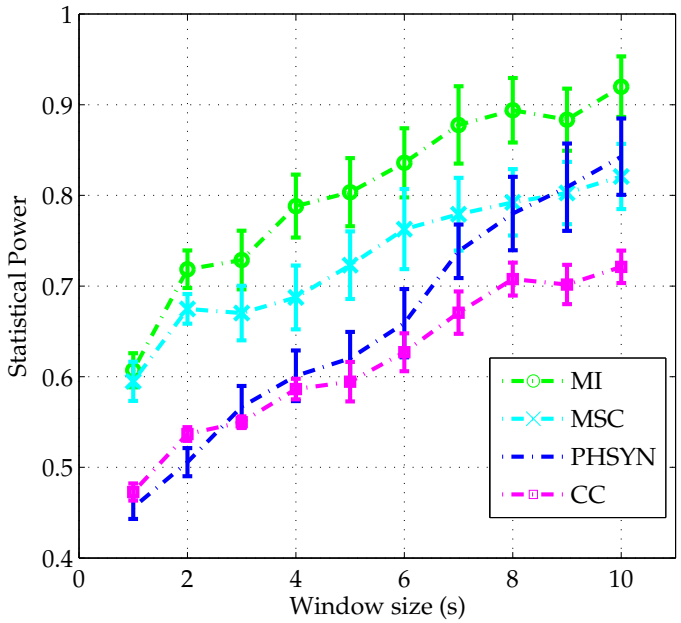
In addition, we also consider the dependence variance as a criterion. Because the dependence value is used to describe the connection strength, the method with a smaller variance is preferred as a more precise connectivity estimator.

4.1.3.3 Simulation results

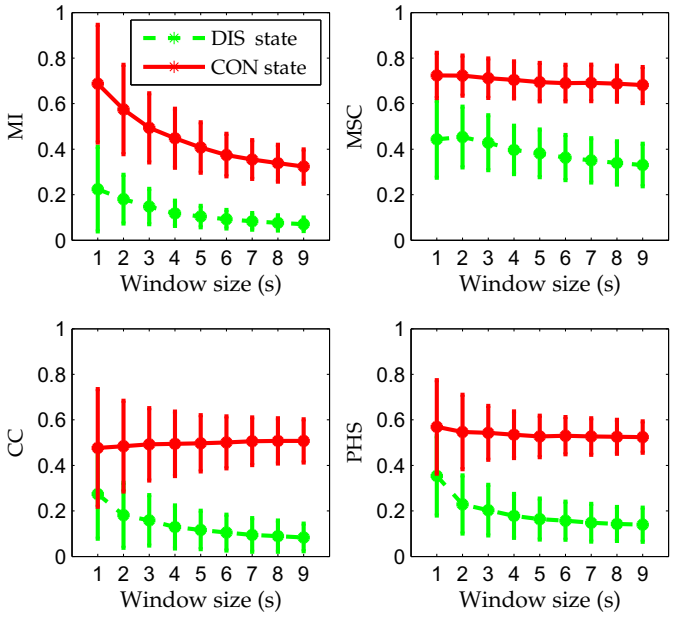
More samples usually enable the measure of functional connectivity with a better accuracy. However, because the time scale of behavior is so short, 100-1000 ms, a statistical method that yields a robust estimator for 10 samples is desired. To compare the performance among different statistical estimators with respect to the window size, we vary the length of the time window applied to the signal from 1 s to 10 s (non-overlapping windows with 10 to 100 samples collected with a period of 100 ms).

We run the simulation in 100 random settings of the synaptic weights between pair of neurons based on uniform distributed realization of the truncated normal distribution $W \sim N(1, 0.1), W \in [0, 2]$. In each setting, the simulation generates 200 subsets of neural activity in DIS state and CON state, respectively. The dependence is measured for each subset with different window sizes. The statistical power of dependence is calculated for each random setting with respect to the window size. Figure 4-3A shows that the statistical power of the coupling detection is a monotonically increasing function of window size. The phase synchronization increases the most, which suggests that phase synchronization is progressively more efficient with longer time windows that characterize stable regimes. In contrast, the mean square contingency and MI have a more gradual increase indicating that their performance is the least sensitive to the window length. The statistical power of MI is the best over all window sizes, while the statistical power for cross-correlation is always lower than that of other dependence estimators, because the nonlinear coupling is not detected well by cross-correlation.

Moreover, a *Kolmogorov-Smirnov Test* (KS -test) is performed on statistical powers of each pair of measures, which indicates that the statistical power of MI and MSC is significantly greater than that of cross-correlation and phase synchronization when



A Statistical powers of the methods with respect to different estimation window sizes. We run the simulation model in 100 random settings. The error bar represents the standard derivation of the statistical power over 100 random settings.



B The mean and standard deviation of each measure with respect to different estimation window sizes.

Figure 4-3. Small sample size performance comparison among four dependence measures: MSC, CC, MI, and PhS, for detecting functional connectivity between spiking neuron A and B.

Table 4-1. Statistical powers of the methods with respect to different average synaptic weight w, when the window size is 10 samples.

Method Weight	1	1.5	2	2.5
MSC	0.34	0.40	0.64	0.77
CC	0.17	0.17	0.37	0.51
MI	0.44	0.64	0.83	0.84
PhS	0.18	0.38	0.38	0.64

the window size is small, e.g., 10 and 20 samples. With 10-sample window size, when we decrease the estimation difficulty by increasing the average synaptic weight W, the statistical power of MI and MSC is significantly greater than that of cross-correlation and phase synchronization, as shown in Table 1. This reflects the robustness of MI and MSC with small window estimation. This robustness suggests that MI and MSC are better candidates to estimate the dynamic functional connectivity.

To further investigate each measure's capability for small window estimation, we present the variance of the estimated connection based on four methods in Figure 4-3B. The standard deviation decreases with increasing window size, as it can be expected. However, if one is interested in small window sizes to quantify the fast dynamics of neural assemblies, we should compare the methods in the small window size case. The standard deviation of MSC (0.10 for CON state and 0.18 for DIS state) is the smallest when window size is 10 samples, while the standard derivation of MI (0.25 for CON state and 0.18 for DIS state) is relatively larger, which implies that MSC is able to estimate the strength of the functional connection more precisely.

In summary, the comparison reveals that MSC is a relatively reliable estimator of functional connectivity with short time resolution and will be our choice in this work.

4.2 Temporal Functional Connectivity Analysis and Interactive Visualization

In this section, the dependence measure is applied to monkey cortical neural activity recorded during a food reaching task with three well defined kinematic states. The temporal functional connectivity is characterized by a dependency graph, where

the nodes (neurons) and edges (the functional connections between pairs of neurons) represent different aspects of the time spatial neuronal interactions. The function connectivity provide a description of the assemblies associated with the time varying kinematic states.

In this section, we characterize the temporal functional connectivity pattern by the transient dependence graph among multiple channels, which is quantify by measuring the statistical dependence. The evolution of transient dependence graph provides a quantifiable representation of the dynamic nature of neural processing and further investigate its association with the underlying behavioral state. The dynamic dependence graph is also interactively visualized with a visualization Java toolkit (Flare), which helps to observe the detail connection of the functional connectivity pattern, its temporal evolution, and its association with behavioral state.

4.2.1 Cortical Neural Data

The data for these experiments was collected in Dr. Nicolelis' primate laboratory at Duke University (see [60] for details). For our experiments, neural data are recorded from an owl monkey's cortex when the animal is performing a food reaching task. Multiple micro-wire arrays are used to record this data from 104 neural cells in the following four cortical areas: posterior parietal cortex ($PP - contra$), primary motor cortex ($M1 - contra$), dorsal premotor cortex ($PMD - contra$) and primary motor & dorsal premotor ($M1/PMD - ipsi$). In tandem with the neural data recording, the 3-D hand positions are digitized when the monkey is performing the task.

For dependence analysis, neuronal spike events are binned in non-overlapping windows of 100 ms. The hand position data sets are digitally low-pass-filtered to avoid aliasing and downsampled to 10 Hz to match the binning utilized in the neural data. Taking the primate's reaction time into account, the spike trains were delayed by 0.230 s with respect to the hand position [103]. Our particular data set contains 104 neural channels recorded for 38.33 min. This time recording corresponds to a data set of

23000×104 time bins, for which about 250 reaches are performed. The time spent in movement and resting are around 11 min and 27 min, respectively.

4.2.2 Kinematic State Analysis

Our testing with real data will elucidate the adequacy of the dependence measures to quantify and analyze neural data given the knowledge of the kinematic variables. Conditioning in the kinematic variables is just a way to segment the data for our statistical tests and does not invalidate the assumption that only neural data interactions are being utilized in the analysis. If the results with this segmentation show statistical different values for the different phases of the movements, then we have demonstrated that the dependence measures may be useful for motor control and other behavior experiments, even when no measured external variables are available. The only extra decision to be made by the experimenter in this more abstract framework is the selection of the baseline, the start of the analysis and the segment length. By brute force one can always use a small window and repeat the analysis in each and then perform clustering on the values obtained. For kinematic state analysis, our hypothesis is that specific neural assemblies are activated given a kinematic state and the goal of the analysis is to characterize assemblies by a set of neurons whose functional connectivity is enhanced significantly to define a state-related neural assembly. Specifically, the functional connectivity analysis of state-related assembly is based on a hypothesis test. Given a kinematic state, the pairwise neuron connections composing the state-related assembly are significantly higher than those in the rest state, which is against the null hypothesis that the connections between a pair of neurons share the same distribution in the rest state and the given kinematic state. Moreover, a graph based on those pairwise functional connections are built to describe a state-related assembly. In the graph, the nodes and the edges represent the index of neurons and the functional connectivity between a pair of neurons.

4.2.2.1 State-related assembly

For the experiments, the neural data are segmented into the rest state and three movement states: rest-to-food (Mv1), food-to-mouth (Mv2), mouth-to-rest (Mv3) according to the previous segmentation of the 3-D hand trajectory [84], as shown in Figure 4-4. The MSC-based measure is applied to estimate the functional connectivity of pairwise neurons in each segment. Since the average duration of each movement state is only 1s, the estimation window size is selected as 1s (10 samples), and we further assume that the interaction within the state-related assembly during each state remains stationary. MSC between pairwise neurons is estimated for 50 trials belonging to each movement state and 100 trials during rest state, which is sufficient to determine the state-related assembly.

