Algorithms for Neural Prosthetic Applications

by

Subash Padmanaban

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2017 by the
Graduate Supervisory Committee:

Bradley Greger, Chair
Marco Santello
Stephen Helms Tillery
Antonia Papandreou-Suppappola
Sharon Crook

ARIZONA STATE UNIVERSITY

May 2017

ABSTRACT

In the last 15 years, there has been a significant increase in the number of motor neural prostheses used for restoring limb function lost due to neurological disorders or accidents. The aim of this technology is to enable patients to control a motor prosthesis using their residual neural pathways (central or peripheral). Recent studies in non-human primates and humans have shown the possibility of controlling a prosthesis for accomplishing varied tasks such as self-feeding, typing, reaching, grasping, and performing fine dexterous movements. A neural decoding system comprises mainly of three components: (i) sensors to record neural signals, (ii) an algorithm to map neural recordings to upper limb kinematics and (iii) a prosthetic arm actuated by control signals generated by the algorithm. Machine learning algorithms that map input neural activity to the output kinematics (like finger trajectory) form the core of the neural decoding system. The choice of the algorithm is thus, mainly imposed by the neural signal of interest and the output parameter being decoded. The various parts of a neural decoding system are neural data, feature extraction, feature selection, and machine learning algorithm. There have been significant advances in the field of neural prosthetic applications. But there are challenges for translating a neural prosthesis from a laboratory setting to a clinical environment. To achieve a fully functional prosthetic device with maximum user compliance and acceptance, these factors need to be addressed and taken into consideration. Three challenges in developing robust neural decoding systems were addressed by exploring neural variability in the peripheral nervous system for dexterous finger movements, feature selection methods based on clinically relevant metrics and a novel method for decoding dexterous finger movements based on ensemble methods.

To

Amma, Bapa, Sashaank, and Avva

"In God we trust, all others must bring data"

- William Edwards Deming

"Data! Data! Data!" he cried impatiently. "I can't make bricks without clay"

- Sir Arthur Conan Doyle
The Adventure of the Copper Beeches

"Not everything that can be counted counts,
and not everything that counts can be counted"

- Albert Einstein

"The human brain named itself"

-My favorite stupid brain fact

ACKNOWLEDGMENTS

I was fortunate enough to have spent graduate school as a member of the Neural Engineering Laboratory under the advisement of Bradley Greger. Dr. Greger has always fostered a work environment built around the idea of asking big questions and guiding us to reach an elegant solution. Dr. Greger, I thank you for giving me the freedom, encouragement, and unwavering support throughout my PhD and helping me reach this milestone in my life. I also thank you for inspiring and teaching me to be a better researcher through your actions and words of wisdom.

My committee members have played a crucial role in shaping my research and adding rigor to the analysis. Dr. Santello, Dr. Helms Tillery, Dr. Papandreou-Suppappola, and Dr. Crook, I thank you for taking time out from your busy schedule to help me stay focused. It truly has been a humbling and rewarding experience learning from your feedback. I would be remiss if I don't thank Laura Hawes for her help throughout graduate school. Thank you for your patience and advise in helping me through all the academic deadlines.

Mom and Dad, I'm lucky to have parents like you for being with me through tough times and providing constant encouragement. I also thank my brother, Sashaank, for being with me through the thick and thin of graduate school.

It has been a pleasure working with my lab mates Kevin and Cody. Kevin, thank you for lending your ears for the brainstorming sessions and I really appreciate your

candor in helping me get to better solutions. I would also like to thank my friends for keeping me grounded to reality and being a much-needed source of distraction.

TABLE OF CONTENTS

Page

LIST OF TABLES vii

LIST OF FIGURES viii

MACHINE LEARNING SYSTEMS FOR NEURAL PROSTHETIC APPLICATIONS: TRENDS AND CHALLENGES FOR DECODING UPPER LIMB MOVEMENTS 1

Introduction 1

Feature Extraction 5

Feature Selection 13

Decoding Methods 15

Factors Affecting Quality of Decode 24

Future Directions and Other Considerations 29

FEATURE SELECTION 34

Abstract 34

Introduction 35

Methods 41

Results 50

Discussion 58

ENSEMBLE METHODS 62

Abstract 62

Introduction 62

Materials and methods 64

Results 79

Discussion 84

NEURAL VARIABILITY 87

Abstract 87

Introduction 88

Page

Methods 89

Results 93

Conclusion 101

REFERENCES 102

LIST OF TABLES

Table Page

1. Feature Selection Algorithms and their Respective Optimal Number of Features on Post-Implantation Day 36 53

2. Chronic Decoding Results 83

LIST OF FIGURES

Figure		Page
1.	Univariate and Multivariate Gaussian Distribution	38
2.	Curse of Dimensionality	39
3.	Neural Decoding System Architecture	42
4.	Raw Neural Recording on Channel 87	50
5.	Mean of Isolated Single-Units Neural Activity on Channel 87	51
6.	Selecting Optimal Number of Multi-Unit Features	52
7.	Selecting Optimal Number of Single-Unit Features	52
8.	Accuracy of Neural Decode on Post-Implantation Day 36	54
9.	Robustness to Simulated Failure of Multi-Unit Features	55
10.	Robustness to Simulated Failure of Single-Unit Features	55
11.	Longevity of Neural Decodes Using Multi-Unit Features	56
12.	Longevity of Neural Decodes Using Single-Unit Features	57
13.	One Standard Error Rule	75
14.	Flow Chart of Decoding Analysis	77
15.	Raw Neural Recording on Channel 54	79
16.	Parameter Selection for Optimal Lambda	80
17.	Kalman Filter Regression	81
18.	Ensemble Regression	82
19.	Chronic Decoding Results	83
20.	Performance of Decoding Algorithms	84

Figure Page

21. Correlation Coefficient for All Channel Pairs 94

22. Mean Firing Rate of Each Finger Movement 94

23. Temporal Variability Across Trials 95

24. Peak and Central Firing Rates.....96

25. Mean Time Difference Between Central and Peak Firing Rates 97

26. Normalized Variance of Firing Rate.....98

27. Mean and Variance of Firing Rate For 6 Random Trials 99

28. Neural Variability on Stimulus Onset.....99

29. Relationship Between Mean and Variance 100

Machine Learning Systems for Neural Prosthetic Applications: Trends and Challenges for Decoding Upper Limb Movements

Introduction

In the last 15 years, there has been a significant increase in the number of motor neural prostheses used for restoring limb function lost due to neurological disorders or accidents. The aim of this technology is to enable patients to control a motor prosthesis using their residual neural pathways (central or peripheral). Recent studies in non-human primates and humans have shown the possibility of controlling a prosthesis for accomplishing varied tasks such as self-feeding, typing, reaching, grasping, and performing fine dexterous movements (Carrozza et al., 2006; Cipriani, Controzzi, & Carrozza, 2011; Collinger et al., 2013; Collinger et al., 2014; Davis, Wark, Hutchinson, Warren, O'Neill, Scheinblum, Clark et al., 2016b; Gilja, Nuyujukian, Chestek, Cunningham, Byron, Fan, Ryu et al., 2012; Velliste, Perel, Spalding, Whitford, & Schwartz, 2008). A neural decoding system comprises mainly of three components: (i) sensors to record neural signals, (ii) an algorithm to map neural recordings to upper limb kinematics and (iii) a prosthetic arm actuated by control signals generated by the algorithm. Machine learning algorithms that map input neural activity to the output kinematics (like finger trajectory) form the core of the neural decoding system. The choice of the algorithm is thus, mainly imposed by the neural signal of interest and the output parameter being decoded.

Neural signals.

There are two types of neural signals mainly used for designing neural prosthetic systems. They are Local Field Potentials (LFP) and Action Potentials. LFP and AP are electrophysiological signals of the nervous system that contain information related to the task being performed. In this review, we surveyed neural prostheses powered by invasive neural signals like LFP and AP. Although recent researches have shown advances in motor prostheses based on Electroencephalography (surface EEG) and Electromyography (EMG), they are not in the scope of this review (Ison & Artemiadis, 2014; Tehovnik, Woods, & Slocum, 2013). LFPs can be recorded by using Intracranial Electroencephalography (iEEG) or Electrocorticography (ECoG or sometimes macro ECoG/standard ECoG) and micro Electrocorticography (micro ECoG).

Local field potential or synchronized postsynaptic potential is an electrophysiological signal generated by the spatial summation of electric current flowing from multiple nearby neurons within a small volume of nervous tissue. LFPs are continuous signals and are believed to carry event related neural information in the 0.2-300 Hz frequency range. Macro ECoG and micro ECoG electrodes have similar operating procedures. The electrodes are placed on the surface of the cortex, sufficiently farther away from the local extracellular space to avoid any individual neuron dominating the recorded signal. A macro ECoG electrode has 3mm contact area diameter and 10mm center-to-center distance while a micro ECoG electrode has 1.5mm in contact area diameter and 4mm center-to-center distance. Due to the difference in size of the electrodes, micro ECoG electrodes provide smaller spatial resolution (area around the tip of the electrode from

where neural signals are recorded) than macro ECoG electrodes. Micro ECoG electrodes are usually available as microelectrode arrays (in a 4x4 grid).

An action potential is an electrophysiological signal generated due to the rapid change in electrical membrane potential of a neuron. APs (discrete events often referred to as "spikes") are point processes and information related to an event (for example, reaching and grasping an object) is encoded through rate coding and temporal coding. The rate coding model states that the rate or frequency of action potential "spiking" is directly proportional to the intensity of a stimulus. With respect to upper limb movements like reaching and grasping, fast and slow type 1 and 2 sensory afferents encode information about manipulated objects (Johansson & Flanagan, 2009). The rate coding model of grasping states that dedicated sensory afferents encode information about various modalities of touch information such as texture, position, movement, and temperature. Based on the rate coding model, neural spike-count rate is a commonly extracted feature. Temporal coding model of neuronal firing states that precise spike-timing based fluctuations encode information about a stimulus. These fluctuations occur in the range of milliseconds. Action potentials are recorded using microelectrode arrays (MEA). The MEA is a 10x10 grid of 1 mm tall electrodes that are capable of recording action potentials in addition to local field potentials (Maynard, Nordhausen, & Normann, 1997).

Motor regions for decode.

Applications of cortical control of a motor prosthesis are designed to record from regions of the cortex that help plan and execute volitional upper limb movements. Cortical

control of a motor prosthesis has been shown possible by recording signals from the motor cortex, premotor cortex, sensorimotor cortex, and parietal cortex (Chestek et al., 2013a; Churchland, Yu, Ryu, Santhanam, & Shenoy, 2006; Klaes et al., 2015; Ohara et al., 2000; K. V. Shenoy, Kaufman, Sahani, & Churchland, 2011). Peripheral control of a motor prosthesis has been shown possible by using neural signals from median and ulnar nerve (Ciancio et al., 2016; Dhillon & Horch, 2005a; Micera, Carpaneto, & Raspopovic, 2010; Navarro et al., 2005). Although these experiments show the feasibility of controlling a motor prosthesis with comparable accuracies, further investigation might be required to compare the reliability of chronic decodes using neural signals originating from the various motor regions. Previous research suggests that neural signals originating in various motor regions might correspond to intention, planning, and execution of volitional motor movements (Schwartz, 2007). In addition to decoding upper limb kinematics, offline decodes have also been used to decode planning and expected reward of a trial (Musallam, Corneil, Greger, Scherberger, & Andersen, 2004).

System architecture.

In neural prosthetic systems, neural data is recorded using an appropriate sensor such as standard ECoG electrodes or a microelectrode array (MEA). The recording electrodes are implanted typically in a motor related area such as the motor cortex, sensorimotor cortex, or parietal cortex in the brain and median or ulnar nerve in the hand. The raw neural data are then high pass filtered and processed using a spike-sorting algorithm, such as PCA based spike-clustering to identify neuronal action potentials. The identified action potentials are used to generate a spike-count rate or firing rate by using a

moving window approach (Cunningham, Gilja, Ryu, & Shenoy, 2009). The raw neural data are low pass filtered to obtain the local field potentials. The raw voltages of local field potentials (sometimes referred to as *locomotor potential*) is used as a feature. This process is repeated for obtaining firing rates and local field potentials of neural signals from individual electrodes. A subset of available channels of neural data features (firing rate and/or local field potentials) is then selected by using a feature selection method such as Mutual Information Maximization or Wilcoxon signed-rank test. The neural data features are then used as input to the decoder to generate control signals for actuating a prosthesis.

The signal of interest and end-goal of decoding mainly drive the system architecture design (Marathe & Taylor, 2011). The signal of interest can be either action potentials, local field potentials or other features extracted from the raw neural signals. End-goal of decoding can be reaching, grasping, typing, or performing fine, dexterous finger depending on the objective of the experiment. The output of the neural decoding system will be discrete if the end-goal is to predict discrete finger movements (for example, typing) and continuous if the end-goal is to predict trajectory of wrist or fingers (for example, grasping). The process of predicting discrete outputs is called classification while, predicting continuous outputs is called regression. Therefore, decoding end-goals impact in selecting specific algorithms that can handle either classification or regression problems. On choosing to decode using either a classification or regression algorithm, experimenters also must identify specific quantities to decode such as position, velocity, or goal. Recent research suggests there is a significant improvement in performance of decoders by adding decoding errors as input. Decoding combinations between position, velocity and goal

yielded improvement in performance in certain pairs of transformations (Marathe & Taylor, 2011).

Feature extraction

Data represented as inputs can be continuous, categorical, or ordinal. Neural data from MEA sampled at 30 kHz contains the instantaneous raw voltage for each channel. Inputs supplied to the machine learning algorithm to decode an end-goal is often referred to as features. Features is synonymous to attributes or inputs. Human expertise is required to remove unwanted noise and extract useful information from data to enhance the performance of a machine learning algorithm. The process of filtering data (to remove noise) and extracting new information from the raw data using some domain expertise is called feature extraction (also called feature engineering or feature construction). The process of low pass filtering raw neural data to obtain LFP potentials is an example of feature extraction. The process of high pass filtering raw neural data, performing spike sorting and computing the spike-count rate or firing rate is also an example of feature extraction. There are algorithms that can perform automatic feature extraction. Artificial neural networks have been proven to be efficient for neural decoding (Sussillo et al., 2012; Tenore, Aggarwal, White, Schieber, & Thakor, 2009). The weights computed in the hidden layers of the artificial neural network can be inferred as automatic features extracted from the raw neural data. In this section, we examine the role of feature extraction and its impact on the performance of the machine learning algorithm.

Pre-processing.

Input features play a key role in influencing the performance of a machine learning algorithm. The process of removing noise and unwanted data from the raw data is called pre-processing (also referred to as data cleaning or data preparation). Pre-processing entails preparing the raw data for feature extraction. Depending on the analysis, neural data is often subjected to pre-processing steps such as standardization, normalization, signal enhancement, non-linear expansions, and feature transformation.

Standardization is the process of changing the scales of each input feature such that they all have similar range. Different inputs can have different range. For example, raw voltage of neural data typically has a range of -2000 to + 2000 μV, neural firing rate has a range of 0 to 200 Hz, and the power spectral density of neural data has a range of 0 to 1. If the feature set used for decoding contains features that have a significant difference in range, the features with higher values might create an undesirable bias on the machine learning algorithm. Input features can be standardized by centering the data and scaling it. This process can be accomplished by subtracting the mean of an input feature from all data points and dividing by standard deviation for a feature.

Normalization is like standardization in its objective to convert all input features to a set range (for example, -1 to +1). There are many methods to normalize a given dataset. One of the widely-used methods to normalize features include dividing all data points of a feature by its maximum value.

Signal enhancement is the process of improving the signal-to-noise ratio of a signal by filtering. Popular operations of signal enhancement include baseline referencing

(common average referencing), de-noising, smoothing or sharpening. Fourier transforms and wavelet transforms are widely used for signal enhancement (cite wavelet and Fourier denoising papers).

Non-linear expansions refer to the process of including higher order terms of the input features to account for non-linear interactions between features. Although, feature selection methods employed in later stages aim at reducing the dimensionality of the feature space, some problems are complex where first order interactions are not sufficient and require the inclusion of higher order terms. For a given feature x, non-linear expansions can be as simple as including x^2, x^3, … x^n terms. Kernel methods in machine learning (such as non-linear kernels in SVM) perform the same operation within the workings of the algorithm. Polynomial and radial basis function (Gaussian) kernels are popular kernel choices, that expand the input features to a higher dimension by non-linear expansions.

Feature transformation is the process of applying transform functions to the input features to represent them in some higher dimensional space. Fourier transform, Hilbert transform, and Principal Component Analysis are some examples of popular transformation techniques. The inherent structure of the transform functions help represents and visualize data in a different dimension which in turn, might be useful for achieving better decodes. For example, Principal Component Analysis is an orthogonal transformation where the principal components capture the variance of the input features in descending order (Jolliffe, 2002).

Statistical measures of raw neural data can also be used as a feature extraction technique. Computing the mean, variance and other higher order moments using a moving

window approach has been widely used in myoelectric prostheses. The application of such statistical measures is yet to be examined for cortical neural prostheses (Guyon & Elisseeff, 2006).

Amplitude, frequency, and phase are three attributes that help characterize a signal. Feature extraction techniques for neural decoding exploit all three attributes to represent information pertaining to upper limb kinematics.

Time domain features.

Time domain features include features extracted from the raw neural data that represent amplitude of the signal. The most commonly used time domain feature for developing cortical neural prosthesis is the neural firing rate (spike-count rate). The steps involved in computing the firing rate from raw neural data is as follows:

1. Filter the data using a high pass filter. Typical cut-off frequencies lie in the range of 1000-2000 Hz.
2. Set a threshold based on some signal-noise metric such as signal-to-noise ratio (SNR). The threshold is set as a scalar multiple of SNR depending on the mean amplitude of probable action potentials (for example, 3-10 times SNR).
3. Detect peaks lying above the set threshold.
4. Extract action potential snippets around the detected peaks. Perform Principal Component Analysis on extracted snippets.

5. Identify distinct clusters by examining the first three principal components using a clustering algorithm like k-means clustering (cite k-means). The cluster centered around the origin is the noise cluster which has similar characteristics to neuronal firing.

6. Label data points belonging to the non-noise cluster as neuronal firing and extract the time stamps of firing. Depending on the principal component analysis clustering, there might be more than one neuronal cluster.

7. Perform a moving window average of the number of neuronal firing i.e., count the number of "spikes" or action potentials in each window. This moving window average represents the neural firing rate of a channel. Repeat the same procedure for all channels.

Neural firing rate is a time domain feature based on action potentials. Time domain features can also be extracted based on local field potentials. The most commonly used time domain feature based on local field potentials is the raw voltage of bandpass filtered signal. The steps involved in computing the local field potential is as follows:

1. Filter the data using a low pass filter with a cut-off frequency of 300-350 Hz.

2. The low pass filtered neural signal corresponds to local field potential.

Factors affecting neural firing rate computation. Spike-sorting vs. thresholding: Christie et al analyzed the effect of various spike-sorting methods and thresholding for

decoding reach direction and hand position in two macaques using Naïve-Bayes algorithm and linear regression respectively. The performance of the decoders was analyzed over 58 sessions by using neural data after spike-sorting and only thresholding. Thresholding refers to the process of setting a threshold and treating all peaks detected above the threshold as an action potential and thus, evading the need for spike-sorting. The results indicated that the performance of the decoding algorithms after spike-sorting was significantly better than thresholding for one monkey but not the other. The optimal range for setting the thresholds for this analysis was identified to be between -3 and -4.5 V_{rms} (Christie et al., 2014).

Effect of spike-sorting algorithms: The performance of a machine learning algorithm for a task using neural firing as an input feature is dependent on the accuracy of the spike-sorting method. Error analysis of sorted action potentials reveals important information about the underlying neuronal populations and oftentimes requires manual guidance to label spike clusters. *Cluster cutting* is one of the widely-used spike-sorting methods based on PCA and k-means clustering. *Bayesian clustering* is the process of modeling action potentials as a multivariate Gaussian distribution using a Maximum Likelihood Estimation. Probable action potentials are assigned a cluster based on the probability and distance from adjacent clusters. Since this method assigns a cluster to an action potential based on probability, it is possible to get confidence intervals of spike classification. *Wavelet based spike sorting* is an unsupervised spike-sorting method based on wavelet coefficients rather than Principal Component Analysis. This method provides better temporal resolution for detection action potentials due to the time scaling property of the wavelet decomposition. *Mixtures of t-distribution based spike sorting* also offers a robust, automatic spike-sorting technique. Action potentials do not follow a Gaussian

distribution and can be efficiently modeled using mixtures of t-distribution. This method combines an Expectation-Maximization method along with the mixtures of t-distribution to sort multi-unit spikes recorded on the same channel (Letelier & Weber, 2000; Lewicki, 1998a; Shoham, Fellows, & Normann, 2003).

Multi-unit vs. Single-unit features: In an MEA, there is a possibility for an electrode to record from more than one underlying neuron. The probability that the underlying neurons encode for the same task (such as same direction or same finger movement) is random. Therefore, there might be a need for treating action potentials from these underlying neurons as individual sources of information. Sorting multi-unit neurons is a computationally intensive procedure and adds to possible lags or delay due to computation during real-time decode. The alternative to this problem is to treat all underlying neurons in an electrode as a single source of information and compute the firing rate. Further research is required to analyze the effect of multi-unit and single-unit based firing rates on machine learning algorithms (Perge et al., 2014).

Frequency domain features.

Frequency domain features are typically used to extract the power spectral density of local field potentials. Different bands of frequencies have been defined in the total frequency range of the local field potentials, with research findings validating the role and correlation of specific frequency bands on various tasks performed. The various frequency bands defined in LFPs are: 0.3 – 2 Hz (δ), 2 – 7 Hz (θ), 7 – 15 Hz (α), 15 – 30 Hz (β), 30 – 60 Hz (γ), 60 – 100 Hz (high γ), 100 – 200 Hz (MUA 1), and 200 – 400 Hz (MUA 2). The narrow band frequencies alpha, beta, gamma, theta, and high gamma correspond to

raw LFP signals. The MUA 1 and MUA 2 frequency bands correspond to multi-unit activity of action potentials. The frequency domain features are extracted by performing a spectrogram analysis of the raw LFP signals. The power spectra of the frequency bands are averaged and used as individual features (Rule, Vargas-Irwin, Donoghue, & Truccolo, 2015).

Phase domain features.

Phase domain features are used to extract information about the lag or lead of one channel over the other. Temporal coding model of neural firing states that the precise action potential timing and fluctuations are useful in neural coding with a temporal resolution of milliseconds. Phase domain features exploit these neuronal fluctuations and interactions that can help potentially decode motor movements. Correlation, covariance, and phase lag index are some of the phase domain features that have been used in studying functional connectivity of various cortical and deep brain structures (Kirsch, Rivlis, & Schieber, 2014; Stam, Nolte, & Daffertshofer, 2007).

Feature selection

High-dimensional data such as neural data is subjected to the curse of dimensionality. Curse of dimensionality refers to the problems that arise while dealing with high-dimensional data. When the number of features increases, the sparsity of data

increases. This increase in sparsity of data is detrimental to many machine learning algorithms that do not scale well in high-dimensional feature space. Feature selection algorithms help select a subset of informative features from the original feature space based on some metric. Feature selection methods reduce redundancy of features and help increase the information density of the feature space (Kohavi & John, 1997a).