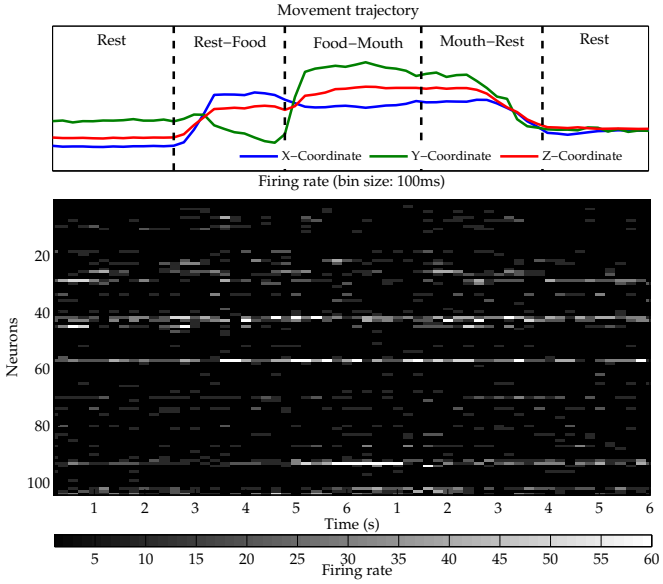
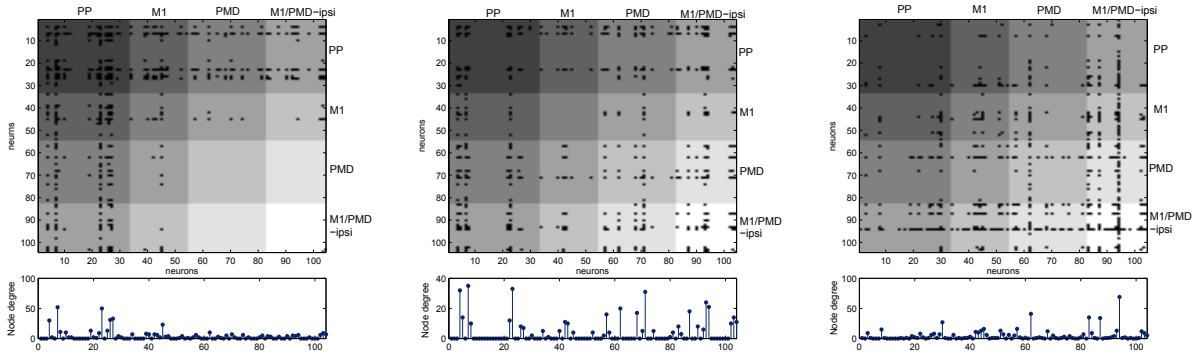


Figure 4-4. Example of reaching movement trajectory. Trajectory is segmented into rest (R), rest-to-food (Mv1), food-to-mouth (Mv2), and mouth-to-rest (Mv3) states.

The estimated dependence for each trial is treated as an observation of the conditional probability distribution given a kinematic state. We perform a *Kolmogorov-Smirnov Test* (KS -test), which gives an indication of the separability of two measurement sets, to check whether the pairwise neuron dependence in a movement state and the

rest state are significantly different. The null hypothesis is that the connection between a pair of neurons share the same distribution in the rest state and the given movement state, which is translated for the KS test on nonsignificant difference of measured functional connection between these two states. Therefore, a rejection of null hypothesis at the 0.0001 significance level indicates that the connection between a pair of neurons is significantly different during the given state. The small significance level 0.0001 is determined by Bonferroni's correction for multiple comparisons where we divide a family wise error rate of 0.05 by the number of comparisons. Given a movement state, all activated connections (*edges*) and the connected neurons (*nodes*) comprise the assembly graph. The *degree* of a neuron is defined as the number of edges incoming to the neuron. Three conditional assembly graphs given three movement states (rest-to-food (Mv1), food-to-mouth (Mv2), and mouth-to-rest (Mv3) states) are plotted based on KS-test results in Figure 4-5. For each graph, the black dots denote the activated functional connections in this movement state with respect to the rest state. There are 104 neurons in the neural data set. Each point (x, y) in the matrix represents the functional connection between neuron x and neuron y. The background color of each graph is used to indicate the cortical area. The bottom plots in each sub-figure show the neuron *degree* (the number of edges incoming to the neuron) in the assembly graph given a movement state.

For each figure, the black dots denote the activated functional connections (*edges*) at the movement state, compared to the rest state. In fact, in any of the movement states the dependence is always higher than in the rest state (Figures not shown), which is consistent with the hypothesis that reciprocal interactions among neurons contribute to the emergence of the behavior [99]. What is more interesting is that the firing rate of neurons during resting and movement is not significantly different, as shown in Figure 4-4.



A Rest-to-food (Mv1) state B Food-to-mouth (Mv2) state C Mouth-to-rest (Mv3) state

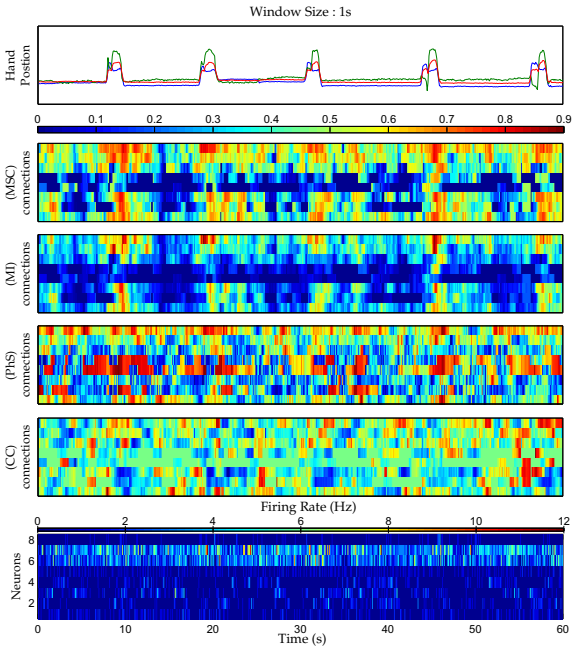
Figure 4-5. 104×104 matrices plot three functional connectivity graphs assessed by KS-test results.

The variability among these three graphs demonstrates the specificity of the state-related assemblies. In other words, the state-related assemblies provide a subset of the cortical neurons that are especially important to emergence and implementation of a specific behavior. For example, in the initiation of the Mv2 movement there is functional connectivity among posterior parietal neurons. This connectivity spreads to multiple cortices (*PMD- contra* and *M1- contra*) during the large excursion reach between the food and mouth. To further investigate the correlation between the functional connectivity in state-related assemblies and behavioral states, we study the evolution of the connection strength and the activation degree of the state-related assemblies in the cortex.

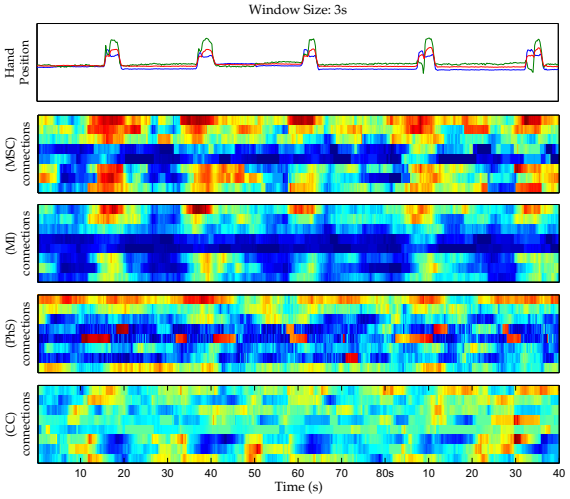
4.2.2.2 Evolution of connection strength

The evolution of functional-connectivity strength is estimated by sliding a 1s or 3 s window by 0.1 s steps over the data. The results reveal that the strength of ensemble connections is enhanced at the movement states. To illustrate this connectivity evolution, the neurons connected to neuron 93 (see Figure 4-5C) are selected from the assembly graph of the Mv3 assembly, which is one of the most connected neuron. The time-varying strengths of these connections are depicted in Figure 4-6A and 4-6B.

In each column sub-figure, the first panel shows the 3-D hand position time series, followed by the connection strength estimated by MSC, MI, PhS, and CC with a 1 or 3 s moving windows (sliding by 0.1s steps). The color represents the connection strength of neuron 93 to its functionally connected neurons. The bottom plot in Figure 4-6A shows the firing rates of neurons 93 and its functional connected neurons.



A Window size: 1 s



B Window size: 3 s

Figure 4-6. Time-varying connection strength of neuron 93 in the Mv3 assembly.

Figure 4-6A demonstrates the dynamics of the connectivity associated with the Mv3 kinematic state with a 1s moving window. The delay, the duration, and the peaks of the connection's strength vary among different neuron pairs. However, the connectivity pattern repeats at every reaching task, with the variance induced by slight movement

differences across trials. These results support the hypothesis that the connection strength in state-related assembly based on KS-test is modulated by the kinematic state.

In Figure 4-6A, it is worth noting that MI and MSC are able to detect the association between the functional connectivity and the kinematic state with a fine time resolution. In contrast, cross-correlation and phase synchronization fail to detect such association with the 1s estimation window. When the window increases to 3 s (Figure 4-6B), the connection strength based on phase synchronization and cross correlation also shows the association with the transition from rest to movement states, but with relatively lower level and worse time resolution. The comparison further demonstrates the robustness of MSC and MI in detecting the functional connectivity with short estimation window.

4.2.2.3 Activation degree of state-related assemblies

So far we have focused on individual neurons, now we move focus to neural assemblies. It is non-trivial to extend the pairwise dependencies to a measure pertaining to the total assembly activity. Hence, we are interested in how to quantify the degree of activation of an assembly by combining the estimated pairwise functional connections reported above, which provides a way to quantify the graph isomorphism between temporal functional connectivity and state-related assembly. To compare the validity of the qualifications, we investigate the correlation between the activation degree of state-related assemblies and the specific kinematic states.

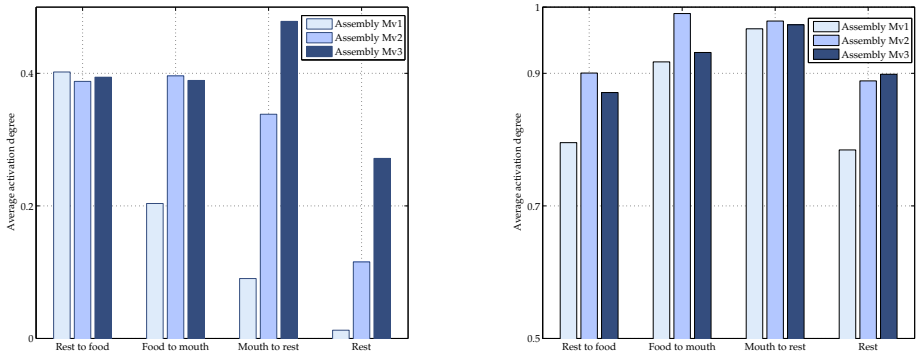
Specifically, the threshold for functional connection between each pair of neurons is estimated from the rest state set of pairwise dependence with a p-value of 0.05. If the dependence of a pair of neurons is over the threshold, the corresponding edge (linking the node pair) and nodes in the assembly graph are activated. There are two possible ways to define the degree of the assembly activation: 1) the probability of the activated edge (*edge method*) defined as:

$$p_{edge} = n_{edge}/N_{edge} \tag{4–16}$$

2) the probability of the activated nodes in the assembly graph (*node method*) defined as:

$$p_{node} = n_{node}/N_{node} \tag{4–17}$$

where n_{edge} and n_{node} represent the number of the activated edges and nodes and N_{edge} and N_{node} are the total number of the edges and nodes in the assembly graph. The edge-based activation degree measures the relative strength of the value of functional connections (interactions) in the given assembly. The node-based activation degree measures the relative number of neurons in the given assembly with enhanced functional connection.



A Average activation degree of the assembly based on the activated edge

B Average activation degree of the assembly based on the activated node

Figure 4-7. Average activation degree of state-related assemblies.

For these three state-related assemblies, their average activation degree is estimated over all trails during Rest-to-food, food-to-mouth, mouth-to-rest, and rest kinematic states, respectively, as shown in Figure 4-7. Figure 4-7A shows the

edge-based activation degree results, where given a state-related assembly, its activation degree always has the highest value during its corresponding kinematic state. From another aspect, during each movement state, its corresponding assembly always has the higher activation degree than the others. Moreover, the activation degrees of all three assemblies are decreased in the rest state. In contrast, there are no such results obtained by the node-based activation degree, as illustrated in Figure 4-7B.

In addition, the temporal evolution of functional activation estimated with the KS-test of three assemblies corresponding to Mv1, Mv2, and Mv3 using the edge and node methods are shown in Figure 4-8, where the upper panel shows the reaching movement trajectory of 3D hand position. The second panel is the activation degree of three neural assemblies based on the activated edge method, estimated by a moving window (sliding 1s window by 0.1 s steps).The y-axis is the three assembly index corresponding the rest-to-food, food-to-mouth and mouth-to-rest state. The third panel demonstrates the activation degree based on the activated node method. The assembly activation based on the edge method detects consistently the repeated food-reaching tasks. Moreover, the highest activation of the Mv1 assembly is only present in the beginning of the Mv1 state. The Mv2 assembly activation peaks during Mv2 and Mv3 state occurs. These results show that the activation of the assemblies is able to reach the peak repeatedly when the specific movement state arrives. These results also suggest that the specific assembly explains the emergence and operation of a specific behavior. In contrast, the assembly activation accessed by the node probability is less effective to detect the specific kinematic state because the same set of neurons is involved in different movements, in spite of the increased activation during the movement states. However, the organization of the interaction network appears more consistent given a specific kinematic state, and this is the reason why edges are better suited for discrimination of functional connectivity. This comparison reveals that the operation of a specific behavior is more closely associated with the network of interactions among the neurons.

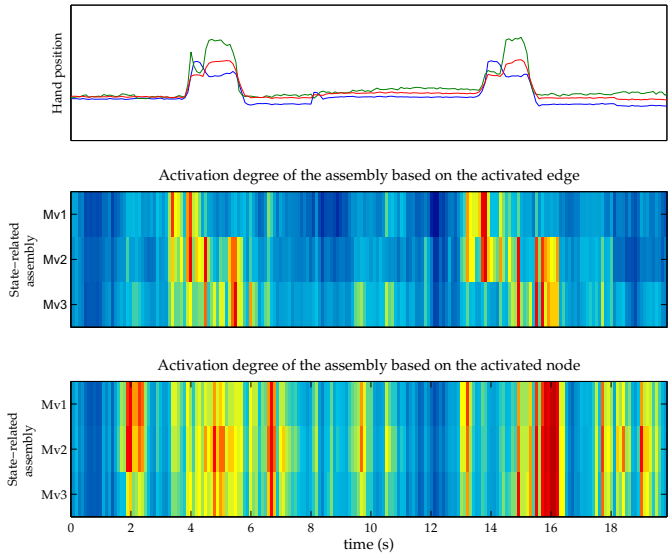


Figure 4-8. The temporal evolution of functional activation of three assemblies corresponding to Mv1, Mv2, and Mv3 states.

4.2.2.4 Dynamics of local functional connectivity

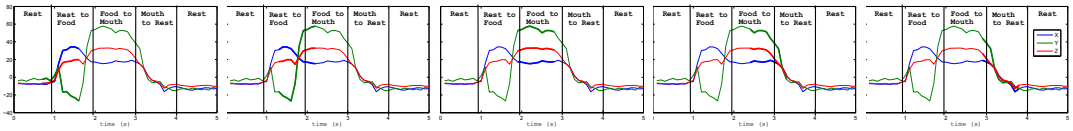
The different organization of the state-related assemblies (Figure 4-5) motivates the question about the varying role of cortical regions in voluntary movement. To further investigate the relationship between cortical neural assemblies and behavior, the Flare toolkit [98] is applied to visualize the local assembly graphs of four cortical areas (*PP-contra*, *M1-contra*, *PMD-contra*, *M1/PMD-ipsi*).

In order to show the evolution of the graphs in time, we further separate the data into 5 sub-states, which are Mv1, Mv2, Mv3 and two middle states (the one between Mv1 and Mv2 and the one between Mv2 and Mv3), as shown in Figure 4-9. For each sub-figure, the five functional connectivity graphs correspond to the 5 movement sub-states. The neurons are paced along a circle and a link indicates that two neurons are connected. The state-related assembly graphs are based on the KS-test results between each sub-state and the rest state, similarly to Section 4.2.2.1. In the interactive visualization of pairwise neural dependence graph created by Flare, the neurons are placed along a circle. A link indicates that two neurons are connected.

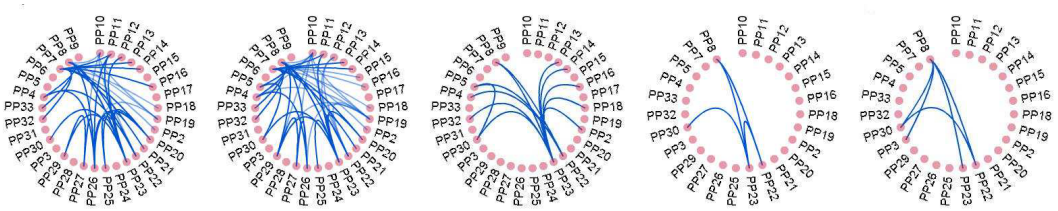
As shown in Figure 4-9, the network in the *PP-contra* becomes highly interconnected before and at the beginning of the reaching task. It supports the role of *PP-contra* in producing planned movement and the linkage between internal to external coordinate systems.

In our results, the local functional connectivity within *M1-contra* is activated by the movement. Figure 4-9C shows that when the velocity approaches zero in sub-state 3, little interaction appears within this region (Figure 4-9). Moreover, for the movements with different joint angles in sub-state 1 and sub-state 5, most connections in assembly network are changed in *M1-contra*, which further supports the hypothesis that the specific assembly contributes to the emergence of the movement with specific kinematic parameters.

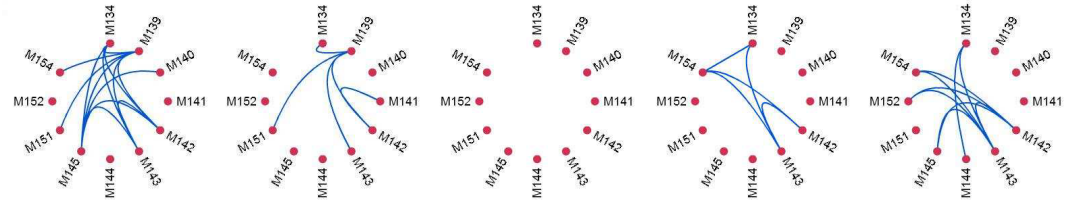
In addition, considerable functional connections are created during the mouth to rest state in *M1/PMD-ipsi*, but not during other kinematic states, which is consistent with the results of the hand trajectory reconstruction in [84]. They show that *M1/PMD-ipsi* accurately captures the mouth/rest regions, but misses the beginning of movement. Although the main organizational principle of primate motor systems is cortical control of contralateral limb movement, motor areas also appear to play a role in ipsilateral limb movements. Our results reveal that the interaction in M1 is modulated with functional connectivity by the ipsilateral limb movements, which support the argument that the motor areas are also able to correlate ipsilateral limb kinematics with high precision [33].