Feature selection in neural decoding starts with exploratory data analysis. The first step in exploratory neural data analysis is to extract trial segments based on average trial duration. Exploratory data analysis provides more insight about the underlying relationship between the neural data and the end-goal that might be valuable in designing various parts of the machine learning system. Some of the widely-used graphing and visualizing techniques in neural data processing include tuning curves, histograms, and Peri-Stimulus Time Histogram (PSTH). *Tuning curves* can be visualized by plotting the neural data to various possibilities of the end-goal (for example, all possible values of direction in a reaching task). Tuning curves are best used for visualizing neural representations for reaching and grasping tasks. *Histograms* are statistical tools used for visualizing the number of occurrences as a function of neural firing rate. The first step in understanding the range and central tendency of neural data is to visualize histograms for all possibilities of end-goal for each channel or neuron. *Peristimulus Time Histogram (PSTH)* are histograms of timings of neural firing. Like histograms, visualizing PSTH requires plotting timings of neuronal firing for all possibilities of end-goal for each channel or neuron. PSTH plots can also be used to visualize trial averaged neural firing rates and understand the correlation and variance of the underlying neural representations for a task (Cunningham et al., 2009).

There are two major types of feature selection methods. They are filter or criteria based feature selection and wrapper based feature selection. *Filter* or *Criteria based feature selection* refers to feature selection methods that rank features based on some criterion. By ranking these features, the feature selection methods help "filter" features that are uninformative or redundant. Criteria based feature selection methods are computationally less intensive. But, they do require domain knowledge of the problem for applying the right criteria based method and choosing the correct metric. Few examples of criteria based feature selection methods include student's t-test, Wilcoxon signed-rank test, Mutual Information Maximization, and Principal Component Analysis.

Wrapper based feature selection refers to feature selection methods that rank features based on its performance on a validation set using a machine learning algorithm. Wrapper based methods can be embedded in the working of an already existing machine learning algorithm. Wrapper based methods rank features by iteratively evaluating the performance of a combination of subset of the features. These feature selection methods are computationally intensive but require no domain knowledge of the problem.

Decoding methods

Many decoding methods have been used in actuating an upper limb prosthesis. Decoding algorithms can be classified based on various factors such as a classifier or a regression algorithm based on the output variable, linear or non-linear method based on the underlying assumptions of the hypothesis and so on.

Linear models.

Linear methods for decoding are algorithms or models that assume a linear relationship between the input (neural data) and output (upper limb kinematics). Linear regression is a common method of regression, with an objective to fit a straight line to the input data points. A simple linear regression is a problem where a dependent variable 'y' is modeled as a function of an independent variable 'x'. When the number of independent variables used in the model is greater than one, it is called a multivariate linear regression. A straight line can be modeled using the equation $y = mx + b$, where 'm' is the slope of the straight line and 'b' is the y-intercept. Since the goal of linear regression is to fit a straight line, the problem is solved by estimating an optimal value for the model parameters slope and y-intercept. *Ordinary least squares* is a popular technique used in estimating the model parameters by iteratively minimizing the sum of squared error between possible values of 'm' and 'b'. *Generalized least squares* is an extension of the ordinary least squares method with less stringent assumptions about the data (for example, allowing heteroscedasticity and correlations between the input variables).

Ridge regression is an extension of linear regression accounting for the multicollinearity of input variables. Ridge regression is modeled by introducing an error term in the straight-line equation and a shrinkage term. *Logistic regression* is a linear, binary classification model. The decision for classifying a point is made using a sigmoid function. In neural prosthetic applications, a linear regression model was used for decoding continuous limb movements in a non-human primate. The end-goal of decoding in this experiment was the limb position. The online decoding was performed using epidural field

potentials. Bougrain et al also used a linear regression model to decode finger flexions in human subjects based on electrocorticogram (ECoG) signals. Depending on the signals and end-goal of decoding, a simple linear regression model can be used to model complex neural representations for decoding upper limb movements (Liang & Bougrain, 2012; van Gerven, Chao, & Heskes, 2012).

State-space models.

A state-space representation is a mathematical model of a system as a set of input, output, and state transition variables. Algorithms represented through state-space models can be used for modeling the system and its output in an iterative manner. Kalman Filter is the most popular state-space model used widely in signal processing and econometrics. Kalman Filter, also known as linear quadratic estimation algorithm, works on the principle that input and output measurements made with noise and inaccuracies can be efficiently modeled by constantly updating the state-transition variables based on the error of the previous prediction. The Kalman Filter is a recursive filter with a predict and update rule in each step. In the predict step, the Kalman Filter predicts the output of a system based on prior and current inputs and prior outputs. In the update step, a Kalman gain variable is used to update the system relationship based on past predicted output and past true output. Kalman Filters have been used in neural prosthetic applications for controlling a motor prosthesis. Neural firing rate recorded from intracortical recordings in humans has been used to decode movement velocity using a Kalman Filter. This study observed a correlation

coefficient of 0.99 between the decoded movement velocity and true movement velocity (Velliste, McMorland, Diril, Clanton, & Schwartz, 2012).

Kalman Filters have also been used for decoding three-dimensional reach and grasp using features extracted from neural firing rate and LFP potentials. The features were extracted from neural signals recorded in the primary motor cortex and ventral premotor cortex of a free-behaving monkey. In this study, it was found that neural firing rate features from the motor cortex was the preferred feature for decoding. In addition, LFPs and combined signals from the motor cortex and ventral premotor areas were used added robustness to the brain-computer interface (Bansal, Vargas-Irwin, Truccolo, & Donoghue, 2011).

One of problems observed when using a Kalman Filter based decoder is the stability of decoding: with fixed training parameters, the performance of the algorithm degrades over time. A stabilized dual Kalman Filter was proposed as an alternative to the simple Kalman Filter (Zhang & Chase, 2013). The dual Kalman Filter was a modified version of the simple Kalman Filter by adding a parameter vector to each neuron to track the evolution of tuning curves over time. This accounted for the non-stationarity and evolution of neural representations that were impeding the performance of a simple Kalman Filter. The dual Kalman Filter with fixed parameters showed improved performance over five consecutive days compared to a simple Kalman Filter which was retrained every day. An Extended Kalman Filter is a modified Kalman filter with non-linear state transition variable. This non-linearity can help account for complexities in the relationship between the neural data and the observed hand kinematics. An Extended Kalman Filter was used to map neural

firing rate to a high degree of freedom output space for controlling a prosthetic arm (Schwartz 2012). Real time control of finger position was achieved using an Extended Kalman Filter with an ability to actuate up to 3 degrees of freedom simultaneously.

Feedback intentions have also been incorporated in the system equations of a simple Kalman Filter. Gilja et al proposed a recalibrated feedback intention-trained Kalman Filter which had novel modifications to the simple Kalman Filter. 50ms bins of neural data were used to generate neural firing rate features from data recorded from the primary motor cortex and dorsal premotor area of a free-behaving monkey. This study proposed an innovative parameter fitting approach to train a Kalman Filter to account for the differences in user intention during offline and online control. Once the training data was recorded, the kinematics of the arm were transformed to a target-centric coordinates where the position of the cursor was labeled as zero when it was on the target. This study also proposed a novel training approach that accounted for high-frequency jitters in online decode. Both position and velocity estimates were decoded simultaneously using neural firing rate for a reaching task while an uncertainty term 'W' was introduced in the system update equations to account for the high frequency jitters. The results showed that it was possible to obtain near-perfect decodes during online control with a success rate of 83 – 100 % (Gilja et al., 2012).

Kim et al compared the use of linear filters such as Wiener Filter, LMS adaptive filters, Gamma Filter, and subspace Wiener Filters for a reaching and target hitting task in a non-human primate. Neural firing rates from 100 – 200 neuronal ensembles were extracted and used as input features for the linear filters. The performance of the linear

filters was compared based on the Correlation Coefficient during movement and rest for a 2D and 3D reaching and target hitting task. The performance of all the algorithms was comparable to the Wiener Filter which was used to establish the baseline performance. These results were also similar in performance to a simple Kalman Filter (S. Kim, Sanchez, Rao, Erdogmus, Carmena, Lebedev, Nicolelis, & Principe, 2006a).

Pattern recognition models.

A Perceptron is a binary, linear supervised learning algorithm (Rosenblatt, 1958). It is a simple model used to distinguish binary classes by constructing a weighted decision boundary based on minimizing a loss function between the predicted and true values in a training set. A squared error loss function is typical is a popular choice for a wide variety of applications of a Perceptron. Gradient Descent is an iterative minimization algorithm that is used to find an optimal decision boundary while training a Perceptron. A single layer perceptron can be thought of as a building block for multi-layer perceptron or a neural network. Neural networks (also referred to as Artificial Neural Nets/Networks) have input, output, and hidden layers to map an input-output relationship. The number of hidden layers and hidden nodes is estimated through cross validation. Back propagation is a better alternative than Gradient Descent to training ANNs. During training, weights or transition parameters of the hidden layers are estimated using cross validation. ANNs have been widely used in developing motor prosthesis based on neural signals (Choi, Hirose, Sakurai, Iijima, & Koike, 2009; Mollazadeh et al., 2009; Sussillo et al., 2012; Tenore et al., 2009). Although ANNs have moderate to high accuracies of decoding, they are computationally

intensive to train. Future directions for ANNs are moving towards developing Deep Learning architectures that are capable of learning truly non-linear relationships between the input and output variables.

Support Vector Machines (SVM) are non-probabilistic, linear, binary instance-based classifiers which have now become an efficient and established method of neural decoding for varied applications (Platt, 1999). SVMs have also taken center stage in many other signal processing applications. SVMs represent the data in higher dimensional space and find the best separating hyperplane in this space. The objective of the SVM is to find a hyperplane that has the maximum distance from a point belonging to any class. Such a classifier is also called a maximum margin classifier whose generalization error is low. During training, each point in the training set is assigned a weight α. Those points with training weights $\alpha \neq 0$ are called the support vectors since, they help forming the hyperplane. In case of linearly non-separable cases, a soft margin classifier is implemented which allows for misclassified instances. Non-linear problems can be solved by using the "kernel trick" in the SVM. Kernel functions map data into a higher dimensional space where, the hyperplane is now formed. Since SVMs are innately binary classifiers, their application can be extended to solve multi-class classification problems by decomposing the problem to many binary problems and apply a majority voting to predict new instances. Popular methods for constructing multi-class SVMs are the one-vs-all and one-vs-one (Hsu & Lin, 2002). Support vector machines have shown promising results in upper extremity decoding tasks using various source signals such as MEG, EEG, ECoG and EMG . SVMs have been used in classifying finger movements using LFP signals collected from ECoG data from human patients (P. Shenoy, Miller, Ojemann, & Rao, 2008). In this study, the

error rate observed using an SVM was 23% for a five-finger flexion classification task. SVMs have also been used to distinguish between different grasp types in monkeys using LFP signals from the dorsal premotor area (Hao et al., 2014a). One of the significant results of this study indicated that the neural representations related to a grasp type could be decoded as early as 200 – 300ms from the time of cue. The results also showed robust, long term stable decodes from the PMd area as a viable option for neuroprosthetic applications. SVMs were used in a robotic hand based on electroneural signals recorded from Long Intra-Fascicular Electrodes (LIFE). The SVM decoder was used to distinguish three different movements: power grip, pinch grip, and flexion of little finger (Micera et al., 2011). In a different study, the optimal number of channels was also evaluated for this decoding analysis (Carpaneto et al., 2011). SVMs have also been used in classifying finger movements using LFP signals recorded from a micro-ECoG grid implanted in human subjects (Vinjamuri et al., 2009). The classification accuracy in this experiment was 73 % (level of chance: 20 %).

K-nearest neighbor (k-NN) is an instance based, multi-class classification or regression algorithm that predicts classification results by majority voting and regression results by averaging. K-NN algorithm computes the distance of data points and based on the ‘k’ nearest neighbors predicts the output of an unknown or test instance. Inverse-distance weighted k-NN algorithm produces a weighted output based on the distance between the unknown point and the neighboring points. This method accounts for relatively nearby points to have a higher influence on the decision making than points that are farther away. There are many ways to compute distance. Based on the underlying statistics of the data and with some domain knowledge, an appropriate distance measure can be used in the

k-NN algorithm. Popular distance measures include Euclidean distance, Mahalanobis distance, Cityblock distance, and Cosine distance. The advantages of a k-NN algorithm are: (i) computationally less intensive, (ii) very less prediction times, and (iii) the algorithm is non-parametric and does not assume about the underlying distribution of the neural data. K-NN algorithm has been used in neural prosthetic applications for predicting upper limb movements. A 4-NN algorithm was used to predict hand positions using LFP signals recorded from the posterior parietal cortex (Klaes et al., 2015). The possible output grasp types were hand positions corresponding to the popular game and its extension Rock, Paper, Scissors, Lizard, and Spock. The k-NN algorithm was used in continuous online decoding. A fuzzy k-NN was implemented to classify grasp movements from LFP and spike data recorded from the PMd in monkeys (Hao et al., 2014a).

Bayesian approaches provide a probabilistic platform to develop neural prosthetic applications. Naïve-Bayes algorithm is a modification of the Bayes' theorem. Bayes' theorem provides a relationship to estimate the conditional probability of two independent but correlated events A and B as

$$P(B|A) = \frac{P(A|B).P(A)}{P(B)}$$

Where P(B|A) is the conditional probability of event B given that event A occurred, P(A) and P(B) are probabilities of A and B respectively. Consider a task of classifying finger movements (for example, five finger flexions) from neural firing rate (for simplicity, from only one channel). Consider a test instance with 10 Hz neural firing rate on channel 1 and true target of thumb flexion. Using the Bayes' theorem, the classification output can

be predicted by estimating the probabilities of all possible outputs for a given input. The predicted output will be the class or finger flexion with the maximum probability.

$$P(output|\ firing\ rate = 10\ Hz)$$
$$= \frac{P(firing\ rate = 10\ Hz|output).P(firing\ rate = 10\ Hz)}{P(output)}$$

The drawback with the above relationship is in estimating the probability of a given output (the denominator term/say, P(thumb flexion)). It is not possible to compute the probability of each individual flexion empirically or theoretically out of a subset of flexions. The rigid relationship between the conditional probabilities of two independent events as described by the Bayes' theorem limits its direct application to developing decoding systems. This problem can be circumvented by making some assumptions about the conditional probability relationship. The Naïve-Bayes algorithm predicts unknown test instances based on the following equation

$$prediction = max_{output}\ \frac{P(firing\ rate = 10\ Hz|output).P(firing\ rate = 10\ Hz)}{constant}$$

The Naïve-Bayes algorithm assumes that the term P(B), the probability of a given flexion, is non-computable and thus, can be assumed as a constant. Thus, we can compare only the numerator terms in the Bayes' relationship since the denominators are equal (constant). The predicted value in this case will be assigned to the type of finger flexion with the highest probability. Naïve-Bayes algorithm was implemented to classify reach directions performed by a non-human primate using neural firing rates extracted from the motor cortex (Bishop et al., 2014).

Bayesian networks are probabilistic graphical models that are based on modeling conditional dependencies of the random variables via a directed acyclic graph. Hidden Markov Models are Bayesian networks modeled assuming the system to be a Markov process with hidden variables. Hidden Markov Models are used in applications requiring temporal pattern recognition. Hidden Markov Models have been used in detecting neural state transitions for motor cortical prostheses (Kemere et al., 2008). In this study, HMMs were used to automatically identify baseline, plan, and perimovement states based on neural data from the motor cortex. A Maximum Likelihood decoder was later used to identify reach directions for a center-out task. Hidden Markov Models have also been employed for decoding finger movements from Electrocorticography signals recorded from the primary motor cortex of four human subjects (Wissel et al., 2013a).

Factors affecting quality of decode

The various parts of a machine learning system like the neural data, pre-processing, feature extraction, feature selection, and the decoding algorithm have an obvious effect on the performance of a neural decoding system. There are other non-algorithmic factors that also influence the performance of a neural decoding system.

Residual efferent and afferent maps.

The principle of "Use it or lose it" governs the neural connections and pathways in our nervous system. Previous studies have shown the reduction in map size of neural representations in the motor regions corresponding to most tasks following neurological diseases or disorders (Kleim, 2011). Neural disorders or amputations can cause the same

effect to lost limb function. The motor maps of skilled tasks such as dexterous manipulation of objects with fingers are speculated to reduce more. The residual sensory maps of hand and arm areas also undergo changes in the map size. Unused or damaged muscles along the pathway undergo atrophy over time. This change in motor map after loss of limb function influences the quality of neural data recorded for neural prosthetic application. Many research studies have examined the residual motor and sensory maps after limb loss in humans (Di Pino et al., 2012; Karl, Birbaumer, Lutzenberger, Cohen, & Flor, 2001; Mercier, Reilly, Vargas, Aballea, & Sirigu, 2006; Reilly, Mercier, Schieber, & Sirigu, 2006). Electrical stimulation of arm stumps elicited non-painful phantom sensations in the somatosensory cortex of 4 human adults as observed with fMRI (Björkman et al., 2012). The researchers also found activation of contralateral posterior parietal cortex and premotor cortex. The activation regions in the various cortices were equivalent corresponding to stimulation of different finger sensations in the arm stump as compared to stimulation in healthy adults. In contrast, the sensation of hand as perceived through electrical stimulation had coarser representations. Mercier et al performed an experiment where they stimulated the primary motor cortex of 4 amputees using Transcranial Magnetic Stimulation (TMS) and found that it elicited sensations like performing hand movements (Mercier et al., 2006). In one subject, the intensity of stimulation positively correlated with the perceived amplitude of movement. Several other studies report similar results of intact sensory and motor maps in patients after amputation (Jarosiewicz et al., 2015; Klaes et al., 2015). While this is the case in amputees, patients who suffered neurological disorders or diseases often undergo cortical reorganization in the motor regions. Traversa et al performed focal TMS to study the somatotopical reorganization of the motor cortex in stroke patients. The results

indicated that the motor evoked potential (MEP) threshold for excitability was significantly higher in the affected hemisphere of stroke survivors than control subjects (Traversa, Cicinelli, Bassi, Rossini, & Bernardi, 1997). Results also indicated that the latency of excitation was significantly higher and extenuation of motor area to the abductor digiti minimi muscle was significantly lower in the affected hemisphere of stroke survivors. Other studies also indicate reduction in the map size of the affected hand in the motor cortex (Ward & Cohen, 2004).

Neuroplasticity and motor learning.

Amputation induced plasticity is a widely-reported phenomenon in amputees that corresponds to aberrant plastic changes in the motor cortex of amputees. Neuroplasticity has direct implications for designing bidirectional motor prosthetic interfaces. Few studies have also hypothesized that using a bidirectional motor prosthesis might revert unwanted plasticity experienced due to sudden limb loss. Donald Hebb presented an underlying principle for understanding plasticity in the nervous system. The Hebbian learning principle states that "cells that fire together, wire together". Repeated use of unused nervous pathways in the central and peripheral nervous system helps re-establish the lost connections (Hebb, 2005). The nervous system looks like a continuously changing structure of which plasticity is an inherent property and a necessary result that evolves with external stimuli. Neuroplasticity is a key component that can be observed when patients reuse their lost limb function. Motor learning is a branch of motor control that is used to study how humans learn and acquire new motor skills. The corticostriatal pathway plays a

crucial role in acquiring new motor skills such as driving (Pisani, Centonze, Bernardi, & Calabresi, 2005; Wickens, Reynolds, & Hyland, 2003). Learning to use a neurprosthetic device also falls under the category of motor learning. Few experiments conducted that used brain machine interface as a paradigm to study motor learning revealed new insights into the role of neuroplasticity in prosthetic control. Ganguly and Carmena used a Kalman Filter based decode for completing a reaching task in a non-human primate using LFP signals from the primary motor cortex (Ganguly & Carmena, 2009). As explained in the previous sections, Kalman Filters used with fixed parameters (without retraining) used over a chronic duration produces poor decoding results. In this study, it was hypothesized that when using a Kalman Filter with fixed parameters, the non-human primate would adapt and learn to use this stable "bad decoder". The results of this experiment indicated that the performance of the non-human primate improved over a chronic period while using a "bad decoder". It was observed that the frequency of modulation of the LFP signals evolved over time which correlated with the improved performance. Thus, neuroplasticity plays a critical role in learning to use a prosthetic device. It is possible to obtain better decoding performance over an extended period even using a mediocre decoding algorithm.

Feedback: sense of embodiment and virtual reality.

Touch perception is imperative for effectively interacting with the environment and improving the fundamental understanding of the world. Reviews of amputees using prosthetics indicate that user acceptance of modern artificial limbs might increase drastically when incorporated with appropriate, graded sensory feedback eliciting distally

referred sensations of touch and joint movements (Dhillon, Lawrence, Hutchinson, & Horch, 2004). Providing sensory feedback gives the user a sense of embodiment of the prosthesis. Chances of associating the prosthetic device as a part of the user's interpersonal space improves drastically when feedback is incorporated as opposed to using a passive prosthesis which is often associated with their peri-personal space. Many research studies have shown the possibility of eliciting sensory feedback of the phantom hand to amputees and patients suffering from neurological disorders by implanting an electrode in the peripheral nervous system or the somatosensory cortex. Dhillon and Horch showed that by implanting microelectrode arrays in the intrafascicular nerve endings of the peripheral nerves, it was possible to produce graded, discrete sensations of touch or movement associated with the phantom hand of amputees (Dhillon & Horch, 2005a). In this study, the authors also reported that even in the absence of visual inputs the subjects could adequately judge and set grip forces for manipulating objects using an artificial prosthetic arm when provided with sensory feedback. Tan et al demonstrated the capability of cuff electrodes to provide touch perception over a chronic duration of up to 24 months (Tan et al., 2014a). Amplitude, duty cycle, and frequency stimulation waveform were some of the factors that affected sensory perception. Peripheral nerve cuff electrodes were used to elicit natural sensations without the "tingling" feeling reported in other literature. This study showed that it was possible to provide chronic sensory feedback over an extended period with repeatable, stable sensory maps of the phantom hand.

Virtual reality provides an immersive environment to interact and manipulate objects in a 3-D virtual space (Ryan, 2001). Virtual reality tools such as Occulus Rift, Microsoft VR, and Leap Motion VR gadgets are currently used to provide next-level

gaming experience. The same tools can be used for rehabilitation and neuroprosthetic applications. Physical hardware prostheses used for motor neuroprosthetic control have limitations in their ability to perform tasks due to hardware related issues. A survey of current motor prostheses (passive, body powered, and myoelectric) analyzed the reasons for prosthetic abandonment in pediatric and adult amputee populations (Biddis and Chau). The study revealed that appearance, function, control, comfort, and durability were contributing factors that affected user acceptability of the motor prosthesis. Physical prostheses used in many studies are bulky and weigh significantly more than a natural, intact hand. Virtual prostheses used by rendering a robotic controller on a 2-D computer screen evaded this problem. However, they are limited in their capacity to provide a realistic experience of using a prosthesis and associating the device in the user's interpersonal space. Virtual reality provides a good trade-off between providing an immersive, point-of-view environment and the ease of use of a virtual prosthesis in this virtual space. They are intuitive and the sense of immersion into this virtual environment is speculated to increase user acceptability and compliance for using prostheses. Although they are not a substitute to physical prostheses, virtual reality coupled with motion tracking can be used for efficient rehabilitation. They can also be used to evaluate possible prosthetic models and test their effectiveness without building a physical prosthesis. In essence, virtual reality provides heightened visual feedback which is critical for motor control. The immersive environment is also hypothesized to improve user attention which is an important contributor for effective rehabilitation and motor learning.