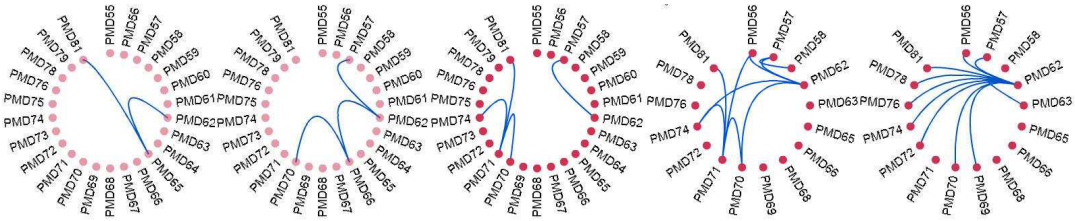
A 5 sub-state of the reaching task trajectory.



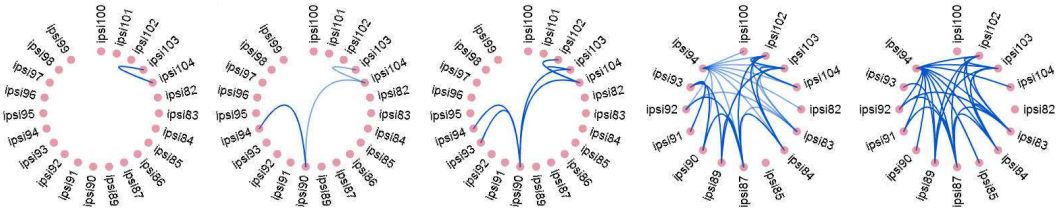
B Local functional connectivity graphs in PP-contra given the 5 movement sub-states



C Local functional connectivity graphs in M1-contra given the 5 movement sub-states



D Local functional connectivity graphs in PMD-contra given the 5 movement sub-states



E Local functional connectivity graphs in M1/PMD-ipsi given the 5 movement sub-states

Figure 4-9. Local functional connectivity graphs in 4 cortical areas: *PP-contra*, *M1-contra*, *PMD-contra*, and *M1/PMD-ipsi* for 5 movement sub-states.

4.3 Neural Modeling via Functional Connectivity Temporal Pattern

In the previous sections, we investigate the estimation and visualization of the temporal functional connectivity, which illustrates its underlying association with behavioral states. The inference of the functional connectivity dynamics with cognitive behavior, disease, stimulation has also been explored in [7, 15, 22, 25, 34, 82], which indicates that the temporal functional connectivity pattern is a good way to describe the spatiotemporal neural activity pattern. However, this functional connectivity pattern is represented by a graph instead of a vector, which make it challenging to use it as input feature in neural system modeling. Instead of directly addressing functional connectivity graphes or matrices, most researches implement graph theory to quantify the topology property of graph pattern including node degree, clustering coefficient, betweenness centrality, and etc. [22, 82]. These topology metrics abstract feature vectors from the functional connectivity graphs/matrices, which allows the implementation of the traditional modeling with the vector-based input space. However, the abstract process can only capture how important a neuron plays a role in the underlying functional communications from certain perspective, but reduces the global graph structure. Therefore, in our work we design a kernel for the functional connectivity matrix, which allow us to implement kernel based machine learning algorithm directly on the temporal functional connectivity pattern. In this section, we first review the statistic topology attribute measure that are widely used to investigate functional connectivity pattern. Then we propose a kernel designed for functional connectivity matrices. In addition, we investigate and compare the modeling performance using graph topologies or functional connectivity matrices with matrix kernel.

4.3.1 Measures of Neural Network Topology

Graph topology can be quantitatively descried by a wide variety of measures, which has been implemented to characterize the functional connectivity pattern in an abstract way and then used as an feature vector in modeling.

1. **Node degree** The degree of a node is the number of connections that link it to the rest of the network, which is the most fundamental network measure.
2. **Clustering coefficients** The weighted clustering coefficient [6] is calculated individually for each node, which measure the degree to which nodes in network tend to cluster together. If the nearest neighbors of a node are also directly connected to each other, then they consist a cluster. The clustering coefficient quantifies the number of connections that exist between the nearest neighbors of a node as proportion of maximum number of possible connections, which defined by

$$C_n = \frac{2e_n}{k_n(k_n - 1)}, \tag{4–18}$$

   where k_n is the number or neighbors of the node n, e_n is the number of connected pairs between all neighbors of the node n, and $k_n(k_n - 1)$ is the maximum number of possible connection among all neighbors.
3. **Betweenness centrality** The centrality of a node measure how many of the shortest paths between all other node pairs in the network pass through it. A node with high centrality indicates that it is crucial to efficient communication. The betweenness centrality is calculated

$$B_n = \sum_{i \neq j \neq n} \frac{n_{ij}(n)}{n_{ij}}, \tag{4–19}$$

   where n_{ij} is the number of shortest paths between i and j, and $n_{ij}(n)$ is the number of shortest path between i and j that traverse n.

All these graph topology measures are able to abstract a vector from the $N \times N$ functional connectivity graph, where N the number of the nodes. The feature vector size is N. Then we are able to use them as input feature in neural model by applying traditional vector-based machine learning algorithm. However, it is not established whether these attributes can sufficiently characterized the functional connectivity pattern.

4.3.2 Kernel for Functional Connectivity Matrix

In order to effectively use temporal functional connectivity pattern as the input feature for neural system modeling, it is important to select a suitable representation. In our case, the global connection pattern, the connection weight of each pair of neurons, and also the neuron's labels in this network are considered as the effective information. Therefore, we represent the temporal pattern as matrix defined by

$$\mathbf{s} = \begin{pmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nn} \end{pmatrix}, \text{for n neurons}$$

where s_{ij} is the functional connection weight between neuron i and neuron j. The functional connectivity matrix is symmetric with respect to the main diagonal.

Conceptually, this kernel of functional connectivity matrix operates in the same way as the kernels operating on data samples in machine learning [86] and information theoretic learning [74]. To take advantage of the framework for statistical signal processing provided by RKHS theory, κ is required to be a symmetric positive definite function. By the Moore-Aronszajn theorem [4], this ensures that an RKHS H_κ must exist for which κ is a reproducing kernel. Since we only care about the interdependence pattern, the kernel κ is further required to be invariant with the artificial order of the neurons, that is, exchanging row and corresponding column should not impact the kernel value.

By analogy to how the Gaussian kernel is obtained from the Euclidean norm, we can define a similar kernel for functional connectivity matrix as

$$\begin{aligned} \kappa(\mathbf{s}, \mathbf{s}') &= \exp(-\frac{\|\mathbf{s} - \mathbf{s}'\|^2}{\sigma^2}) \\ &= \exp(-\frac{\|\mathbf{D}\|^2}{\sigma^2}) \end{aligned} \tag{4–20}$$

where σ is the nonlinear weighting kernel size parameter and $\mathbf{D} = \mathbf{s} - \mathbf{s}'$. The matrix norm used in this work is Frobenius norm defined by

$$\|\mathbf{D}\|_F = \sqrt{\text{trace}(\mathbf{D}^*, \mathbf{D})} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} |d_{ij}|^2}, \tag{4–21}$$

We investigate properties of the kernel that we designed for functional connectivity matrix, which is essential to indicate that this kernel is well defined and able to induce an RKHS with necessary mathematical structure for computation.

Property 1. The kernel of functional connectivity matrix is a symmetric positive definite kernel of matrix.

Proof: The function's symmetry is inherited directly from property of the norm. In addition, the kernel with the matrix norm definition (Equation 4–20) can be rewritten as

$$\kappa(\mathbf{s}, \mathbf{s}') = \prod_{i}^{n}\prod_{j}^{n} \exp(-\frac{\|s_{ij} - s'_{ij}\|}{\sigma^2}), \tag{4–22}$$

which is the product of Gaussian kernel. Since Gaussian kernel is positive definite kernel, the product of gaussian kernel is also positive definite.The positive definiteness of the kernel indicates that the kernel induce an RKHS.

Property 2. The kernel of functional connectivity matrix is invariant with the row/column permutation.

Proof: The kernel is only sensitive to the difference between two matrices. Consequently, we can write $\kappa(\mathbf{s}, \mathbf{s}') = \kappa(\mathbf{s}-\mathbf{s}') = \kappa(\mathbf{D}) = \kappa(\sum_{i=1}^{n}\sum_{j=1}^{n} |d_{ij}|^2)$, which is a entrywise operator. Therefore, it is unrelated to the row/column order.

4.3.3 Experiment

In order to investigate the kernel capability to identify the mapping from the functional connectivity to its underlying stimulation or its reasoning behavioral state, temporal pattern of functional connectivity is used as the input feature to classify the

multiple micro-stimulation patterns. Here we use two category representations of functional connectivity pattern, that is the connectivity matrix and the topology vector including: node degree, clustering coefficient and betweenness centrality. For the topology vector, standard vector-based Gaussian kernel is used. The matrix kernel proposed in the previous section is applied to the connectivity matrix. In this way, all the input features are mapped into the RKHSs, respectively. The standard support vector machine classification algorithm is used to identify the map from a sample in RKHS to their corresponding stimulation class.

The data set is the same data we used in Chapter 3, which recorded from somatosensory cortex during the bipolar micro-stimulation being imprinted in Thalamus, which contains 24 patterns: 8 different locations, 3 different amplitude levels for each location, and 125 events for each pattern, as shown in Figure 2-8. The spike data are collected from 50 Channels. The neural response is selected right after each stimulus with the 20ms window length.

Considering the short time scale of neuron response, Schreiberg et al. similarity measure defined in Equation 4–2 is used to estimate the functional connection weight between each pair of neurons, which constitute a functional connectivity matrix. In Schreiberg et al. similarity measure for spike trains, the functional representation of a spike train is defined by

$$\hat{\lambda} = \sum_{n=1}^{N} \exp(-\frac{t - t_n}{\tau}) U(t - t_n), \tag{4–23}$$

In this functional representation, τ is a free parameter, which is important for the representation accuracy. Therefore, we optimize τ by minimizing $E(\hat{\lambda} - \lambda)^2 = E(\hat{\lambda}^2 + \lambda^2 - 2\hat{\lambda}\lambda)$, which can be equalized to maximize $E(\hat{\lambda}^2 - 2\hat{\lambda}\lambda)$, where $\lambda(t) = E(s(t)) = 1/N \sum_{n=1}^{N} s_n(t)$. In this data set, we estimate $\tau = 0.01$, which is average τ over all the channels. The average functional connectivity matrix for each stimulation pattern is shown in Figure 4-10, where the difference among different stimulation patterns is

observable. In order to quantify this difference, we estimated the Frobenius distance of each pair of functional connectivity matrices (3000 graphs) defined in Equation 4–21, which is shown in Figure 4-11.