Future directions and other considerations

There have been significant advances in the field of neural prosthetic applications. But there are problems that haven't been addressed while translating neural prostheses from a laboratory setting to a clinical environment. To achieve a fully functional prosthetic device with maximum user compliance and acceptance, these factors need to be addressed and taken into consideration.

Longevity of decode.

A meta-analysis of existing literature reporting advances in neural prosthetic applications might reveal that most of the studies are tested over a shorter duration of time. Various reasons accounting for such a short duration range from loss of recording active neuronal units, surgical or clinical mishaps post-implantation to IRB or FDA restrictions on experiment durations. To truly validate the functionality and feasibility of such a prosthesis, it is imperative to establish repeatability of high accuracies of neural decodes over an extended period.

Stability of algorithms: necessity to obviate the need for re-calibration.

Patient time is an important metric that must be considered while planning experiments. Shorter experiment durations can help keep the patient attentive and interested in the experimental paradigm. Current protocols of experiments and working procedures of machine learning algorithms require data collection to train the algorithm.

Training an algorithm at the start of every session is necessary to account for various factors of variability in neural data. Small movements of electrodes at the site of implantation, change in neural representations pertaining to the task, or loss of active neural units due to neural edema and/or glial scarring are possible causes for variability of neural data. Experiment time can be reduced by removing the need for training an algorithm prior to each session. Methods to factor change in neural representations have been proposed to continue using an algorithm with fixed parameters (cite Chase tuning factor).

Unsupervised learning methods.

Unsupervised learning algorithms are adaptive algorithms that do not require training. These algorithms learn the input-output relationship by updating their errors while predicting. Unsupervised clustering methods such as variants of Gaussian Mixture Models (GMMs) work based on Expectation Maximization algorithm to determine the mixture parameters. GMMs are efficient unsupervised methods for modeling non-stationary time series data. Since unsupervised methods do not require explicit training, they are adaptive to dealing with changes in neural representations mentioned previously. Unsupervised methods might be a viable solution to dealing with the stability of an algorithm by obviating the necessity for re-calibration prior to each session.

Decrease in number of active neurons and coping mechanisms.

The performance of the brain machine interface (BMI) can be influenced by the quantity of neural information available for decode. Previous research has shown that there is a significant decrease in the signal to noise ratio of the neural signals and a steady decrease in impedance of the recording electrodes over time (House, MacDonald, Tresco, & Normann, 2006; Vetter, Williams, Hetke, Nunamaker, & Kipke, 2004). There can be a steady decrease in the number of electrodes that record action potentials, which can have a deleterious effect on BMI performance. Action potential recordings can also be affected due to glial scarring or electrode location changes (Berens, Keliris, Ecker, Logothetis, & Tolias, 2008; Frien & Eckhorn, 2000; Jia, Smith, & Kohn, 2011; Leopold & Logothetis, 2003; O'Leary & Hatsopoulos, 2006; Stark & Abeles, 2007). There is a need for advances in the field of machine learning to develop coping mechanisms to account for loss of information from previously contributing neurons.

Multi-modal brain machine interfaces.

Brain machine interfaces based on different neural signals have been shown to efficiently generate control signals for upper limb prosthesis. New attempts at building brain machine interfaces incorporate multi modal signals i.e., using more than one type of signal as input to the neural decoding system. Additional signals can also be extracted to detect other attributes of motor control like attention and intention. Eye tracking can help extract information about attention of the user. The aim of integrating multiple modalities of neural and other physiological signals is to account for factors related to motor control

such as information processing, coordination, cognition, and perception. Incorporating these features into a machine learning algorithm might help add robustness and improve detection of various movement states (McMullen et al., 2014; Suminski, Tkach, & Hatsopoulos, 2009).

Internal models of motor control.

In the field of motor control, the concept of internal model as a system which mimics the behavior of a natural process has become an important theoretical concept. Internal models can be classified into two groups. First, internal forward models that aim to represent behavior of the motor system in response to outgoing motor commands. Second, internal inverse models that are thought to represent the behavior of a normal motor system by determining the commands required to achieve a desired movement trajectory. There is also a variation of the forward models which models the behavior of the external environment. The end goal of a forward model is to generate an estimate of the sensory consequences of motor commands. The rationale behind hypothesizing the brain has internal models for carrying out various tasks is that it would be impossible to make fast, smooth, and accurate motor execution solely dependent on actual sensory feedback. Sensory feedback is a delayed feedback signal due to time delays associated with receptor transduction, neural conduction, central processing, and muscle activation. Therefore, skilled manipulations depend on predicted properties of objects and the motor system itself. Incorporating internal models of motor control into designing machine

learning algorithms might be helpful in decoding anticipatory control which might reduce possible lags in the decoding system (Miall & Wolpert, 1996).

Feature selection

Abstract

Objective The performance of machine learning algorithms used for neural decoding of dexterous tasks may be impeded due to problems arising when dealing with high-dimensional data (curse of dimensionality). The objective of feature selection algorithms is to choose a near-optimal subset of features from the original feature space to improve the performance of the decoding algorithm. The aim of our study was to compare the effects of four feature selection techniques, Wilcoxon signed-rank test, Relative Importance, Principal Component Analysis, and Mutual Information Maximization on SVM classification performance for a dexterous decoding task.

Approach A nonhuman primate (NHP) was trained to perform small coordinated movements – similar to typing. An array of microelectrodes was implanted in the hand area of the motor cortex of the NHP and used to record action potentials during finger movements. A Support Vector Machine (SVM) was used to classify which finger movement the NHP was making based upon action potential firing rates. We used the SVM classification to examine the functional parameters of (i) robustness to simulated failure and (ii) longevity of classification. We also compared the effect of using isolated-neuron and multi-unit firing rates as the feature vector supplied to the SVM.

Main results The best classification performance was on post-implantation day 36, when using multi-unit firing rates the worst classification accuracy resulted from features selected with Wilcoxon signed-rank test (51.12 ± 0.65%) and the best classification accuracy resulted from Mutual Information Maximization (93.74 ± 0.32%). On this day

when using single-unit firing rates, the classification accuracy from the Wilcoxon signed-rank test was 88.85 ± 0.61 % and Mutual Information Maximization was 95.60 ± 0.52% (degrees of freedom =10, level of chance =10%).

Significance These results suggest improved decoding performance can be achieved by using optimally selected features. The results based on clinically relevant performance metrics also suggest that the decoding algorithm can be made robust by using optimal features and feature selection algorithms. We believe that even a few percent increase in performance is important to the patient and can help improve the ease and experience of using the neuroprosthesis.

Introduction

Microelectrode array brain machine interfaces (BMI) have shown the potential to alleviate various neurological disorders. BMIs utilizing advances in robotics and machine learning can restore limited lower and upper extremity motor function. Several research studies have investigated the viability of a cortical brain machine interface in humans and NHPs (Aflalo et al., 2015; Collinger et al., 2013; Gilja, Nuyujukian, Chestek, Cunningham, Byron, Fan, Churchland et al., 2012; Gilja et al., 2015; Hwang & Andersen, 2013; S. Kim, Simeral, Hochberg, Donoghue, & Black, 2008; Musallam et al., 2004). Brain machine interfaces for controlling a robotic limb or moving a cursor have been successfully demonstrated in humans and non-human primates. These systems provided real time control of a neuroprosthetic system by decoding neural signals moment by moment with an objective to provide certain functionality to replace the native arm. Communication prostheses focus on achieving discrete goals like moving cursor to specific targets (Kellis

et al., 2010; Santhanam, Ryu, Byron, Afshar, & Shenoy, 2006; K. V. Shenoy et al., 2003). These systems are based on decoding the endpoint goal of reach and map the neural signals to spatially distributed targets.

BMIs can be broadly classified based on the type of bio-signal used to control the prosthesis. Electroencephalogram (EEG), Local field potential (LFP) and Action potential (AP) constitute the majority of source signals used in brain machine interfaces. APs are discrete spiking events of an individual neuron. In statistics terms, APs or neural "spiking" can be thought of as a non-stationary point process in which neural information is largely encoded by changes in the AP firing rate coding (frequency of action potentials/spiking) (Truccolo, Eden, Fellows, Donoghue, & Brown, 2005). In this paper, we utilize neural recordings of APs from individual neurons to classify various movements of the fingers. Finger movements associated with tasks performed in daily life can be broadly classified into continuous and discrete movements. One of the important characteristics of the human upper extremity functioning is the ability to perform coordinated and dexterous finger movements. Typing, eating with a spoon, writing with a pen and opening a lock with a key are some of the examples in our daily life that require such dexterous manipulations using individual or combined finger movements. Incorporating dexterity as a feature in a neuroprosthesis would help amputees and paralyzed persons to carry out a wider range of tasks. To achieve such dexterous control requires a neural decoding algorithm that can map high-dimensional neural signals onto a high-dimensional hand prosthesis. Optimizing algorithms for decoding neural signals will be critical for providing useful control of upper extremity neuroprosthesis. Feature selection is an important step in designing a machine

learning system. Choosing a *w*-dimensional subset from a *p*-dimensional feature space consisting of '*p'* predictors using an objective metric is the aim of feature selection. Feature selection also reduces the dimensionality of feature space, inundating it with more "informative" features thus, removing lesser contributing ones that might occlude the feature space.

Curse of Dimensionality.

Certain machine learning algorithms fail to scale well in high dimensional feature space. These algorithms suffer from the "curse of dimensionality", which refers to the problems that arise when analyzing and organizing high-dimensional data. Consider a univariate, independent variable 'X' which follows a Gaussian distribution with mean 'μ' and variance 'σ' ($X \sim N(\mu, \sigma)$). According to the properties of Gaussian distribution, ~68% of the data is enclosed in the region surrounded by the mean ± 1 standard deviation (Figure 1a). Consider two independent variables X_1 and X_2 which follow Gaussian distributions with means 'μ_1' and 'μ_2', and variances 'σ_1' and 'σ_2' respectively ($X_1 \sim N(\mu_1, \sigma_1)$ and $X_2 \sim N(\mu_2, \sigma_2)$). For a bivariate, Gaussian distribution only ~40% of the data is enclosed within the same region (Figure 1b). For a 50-dimensional multivariate normal distribution, only ~1/250,000,000th of the data lie within the mean ± 1 standard deviation region. As the number of dimensions (variables) increase, the amount of data bounded by the mean ± 1 standard deviation region decreases exponentially (Figure 1c). In neural decoding, data from each electrode is treated as an individual feature. A microelectrode array usually consists of 96 electrodes thus, making the feature space 96-dimensional. In case of a 96-

dimensional feature space, only an infinitesimally small proportion of data points are enclosed in the mean ± 1 standard deviation region. Results of Figure 1 were generated using a novel approach to constructing the Multi-dimensional standard deviation ellipsoid based on spectral decomposition of the sample covariance (Wang, Shi, & Miao, 2015).

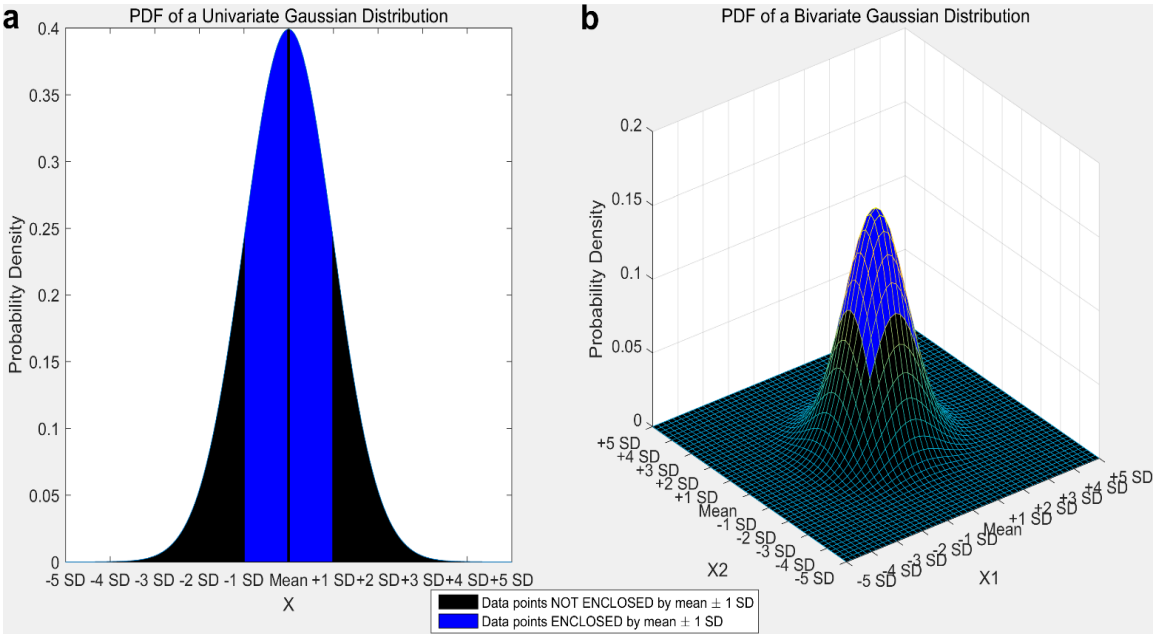


Figure 1a. Univariate Gaussian Distribution. The area shaded in red shows the data points bounded by mean ± 1 standard deviation. 68.27% of the data is enclosed in this region. Figure 1b. Bivariate Gaussian distribution. The area shaded in blue shows the data points bounded by mean ± 1 standard deviation. Only 39.35% of the data is enclosed in this region. As the number of dimensions increase from a univariate to a bivariate distribution, the amount of data bounded by mean ± 1 standard deviation reduces by ~42%.

In high-dimensional space, almost every point is closer to the edge of a hypercube that encloses the points than to another sample point. For a sample of size 'n', the expected average distance between the sample points 'D' in a 'd'-dimensional feature space can be estimated using the following equation:

$$D(d,n) = \frac{1}{2}.(\frac{1}{n})^{\frac{1}{d}}$$

For a two-dimensional space with 10,000 points, the average expected distance between the sample points is 0.005 and for a 100-dimensional space with the same number of points, the expected distance is 0.45. It should be noted that the maximum distance from any point to the edge is 0.5 for normalized values of dimensions (Kantardzic, 2011).

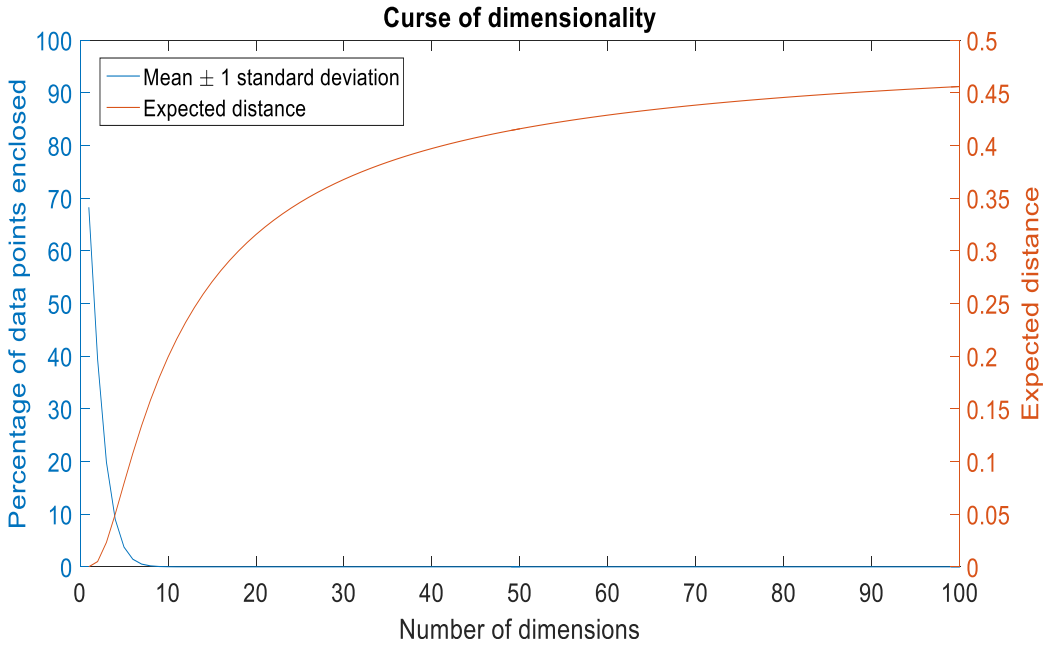


Figure 2. Curse of dimensionality. The plot in blue shows the percentage of data points enclosed by the mean ± 1 standard deviation for 100 dimensions. The amount of data enclosed by the mean ± 1 standard deviation region asymptotes to zero from an octa-variate (8-dimensional Gaussian distribution contains only 0.18% of the data in mean ± 1 standard deviation region) distribution. The plot in red shows the expected distance for 100 dimensions. Expected distance is defined as the distance between the center to the edge of a hypercube.

It can be seen that the percentage of data points enclosed by the mean ± standard deviation region decreases as the number of dimensions increase (Figure 2). Also, the expected distance between points increases exponentially as the number of dimensions increase (Figure 2).

The above two examples illustrate the sparsity of finite data in high-dimensional space. In high-dimensional space, most data points act as outliers. This sparsity in data distribution deters the efficacy of certain machine learning algorithms in high-dimensions. Feature selection is one of the methods to cope with "curse of dimensionality".

Using machine learning algorithms for multivariate, high-dimensional data is often computationally expensive. Due to the complexity of feature space and rigorous numerical computations involved in defining the hyperplane in this high-dimensional feature space, the performance of the machine learning algorithm is deterred. Feature selection is the process of selecting an *O*-dimensional subset feature space from a *P* – dimensional original feature space where '*p*' is the number of predictors.

Feature selection is usually applied to reduce information redundancy and trim the input space to better predict the responses. Some of the advantages of feature selection are:

- Facilitate data visualization and data understanding
- Reduce data measurement and storage requirements
- Reduce training and utilization times
- Simplify the learning model and aid in better understanding and interpretation by researchers
- Enhance generalization by reducing overfitting
- Defy the curse of dimensionality to improve predictor performance (Guyon & Elisseeff, 2003).

Identifying the best subset of features is a sub-optimal problem to solve. The only method to do this is through exhaustive grid search, i.e. exhaustively searching through every permutation of predictors available. Mathematically, there exists 2^p permutations of features that can be selected from 'p' features. In case of our neural data, this results in iterating through 2^{96} (96 features for multi-unit firing rate and >96 features for single-unit firing rate based feature vector) permutations of features to identify the "best" subset.

When dealing with multivariate, time-series signals like neural signals, it is imperative to judge where the learning algorithm must focus its attention. *Filter* or *Criterion* based feature selection and *Wrapper* based feature selection are two broad categories of feature selection that are commonly applied in machine learning (Kohavi & John, 1997b). Application of statistical, empirical or other "criteria" based methods such as mean, variance, student's t-test and correlation are some examples of criterion based feature selection. Applying criterion based feature selection requires some domain expertise in order to determine what qualifies as a useful criteria. Wrapper based feature selection iteratively uses various combinations of features as input to a machine learning algorithm and evaluates the importance of each feature based on some evaluation criteria from the prediction such as coefficient of determination (r^2). Ideally, it is advisable to use the same machine learning algorithm as a classifier and a wrapper for feature selection. Oftentimes, it is also valuable to use a simpler, computationally efficient machine learning algorithm as a substitute wrapper. For example, SVMs are an efficient but computationally intensive solution to solve the problem of face recognition by computing key points (that act as features) on the face. Using SVM as a wrapper in this case would demand access to a lot of resources (in terms of clusters) and still be time consuming. An alternative to using

SVM in this case would be using a simpler algorithm such as Logistic regression. Care should be taken to ensure both the algorithms have similar assumptions about the data such as nonlinearity or heteroscedasticity of noise.

Methods

Approval for the animal use protocol in this study was obtained from the University of Utah Institutional Animal Care and Use Committee (IACUC). All procedures conformed to National Institute of Health (NIH) standards for animal care. The recording setup, behavioral task, data collection and preliminary data processing approaches are explained elsewhere (Baker et al., 2009). A 96 channel microelectrode array (MEA, Blackrock Microsystems) was implanted in the hand area of primary motor cortex of a male macaca mulatta. The non-human primate (NHP) was trained to perform cued combined flexions of the thumb, index and middle finger and individual flexions and extensions of the same digits using a manipulandum. Visual cues were provided using a computer screen placed in front of the monkey. In order to start a trial, the monkey had to relax all its fingers moving all of the finger switches in the manipulandum to the open state. After a randomized wait time of 1000-3000ms, a visual cue indicating which finger(s) to flex/extend appeared on the computer screen. The monkey then had 2000ms to react to the visual cue and depress the associated switch. Once the correct switch was pressed, the monkey had to hold the switch for 500ms. The trial was deemed successful if the monkey pressed the correct switch and adhered to the time constraints. The behavioral task was implemented using a real-time operations systems in a custom LabVIEW (National Instruments) program.

Neural Decoding System Architecture.

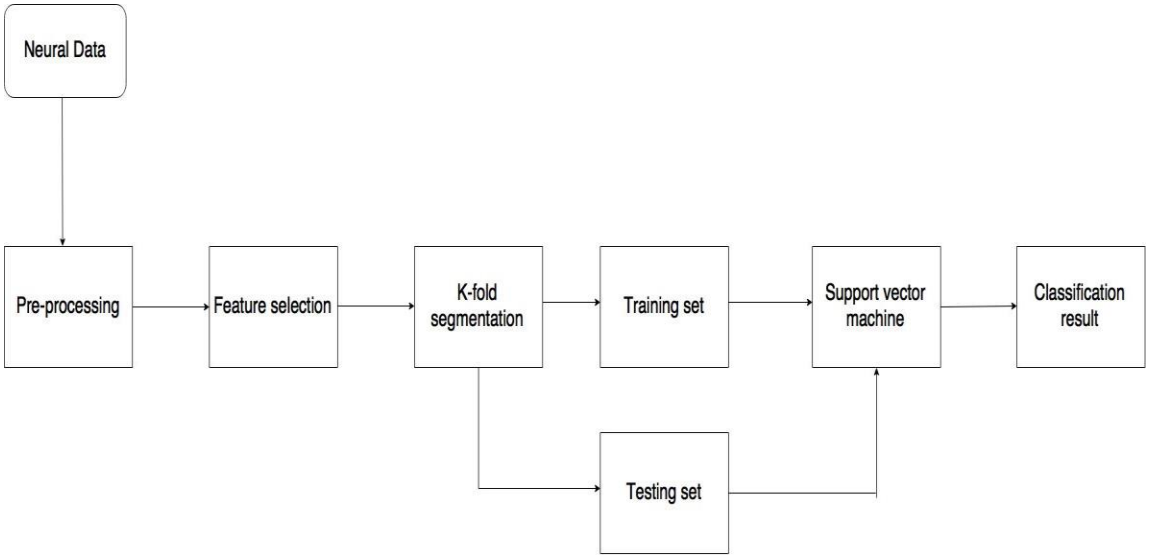


Figure 3 Neural decoding system architecture. After pre-processing the neural data, feature selection was performed. During K-fold segmentation, the entire data set was divided into 10 folds. 9 folds were used for training and the 10^{th} fold was used for validation. This process was repeated 10 times until each fold served as a validation fold. Validation error and parameters of the SVM were estimated using a nested cross-validation loop.