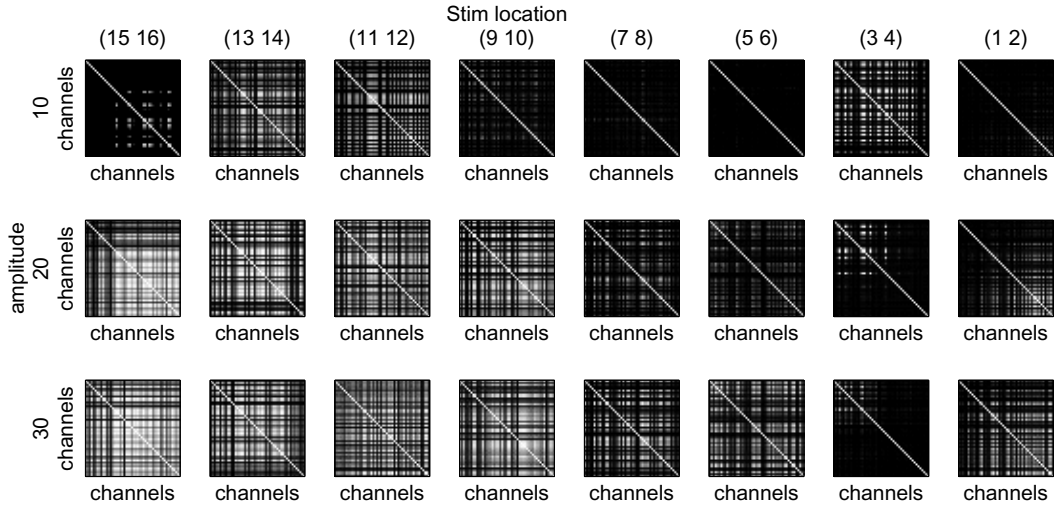


Figure 4-10. Functional connectivity matrices for each stimulation pattern

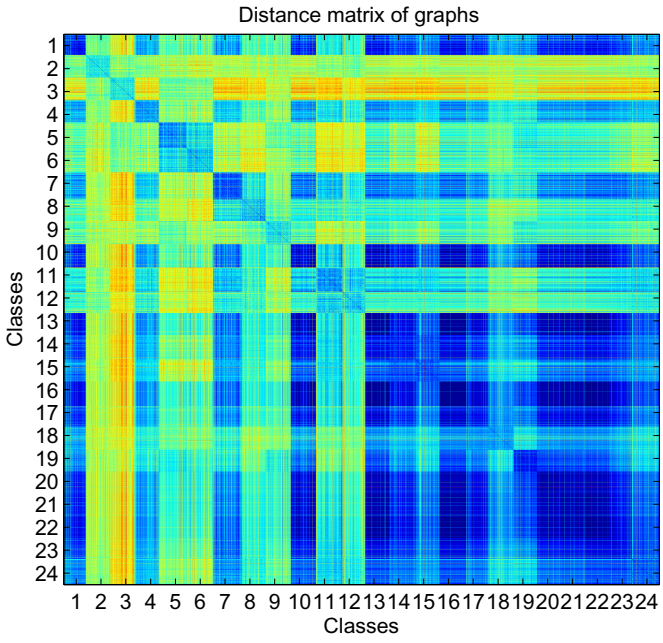


Figure 4-11. Pairwise Frobenius distance of functional connectivity patterns

Table 4-2. Classification comparison over different input features.

	Matrix kernel	Graph Topology			Temporal data
		Clustering coefficient	Node degree	Betweenness centrality	
Test error	0.16	0.19	0.34	0.51	0.23

In Figure 4-11, the x-axis and y-axis were sorted by the index of stimulation pattern. It was observed that the distance value is small within 125125 square lying on the diagonal, which indicates that the distance between the samples within the same class is relatively smaller than the cross-class distance in RKHS. For classification task, we perform a ten-fold cross-validation test, where 2700 samples are used as the training data and the rest 300 samples are used as the test data. The testing results of different input features are shown in Table 4-2, which illustrates that the matrix kernel based method has the best classification performance. Comparison among different graph topology vectors shows that clustering coefficients have the best classification accuracy. The clustering coefficients provide a description of the locality of functional connectivity pattern, which is important in neural coordinate mechanism [61]. However, the clustering coefficient vector loses the information of which assembly a certain neuron belongs to, which changes cross different stimuli and is informative to the stimulation pattern.

We further investigate the test results of the matrix kernel as shown in Figure 4-12, where the most classification mistakes happens among the stimulation patterns that shared the same stimulation location but have different stimulation amplitude. Moreover, the samples corresponding to the stimulation with small amplitude are easier to be misclassified, since the variability of spike trains is large especially for small stimuli that correspond to small signal noise ratio.

4.4 Discussion

In this chapter, we utilize dependence measures to quantify and study the dynamics of functional connectivity in motor/sensory cortex exclusively from the neural activity.

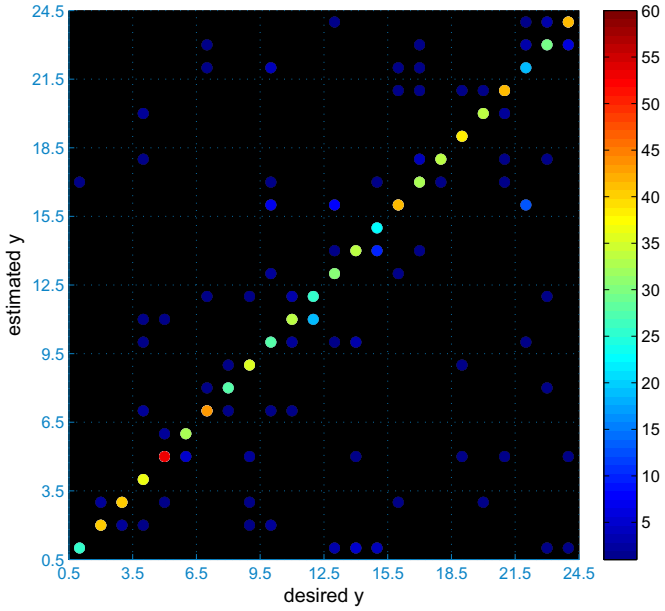


Figure 4-12. Classification results of functional connectivity matrix kernel

Because the estimation window is limited by the time scale of kinematic/stimulation states, the robustness of the functional-connectivity estimation with a small window size is key to detect the dynamic nature of neural activity. The comparison among four dependence measures demonstrates that MSC and MI are the most reliable estimators for one second windows which contain 10 samples of binned neural activity (100 ms). This bin size has been preferred for motor BMIs, but future studies should be conducted to verify that it is appropriate for functional studies as the one presented in this work and if not, determine its optimal value. In addition, both MSC and MI are dependence measures that can be generalized to study multichannel data, so this extension will be investigated in our future work.

In this study, we test this algorithm with microelectrode array data for a food reaching task. Three state-related assemblies are assessed corresponding to the rest-to-food, food-to-mouth, and mouth-to-rest kinematic state, respectively. In spite of the coarse sampling of neural structures and the relatively coarse time resolution

of the method, which imposed the grouping of all activated neurons in a single neural assembly, the graphs of conditional function connectivity show distinct assemblies.

Our approach detects a close association between the increase of functional connection strength in state-related assembly and the transition from rest to movement. Moreover, the activation degree of the state-related assembly repeatedly reaches the peak when the corresponding movement state occurs. In future work, the activation degree can be further investigated as a quantifiable way to estimate the specific movement state with functional connectivity. These results also reveal that the network of interactions among neurons seems the key factor for the operation of a specific behavior. Although in these experiments we conditioned on the kinematic variables, the method extracts the information solely from neural data, transcends motor tasks and can be applied to any brain area. In this more general scenario, the experimenter will have to select a baseline state, a starting point and a data length for statistical robustness (1 sec currently), and apply the dependence measures for each window of neural data. Clustering of the dependency measures, or building the graphs to find consistency between dependencies across sequential segments can be used to abstract different interactions that are the signature for different kinematic states. The relationship to kinematics is merely used to support our conclusions.

The circle tree graphs of the local functional connectivity within each cortical area *PP-contra*, *M1-contra*, *PMD-contra*, and *M1/PMD-ipsi* help understand the time varying spatial assemblies that are in the cortex. Moreover, the dynamics of the local assembly graphs for each cortical area corroborate the regional contribution to the movement implementation. Since *PP-contra* localizes the body in space and plans the movement, neurons are highly interactive at the beginning of the reaching task. The interaction in *M1-contra* is only expressed when the movement velocity is large. These results also show that the functional connectivity is associated with ipsilateral limb movement.

Although this was unexpected, we have shown by different analysis [84] that this only occurs during the latter portion of the trajectory.

In summary, the mean square contingency seems to be a valuable technique to inquire about functional activation of neural assemblies during behavior because of its statistical power at small sample size and its easy computation. One big challenge is how to shorten the observation window (now at least 1 s) to increase the temporal resolution needed to study behavior. The other big challenge is how to fully utilize the information contained in the pairwise activations. Our approach was to apply a statistical significant threshold to simply the connectivity matrix and provide a first level analysis of functional connectivity. However, there is potentially much more information in the connection matrix to be quantified by for instance performing spectral clustering on this matrix to find and track clusters of functional activation over time. This will be pursued in further studies.

The dependence analysis of neuronal functional connectivity dynamics can provide an unsupervised estimation of the hidden neuronal states and their transitions related to the kinematic states. In addition, the state-related assembly also reveals the internal functional structure present in the multichannel of neural signals, which allows a more efficient utilization of the neural data in BMI applications, since neural modeling using multichannel data normally require an independence assumption amongst neural signals.

However, functional connectivity pattern can only be characterized matrices or graphes, which induces the challenge when we decode the kinematic/stimulation information from it, that is, no conventional vector-based machine learning can not be directly applied on it. The graph theory provide a way to extract feature vector from a graph/matrix with losing some distribution and global structure of the connectivity pattern. The matrix kernel proposed in this work enable decoding kinematic/stimulation states from the functional connectivity pattern with no feature extraction. It bypass

the reduce of the information induced by feature extraction, which contributes to the enhancement of the decoding accuracy in the stimulation pattern classification test. In our classification cost, both matrix kernel-based approach or topology -based approach is able to obtain better classification performance than the temporal pattern based approach, which suggested that some important kinesmetic/stimulation state is coded in the distributed functional network structures.

CHAPTER 5
CONCLUSIONS

Brain is a complex system, which contains a large number of functional interacting neurons. The neurons communicate through spike trains. They select and coordinate their distributed activity, which is associated with movements, stimuli, and some other cognitive actions [61]. In this work, our goal is to decode the behavior/stimulation information from the neural activity. Considering the complexity of the system, instead of modeling each component in the neural system and combining them to build a whole model, we treat the neural system as a black box with observable inputs (neural activity) and outputs (corresponding behavioral/stimulation state). A machine learning framework is designed to identify the functional mapping from the neural activity and behavior states. As can be expected, it is challenging to extract information from the activity in a highly distributed, dynamic and complex system using traditional modeling approaches.

5.1 BMI Application

From BMI application perspective, our work proposes the kernel-based machine learning framework to, which provides a better way to decode information from the temporal and spatial pattern of neural activity. The kernel-based framework overcomes challenges induced by the special signal format and thus allowed us to model and control a neural system. Because the neurons communicate with spike trains, the information is not coded in the amplitude but in the timing of spikes. Therefore, no conventional machine learning approach can be directly applied to it. The existing approach is to transform spike trains to amplitude data by counting the number of spikes in a short bin and use this number to represent the value at this time. After transforming a spike train to a discretized vector, a number of existing signal processing technologies have been extended and then applied on the discretized firing data. However, the outstanding drawbacks are when a fine time resolution (less than 10ms) is required. The fine time resolution requires a small bin size, which causes the sparsity

and high-dimensionality of the decoder input and then worsen the modeling difficulty and the computation cost. In our work, we investigate kernels designed on the space of spike trains, which are able to projection the spike train into a RKHS. Although there is no algebraic structure in the original spike train space, the regression method applied in RKHS can be implemented by many kernel-based regressor. Therefore, we take advantage of the kernel designed for spike train and kernel-based regressor, and propose a decoder that can directly work on the spike times, which bypasses limitations of the discretized representation based approaches. The experiment indicates that our approach outperforms GLM and the spikernel with respect to both decoding performance and computation time when the time resolution is required to be small.

However, in the rat sensory stimulation experiment, it is observed that the firing pattern of neurons has marked variability, that is, for the same stimuli, the neural response is not repeated, which causes the fluctuations in the decoder output. In addition, a functional unit of the brain contains thousands of neurons. Only the activity of a small subset neurons can be recorded in the experiment, which fails to cover the complete information that gives rise to the intentional states. This is illustrated by lack of ability to discriminate different stimulation channel when we decode the multi-channel micro-stimulation pattern. In contrast with spike trains, LFPs as an alternative neural signal with larger spatial and time scale are recoded simultaneously with spike train from the same electrode array, which contains complementary information of the intentional state of neural system. Therefore, it is worth to combine them in the same model to enhance the robustness and accuracy of the decoder. Considering the challenges posed by the different signal format, spatiotemporal scale, and unknown relationship, tensor product kernel based decoder is proposed for the multi-scale neural decoding. The results of reconstructing the tactile/ micro stimulation show that this decoder is able to effectively extract the complementary information from multi-scale neural responses.

The ultimate goal of our work is to emulate ”nature touch” with micro stimulation. With this novel decoding methodology, a MIMO adaptive inverse control scheme is built in the spike train RKHS to control neural responses to match the target spatiotemporal firing pattern. The experimental results are presented in a synthetic neural system built from LIF neurons but the excitation is diverse and broad to mimic more realistic conditions. They show that the controller adaptation allows the control system to provide the desired firing patterns with high accuracy. Moreover, the controller is also able to adapt to perturbation of the underlying neural system organization, which is an essential quality for the control system, otherwise the control system will fail to track the target signal when perturbations happen. In this more realistic case, the desired stimulation pattern may not exist to reproduce the target signal but the linear structure of the controller in RKHS and its convex cost function guarantee that this control scheme is able to find the global optimal spatiotemporal pattern of stimulation that can minimize the dissimilarity between the system output and the target signal. These preliminary results show that this kernel-based control diagram may be applicable in real stimulation control with the multi-scale neural decoding methodology. In the current mode, only open loop control is applied on the rat because of the animal experiment setting up. It is observed that the controlled system output have the similar spatiotemporal pattern with the target one induced by the tactile stimulation. Close loop control will be implemented in future.

In addition, most existing decoding framework are based temporal pattern without considering the joint information cross channels, because of its complexity. An alternative way to characterize the joint information is to estimate functional connectivity pattern, where the dependence between each fair of neurons is quantified. In our work, we estimated temporal functional connectivity pattern. The association between the temporal functional connectivity pattern and the animal intentional states is observed, which is not directly visible between the temporal pattern of neural activity and intentional states. The association is also established in the past using a supervised

framework (i.e. with the knowledge of the desired response [84] or using a Hidden Markov Model trained with pre segmented data [23]), however here it is decided with a statistical test exclusively from neural data dependencies. In addition, matrix kernel is designed for the functional connectivity matrix, which allows to use the temporal functional connectivity pattern as the input feature of the decoder. The results of rat experiment show that using temporal functional connectivity is able to classify the micro-stimulation pattern and have less misclassification error than using temporal pattern of the data with independent assumption. However, there are considerable limitations of decoding information from the temporal functional connectivity patterns. Measuring temporal dependency requires a truck of data, which limits its time resolution. It may suggest that the functional connectivity is not an appropriate feature for decoding task that requires short time resolution.

5.2 Methodology Application

The kernel based framework we designed in this work is a general computational framework, whose application area is not constrained to the neural engineering. For example, the Schoenberg kernel based regressor of spike trains is suitable for general point process data. In finance and economics area, we can model the arrival of events such as corporate bankruptcies, mergers and acquisitions, or security trades as point process data. Moreover, the customer claim arrivals for insurance, the arrivals or departures of customers in queuing problem, and the occurrences of diseases in health care are all point process data, which can be modeled by the Schoenberg kernel. Another potential application area is spiking neural network, which falls into the third generation of neural network models. The spiking neural network take advantage of the pulse coding and the principles of reservoir computing, and thus have better ability to model the nonlinear mapping from the system input to system output in theory. However, in practice there is no simple and practical way to tune the parameters of the neural circuits because the elements of neural circuit are interacted with spike

trains instead of amplitude vectors. As a result there has been few applications of spiking neural networks to solve computational tasks that are commonly addressed by applying conventional machine learning method on the rate coded output layer of neural networks. With the Schoenberg kernel based regression method, the transformation from spike trains to the rate vector is not required anymore in the output layer.

In addition, kernel-based adaptive inverse control diagram proposed in this work provide a general control framework that allows controlling a nonlinear dynamic system fully in the RKHS. The advantages of control in RKHS lie on the following aspects. First, it can be directly applied to any type of data formate, as long as we can define a kernel that can map the data from the original input space to RKHS. Even for the data space with no algebraic structure, the controller parameter can be adjusted to minimize the distance between target signal and the system output in RKHS. In addition, the control diagram is able to control the output of a nonlinear system to follow the target pattern with linear control in RKHS. Its linear structure in RKHS guarantees that the controller will not converge to local minima during the control process. In our work, we show two examples: spike trains and heterogenous multi-source data.

REFERENCES

[1] A. M. Aertsen, G. L. Gerstein, M. K. Habib, and G. Palm, "Dynamics of neuronal firing correlation: modulation of "effective connectivity"," *Journal of Neurophysiology*, vol. 61, no. 5, pp. 900–917, 1989.

[2] M. Aghagolzadeh, S. Eldawlatly, and K. Oweiss, "Synergistic coding by cortical neural ensembles," *IEEE Transactions on Information Theory*, vol. 56, pp. 875 – 889, 2010.

[3] Y. Ahmadian, A. M. Packer, R. Yuste, and L. Paninski, "Designing optimal stimuli to control neuronal spike timing," *Journal of Neurophysiology*, vol. 106, pp. 1038–1053, 2011.

[4] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950.

[5] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Prentice Hall, Dec. 1994.

[6] A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks," *PNAS*, vol. 101, pp. 3747–3752, 2004.

[7] D. S. Bassett and E. Bullmore, "Small-world brain networks," *Neuroscientist*, vol. 12, p. 512, 2006.

[8] A. Belitski, S. Panzeri, C. Magri, N. K. Logothetis, and C. Kayser, "Sensory information in local field potentials and spikes from visual and auditory cortices: time scales and frequency bands." *Journal of computational neuroscience*, vol. 29, no. 3, pp. 533–545, Dec. 2010.

[9] P. Berens, G. Keliris, A. Ecker, N. Logothetis, and A. Tolias, "Feature selectivity of the gamma-band of the local field potential in primate primary visual cortex," *Frontiers in Neuroscience*, vol. 2, p. 199207, 2008.

[10] P. Berkes, F. Wood, and J. Pillow, "Characterizing neural dependencies with copula models," *Advances in Neural Information Processing Systems*, vol. 21, pp. 129–136, 2009.

[11] W. Bialek, F. Rieke, R. de Ruyter van Steveninck, and D. Warland, "Reading a neural code," *Science*, vol. 252, no. 5014, pp. 1854–1857, 1991.