Neural data recorded from the NHP was spike sorted. The timestamp of spike events was obtained from the offline sorter. Pre-processing also included binning/moving average windowing of the point process using a boxcar window. After applying the moving average technique, neural "firing rate" for each single or multi-unit was obtained. Neural firing rate was used as the feature vector (input) to the SVM. Neural activity corresponding to each successful finger movement trial was extracted and concatenated. The entire dataset was randomly divided into 10 folds. Each fold served as the testing set once while data from the remaining folds was used for training. Model parameters such as box constraint(C) and sigma (for the RBF kernel) were estimated using an exhaustive grid search algorithm with exponentially increasing values from 1e-5 to 1e5. Classification accuracy was calculated after predictions were made on the unseen test set. This process was repeated 20 times to reduce generalization error of the SVM.

Pre-processing.

The MEA is a 10x10 grid of 1 mm tall electrodes that are capable of recording action potentials in addition to local field potentials (House et al., 2006). The MEA data were sampled at 30 kHz. Neural data collected using the MEA were sorted offline using an expectation-maximization based competitive mixture of *t*-distributions decomposition algorithm (Shoham et al., 2003). Data were then imported to Matlab (Mathworks) for further analysis. The time stamps of action potentials recorded at 30 kHz were downsampled to 600 Hz. A boxcar moving average window of 300ms width and 33.3 ms step size was used to obtain a moving average firing rate. Electrodes in the motor cortex can record from more than one neuron. The features extracted from neural signals recorded from such electrodes are called "multi-unit" firing rate. However, the neural activity recorded on such electrodes can be separated using techniques such as Principal Component Analysis, Expectation-Maximization algorithm or Independent Component Analysis (Lewicki, 1998b). Features extracted from such individual, isolated neurons are called "single-unit" firing rate. The moving average firing rate was downsampled in order to reduce data size. A 4^{th} order low pass Butterworth filter with a cut-off frequency of 10 Hz was used prior to downsampling the neural firing rate to 20 Hz and the neural firing rate was obtained as a time varying vector. This process was repeated for all 96 electrodes to obtain multi-unit neural firing rate, i.e. the cumulative firing rate of all neurons recorded on a particular electrode. An average of 142.2 ± 36.3 neural units were recording from 96 electrodes during each session.

Data from individual trials was aligned in time on switch closure times of successful trials. A movement period was defined as the duration corresponding to 450ms prior and

1000ms after the switch closure. A baseline period (resting state) for a trial was defined as the duration corresponding to 2500ms to 1000ms prior to switch closure. Baseline and movement period data was obtained for all available degrees of freedom and all successful trials for each day experiments were conducted and represented a vector of time-series data.

Feature selection.

In this study, we have limited our comparisons to criteria based feature selection methods.

Wilcoxon signed-rank test. Wilcoxon signed-rank test is a non-parametric alternative to the student's t-test. This non-parametric test can be used to identify if samples from two independent yet related distributions are significantly different (Randles, 1988). In the context of selecting single or multi-unit data as input to the SVM, the difference between baseline and movement related firing rate was computed. The null hypothesis was that the data came from a continuous, symmetric distribution with a median equal to zero (i.e. no electrode recorded increased firing rates in the movement period as compared to the baseline period). Electrodes for which the null hypothesis was rejected ($p<0.001$) with a positive median difference from baseline were kept. These electrodes were then sorted in order of increasing median difference. For the purpose of feature selection, the median difference was computed as a scalar to select features (single unit/multi-unit).

Relative importance. Relative importance was a feature selection technique initially developed for selecting neurons in the primary motor cortex for decoding (H. Kim et al., 2012). First the movement only firing rate (difference of movement and baseline firing rate) was computed. The trial averaged firing rate for each neuron for all the successful trials was

calculated. Then, the inter-movement variance was computed as the difference of trial averaged firing rate and the average firing of a neuron for a particular movement. The neural recordings were then ranked in descending order of inter movement variance. For the purpose of feature selection, the inter movement variance was computed as a scalar to rank features (single/multi-unit).

Principal Component Analysis. Principal component analysis (PCA) can be used as a feature transformation technique, where a transform function is applied to the data to represent it in a higher dimensional transform space. For an '*n'* dimensional possibly correlated data, PCA represents the data in a (n-1) dimensional space in linearly uncorrelated principal component coordinates (Jolliffe, 2002; Lu, Cohen, Zhou, & Tian, 2007). The transformation is carried out in such a way that the first principal component contains the maximum possible variance of the data. The succeeding principal components are ordered in descending order of variance. This transformation of data according to the variance at each time point can be used to eliminate noise, but does not necessarily extract discriminative features. Neural firing rates corresponding to each movement was provided as an input to PCA. The operation of PCA can be thought of as revealing the internal structure of the data based on its variance. For a multivariate dataset that can be represented in a high-dimensional space, PCA provides a better representation in low-dimensional space from an "informative" viewpoint. This is done by considering only the first few principal components and thus, PCA serves as a dimensionality reduction method.

Mutual information maximization. Mutual information is the mutual dependence of two random variables. Unlike correlation, mutual information is not limited to real-valued

random variables and estimates how similar the joint distribution P(X|Y) is to the products of the factored marginal distribution P(X) and P(Y) (Torkkola, 2003). Entropy of a random variable C can be defined as

$$H(C) = -\sum_{c} P(c)\log(P(c))$$

The conditional entropy of two random variables C and Y can be defined as

$$H(C|Y) = -(\sum_{c} P(c|y)\log\big(P(c|y)\big)\,dy$$

Then, the mutual information of random variables C (neural firing rate) and Y (movement type) can be defined as the I(C;Y) = H(C) – H(C|Y) and can be represented as

$$I(C|Y) = \sum_{c}\sum_{y} P(c|y)\log\frac{P(c|y)}{P(c)*P(y)}$$

Mutual Information maximization was implemented using the FEAST Toolbox available for MATLAB (Brown, Pocock, Zhao, & Luján, 2012). For a class label Y, the mutual information score of feature C is defined as:

$$J(C) = I(C;Y)$$

This score J(C) is referred to as mutual information maximization and we rank the features in descending order of the mutual information score. Neural firing rates corresponding to movement period for each degree of freedom was used as the input to Mutual Information Maximization algorithm.

Support vector machine.

Support vector machines have shown promising results in upper extremity decoding tasks using various source signals such as MEG, EEG, ECoG and EMG (Bitzer & van der Smagt, 2006; Demirer, Ozerdem, & Bayrak, 2009; Quandt et al., 2012; Wissel et al., 2013b). Support vector machine is a class of non-probabilistic, binary, linear classifier (Platt, 1999). Support vector machines represent the data in higher dimensional space and find the best separating hyperplane in this space. The objective of the SVM is to find a hyperplane that has the maximum distance from a point belonging to any class. Such a classifier is also called a maximum margin classifier whose generalization error is low. During training, each point in the training set is assigned a weight α. Those points with training weights $\alpha \neq 0$ are called the support vectors since, they help forming the hyperplane. In case of linearly non-separable cases, a soft margin classifier is implemented which allows for misclassified instances. Non-linear problems can be solved by using the "kernel trick" in the SVM. Kernel functions map data into a higher dimensional space where, the hyperplane is now formed. Gaussian (radial basis function) kernel was employed in our classification problem to account for non-linearity in the input-output relationship. Gaussian kernel K(x,x') for two samples x and x' defined as a feature vector in some predictor space is defined by,

$$K(x, x') = exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right)$$

where σ is a free parameter that defines the smoothness of the Gaussian kernel.

SVMs are inherently binary classifiers i.e. they can distinguish between only two classes. Their functionality can be expanded to solve multiclass problems by decomposing it into multiple binary sub-problems (Duan & Keerthi, 2005; Hsu & Lin, 2002). We used a

one-vs-one multiclass implementation of the SVM to differentiate between the many available movements. For a problem of classifying *'k'* classes, we require $\frac{k(k-1)}{2}$ binary SVM classifiers for each pair of the 'k' classes. The class of a test instance is predicted by taking the mode of predictions of all the one-vs-one SVM pairs.

In addition to extracting neural activity corresponding to valid trials for all available degrees of freedom for a particular session, we included 30 random baseline periods as a "rest" phase (11^{th} degree of freedom). During the training phase of supervised learning algorithms such as Support Vector Machine, the algorithm must be provided with corresponding outputs (class labels). The class labels were created depending on the movement type. For example, thumb flexion was encoded as 1, index flexion as 2, middle finger flexion as 3 and so on.

Performance metrics.

The first step in assessing the performance of feature selection methods was to find the optimal number of features for each feature selection algorithm that best classified the different finger movements and the resting state. For this purpose, all available successful trials in a session were split into a 70% for training and the remaining 30% for testing. A 10 fold cross validation routine was performed to reduce variability in performance estimates during validation. For a given input data (multi-unit or single-unit firing rate), the features were ranked based on the results of the feature selection algorithms. We iteratively incremented one feature (neural firing rate on a single electrode or from an isolated neuron) at a time and used it as an input to the classifier to identify the optimal number of features.

In order to evaluate the performance of features selected at random, we also included random multi-unit and isolated unit firing rate feature to compare with the other methods.

Robustness to simulated failure. The performance of the brain machine interface (BMI) can be influenced by the quantity of neural information available for decode. Previous research has shown that there is a significant decrease in the signal to noise ratio of the neural signals and a steady decrease in impedance of the recording electrodes over time (House et al., 2006; Vetter et al., 2004). There can be a steady decrease in the number of electrodes that record action potentials, which can have a deleterious effect on BMI performance. Action potential recordings can also be affected due to glial scarring or electrode location changes (Berens et al., 2008; Frien & Eckhorn, 2000; Jia et al., 2011; Leopold & Logothetis, 2003; O'Leary & Hatsopoulos, 2006; Stark & Abeles, 2007). Feature selection algorithms should be robust enough to handle the sudden losses in neural information over time. In order to test the endurance of the feature selection algorithms, we randomly dropped 10's of percent of the available neural firing rate and tested its performance. The random removal procedure was repeated 20 times to reduce generalization bias.

***Longevity of neural decodes*.** Brain machine interfaces are devices which will be used over an extended period of time. In order to be useful the neuroprosthetic device must be capable of accurate performance over this extended period of time. We present here the chronic decoding results of 47 sessions collected over 142 days. For a given session the optimal number of features was computed. Decoding accuracy for a feature selection algorithm on a particular day was then calculated using the cross validated optimal features.

Results

Quality of neural recordings.

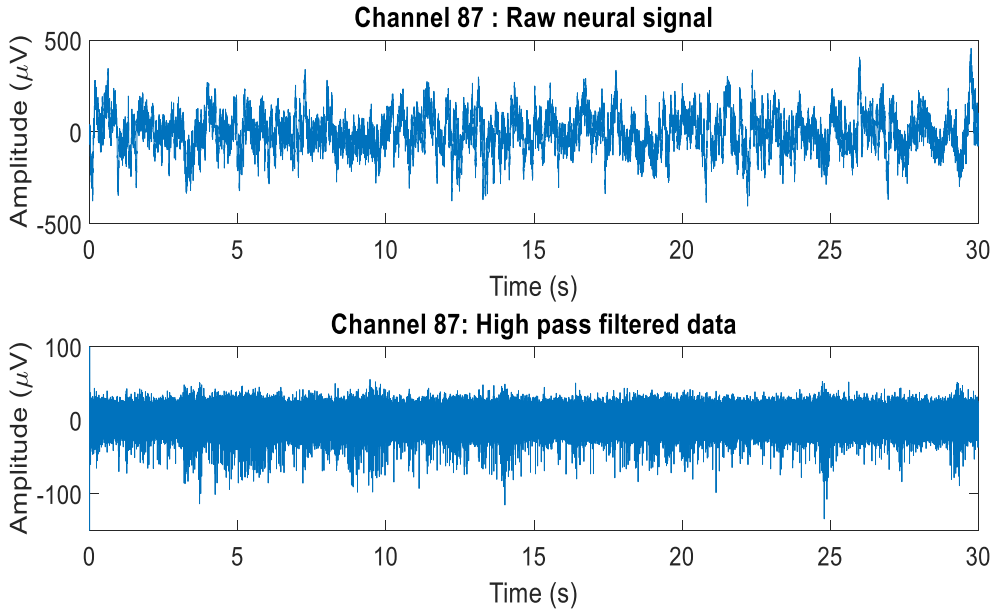


Figure 4. Raw neural recording on channel 87. The 30 second recording consisted of multiple different trials. The raw neural recording on channel 87 was filtered using a high-pass Butterworth of cut-off frequency 250Hz. The MATLAB function *"filtfilt"* was used to filter the neural signal to ensure zero-phase distortion.

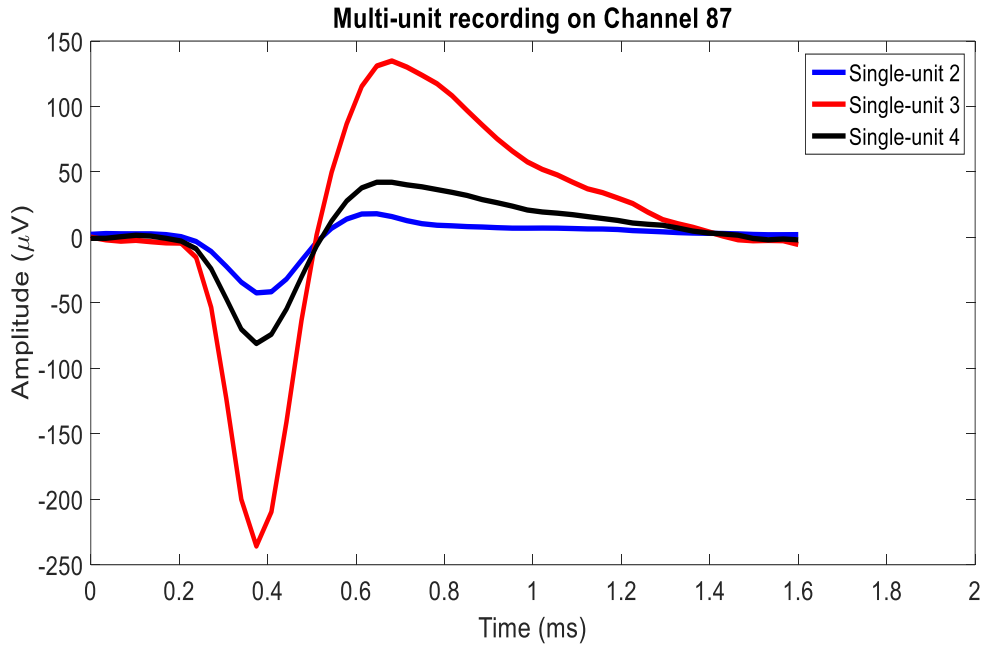


Figure 5. Mean of isolated single-units neural activity on Channel 87. Channel 87 contained three individual single-units post spike-sorting. Since there was more than one single-unit recorded on channel 87, the single-unit and multi-unit firing rates were different. For single-unit firing rate, each single-unit was treated as an individual source of information. Whereas for multi-unit firing rate, the single-units were treated as one and the firing rate was computed.

In order to demonstrate the quality of neural recordings used in this analysis, we extracted a 30 seconds neural recording. Principal Component Analysis followed by k-means clustering was performed to separate the isolated units and noise. In Channel 87, there were 3 isolated single-units (Figure 5). Single-unit firing rate was obtained by treating the isolated neural units as individual sources of information. Therefore, we obtained three single-unit firing rates for channel 87 by treating single-units 2, 3 and 4 as individual sources of information. Multi-unit firing rate was obtained by treating the individual single-units as one source of information. Therefore, we accrued the neural activity of single-units 2, 3 and 4 and obtained one multi-unit firing rate for channel 87. For channel 33, the single and multi-unit firing rate are the same. To summarize, the number of multi-unit features is equal to the number of active electrodes (irrespective of the number of isolated units it was

recording). Whereas, the number of single-unit features is equal to the number of isolated units.

Selecting optimal number of features.

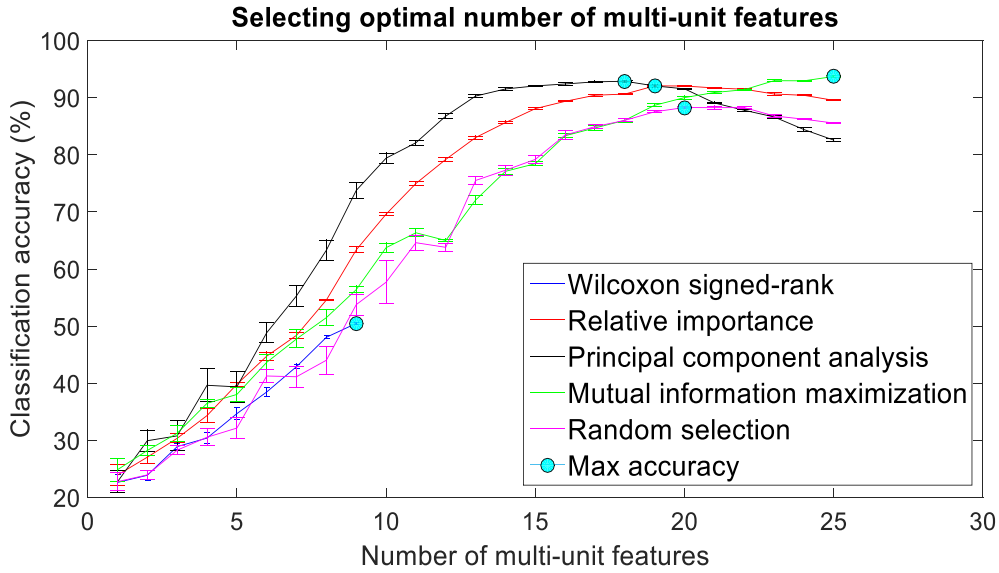


Figure 6 Selecting optimal number of channels.

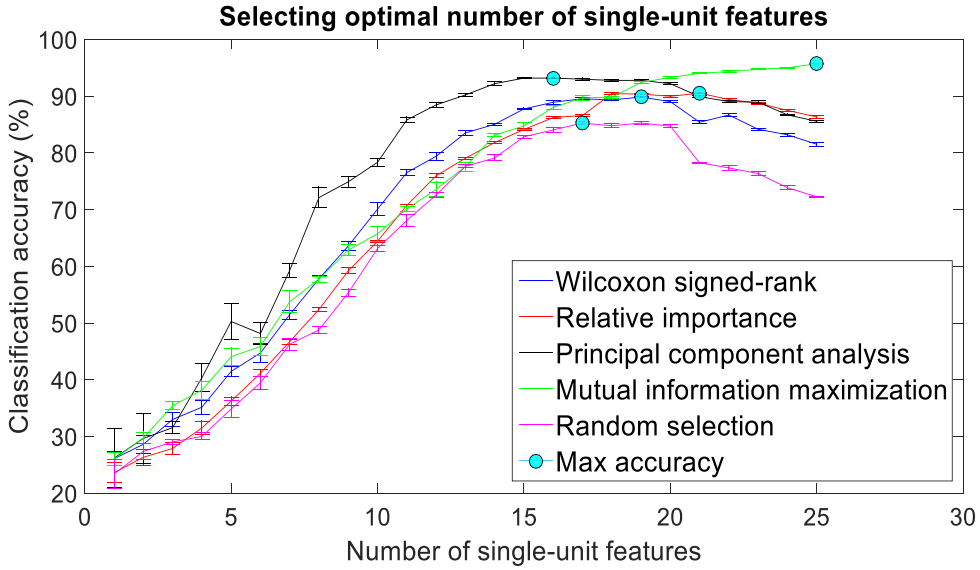


Figure 7 Selecting optimal number of units. The plots above show the cross validated accuracy of feature selection algorithms for increasing number of multi-unit features (Fig. 6) and single-unit features (Fig. 7). The solid circle (cyan) in each graph shows the maximum cross-validated accuracy for a feature selection algorithm. The number of single or multi-unit features corresponding to this accuracy was chosen as the optimal number of features.

The optimal number of features for various feature selection algorithms on post-implantation 36 was differing (Table 1).

Table 1.

Feature selection algorithms and their respective optimal number of features on post-implantation day 36.

	Wilcoxon signed-rank test	Relative Importance	PCA	MIM	Random features
Multi-unit	9 (50.48%)	19 (92.07%)	18 (92.88%)	25 (93.71%)	20 (88.28%)
Single-unit	19 (89.84%)	21 (90.53%)	16 (93.19%)	25 (95.71%)	17 (85.27%)

The values in Table 1 correspond to the optimal number of features (values outside the parentheses) and maximum cross-validated accuracy (values within the parentheses). With an exception of Wilcoxon signed-rank test, the other feature selection algorithms did not show significant changes (Two sample t-test, $p < 0.05$) from using multi-unit and single-unit firing rate both in terms of number of optimal features and classification accuracy (less than ± 3% difference in classification accuracy and ± 1 feature). In case of Wilcoxon signed-rank test, the number of optimal features increased from 9 features for multi-unit firing rate to 19 feature for single-unit firing rate. The classification accuracy improved from 51.12 ± 0.65 % for multi-unit firing rate to 88.12 ± 0.61 % for single-unit firing rate (Figure 6 and 7).

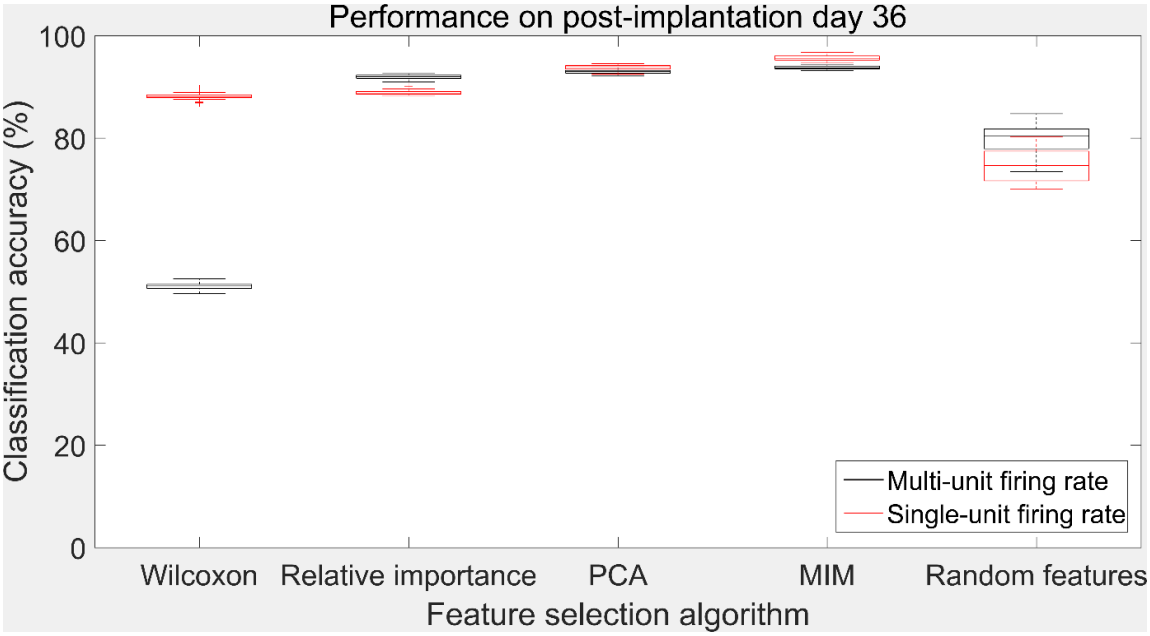


Figure 8 Accuracy of neural decode on post implantation day 36. Classification accuracy of feature selection algorithms on the test set using cross validated optimal number of features. The plots in black and red correspond to classification accuracy obtained using multi-unit firing rate and single-unit firing rate respectively. Level of chance was 10% (10 degrees of freedom). The central box represents the central 50% of the data with the top and bottom sides of the central box representing the 75% quantile and 25% quantile respectively. The central line in the central box represents the median of the central 50% of the data. The vertical lines extending above and below the central box represent the remaining data that are not regarded as outliers.