[12] A. J. Brockmeier, S. Member, J. S. Choi, M. M. Distasio, and J. T. Francis, "Optimizing microstimulation using a reinforcement learning framework," *IEEE Engineering in Medicine and Biology Magazine*, pp. 1069–1072, 2011.

[13] E. N. Brown, R. E. Kass, and P. P. Mitra, "Multiple neural spike train data analysis: state-ofthe-art and future challenges," *nature neuroscience*, vol. 7, pp. 456–461, 2004.

[14] D. Brugger, S. Butovas, M. Bogdan, and C. Schwarz, “Real-time adaptive microstimulation increases reliability of electrically evoked cortical potentials,” *Biomedical Engineering, IEEE Transactions on*, vol. 58, pp. 1483 –1491, 2011.

[15] E. Bullmore and O. Sporns, “Complex brain networks: graph theoretical analysis of structural and functional systems.” *Nature reviews. Neuroscience*, vol. 10, no. 3, pp. 186–198, 2009.

[16] G. Buzsáki and A. Draguhn, “Neuronal oscillations in cortical networks,” *Science*, vol. 304, no. 5679, pp. 1926–1929, Jun. 2004.

[17] J. M. Carmena, M. A. Lebedev, R. E. Crist, J. E. O’Doherty, 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,” *PLoS Biol*, vol. 1, no. 2, pp. 193 – 208, 2003.

[18] J. K. Chapin, K. A. Moxon, R. S. Markowitz, and M. A. Nicolelis, “Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex.” *Nature neuroscience*, vol. 2, no. 7, pp. 664–670, Jul. 1999.

[19] B. Chen, S. Zhao, P. Zhu, and J. C. Principe, “Quantized kernel least mean square algorithm,” *Transactions on Neural Networks*, vol. 23, pp. 22–32, 2012.

[20] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[21] L. Csato and M. Opper, “Sparse online gaussian processes,” *Neural Computation*, vol. 14, pp. 641–668, 2002.

[22] I. Daly, S. J. Nasuto, and K. Warwick, “Brain computer interface control via functional connectivity dynamics,” *Pattern Recognition*, vol. 45, pp. 2123–2136, 2012.

[23] S. Darmanjian, “Design and analysis of generative models for brain machine interfaces,” Ph.D. dissertation, University of Florida, 2009.

[24] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT Press, 2001.

[25] L. Deuker, E. T. Bullmore, M. Smith, S. Christensen, P. J. Nathan, B. Rockstroh, and D. S. Bassett, “Reproducibility of graph metrics of human brain functional networks,” *NeuroImage*, vol. 47, no. 4, pp. 1460 – 1468, 2009.

[26] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez, “Co-adaptive brain machine interface via reinforcement learning,” *IEEE Transactions on Biomedical Engineering (Special issue on Hybrid Bionics)*, vol. 56, pp. 54–64, 2009.

[27] A. S. Ecker, P. Berens, G. A. Keliris, M. Bethge, N. K. Logothetis, and A. S. Tolias, "Decorrelated neuronal firing in cortical microcircuits," *Science*, vol. 327, pp. 584–586, January 2010.

[28] S. Eldawlatly, R. Jin, and K. G. Oweiss, "Identifying functional connectivity in large-scale neural ensemble recordings: A multiscale data mining approach," *Neural computation*, vol. 21, pp. 450–477, 2009.

[29] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, pp. 2275–2285, 2003.

[30] X. Feng, B. Greenwald, H. Rabitz, E. S. Brown, and R. Kosut, "Toward closed-loop optimization of deep brain stimulation for parkinson's disease: concepts and lessons from a computational model," *Journal of Neural Engineering*, vol. 4, no. 2, p. L14, 2007.

[31] N. A. Fitzsimmons, W. Drake, T. L. Hanson, M. A. Lebedev, and M. A. L. Nicolelis, "Primate reaching cued by multichannel spatiotemporal cortical microstimulation," *The Journal of Neuroscience*, vol. 27, no. 21, pp. 5593–5602, 2007.

[32] J. T. Francis, S. Xu, and J. K. Chapin, "Proprioceptive and cutaneous representations in the rat ventral posterolateral thalamus," *Journal of Neurophysiology*, vol. 99, no. 5, pp. 2291–2304, 2008.

[33] K. Ganguly, L. Secundo, G. Ranade, A. Orsborn, E. F. Chang, D. F. Dimitrov, J. D. Wallis, N. M. Barbaro, R. T. Knight, and J. M. Carmena, "Cortical representation of ipsilateral arm movements in monkey and man," *The Journal of Neuroscience*, vol. 29, pp. 12 948–12 956, 2009.

[34] C. E. Ginestet and A. Simmons, "Statistical parametric network analysis of functional connectivity dynamics during a working memory task," *NeuroImage*, vol. 55, no. 2, pp. 688 – 704, 2011.

[35] F. Grammont and A. Riehle, "Spike synchronization and firing rate in a population of motor cortical neurons in relation to movement direction and reaction time," *Biological Cybernetics*, vol. 88, pp. 360–373, 2003.

[36] S. Grün, M. Diesmann, and A. Aertsen, "Unitary events in multiple single-neuron spiking activity: I. detection and significance," *Neural Computation*, vol. 14, no. 1, pp. 43–80, January 2002.

[37] J. R. Huxter, T. J. Senior, K. Allen, and J. Csicsvari, "Theta phase-specific codes for two-dimensional position, trajectory and heading in the hippocampus," *Nature Neuroscience*, vol. 11, pp. 587–594, 2008.

[38] *Csim: a neural circuit simulator.*, The IGI LSM Group, 2006.

[39] B. James and J. Ranck, “Which elements are excited in electrical stimulation of mammalian central nervous system: A review,” *Brain Research*, vol. 98, no. 3, pp. 417 – 440, 1975.

[40] R. S. Johansson and G. Westling, “Roles of glabrous skin receptors and sensorimotor memory in automatic control of precision grip when lifting rougher or more slippery objects,” *Experimental Brain Research*, vol. 56, no. 3, pp. 550–564, Oct. 1984.

[41] R. E. Kass, V. Ventura, and E. N. Brown, “Statistical issues in the analysis of neuronal data,” *Journal of Neurophysiology*, vol. 94, no. 1, pp. 8–25, 2005.

[42] R. C. Kelly, M. A. Smith, R. E. Kass, and T. S. Lee, “Local field potentials indicate network state and account for neuronal response variability,” *Journal of computational neuroscience*, vol. 29, pp. 567–579, 2010.

[43] J. P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, “Measuring phase synchrony in brain signals,” *Human Brain Mapping*, vol. 8, pp. 194–208, 1999.

[44] L. Li, I. M. Park, S. Seth, J. S. Choi, J. T. Francis, J. C. Sanchez, and J. C. Principe, “An adaptive decoder from spike trains to micro-stimulation using kernel least-mean-squares (klms),” in *Machine Learning for Signal Processing (MLSP), 2011 IEEE International Workshop on*, Sept. 2011, pp. 1–6.

[45] L. Li, S. Seth, I. Park, J. C. Sanchez, and J. Principe, “Neuronal functional connectivity dynamics in cortex: An msc-based analysis,” in *IEEE International EMBS Conference*, 2010.

[46] J. Liu and W. Newsome, “Local field potential in cortical area mt: Stimulus tuning and behavioral correlations,” *Journal of Neuroscience*, vol. 26, pp. 7779 – 7790, 2006.

[47] J. Liu, H. K. Khalil, and K. G. Oweiss, “Model-based analysis and control of a network of basal ganglia spiking neurons in the normal and parkinsonian states,” *Journal of Neural Engineering*, vol. 8, no. 4, p. 045002, 2011.

[48] W. Liu, I. Park, and J. C. Principe, “An information theoretic approach of designing sparse kernel adaptive filters.” *IEEE Transactions on Neural Networks*, pp. 1950–1961, 2009.

[49] W. Liu, P. Pokharel, and J. Principe, “Correntropy: Properties and applications in non-gaussian signal processing,” *IEEE Transactions on Signal Processing*, vol. 55, pp. 5286– 5298, 2007.

[50] W. Liu, J. C. Príncipe, and S. Haykin, *Kernel adaptive filtering*, S. Haykin, Ed. John wiley sons, Inc., 2010.

[51] W. Maass, T. Natschlager, and H. Markram, “Computational models for generic cortical microcircuits,” *Computational Neuroscience: A Comprehensive Approach*, 2003.

[52] W. Maass, T. Natschläger, and H. Markram, “Real-time computing without stable states: A new framework for neural computation based on perturbations,” *Neural Computation*, vol. 14, pp. 2531–2560, 2002.

[53] S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, 2008.

[54] V. Z. Marmarelis, *Nonlinear Dynamic Modeling of Physiological Systems*. Wiley-IEEE Press, 2003.

[55] N. Mesgarani, S. V. David, J. B. Fritz, and S. A. Shamma, “Influence of context and behavior on stimulus reconstruction from neural activity in primary auditory cortex.” *Journal of Neurophysiology*, vol. 102, no. 6, pp. 3329–3339, Dec. 2009.

[56] M. Mesulam, “Large-scale neurocognitive networks and distributed processing for attention, language, and memory,” *Annals of Neurology*, vol. 28, pp. 597–613, 2004.

[57] J. Moehlis, E. Shea-Brown, and H. Rabitz, “Optimal inputs for phase models of spiking neurons,” *Journal of Computational and Nonlinear Dynamics*, vol. 1, pp. 358–367, 2006.

[58] S. Monaco, G. Kroliczak, D. Quinlan, P. Fattori, C. Galletti, M. Goodale, and J. Culham, “Contribution of visual and proprioceptive information to the precision of reaching movements,” *Experimental Brain Research*, vol. 202, pp. 15–32, 2010.

[59] F. Mormann, K. Lehnertz, P. David, and C. E. Elger, “Mean phase coherence as a measure for phase synchronization and its application to the eeg of epilepsy patients,” *Physica D: Nonlinear Phenomena*, vol. 144, no. 3-4, pp. 358–369, 2000.

[60] M. A. L. Nicolelis, “Brain-machine interfaces to restore motor function and probe neural circuits,” *Nature Reviews Neuroscience*, vol. 4, pp. 417–422, 2003.

[61] M. A. L. Nicolelis and M. A. Lebedev, “Principles of neural ensemble physiology underlying the operation of brainmachine interfaces,” *Nature Reviews Neuroscience*, vol. 10, no. 7, pp. 530–540, July 2009.

[62] J. E. ODoherty, M. A. Lebedev, T. L. Hanson, N. A. Fitzsimmons, and M. A. Nicolelis, “A brain-machine interface instructed by direct intracortical microstimulation.” *Frontiers in integrative neuroscience*, vol. 3, 2009.

[63] J. E. OḊoherty, M. A. Lebedev, P. J. Ifft, K. Z. Zhuang, S. Shokur, H. Bleuler, and M. A. L. Nicolelis, “Active tactile exploration using a brain-machine-brain interface,” *Nature*, vol. advance online publication, Oct. 2011.

[64] J. E. ODoherty, M. A. Lebedev, P. J. Ifft, K. Z. Zhuang, S. Shokur, H. Bleuler, and M. A. L. Nicolelis, “Active tactile exploration using a brain-machine-brain interface,” *Nature*, vol. advance online publication, 2011.

[65] A. Paiva, I. Park, and J. Principe, “A comparison of binless spike train measures,” *Neural Computing & Applications*, vol. 19, pp. 405–419, 2010.

[66] A. R. C. Paiva, I. Park, and J. C. Principe, “A reproducing kernel hilbert space framework for spike train signal processing,” *Neural Computation*, vol. 21, pp. 424–449, 2009.

[67] I. M. Park, “Capturing spike train similarity structure:a point process divergence approach,” Ph.D. dissertation, University of Florida, 2010.

[68] I. M. Park, S. Seth, M. Rao, and J. C. Principe, “Strictly positive definite spike train kernels for point process divergences,” *Neural Computation*, accepted.

[69] K. Pearson, J. Harris, A. Treloar, and M. Wilder, “On the theory of contingency,” *American Statistical Association*, vol. 25, no. 171, pp. 320–327, Sep 1930.

[70] E. Pereda, R. Q. Quiroga, and J. Bhattacharya, “Nonlinear multivariate analysis of neurophysiological signals,” *Progress in Neurobiology*, vol. 77, pp. 1–37, 2005.

[71] J. Pillow, Y. Ahmadian, and L. Paninski, “Model-based decoding , information estimation , and change-point detection in multineuron spike trains,” *Statistics*, pp. 1–27, 2006.

[72] G. Pipa, D. W. Wheeler, W. Singer, and D. Nikolić, “NeuroXidence: reliable and efficient analysis of an excess or deficiency of joint-spike events.” *Journal of computational neuroscience*, vol. 25, no. 1, pp. 64–88, August 2008.

[73] J. Platt, “A resource-allocating network for function interpolation,” *Neural Comput.*, vol. 3, no. 2, pp. 213–225, 1991.

[74] J. Principe, J. W. Fisher, and D. Xu, *Unsupervised Adaptive Filtering*. Wiley, 2000, ch. Information Theoretic Learning, pp. 265–319.

[75] R. Q. Quiroga, A. Kraskov, T. Kreuz, and P. Grassberger, “Performance of different synchronization measures in real data: A case study on electroencephalographic signals,” *Physical Review E*, vol. 65, no. 4, pp. 903–941, Mar 2002.

[76] H. Ramlau Hansen, “Smoothing counting process intensities by means of kernel functions,” *The Annals of Statistics*, vol. 11, no. 2, pp. pp. 453–466, 1983.

[77] M. J. Rasch, A. Gretton, Y. Murayama, W. Maass, and N. K. Logothetis, “Inferring spike trains from local field potentials,” *journal of neurophysiology*, vol. 99, pp. 1461–1476, 2007.

[78] J. Rigosa, D. J. Weber, A. Prochazka, R. B. Stein, and S. Micera, “Neuro-fuzzy decoding of sensory information from ensembles of simultaneously recorded dorsal root ganglion neurons for functional electrical stimulation applications.” *Journal of Neural Engineering*, vol. 8, no. 4, p. 046019, 2011.

[79] M. Rosenblum, A. Pikovsky, J. Kurths, C. Schafer, and P. A.Tass, *Phase synchronization: from theory to data analysis*. Elsevier Science, 2001, ch. 9, pp. 279–321.

[80] Y. Roudi, S. Nirenberg, and P. E. Latham, “Pairwise maximum entropy models for studying large biological systems: When they can work and when they can’t,” *PLoS Comput Biol*, vol. 5, no. 5, 05 2009.

[81] M. S. Roulston, “Estimating the errors on measured entropy and mutual information,” *Physica D: Nonlinear Phenomena*, vol. 125, pp. 285–294, 1999.

[82] M. Rubinov, S. A. Knock, C. J. Stam, S. Micheloyannis, A. W. Harris, L. M. Williams, and M. Breakspear, “Small-world properties of nonlinear brain activity in schizophrenia,” *Human Brain Mapping*, vol. 30, p. 403416, 2009.

[83] J. Sanchez and J. C. Principe, *Brain Machine Interface Engineering*. Morgan and Claypool, 2007.

[84] J. C. Sanchez, “From cortical neural spike trains to behavior: Modeling and analysis,” Ph.D. dissertation, University of Florida, 2004.

[85] B. Schlkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MA, Ed. MIT Press, 2001.

[86] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[87] B. Schölkopf and A. J. Smola, *Learning with kernels : support vector machines, regularization, optimization, and beyond*, ser. Adaptive computation and machine learning. MIT Press, 2002.

[88] S. Schreiber, J. Fellous, D. Whitmer, P. Tiesinga, and T. Sejnowski, “A new correlation-based measure of spike timing reliability,” *Neurocomputing*, vol. 52-54, pp. 925 – 931, 2003.

[89] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, “Brain-machine interface: Instant neural control of a movement signal,” *Nature*, vol. 416, no. 6877, pp. 141–142, Mar. 2002.

[90] L. Shpigelman, Y. Singer, R. Paz, and E. Vaadia, “Spikernels: Embedding spiking neurons in inner product spaces,” in *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

[91] ——, “Spikernels: Predicting arm movements by embedding population spike rate patterns in inner-product spaces,” *Neural Computation*, vol. 17, no. 3, pp. 671–690, March 2005.

[92] A. J. Smola and B. Schölkopf, “A tutorial on support vector regression,” *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[93] D. L. Snyder and M. I. Miller, *Random Point Processes in Time and Space*. Springer-Verlag, 1991.

[94] R. Sokal and F. Rohlf, *Biometry: The principles and practice of statistics in biological research*, 3rd ed. New York, 1995.

[95] R. Storchi, A. G. Zippo, G. C. Caramenti, M. Valente, and G. E. M. Biella, “Predicting spike occurrence and neuronal responsiveness from lfps in primary somatosensory cortex,” *PLoS ONE*, vol. 7, no. 5, p. e35850, 5 2012.

[96] P. Tass, M. G. Rosenblum, J. Weule, J. Kurths, A. Pikovsky, J. Volkmann, A. Schnitzler, and H. J. Freund, “Detection of $n : m$ phase locking from noisy data: Application to magnetoencephalography,” *Phys. Rev. Lett.*, vol. 81, no. 15, pp. 3291–3294, Oct 1998.

[97] D. M. Taylor, S. I. Tillery, and A. B. Schwartz, “Direct cortical control of 3d neuroprosthetic devices,” *Science*, vol. 296, no. 5574, pp. 1829–1832, Jun. 2002.

[98] UC Berkeley Visualization Lab, 2009.

[99] F. Varela, J. P. Lachaux, E. Rodriguez, and J. Martinerie, “The brainweb: phase synchronization and large-scale integration,” *Nature Reviews Neuroscience*, vol. 2, pp. 229–239, 2001.

[100] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, “Cortical control of a prosthetic arm for self-feeding.” *Nature*, vol. 453, no. 7198, pp. 1098–101, 2008.

[101] F. W. Volterra, R. Kelly, and T. S. Lee, “Decoding v1 neuronal activity using particle filtering with volterra kernels,” in *Advances in Neural Information Processing Systems*, 2003, pp. 15–1359.

[102] H. Wang, D. Pi, and Y. Sun, “Online svm regression algorithm-based adaptive inverse control,” *Neurocomput.*, vol. 70, pp. 952–959, 2007.

[103] Y. Wang, “Point process monte carlo filtering for brain machine interface,” Ph.D. dissertation, University of Florida, 2008.

[104] D. K. Warland, P. Reinagel, M. Meister, D. K, and P. Reinagel, “Decoding visual information from a population of retinal ganglion cells,” *J. Neurophysiol*, vol. 78, pp. 2336–2350, 1997.

[105] B. Widrow and E. Walach, *Adaptive Inverse Control*, E. Cliff, Ed. Prentice-Hall, 1995.

[106] D. Xing, C. Yeh, and R. Shapley, "Spatial spread of the local field potential and its laminar variation in visual cortex." *Journal of Neuroscience*, vol. 29, pp. 11 540 – 11 549, 2009.

[107] S. Yamada, K. Matsumoto, M. Nakashima, and S. Shiono, "Information theoretic analysis of action potential trains ii. analysis of correlation among yt neurons to deduce connection structure," *Journal of neuroscience methods*, vol. 66, pp. 35–45, 1996.

[108] J. H. Zar, *Biostatistical analysis*, T. Ryu, Ed. Prentice Hall, 1999.

BIOGRAPHICAL SKETCH

Lin Li received the B.E. degree in electrical and information engineering from University of Science and Technology, Beijing, China, in 2007, the M.E. degree in electrical and computer engineering from University of Florida, Gainesville, in 2010. She is currently pursuing the Ph.D. degree in electrical and computer engineering at University of Florida, Gainesville. She has been working in the Computational NeuroEnginnering laboratory at the University of Florida under the supervision of Dr. J.C. Principe since 2009. Her research interests are machine learning, statistical modeling, and computational neuroscience.