On post implantation day 36, Mutual Information Maximization performed significantly better than the other algorithms and random selection (Two sample t-test, $p<0.05$). There was no significant difference in the performance of single-unit and multi-unit features selected using Relative Importance and Mutual Information Maximization (Two sample t-test, $p<0.05$). Whereas, single-unit features selected using Wilcoxon signed-rank test and Principal Component Analysis performed significantly better than the multi-unit features selected using the respective algorithms.

Robustness to simulated failure.

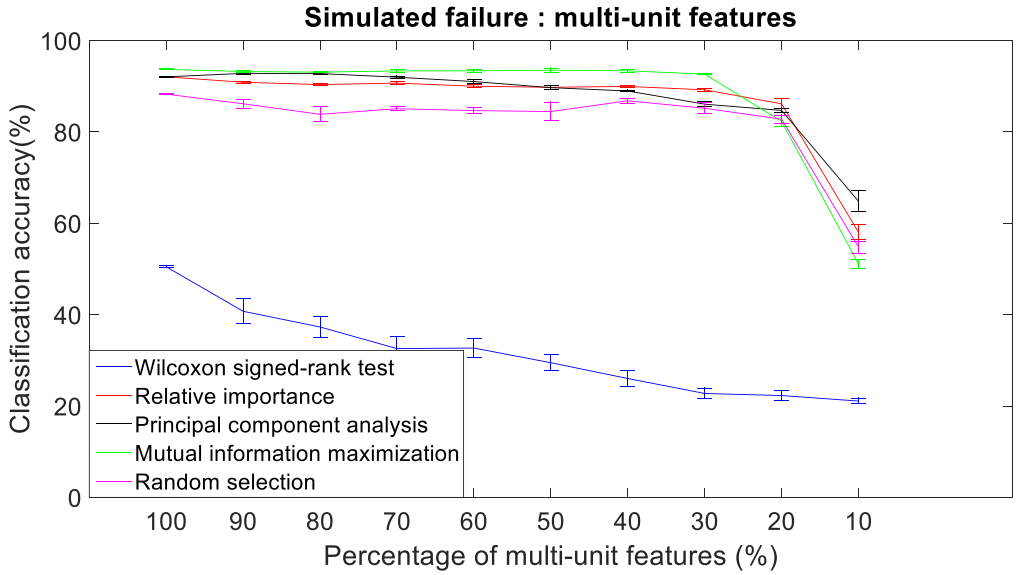


Figure 9 Robustness to simulated failure of multi-unit features. Error bars indicate standard error of the cross validation folds.

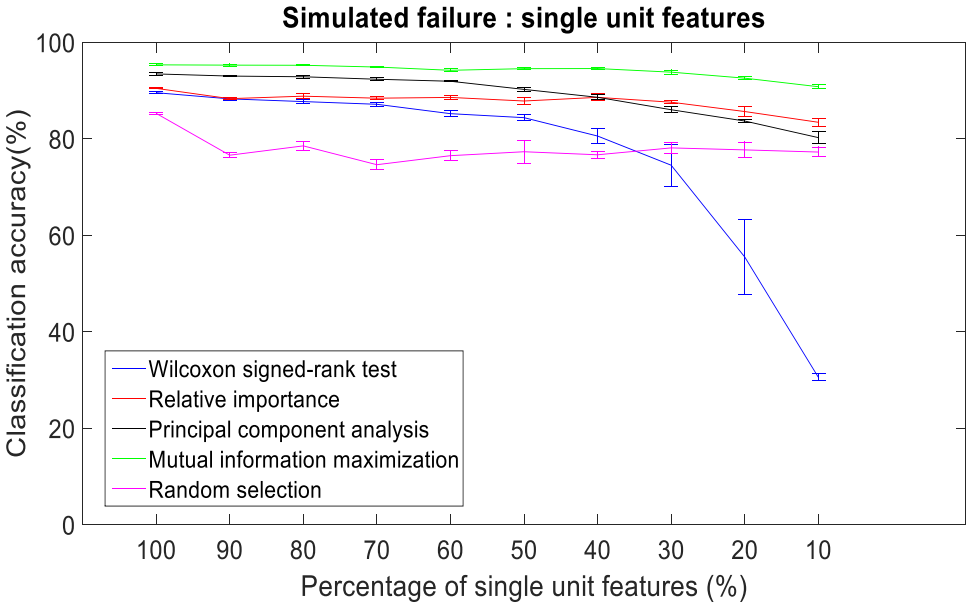


Figure 10 Robustness to simulated failure of units. Mutual information maximization based feature selection had a classification accuracy of 90.79% with just 10% of the neural units as feature vector. Error bars indicate the standard error of classification accuracy.

There is a general trend of decrease in the performance of feature selection algorithms when we decrease the number of features from 100% to 10%. While using multi-unit firing rate, the performance of Principal component analysis was best at 64.82 ± 2.27 % for 10% of channels, whereas the performance of Wilcoxon signed-rank test was 21.08 ± 0.63 %. When we used single-unit firing rate as the feature vector, the robustness to simulated failure was higher for all feature selection algorithms when compared to their respective multi-unit firing rate. In case of Wilcoxon signed-rank test there was a ~10% increase in classification accuracy while there was a ~40% increase in classification accuracy for Mutual information maximization based feature selection. The performance of mutual information maximization feature selection for single-unit firing rate stayed above 90% classification accuracy even while using only 10% of the available units. Mutual Information Maximization based single-unit features performed significantly better than all of the other algorithms for all levels of simulated failure (100% to 10%) (Two sample t-test, $p < 0.05$.)

Longevity of neural decodes.

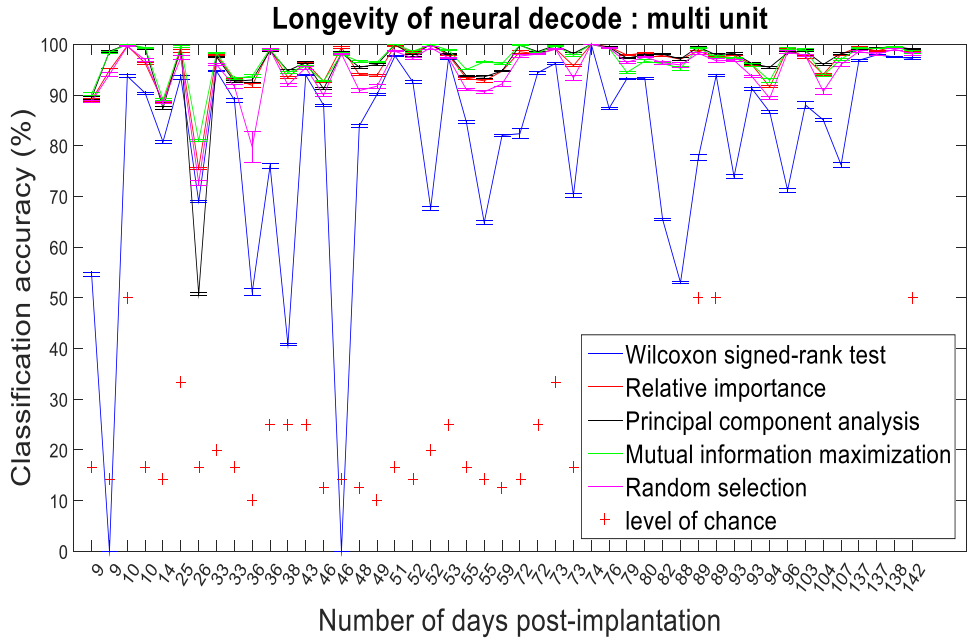


Figure 11 Longevity of neural decodes using multi-unit firing rate. High levels of accuracy over an extended period of time is imperative for a fully functional neuroprosthesis. On certain days, two sessions of recordings were conducted. Repeated x-axis indices (Number of days post-implantation) in the above figure correspond to different sessions conducted on the same day.

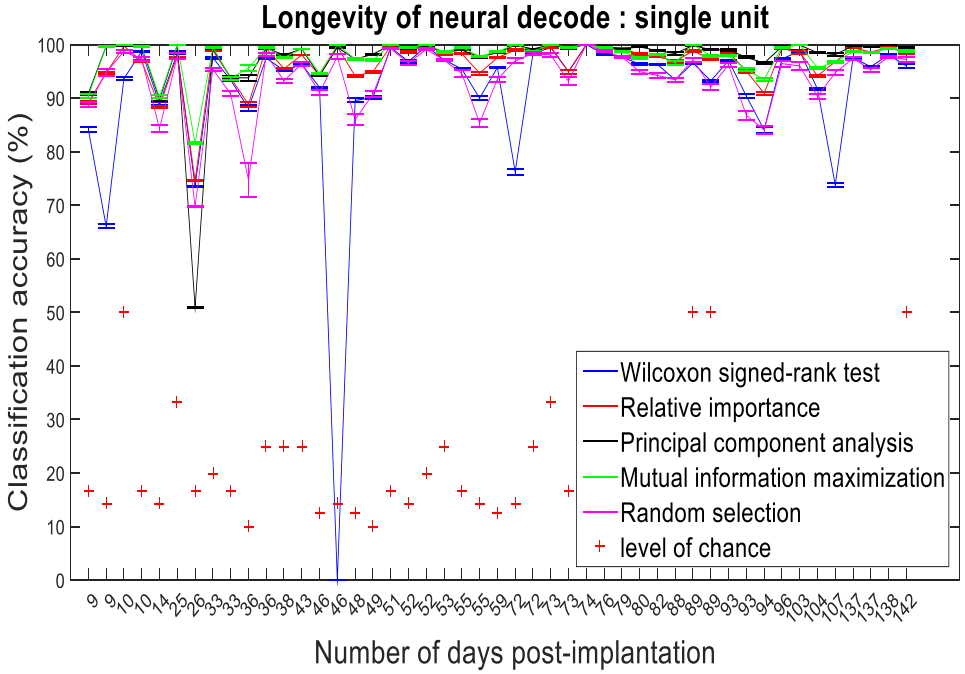


Figure 12 Longevity of neural decodes using single-unit firing rate. The optimal number of features for each feature selection technique was identified using an iterative cross validation scheme. For a given session, 70% of all available successful trials were used as the training set and the remaining 30% were used for testing.

The standard deviation of prediction for mutual information maximization was 7.09 while it was 18.51 for Wilcoxon signed-rank test (for single-unit firing rate). The improvement in classification accuracy from multi-unit to single-unit firing rate requires an action potential isolating pre-processing procedure. Relative importance, principal component analysis and mutual information maximization had comparable accuracies across 47 sessions and performed better than randomly selected features (Figure 11 and 12).

On an average, for 40% of the sessions the feature selection algorithms had no significant difference in performance between single and multi-unit features. For the remaining 60% of the sessions with significant difference in performance between single and multi-unit features, there was an average of 5.8% improvement in accuracy for single-unit features over multi-unit features. Assessing the chronic decoding capability of various feature selection methods, Mutual Information Maximization produced the best results for both single-unit and multi-unit based firing rate. In general, single-unit firing rate feature vector yielded slightly better (~3-4% on average) performance compared to multi-unit firing rate feature vector for all feature selection methods except Wilcoxon signed-rank test. The chronic decoding results also validate the viability of using a neuroprosthetic device with high classification accuracies (> 90% on average and several folds better than level of chance).

Isolating the action potentials from individual neurons is routinely performed on neural recordings from microelectrodes. We have shown that by applying feature selection

techniques to single-unit and multi-unit firing rates, we can get comparable performance over a chronic level. However, utilizing single-unit firing rates demonstrated better performance than multi-unit firing rates when the number of active electrodes decreased.

Discussion

Feature selection is an efficient method to cope with the "curse of dimensionality". As explained in the previous sections, performing feature selection increases the amount of data that is bounded by the mean ± 1 standard deviation region. Reducing the dimensionality of neural data from a few hundred features to an average of 20 features, increases the amount of data bounded by the mean ± 1 standard deviation region exponentially. Therefore, the sparsity of data points in the feature space is reduced. In addition to reducing the sparsity, feature selection algorithms also inundate the feature space with more relevant information based on some criteria. In a way, feature selection can be thought of as a procedure to "prune" the feature space with only "informative" features. All of the feature selection algorithms consistently performed better than the randomly selected features. This significant improvement in performance adds ~10% accuracy in case of both multi-unit and single-unit features compared to randomly selected features. Ideally in real world applications, we would expect the prosthesis to work with 100% accuracy for all different types of movement. In order to increase user compliance and ease of use, feature selection algorithms must yield accuracies as close to 100% as possible. Misclassifications in prediction can impede or in the worst case cause physical damage to the user and/or people around them. Misclassifications in real time prediction can lead to undamaging mishaps that might still be critical in accomplishing tasks such as slips while holding a cup of coffee or other objects that might steer the user away from efficiently using the prosthesis for

activities of daily living. We also speculate that with increasing misclassifications, user acceptance and performance might deteriorate non-linearly.

Feature selection algorithms operate in various mechanisms and perform significantly better than level of chance and randomly selected features. While Wilcoxon signed-rank test, Relative Importance and Mutual Information Maximization retain the innate properties of the feature space (in terms of retaining it in time domain), Principal Component Analysis transforms the features to uncorrelated, orthogonally located principal component axes. It is interesting to note that for certain sessions, PCA has comparable performance as MIM. Exploiting this property of PCA and the noise reduction it provides innately, it will be interesting to program algorithms that do not require re-training for each session. This would be a significant improvement in terms of user experience since training time is usually of the order of 20-30 minutes (performing the training trials and parameter selection for the feature selection and machine learning algorithm) which might be monotonous and tiresome.

In this study, we have tested the feature selection algorithms based on scenarios encountered with real-world neural data. Loss of active single and multi-units over a long duration of time has been observed and reported in various studies. In order to make a neuroprosthesis commercially and practically viable, the algorithm must be robust to handle reduction of available features. We have reported the performance of feature selection algorithms when subjected to a reduced subset of features. We were able to achieve accuracies several folds above level of chance with only 10% of the single-unit features using Mutual Information Maximization based feature selection. One of the main reasons

for the loss of active single and multi-units is due to physiological interactions at the tissue-electrode interface. With technology available at our disposal today, the only way to cope with such physiological interactions might be to replace the micro-electrode array itself. Feature selection algorithms help sustain the performance of the neural decode and maximize the classification accuracy when encountering such intractable conditions.

Neural decoding algorithms must also be reliable over a long duration of time. In this paper, we also present results of various feature selection algorithms over 47 sessions of neural decode. Across all of the sessions, single-unit and multi-unit features had comparable performances for multiple types of movements. According to our results, for 60% of the sessions there was a significant difference between the performance of single and multi-unit features. Although there is an average ~5% increase in performance when using single-unit features, it comes with a trade-off of expensive computation. This computational latency can also manifest in the form of execution delays of the neuroprosthesis while performing a task which might directly affect user performance. We speculate that Mutual Information Maximization performs better across all three performance metrics as it maximizes the class conditional entropies of features in the predictor space. Future analysis will investigate the stability of neural decodes. Stability of neural decodes refers to the performance of a trained model over time without updating the model. The stability of neural decoding models will impact how often a user will need to retrain the classifier model.

Acknowledgements

This research was supported by DARPA Revolutionizing Prosthetics 2009, BAA05-26 and ASU-Mayo Seed Grant. Results included in this paper were presented as part of Master's thesis at Arizona State University (Padmanaban, 2015)

Ensemble methods

Abstract

Objective Peripheral nerve interface based neural prostheses provide a promising platform to accomplish dexterous control in people with an amputation without the surgical risks associated with intracortical neural prosthesis. We investigated the offline implementation of boosted ensemble regression for predicting movement trajectories. Strategies for achieving robust dexterous control using peripheral neural signals from two trans-radial amputees are discussed. *Approach* Peripheral neural signals were recorded using intrafascicular microelectrode arrays implanted in the median (patient 1) and ulnar (patient 2) nerve while subjects performed volitional movements of their phantom fingers. We predicted finger position using an ensemble based approach with regression trees as base learners and neural firing rate as the feature vector. *Main Results* The average mean squared error of the movement trajectory predicted using Ensemble regression on the best decode day (post-implantation day 24) was 0.01 ± 0.0002 while that of Kalman Filter was 0.08 ± 0.03. *Significance* Mean squared errors from the ensemble method were consistently lower than the mean squared errors from a Kalman Filter during the entire duration of implantation. The performance of ensemble regression model may prove to be an approach to dealing with complex/non-linear/non-stationary/non-ergodic neural data that provides significantly better performance (decoding accuracy) than linear approaches, e.g. Kalman Filter for brain-machine interfaces.

Introduction

Volitional dexterous finger movements play an important role in performing a wide variety of tasks in daily life. The functionality to perform individual and coordinated finger movements is in part mediated through the primary motor cortex, the pyramidal tracts to the ventral horn of the spinal cord, and the finally the peripheral nerves originating in the ventral horn.

Neural decoding for transforming neural signals to control signals for actuation of prosthesis is a critical component in designing a brain-machine interface. With a plethora of decoding algorithms, experimenters design their neural decoding system based on the decoding parameter (velocity, position, hand postures), neural signal to be processed (EMG, ECoG, LFP, Action Potentials), and decoding algorithm (instance based learners, probabilistic methods, tree based learners, neural networks). From a neural decoding perspective, early attempts to establish a brain-machine interface aimed at controlling a cursor (Ganguly & Carmena, 2010; Gilja et al., 2010; S. Kim et al., 2008) which used relatively simple decoding algorithms like linear filter, Kalman filter, Wiener filter and Linear Discriminant Analysis. Moving forward in terms of complexity of decoding parameter, gross upper limb motor movements like reaching and grasping (Hao et al., 2014b; S. Kim, Sanchez, Rao, Erdogmus, Carmena, Lebedev, Nicolelis, & Principe, 2006b; Marathe & Taylor, 2013; Shimoda, Nagasaka, Chao, & Fujii, 2012) were decoded using a variety of algorithms such as Wiener Filter, Kalman filter, Linear regression, multivariate partial least squares regression and Support vector machine. Recently, several brain-machine interfaces have aimed at decoding more distal and subtle parameters such

as hand postures, position and velocity of finger movements(Chestek et al., 2013b; Flint, Lindberg, Jordan, Miller, & Slutzky, 2012; Micera et al., 2010; Wissel et al., 2013b) using algorithms such as Linear Discriminant Analysis, Naïve Bayes decoder, Support Vector Machine and Hidden Markov Models.

Amputees without underlying neurological disorders constitute a significant patient population that can benefit from a neuroprosthesis. In such cases, the peripheral nerve interface offers a perfect platform to restore motor functions using a neuroprosthesis without the surgical risks and hazards of an intracortical brain-machine interface. Peripheral nerve interface based neuroprostheses have gained more attention in recent times to restore motor functions in patients with upper extremity amputation(Clark et al., 2014; Dhillon & Horch, 2005b; Garde, Keefer, Botterman, Galvan, & Romero-Ortega, 2009; Horch, Meek, Taylor, & Hutchinson, 2011; K. H. Polasek, Hoyen, Keith, & Tyler, 2007; K. Polasek, Schiefer, Pinault, Triolo, & Tyler, 2009; Tan et al., 2014b). Targeted muscle reinnervation is a surgical technique which transfers residual arm nerves to alternative muscle sites where EMG signals recorded from the surface of the reinnervated skin can be used to control a prosthesis (Kuiken et al., 2009). Finite state controllers have been used in myoelectrically controlled ankle foot prosthesis to assist trans-tibial amputees (Au, Berniker, & Herr, 2008). In our previous study, we demonstrated the viability of a neural prosthesis that interfaced the prosthetic limb directly with efferent and afferent fibers in the median/ulnar nerve using an array of intrafascicular microelectrodes (Davis et al., 2016). In this study, finger movement trajectory was predicted from neural activity recorded for 13 different finger movements using a Kalman filter. Providing dexterous control can help improve the ease and usability of a neuroprosthesis. We present here an

offline implementation of a decoding model with a potential to control 13 different finger movements through a peripheral nerve based control of a prosthesis.

Materials and methods

This study was approved by the University of Utah Institutional Review Board and the Salt Lake City Veterans Affairs Hospital Research and Development Service Center.

Data Collection and Pre-processing.

The results reported in this paper were obtained by using the data from the previous study. The pre-study enrollment period, nerve electrode arrays, surgical procedures, data collection and training methods used in this study are described in the previous study (Davis et al., 2016). The Utah Slanted Electrode Array has been described elsewhere and consists of ninety-six electrodes that project out from a 4 x 4 x 0.3 mm substrate (Branner, Stein, & Normann, 2001). The two volunteers with previous transradial amputations underwent implantation for one month with a USEA into their median (Subject 1, 31 yrs post-amputation) and ulnar (Subject 2, 1.5 yrs post-amputation) nerves. On an average of three times a week, patients 1 and 2, underwent individual sessions of electrophysiological recordings and microstimulation. Neural signals were recorded and amplified using active head-stage cables (ZIF – Clip 96 channels, Tucker Davis Technologies, Inc., Alachua, FL). The head-stage cables connected to a custom-built board used to interface the TDT-

connector with a Neuroport data acquisition system (Blackrock Microsystems, Salt Lake City, UT, USA). Data was collected using Cerebus software (Blackrock Microsystems, Salt Lake City, UT, USA). Neural signals were band pass filtered with cutoff frequencies of 0.3 Hz (1^{st} order high-pass butterworth filter) and 7500 Hz (3^{rd} order low-pass butterworth filter) and sampled at 30 kHz. Online multi-unit activity was extracted from high-pass filtered data (250 Hz 4^{th} order butterworth filter) by setting a threshold using the auto threshold setting in the Cerebus data acquisition software. Neural firing rates were calculated using unsorted spikes and a moving box-car average of 300 ms with an update period of 33 ms. The patient data used in this offline analysis was obtained from transradial amputees who performed volitional phantom finger movements based on mirroring the healthy hand or by following a virtual robotic hand on screen. Thus, the regression was performed on instruction variables provided to the patients and not on the true position of the fingers.

Previous study.

In the previous study, we constructed a closed-loop peripheral nerve interface based neural prosthetic system by having individual sessions of electrophysiological recording and microstimulation. A Kalman filter based decoding algorithm was employed for controlling a virtual robotic hand on-screen. Wilcoxon signed rank test was used to select channels for neural data from 20 trials to train the Kalman filter. The Kalman filter showed good (R = 0.9) Pearson's correlation coefficient for 2 DOF in patient 1 and 4 DOF in patient 2. As the algorithm inherently assumes a linear relationship between the input (neural data)

and output (movement trajectory) and due to variation in neural firing rates on different channels for different degrees of freedom, Pearson's correlation coefficient of the neural decode dropped for higher degrees of freedom. Also when including higher degrees of freedom for online decoding, the experimenters found that the Kalman filter decode produced unstable, random perturbations in the rest phase.

Feature Selection.

Using machine learning algorithms for continuous data can be computationally expensive when the size of the predictor space is huge (>=100). Feature selection is a commonly used technique where we select only the near optimal *o*-dimensional subset of features (channels of neural data) from a *w*-dimensional feature space that help predict responses with maximum accuracy based on some parameter. In case of our neural data, there were 96 features each corresponding to the neural firing rate on each electrode. In order to reduce information redundancy and trim the feature space, we applied a feature selection technique to select channels that would help the algorithm better predict the responses. As explained in the previous study (Davis, Wark, Hutchinson, Warren, O'Neill, Scheinblum, Clark et al., 2016a), we employed Wilcoxon-signed rank test to determine significant changes in the neural firing rate of channels between "baseline" and "movement" periods for all the available degrees of freedom. Application of statistical and empirical methods to select variables is termed as *Filter based feature selection.* Filter or criteria based feature selection methods are independent of the learning algorithm itself.

K-Fold Segmentation.

The performance of a machine learning algorithm is evaluated by dividing the data into training and testing sets. The regression model was trained using data from the training set. The performance of the algorithm was evaluated based on its ability to correctly classify unseen data points from the test set. Mean squared error was used as a metric to quantify the performance of the regression. The method of cross validation is an intuitive approach where the entire dataset is divided into chunks or folds. If the data is segmented into *k-folds,* then the performance metric is measured 'k' times where every 'k^{th} fold' serves as a testing set once while we use the remaining 'k-1 folds' as the training set. Thus, the performance of the classifier can be reported by averaging across *k-folds.* The k-fold cross validated classification accuracy is given by

$$\text{CV accuracy} = \frac{\sum_{i=1}^{k} g(i)}{K} \tag{2.1}$$

Where, K – number of folds

$$g(i) = \frac{\sum f(y=x)}{N} \tag{2.2}$$

Where, $\sum f(y = x)$ - number of correctly classified points from the test set.

N – Number of points in the test set

Throughout the decoding analysis, a ten-fold cross validation scheme was followed. This approach of cross validation is called as complete k-fold cross validation and it is a good approach to measure the true generalization error of the classifier (Kohavi, 1995). We

used a 10-fold cross validation and averaged the performances of the individual folds to obtain the final classification accuracy.

Neural decoding system.

The neural decoding system consisted of an ensemble of classification and regression trees to predict the movement trajectory from neural firing rate. The ensemble of classification trees predicted the discrete class of finger movement performed, while the ensemble of regression trees predicted the continuous movement trajectory (position of the virtual robotic finger). Such a cascade of ensemble of classification and regression trees is essential for producing an *n*-dimensional output, where 'n' is the number of different finger movements available in a session. An ensemble is an aggregation of many weak learners, where the objective of building each weak learner is to capture some of the input-output relationship which is not necessarily captured by other weak learners. Finally, the output of the weak learners was averaged (or weighted depending on the importance of each weak learner).

Decision tree induction. A decision tree is a supervised learning algorithm that generates a set of rules based on some criteria and performs stepwise splitting to divide the feature space into smaller partitions until a stopping condition is met. Decision trees can be classified into classification trees and regression trees depending on the output 'y' being predicted. If the output 'y' is a continuous variable such as housing prices, temperature or stock prices, then the decision tree is called a regression tree. If the output 'y' is a discrete variable such as blood group type, email segregation (primary, promotions and updates) or

user ratings (scaled along 1-5), then the decision tree is called a classification tree. The most widely used approach to construct a decision tree is through a greedy algorithm called C4.5. Nodes are the constituent blocks of a decision tree where some operation is performed. Based on the operation performed, nodes can be divided as root node, internal nodes and terminal/leaf nodes. The root node is a collection of all training examples where there are no incoming splits (root node is the structure of a tree before any splitting is performed). A terminal or leaf node is the end point of splitting along a branch, where there are no outgoing splits. All nodes that are not a root node or a terminal node are called internal nodes where decisions are performed. An internal node on which a split is performed is called a parent node while the nodes resulting from the split are called child nodes. The construction strategy of this approach can be summarized as follows:

1. Start at the root node by collecting all training examples. Let us assume we have an input X which is m x n dimensional, where 'm' is the number of features and 'n' is the number of instances (samples or seconds) and an output Y which is 1 x n dimensional.

2. A decision tree is built by performing iterative splitting based on some impurity measure. Popular impurity measures include entropy, misclassification rate and GINI index for classification and standard deviation for regression. Since the C4.5 algorithm uses a greedy approach, at each step the best split is identified. For example, in our training data we have 'm' features X_1, X_2, … X_m from which the algorithm chooses the best split by iteratively trying each feature and computing the impurity measure of choice.

3. The algorithm continues recursively splitting nodes until some stopping condition is met.

4. Popular stopping conditions include assigning a threshold on the number of instances in an internal node below which splitting is not performed. The rationale behind doing this is to prevent data fragmentation, which is a phenomenon that arises when decision is made based on too few instances.

5. Once a stopping condition is met, the value for the particular (leaf) node is assigned as the mode of output values in that node for a classification tree and average of output values in that node for a regression tree.

6. Recursively split all nodes until the stopping condition is met.

7. To predict the response of a test instance using decision tree, traverse through the structure based on the node rules until you reach a leaf node. The response for the test instance is the value assigned to this leaf node

Ensemble methods. Let variables 'x' and 'y' denote the true input and output respectively. Assume we are trying to fit a function 'f(x)' to capture the relationship between 'x' and 'y' (which is slightly non-linear). A straight line might not be complex enough to capture the relationship between 'x' and 'y'. A model is said to "underfit" when the predicted response is not complex enough for the actual x-y relationship. In contrast, a 4^{th} order polynomial might be too complex for our set of data points (Figure 5C). Thus, the model is "overfitting" the x-y relationship. A quadratic fit might be balanced and provides a fair trade-off between complexity and quality of fit. Error (or noise) introduced due to underfitting is called *Bias,* while the error caused due to overfitting is called *Variance.*

Mathematically, bias and variance is defined as

$$Bias = [E[f(x)] - y]^2$$

$$Variance = \mathrm{E}[(\mathrm{f(x)} - \mathrm{E[y]})^2]$$

Therefore, prediction error can be divided into three parts: bias, variance and irreducible error. Bias and variance are dependent on the complexity of the model while irreducible error is dependent on the inherent properties of the system being modeled. Irreducible errors are often intractable errors introduced into the input or output data that influence the quality of prediction (for example measurement error from the sensors, precision of a device).

Prediction error = Bias + Variance + Irreducible error

Prediction errors can be reduced by incorporating methods that focus on reducing the variance of an algorithm. One such approach is to build an ensemble of *base learners*, whose predictions when averaged results in a reduced variance model. Base learners are the building blocks of an ensemble. Depending on the problem, hundreds (sometimes, thousands) of base learners are trained and the final prediction of this ensemble is obtained by averaging across the individual predictions of the base learners. Averaging the responses of the individual base learners reduces the variance component of prediction error, thus, improving the prediction quality of the ensemble.

Bagging (**b**ootstrap **ag**gregating) and boosting are two common methods to building an ensemble of base learners. In *bagging,* a subset of the training set is randomly selected with replacement and used for training a base learner. Many such bootstrap samples are

generated and used to train the base learners which will be slightly different from each other. The final result of the ensemble is obtained by averaging the individual predictions of the base learners. We employed a bagging ensemble of classification trees to predict the type of finger movement performed for every testing instance. The algorithm for a bagging classification ensemble can be summarized as follows:

1. Let 'k' be the number of bootstrap iterations.
2. Create bootstrap datasets D_1, D_2 … D_k.
3. For every bootstrap dataset, train a classification tree C_i.
4. For a given test instance, predict the mode of the predictions of the individual classification trees.

The bagging ensemble classifier was built using 100 classification trees. The individual trees were built on the bootstrap datasets obtained from the training set by using sampling with replacement.

In *boosting,* the entire training set is used to train the first base learner. Initially, the weights of each training instance are initialized to 1 (equal). Depending on the training error of the first base learner, the weights of those instances which were misclassified are increased. For the second base learner, a weighted sampling is performed which results in the misclassified instances having a higher probability of selection. The same procedure is repeated until all training instances are classified correctly by at least one base learner. The final prediction of the ensemble is obtained by weighting the responses of the individual learners based on their training errors (Dietterich, 2000; Drucker, Cortes, Jackel, LeCun, & Vapnik, 1994; Opitz & Maclin, 1999; Quinlan, 1996). We employed a boosting ensemble

of regression trees to predict the position of finger movement. The algorithm for boosting can be summarized as follows:

1. Initialize weights for each training instance to 1.
2. Let ‘k’ be the number of boosting rounds.
3. For every round of boosting, create a dataset D_i by weighted sampling with replacement from the original training set.
4. Train a regression tree R_1 by using dataset D_1.
5. Obtain the training error, ε_1, of classifier R_1.
6. Update the weights of the training instances that was misclassified by R_1.
7. Compute the ‘importance’, α_1, of the regression tree R_1.

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1 - \varepsilon_i}{\varepsilon_i}\right)$$

8. Repeat the procedure for the subsequent rounds of boosting.
9. For a given test instance, obtain the final response of the boosting ensemble by weighted (based on the importance of each regression tree) averaging of responses of the regression trees.

Regularization. One of the disadvantages of predictive models built using data is the issue of overfitting. Regularization is a method employed to reduce the variance (overfit) of the model. In the context of boosted ensemble regression, we chose *shrinkage* method for regularizing the ensemble. Shrinkage reduces the impact of each additional weak-learner added to the ensemble model. The simplest form of regularization through

shrinkage is the direct proportional shrinkage. In this case, the effect of shrinkage is directly proportional to parameter $\lambda \in [0,1]$. In general, the smaller λ, the lower is the shrinked boosted increments are, the better is the generalization capability of the ensemble. Parameter λ was determined by exponentially stepping through various values. The λ with least mean squared error was identified and used for determining the shrinkage level. The downside of a good regularization measure is the increase in training time. We did not observe any significant changes in training duration with the values of λ obtained during cross validation. The procedure for determining the optimal λ is as follows:

1. Select values of $\lambda \in [0,1]$ by exponentially stepping between the defined limits (10^{-5}, 10^{-4}, …, 1).
2. For a trained model, multiply the value of λ to the importance, α, of the individual trees.
3. Evaluate the mean squared error using 3-fold cross validation of the training set.
4. Repeat steps 2 and 3 for all values of λ and select optimal λ.

Performance metric. We chose mean squared error as a performance metric to quantitatively assess the decoding results of the ensemble regression. Mean squared error in its simplest form refers to the mean of squared difference between the *known* (actual movement trajectory) value and *predicted* (output of the ensemble regression) value.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \theta_i)^2$$

Where Y_i is the known value and θ_i is the predicted value.

Parameter selection.

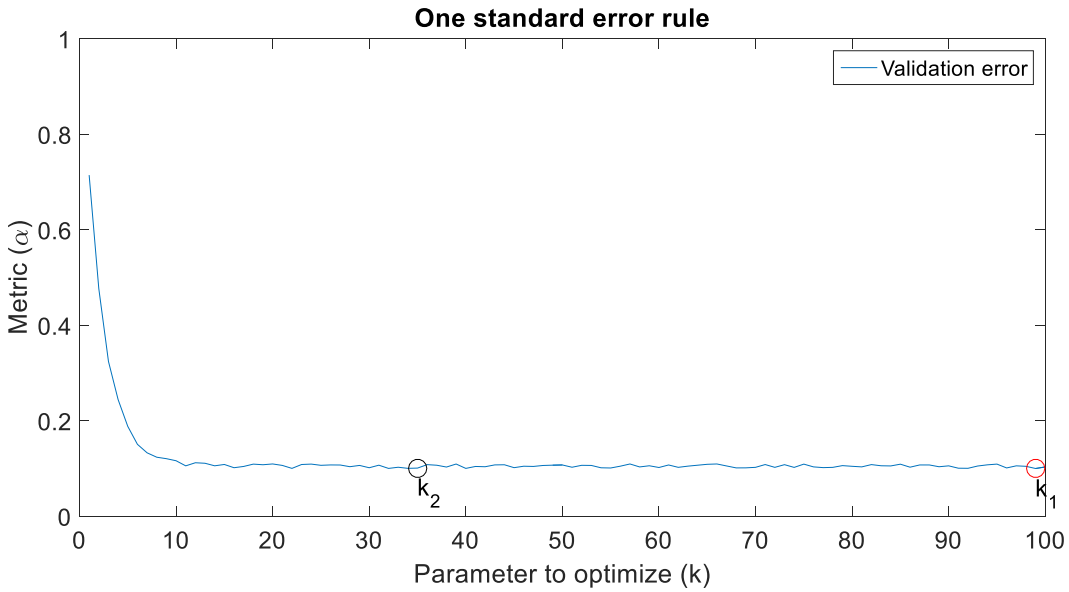


Figure 13. One standard error rule. The plot above shows a simulated response for selecting a parameter 'k' (for example, number of trees in an ensemble method, regularization parameter 'λ' of a regression ensemble or parameter 'σ' of an RBF kernel) based on some metric 'α' (for example, metrics such as misclassification rate, mean squared error or correlation coefficient). K_1 marked with a red circle shows the minimum value of 'k'. K_2 depicted with a black circle shows the minimum value of 'k' that is one standard error within K_1 in the direction of a simpler model. Choosing K_2 over K_1 results in building a simpler model with better generalization.

Oftentimes when constructing a machine learning algorithm, we have to estimate optimal values for various parameters of that algorithm. The efficiency and generalization capability of an algorithm depends on the parameters tuned for a particular problem. Unfortunately, the only way to estimate the optimal parameters of a machine learning problem requires monitoring the performance of the algorithm on a validation dataset which can be expensive in terms of time and resources. As a general rule of thumb, the training and validation error improve with training (Figure 13). For example, let us assume

we are trying to select the optimal value of an arbitrary parameter 'k' of a machine learning algorithm (for example, number of trees in an ensemble method, parameter 'λ' for regularization or parameter 'C' which controls the trade-off between misclassification rate and maximum-margin distance for an SVM) by measuring its performance by computing a metric 'α' (for example, metrics such as misclassification rate, mean squared error or correlation coefficient) on a validation dataset. The validation error decreases (almost exponentially) to a certain point and then metric 'α' asymptotes to 0.1 from k = 9. K_1 (marked with a red circle) refers to the minimum value of parameter 'k'. Selecting the minimum value as the optimal parameter in many cases will result in building a complex model for negligible increase in performance. There are disadvantages to building a complex model. Complex models overfit the data, which means they do not have good generalization of the input-output relationship and are relatively longer to train than simple models.

One of the approaches to selecting the optimal parameters of a machine learning algorithm to promote choosing a simpler model is by using the *One-standard error rule*. According to the one-standard error rule, we move in the direction of simpler model until the parameter 'k' is within one standard error from the minimum (K_1 in this case). K_2 (marked with a black circle) denotes the value of parameter 'k' which is one standard error farthest away from K_1. For ensemble regression, we selected optimal parameters for identifying the number of trees in the ensemble and parameter 'λ' for regularization of the ensemble.

The parameter selection was performed in a nested cross validation loop. The entire data set was divided into ten folds. For each iteration, one fold was used as a testing set while the remaining nine folds were used as the training set. Parameter selection was performed on the training set using k-fold cross validation by dividing it into 3 folds. The parameters estimated using the training set was used to predict the movement trajectory for data in the tenth fold. This process was repeated ten times until each fold was used as a testing set once.

Neural decoding architecture.

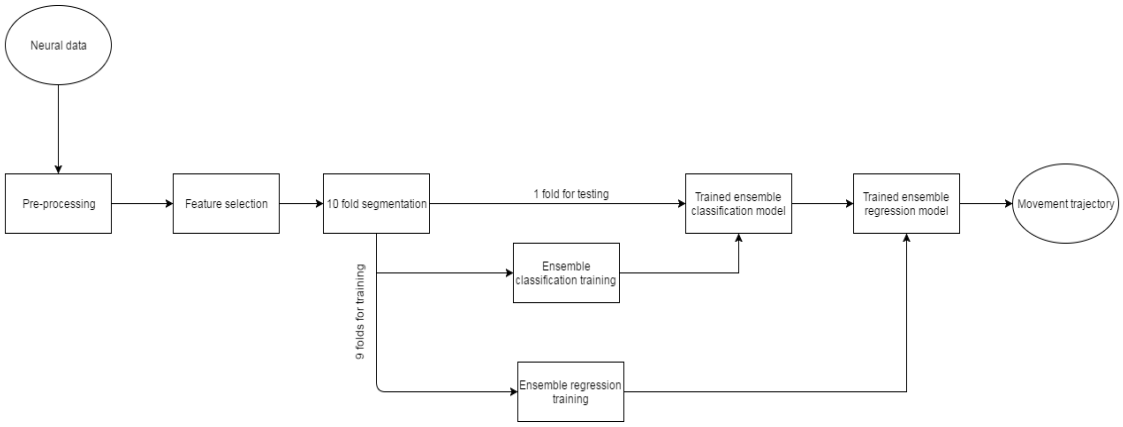


Figure 14. Flow chart of offline decoding analysis of peripheral neural signals. Neural data was used as input. Pre-processing involved using an offline software Offline Sorter (Plexon, Inc) for sorting neural units from noise. Neural firing rate was computed using a moving average box-car window of size 300 ms and step width 33 ms. Wilcoxon signed rank test was used as a feature selection metric to sort channels based on the difference between neural firing rate during movement period and baseline period. 10 fold segmentation was applied to split the data into training and testing set. The boosted ensemble regression was trained using the training set. The trained model was later used to predict movement trajectories for the testing set.

1. Neural data: Patients performed 13 different finger movements instructed based on visual cue. The recorded neural data was sampled at 30 kHz. Instruction variables used for cuing the patients was also recorded. For patient 1, the position of

finger pads of a small manipulandum was recorded. For patient 2, the instruction signals provided to the virtual robotic hand was recorded.

2. Pre-processing: Action potentials were sorted offline using an expectation-maximization based competitive mixture of t-distributions decomposition algorithm. The time stamp of action potentials was downsampled to 15 Hz and a boxcar moving average was performed to obtain the neural firing rate.

3. Feature selection: Neural firing rate on each electrode corresponding to each finger movement period was compared with rest/baseline periods using a Wilcoxon signed-rank test. Based on the test scores, the top 90% electrodes were selected as input to the machine learning algorithm.

4. K-fold segmentation: Neural firing rates corresponding to successful trials of 13 different finger movements and rest period were used as input to the machine learning algorithm. Position of finger pads of the manipulandum (patient 1) or instruction signals to the virtual robotic prosthesis (patient 2) corresponding to respective trials was used as output. Stratified 10-fold segmentation was performed on the available successful trials of each finger movement. During one iteration, trials corresponding to 9 folds were used for training the algorithm while the 10^{th} fold was used for testing. The process was repeated until each fold was used as a testing set. Final mean squared error was obtained by averaging the performance across all testing

sets. During training, validation was performed using a three-fold cross validation of the training data to estimate optimal parameters for the ensemble regression.

5. A. Ensemble classification: Information about type of finger movement (thumb flexion, ring extension and so on) and position of that finger movement is required to obtain complete dexterous control. Boosted classification tree ensembles were used for identifying the type of finger movement performed. The output labels were created manually by labeling each finger movement (Thumb flexion corresponds to '1', ring flexion to '2' and so on). This multi-class classification problem was solved by using neural firing rates as input and manually created labels as output to a boosted classification tree ensemble.

B. Ensemble regression: Information about position of the finger movement was obtained by using a boosted regression tree ensemble. Neural firing rates were used as inputs and instruction variables were used as outputs to this ensemble.

The results for Kalman filter decode were generated using the same architecture. For a given test instance, the type of finger movement was obtained from the trained ensemble classification while the position of the finger movement was obtained by using the trained ensemble regression. Cascading ensemble classification and ensemble regression helped alleviate the problem of generalization across multiple degrees of freedom.

Results

Quality of neural recording.

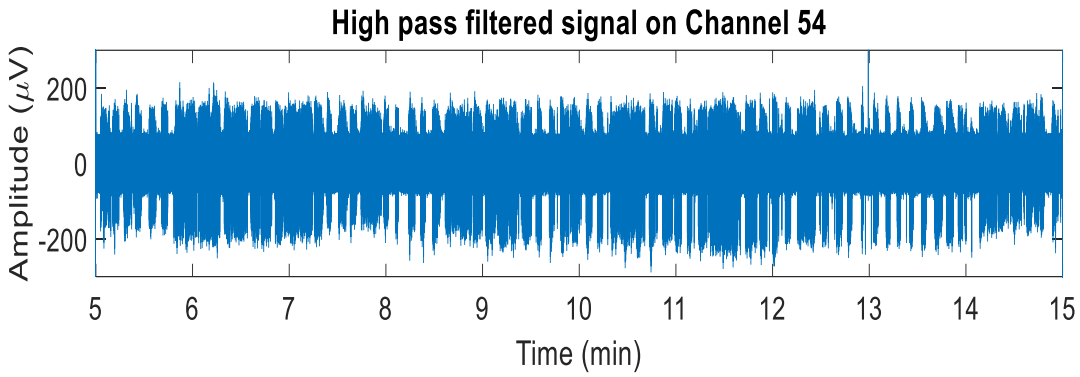


Figure 15. Raw neural recordings from post-implantation day 24. Raw neural recording during a session corresponding to multiple trials over 10 minutes was plotted to illustrate the quality of raw neural signals. The neural signal was filtered using a Butterworth high pass filter with a cut-off frequency of 300 Hz.

15 sessions of neural recording and microstimulation sessions were performed on patient 2. For patient 2, the two-stage decoding model was employed offline on all available days of neural recording (7 sessions). We could not perform offline analysis on data collected from patient 1 as the patient's connectors were disconnected due to a mishap. The impedances, signal to noise ratio and isolated neural units are described elsewhere (Davis et al., 2016a).

Parameter selection.

Boosted Ensemble regression

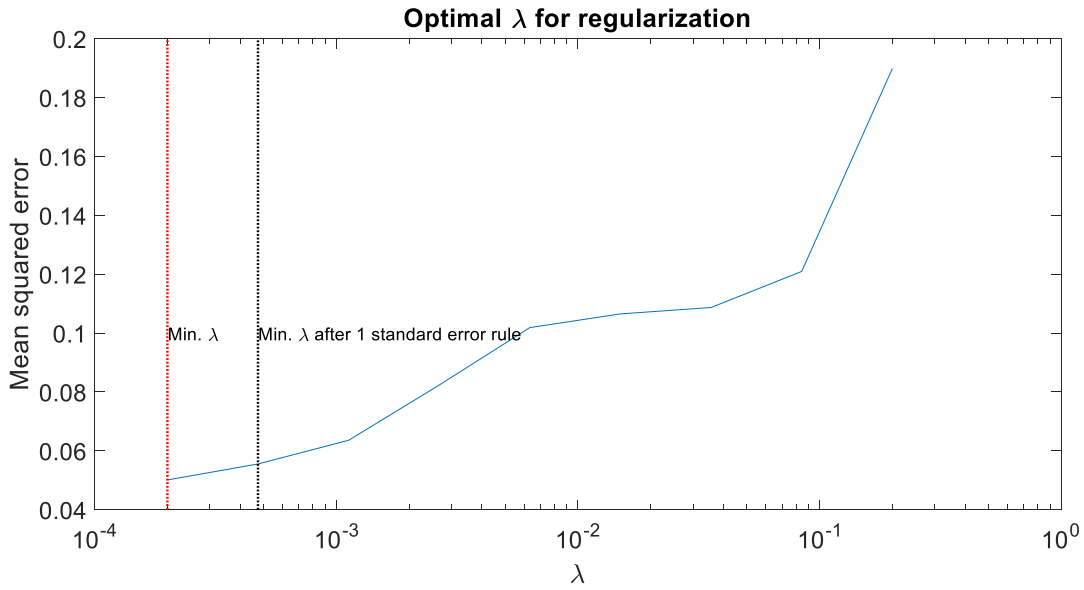


Figure 16. Parameter selection for optimal λ. Optimal values for parameter 'λ' was selected using the one standard error rule for regularizing the Ensemble regression. A 3 fold cross validation was performed on the training dataset to determine the generalization error. Optimal λ was selected based on mean squared error as the performance metric.

For the boosted ensemble regression, we performed parameter selection to determine the optimal value of regularization. Regularization controls the amount of impact each weak learner has on the overall ensemble. Optimal value of λ was chosen by computing the mean squared error within a range of 10^{-4} to 10^{0}. We identified the optimal value of λ by using the one standard error rule. After exponentially stepping through various value of λ, we computed the minimum error (denoted by the red dashed line). Then, we moved in the direction simpler model and found a value within one standard error of the minimum error. In this case, moving along the direction of increasing λ means more regularization of the weak learners, which corresponds to building simpler models.

Introducing λ terms can be thought of as adding an additional weighting term to the output of each individual weak learner.

Decoding performance.

Kalman Filter.

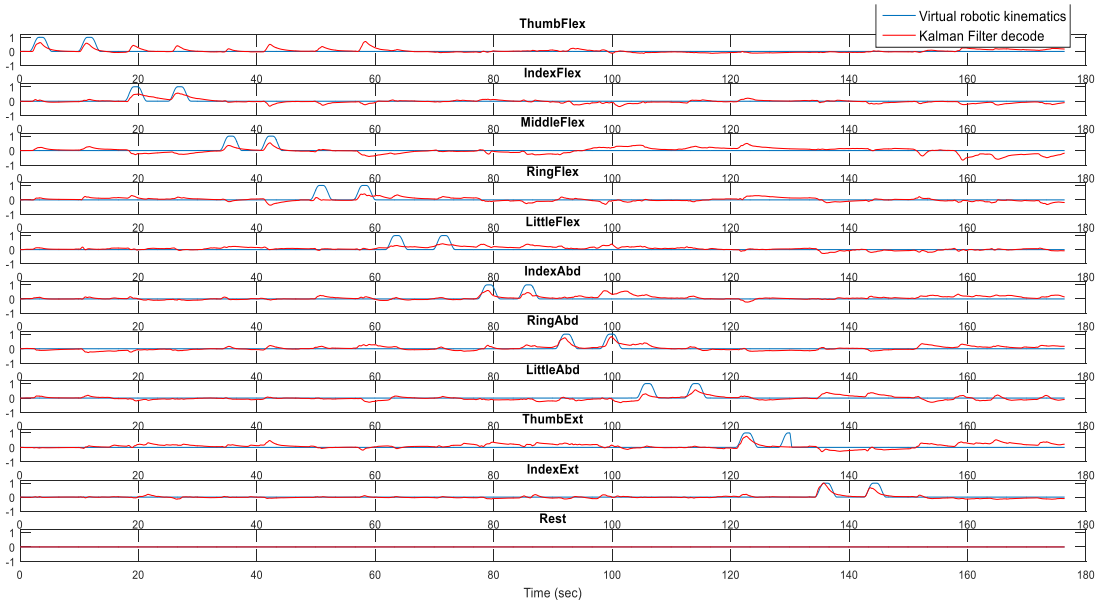


Figure 17. Kalman filter regression. Decoding 11 degrees of freedom including rest phase using Kalman Filter on neural data from post-implantation day 24 in patient 2. The blue lines correspond to the virtual robotic kinematics which was used to move the cursor on-screen. The red lines correspond to the Kalman filter decode performed on a validation set.

The Kalman filter was trained and testing using all the available degrees of freedom in a given session. It is important to note that the output of the Kalman filter is multidimensional with the number of dimensions of the output equal to the number of degrees of freedom available (Figure 17). In other words, the Kalman filter maps a $\mathbb{R}^{mxn}$ input where 'm' is the number of channels (electrodes) and 'n' is the number of training dataset observations (instances) to a $\mathbb{R}^{p\ x\ q}$ output where 'p' is the number of types of movement and 'q' is the number of testing dataset observations (instances). On post-implantation day 24, we used data from 15 electrodes and decoded 12 different types of

movement. The number of training observations for a given fold was 21000 while the number of testing data had 2300 observations. Therefore, the Kalman filter was mapping a $\mathbb{R}^{15 \times 21000}$ input data to a $\mathbb{R}^{12 \times 2300}$ output data. The mean squared error of the neural decode on post-implantation day 24 was 0.08 ± 0.003.

Ensemble Regression Model.

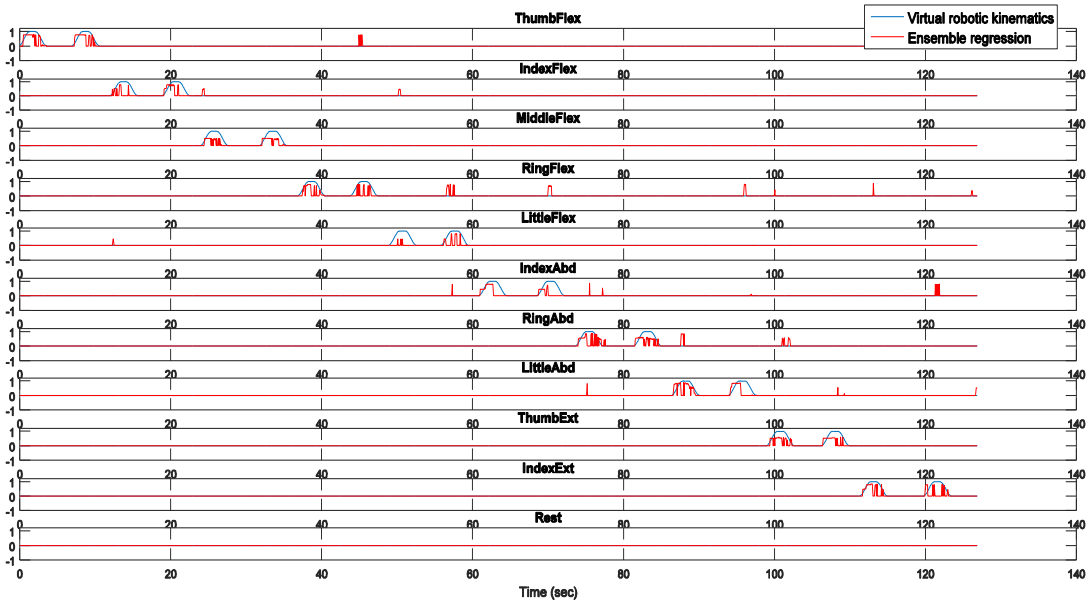


Figure 18. Ensemble regression decoding model. Decoding 11 degrees of freedom using ensemble regression on neural data from post-implantation day 24 in patient 2. The blue lines correspond to the virtual robotic kinematics while the red lines correspond to the ensemble regression.

The ensemble regression model was trained and tested on 11 DOF available on post-implantation day 24. Unlike the Kalman filter, the output of ensemble regression is unidimensional. The ensemble regression maps a $\mathbb{R}^{m \times n}$ input where 'm' corresponds to the number of channels (electrodes) and 'n' corresponds to the number of training dataset observations (samples) to a $\mathbb{R}^{1 \times q}$ output where 'q' corresponds to the number of testing dataset observations. On post-implantation day 24, we used data from 15 electrodes and decoded 12 different types of movements. Therefore, the ensemble regression was mapping

an input data of size $\mathbb{R}^{15 \times 21000}$ to an output data of size $\mathbb{R}^{1 \times 2300}$. Since the output of the ensemble regression is unidimensional, the ensemble regression conveys information about the finger movement position not the type of finger movement at an instance. Results from the ensemble classification were used to determine the type of finger movement associated with the predicted finger position (from the ensemble regression). On post-implantation day 24, the mean squared error of the ensemble regression model was 0.04 ± 0.004 (Figure 18).

Chronic decoding results.

The performance of the algorithms was averaged across ten folds to obtain the final mean values (Table 2 and Figure 19).

Table 2.

Chronic decoding results

Post-implantation day	Kalman Filter	Ensemble Regression	Significance
3	0.26 ± 0.02	0.10 ± 0.006	$p < 10^{-4}$
7	0.06 ± 0.008	0.05 ± 0.008	$p < 0.01$
13	0.15 ± 0.01	0.08 ± 0.002	$p < 10^{-5}$
17	0.17 ± 0.009	0.07 ± 0.006	$p < 10^{-7}$
20	0.14 ± 0.004	0.04 ± 0.004	$p < 10^{-11}$
24	0.08 ± 0.003	0.04 ± 0.004	$p < 10^{-6}$
29	0.12 ± 0.004	0.05 ± 0.003	$p < 10^{-9}$

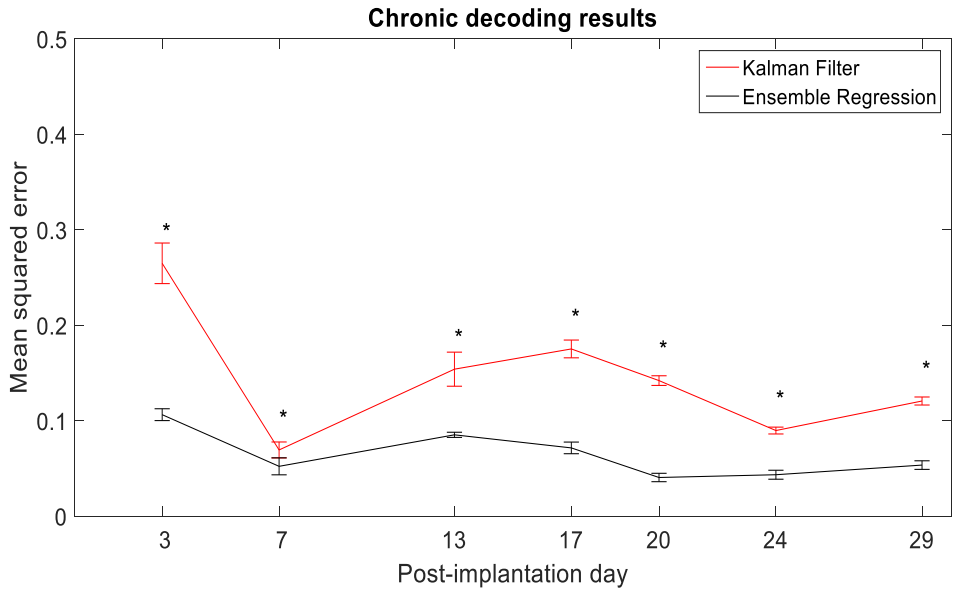


Figure 19. Chronic decoding results. The Kalman Filter and the ensemble regression were employed across all sessions to compute their respective mean squared error. The y-axis corresponds to the mean squared error of the Ensemble regression and Kalman filter regression decodes.

In order to better illustrate the improvement in performance obtained using Ensemble regression, we plotted the results of the decoding algorithms as circles with the radius of the circle equal to the average mean squared error on that particular day (Figure 20). Larger circles depict larger mean squared errors of prediction. Essentially, the metric 'mean squared error' encapsulates the average deviation between what is predicted and what is being estimated. The ensemble regression has smaller area of circles on all days of decode when compared to the Kalman Filter.

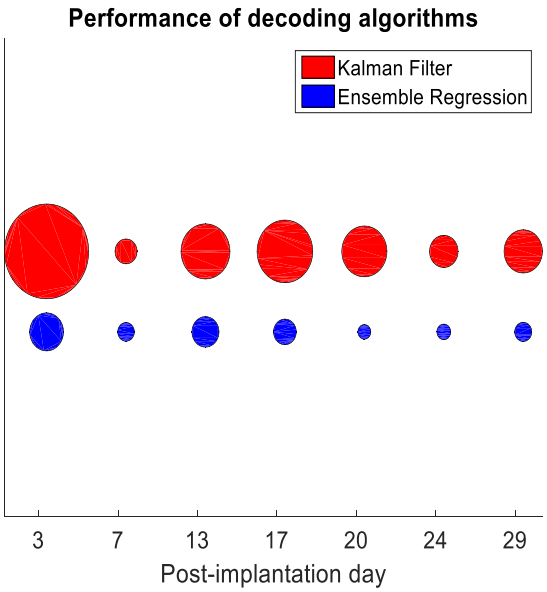


Figure 20. Performance of decoding algorithms. The radius of each circle is equal to the mean squared error of decode for that particular day. If the center of the circle is the point we are trying to predict, the circle denotes an area of equal probability where the decoding algorithm could predict. Therefore, larger the area of the circle higher the uncertainty in predicting the required target (in this case, center of the circle).

A right-tailed two sample t-test was performed to analyze if the mean squared error of the Kalman filter was greater than the mean squared error of the Ensemble regression ($p < 0.05$).

Discussion

In the previous study, we validated a closed-loop peripheral interface based neuroprosthetic control in two transradial amputees. However, the performance (Pearson's correlation co-efficient) of the virtual neuroprosthesis decreased when the patients tried to control with higher degrees of freedom. We speculate that, an increase in number of degrees decreased the generalizing capacity of the Kalman filter.

Multiple degrees of freedom were decoded using peripheral neural signals in an offline implementation of a new decoding model. The decoding model utilizes a hybrid

approach by incorporating both classification and regression algorithms. Classification results produced the discrete control signals which were used as pre-cursors in selecting the correct individual regression model to predict the movement trajectory. The previous study aimed at creating a "global model" for multiple degrees of freedom using a Kalman filter. The degradation in performance in higher degrees of freedom in the previous study was overcome by adopting an explorative approach which had an individual regression model for each degree of freedom, therefore, not compromising on the decoding quality.

A recent survey of adult and pediatric upper-extremity prosthesis users reported challenging activities of daily living as described by the study participants (Biddiss and Chau, 2007, Consumer design priorities for upper limb prosthesis). 40% of the participants reported household chores such as repairs and household maintenance, 30% of the participants reported sports activities such as cycling, swinging sports such as golf, tennis, baseball, 22% of the participants reported hobbies such as playing musical instruments such as guitar, piano, and 19% reported activities of daily living such as food preparation, dressing, typing, hair styling as challenging activities using a prosthesis in everyday life among others. Thus, prosthesis capable of producing dexterous movements can serve as a solution to the demands and challenges experienced by upper-extremity prosthesis users.

This can affect the quality of decode drastically. This problem paves way for investigating the quality of decodes of peripheral nerve based prosthesis in healthy humans and/or non-human primates where the true position of fingers can be measured. From the wrist, the distal part of the hand has 24 different movements (verify) and 13 degrees of freedom (verify) which can be represented by much smaller number of postural muscle

synergies (Santello, Soechting, Flanders). Investigating the peripheral neural basis of postural muscle synergies and incorporating the same in decoding algorithms might provide higher degrees of freedom control with lesser dimensions of representation and smooth transitions between various grip types in an intuitive manner.

Abstract

Objective Peripheral nerve interface based neural prostheses provide a viable platform for restoring motor function lost due to accidents or other neurological disorders. Compared to cortical prostheses, peripheral neuroprostheses provide a safer alternative by evading the need for craniotomy and other surgical risks associated with the procedure. In this offline analysis, we investigated patient induced neural variability in experiments and the relationship between mean and variance of neural variability in the peripheral nervous system and its implications on future experimental design. *Approach* Peripheral neural signals were recorded using intrafascicular microelectrode arrays implanted in the ulnar nerve while the subject performed volitional movement of their phantom fingers. The type of finger movement and finger position was predicted using ensemble methods. Temporal and amplitude variability of various finger movements was analyzed and compared to the decoding performance. The effect of stimulus onset on the mean and variance of raw neural signal, neural firing rate and extracellular neural signals were also compared. *Results* The mean duration of trials performed for all finger movements was 1536ms ± 73ms. The mean normalized variance of firing rate for all finger movements was 49.35 Hz ± 74.66 Hz. The Pearson's correlation coefficient and mean displacement in cross correlation product between mean and variance of firing rate after stimulus onset was 0.77 and 130ms respectively. *Significance* Variability in trial performance was consistently observed across all 7 days of recording. The relationship between subject induced trial performance variability and

the decoding performance is blah. Systems level variability in mean and variance of various neural signals was observed to be linear.

Introduction

One of the crucial aspects while studying systems neuroscience to understand neural encoding is by analyzing the neural response to a stimulus using multiple trials. Most of the neural responses are analyzed by computing the trial-averaged response to a stimulus. To study the trial-averaged neural response meaningfully, we must understand the sources of neural variability. Cell-intrinsic variability, subject induced variability and other noise induced variability is three common sources of neural variability. Cell-intrinsic variability refers to the innate characteristic of a neuron to produce varying neural responses to same and different stimuli. Subject induced variability pertains to variability introduced by the subject due to difference in quality of trial performances. For example, amputees performing volitional phantom finger movements might not perform trials corresponding to the same stimulus consistently due to lack of feedback methods for tracking trial performance. Noise induced variability in neural response refers to the variability introduced by the measuring device (improper reference or grounding). Averaging neural responses (common average/ baseline referencing) helps reduce variability introduced due to noise.

Neural responses corresponding to visual cues for various upper extremity movements provide signature responses in an ensemble of neurons. This phenomenon has been investigated in various motor regions in the cortex, the peripheral nervous system,

and the muscles. Prostheses capable of decoding various upper limb movements have been shown to be viable and effective for restoring limb function lost due to neurological disorders or accidents. Neural variability in various trials can affect the quality of data supplied to the algorithm that can subsequently affect the performance of the prosthesis. One of the challenges while obtaining data from amputees for training an algorithm in a peripheral nerve based prosthesis is the lack of tools to monitor the trials and provide feedback to the subject on the quality of trial performance.

The mean and variance of neural responses to a stimulus and their evolution through time provide a complete characterization of neural variability. The brain area, stimulus, and task influence neural variability. Many studies analyzing the response to stimuli from neural signals recorded in the visual cortex and pre-motor region reported a decrease in neural variability on stimulus onset. Studies have also reported the same phenomenon in other brain areas suggesting that this is a common cortical response. The phenomenon of decrease in neural variability on stimulus onset in the peripheral nervous system has not been reported in any literature to the knowledge of the authors and its effect could reveal significant differences in the network and systems architecture in the peripheral nervous system compared to other brain areas.

The objective of this paper is to study subject induced variability by reducing the effect of cell-intrinsic variability and to study the relationship between mean and variance of raw neural signal, neural firing rate, and extracellular neural signal. These results can be used to analyze the effect of neural variability on decoding performance and its implications on future experimental designs.

Methods

This study was approved by the University of Utah Institutional Review Board and the Salt Lake City Veterans Affairs Hospital Research and Development Service Center.

Data Collection and Pre-processing.

The results reported in this paper were obtained by using the data from the previous study. The pre-study enrollment period, nerve electrode arrays, surgical procedures, data collection and training methods used in this study are described in the previous study (Davis et al., 2016). The Utah Slanted Electrode Array has been described elsewhere and consists of ninety-six electrodes that project out from a 4 x 4 x 0.3 mm substrate (Branner et al., 2001). The two volunteers with previous transradial amputations underwent implantation for one month with a USEA into their median (Subject 1, 31 yrs post-amputation) and ulnar (Subject 2, 1.5 yrs post-amputation) nerves. On an average of three times a week, patients 1 and 2, underwent individual sessions of electrophysiological recordings and microstimulation. Neural signals were recorded and amplified using active head-stage cables (ZIF – Clip 96 channels, Tucker Davis Technologies, Inc., Alachua, FL). The head-stage cables connected to a custom-built board used to interface the TDT-connector with a Neuroport data acquisition system (Blackrock Microsystems, Salt Lake City, UT, USA). Data was collected using Cerebus software (Blackrock Microsystems, Salt Lake City, UT, USA). Neural signals were band pass filtered with cutoff frequencies

of 0.3 Hz (1st order high-pass butterworth filter) and 7500 Hz (3rd order low-pass butterworth filter) and sampled at 30 kHz. Online multi-unit activity was extracted from high-pass filtered data (250 Hz 4th order butterworth filter) by setting a threshold using the auto threshold setting in the Cerebus data acquisition software. Neural firing rates were calculated using unsorted spikes and a moving box-car average of 300 ms with an update period of 33 ms. The patient data used in this offline analysis was obtained from transradial amputees who performed volitional phantom finger movements based on mirroring the healthy hand or by following a virtual robotic hand on screen. Thus, the regression was performed on instruction variables provided to the patients and not on the true position of the fingers.

Neural signals of interest.

Raw neural signal.

1. A 100ms boxcar moving window was used to compute the mean and variance of the raw neural signal.
2. A Butterworth low pass filter with a cut-off frequency of 2000 Hz was used before downsampling the mean and variance of the raw neural signal to 4000 Hz.

Neural firing rate. Mean and variance of the neural firing rate was computed using the following method:

1. Neural data recorded at 30 kHz was used to identify possible neural spikes on each electrode using the Offline Sorter software (Plexon, Inc).

2. Time stamp of sorted action potentials was downsampled to 600 Hz.

3. A 300ms boxcar window was used to compute the moving mean and variance of the neural firing rate on each electrode.

4. A Butterworth low pass filter with a cut-off frequency of 7.5 Hz was used before downsampling the mean and variance of neural firing rate to 15 Hz.

Parameters for analysis.

The analysis presented in this paper can be divided into two sections: (i) effect of patient induced variability on trial performance and (ii) effect of stimulus onset on neural variability in the peripheral nervous system.

Subject induced variability. Temporal and amplitude variability of trials constitute the two main sources of subject induced variability.

(i) Temporal variability: As explained in the previous section, each trial was designed to be 2 seconds in duration. But due to lack of tools to track the kinematics of finger movement for a trial and provide appropriate feedback to the patient, each trial lasted less than 2 seconds. The mean and variance of trial duration was computed across all trials of each finger movement based on the mean firing rate across all electrodes.

(ii) Amplitude variability: To reduce the effect of cell-intrinsic variability, the Normalized Variance (NV) of a neuron for a trial was computed. Normalized variance of a trial for a finger movement is defined as the ratio of variance of a trial to the mean firing rate for all trials of the same finger movement on an electrode. In addition to NV, peak firing rates and

central firing rates of each trial was also computed. Peak firing rate of a trial refers to the maximum amplitude of the firing rate in a trial. Central firing rate of a trial refers to the amplitude of firing rate at the midpoint of a trial i.e., 1 second after trial onset.

Temporal and amplitude variability metrics were computed by using only the neural firing rate.

Effect of stimulus onset on neural variability. In the central nervous system, previous studies have shown a decrease in neural variability after stimulus onset (Churchland et al., 2010). To study the relationship between the mean and variance of various neural signals after stimulus onset, the correlation coefficient and the cross-correlation product was computed. The correction coefficient is expressed between [-1, +1] with a +1 indicating a linear dependence of two variables. The cross-correlation product of two functions shows the similarity between them as a measure of displacement of one relative to the other. The relationship between mean and variance of neural signals after stimulus onset was analyzed using the raw neural signal and neural firing rate.

Results

Correlation coefficient of channel pairs.

While analyzing neural variability, it is important to remove the effects of spurious correlations or random coincidences of channel pairs. To address this phenomenon, Pearson's correlation coefficient was computed between all channel pairs.

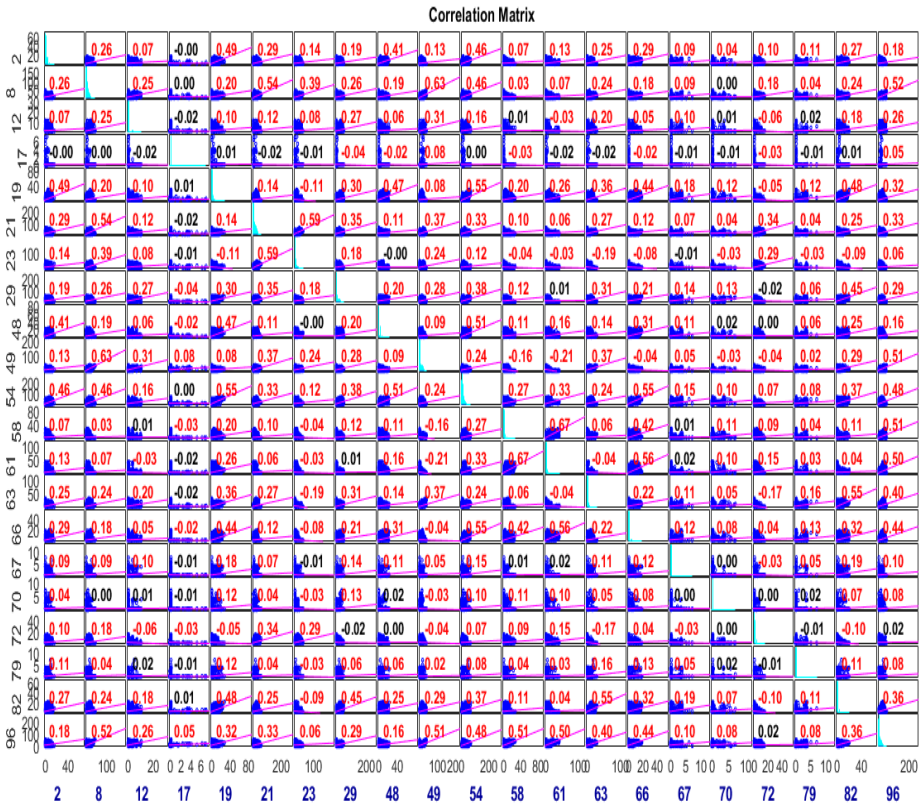


Figure 21. Correlation coefficient for all channel pairs. The correlation coefficient of all pairs of channels revealed that there was no significant linear proportional or linear inverse-proportional relationship between channels due to random coincidences. Correlation coefficients marked in red indicate statistical significance at $p<0.05$ significance.

Temporal variability in trial performance.

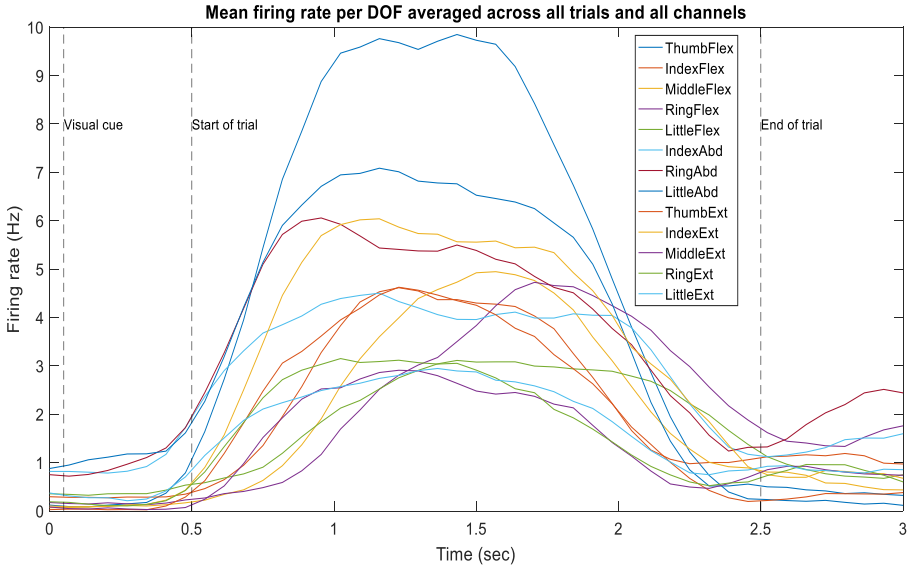


Figure 22. Mean firing rate for each finger movement averaged across all channels and trials.

Mean firing rate of a finger movement was obtained by averaging across all trials and channels. The visual cue was presented at 0ms. A trial is defined as a 2000ms long event, 500ms after presenting the visual cue. 500ms accounts for the reaction period of the subject once the visual cue is presented. For this analysis, a channel and its underlying neuron is said to be actively encoding for a finger movement if the channel mean firing rate for a trial increases above 1 Hz. Theoretically, when a channel and its underlying neuron is not encoding for a finger movement the expected firing rate on across all channels will be 0 Hz. There seems to be insignificant firing in the 0-1 Hz range which is an artifact of moving window calculations.

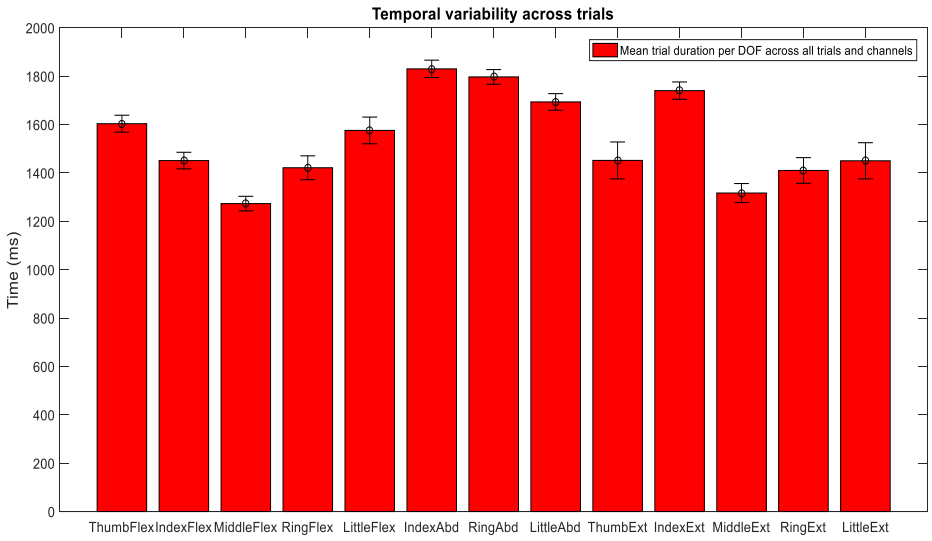


Figure 23. Mean effective duration of movement for each finger movement averaged across all trials as determined using the mean firing rate across all channels for a trial.

Trial duration for a finger movement as extrapolated from the neural firing rate was defined as a period within the 2000ms trial window where the mean firing rate across all channels was greater than 1 Hz. Using this definition, mean trial durations of all finger movements was obtained by averaging across all trials and channels. Across all 13 finger movements, the mean trial duration performed by the patient was 1536ms ± 73ms. In other words, on an average across all finger movements and trials, each trial lasted for only about 76.8% of the 2000ms trial window. This shows the temporal variability of trials and a need for tracking trial performance during training and providing feedback to the patient. This variability also results in including a significant portion of non-trial periods during training and could be detrimental to the accuracy of the machine learning algorithm.

Amplitude variability in trial performance.

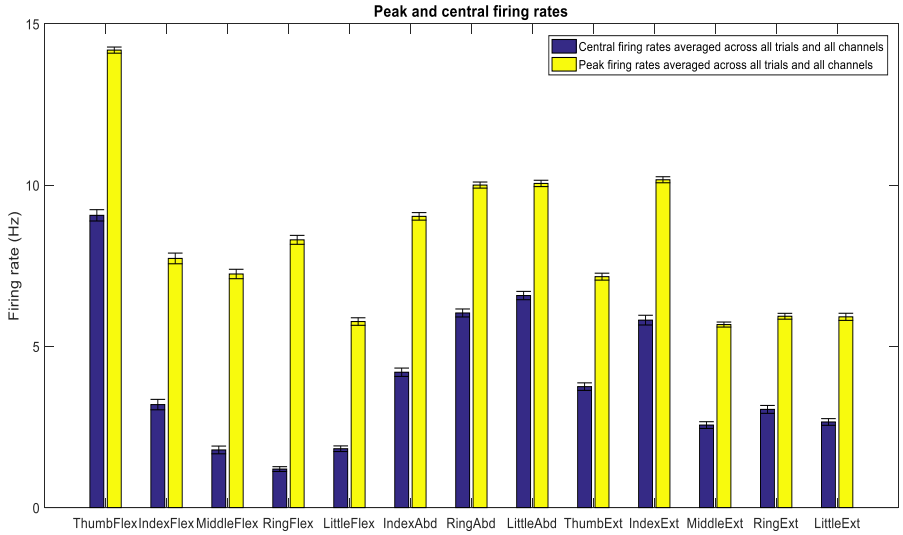


Figure 24. Mean central and peak firing rates for each finger movement averaged across all trials and channels.

Amplitude variability can be expressed in terms of peak firing rate and central firing rate for a finger movement. Peak firing rate corresponds to the maxima of firing rate on a channel as observed in the 2000ms trial window. Central firing rate corresponds to the firing rate at 1000ms after start of trial. This 1000ms point refers to the midpoint of a trial. For an ideal trial, the central firing rate and peak firing rate must be equal. This ensures that there is an equal rise while performing a phantom finger movement to the target and while bringing the phantom finger back to its resting state. For all finger movements, the central firing rate was significantly lesser than the peak firing rate (include p-values from t-test). This difference in central and peak firing rates shows that phantom finger movements during training were performed at varying speeds.

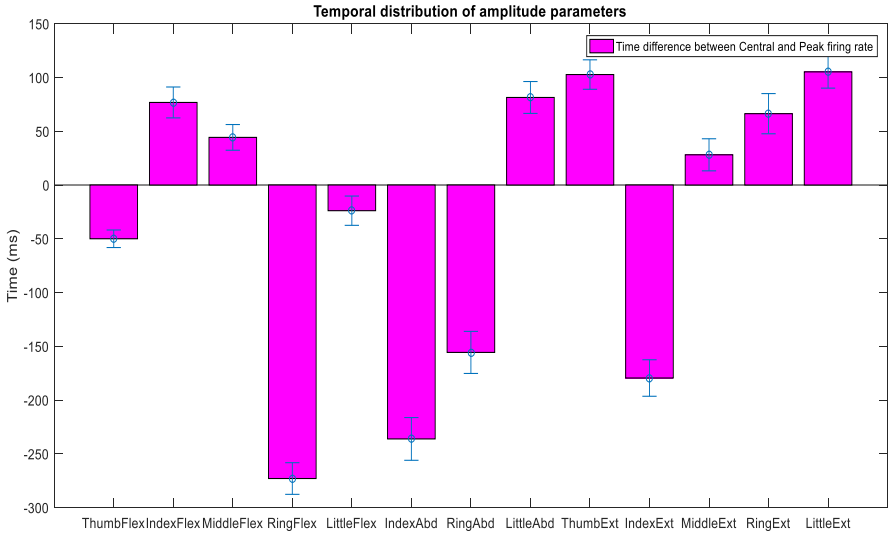


Figure 25. Mean difference in time duration between central and peak firing rates averaged across all trials and channels.

The temporal difference between peak and central firing rate was calculated by subtracting the time stamp of the peak firing rate from 1000ms (central firing rate is the midpoint of a trial). For 6 finger movements (Thumb flexion, ring flexion, little flexion, index abduction, ring abduction, and index extension) the mean peak firing rate was observed after the midpoint of a trial. For the remaining finger movements, the mean peak firing rate was observed before the midpoint of a trial. This shows the temporal variability and variability in speed of trial performance.

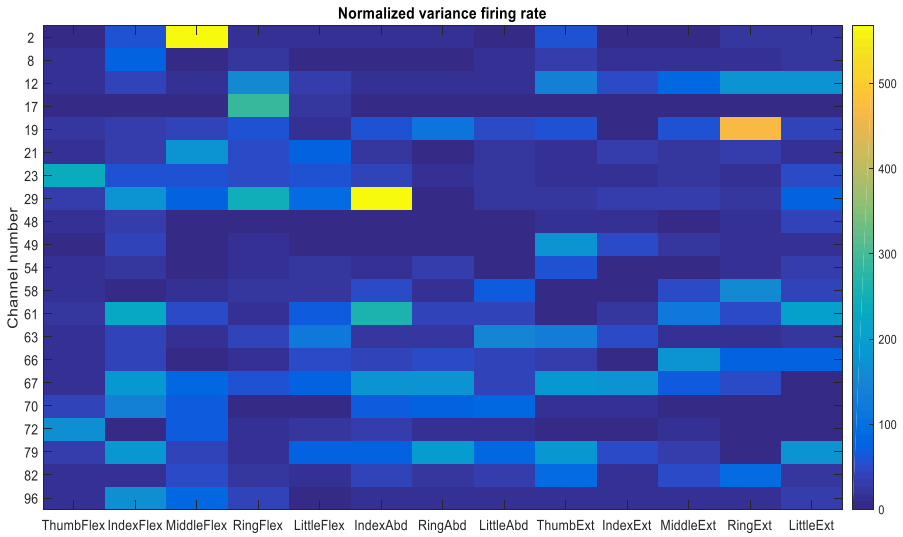


Figure 26. Nomarlized variance of firing rate averaged across all trials for each finger movement and channel.

Amplitude variability of trial performance can also be analyzed using normalized variance as a metric to measure patient induced variability. The normalized variance was computed for a finger movement by computing the ratio of average of variance across all trials to the mean firing rate for each channel. Dividing the mean firing rate of all trials for a finger movement reduces the cell intrinsic variability and helps discern the effect of patient induced variability across trials. The mean normalized variance of firing rate was 49.35 Hz ± 74.66 Hz.

Neural variability on stimulus onset.

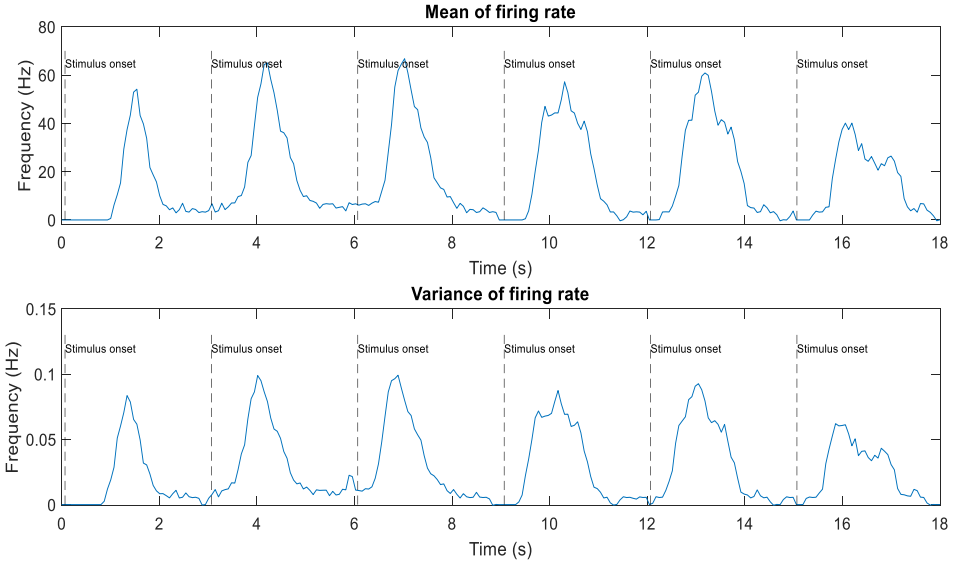


Figure 27. Mean and variance of firing rate for 6 random trials.

The mean and variance of firing rate extracted based on stimulus onset was plotted. The mean and variance were linearly coupled upon visual inspection.

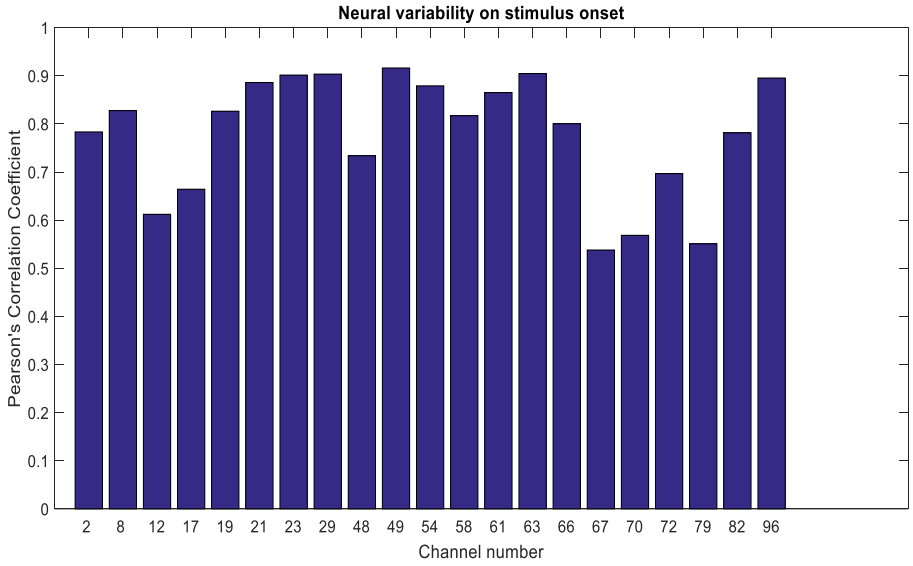


Figure 28. Pearson's correlation coefficient between mean and variance of firing rate on each channel averaged across all finger movements and trials.

To quantify the relationship between the mean and variance of firing rate, Pearson's correlation coefficient and cross correlation displacement between all pairs of mean and variance of firing rate was computed. The mean Pearson's correlation coefficient of mean and variance of firing rate was 0.77. The correlation coefficients were significant at $p < 0.05$ significance value. By interpreting the Pearson's correlation coefficient, the relationship between mean and variance of firing rate is mostly linear. The results of Pearson's correlation coefficient show that the mean and variance of firing rate is mostly linearly coupled.

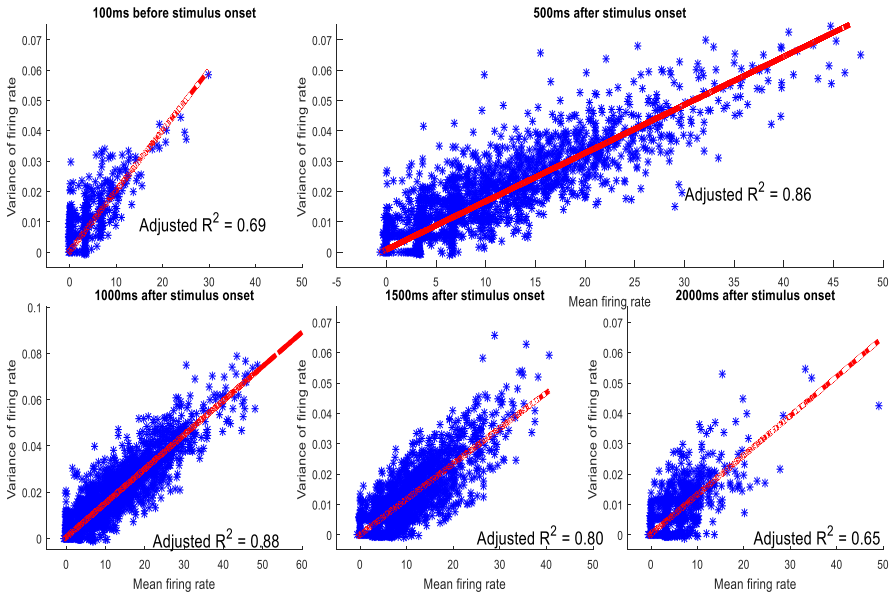


Figure 29. Relationship between mean and variance of firing rate at 100ms before, 500ms, 1000ms, 1500ms, and 2000ms after stimulus onset.

The mean and variance of neural firing rate on all channels was used to analyze the linear relationship 100ms before, 500ms, 1000ms, 1500ms, and 2000ms after stimulus onset for all finger movements. A least squares linear regression was performed

to fit a straight line to the mean-variance relationship. It was found that the linear relationship between the mean and variance was consistent across all trial durations.

Conclusion

Neural variability in trial performance is a challenge that needs to be addressed for designing better experiments. The experimental design factors affect the quality of data obtained from the patient. Neural variability stems from two sources: (i) patient induced variability and (ii) neural variability in the mean-variance relationship. Patient induced variability can manifest in the form of temporal and amplitude variabilities. These factors significantly affect the quality of trials. Based on the results explained in this section, there is a need for tracking trial performance during training and providing feedback to the patient to reduce variability.

In the central nervous system, stimulus onset quenches neural variability. After stimulus onset, the mean of neural signals increases while there is a pronounced decrease in variance. This does not happen in the peripheral nervous system. The mean-variance relationship is consistently linear through the duration of trials for all finger movements. This difference in mean-variance relationship as observed in the CNS and PNS could affect neural decoding system design. The effects of neural variability as explained in this section have implications on future experimental design.

References

Aflalo, T., Kellis, S., Klaes, C., Lee, B., Shi, Y., Pejsa, K., . . . Andersen, R. A. (2015). Neurophysiology. decoding motor imagery from the posterior parietal cortex of a tetraplegic human. *Science (New York, N.Y.), 348*(6237), 906-910. doi:10.1126/science.aaa5417 [doi]

Au, S., Berniker, M., & Herr, H. (2008). Powered ankle-foot prosthesis to assist level-ground and stair-descent gaits. *Neural Networks, 21*(4), 654-666.

Baker, J., Bishop, W., Kellis, S., Levy, T., House, P., & Greger, B. (2009). Multi-scale recordings for neuroprosthetic control of finger movements. *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, 4573-4577.

Bansal, A. K., Vargas-Irwin, C. E., Truccolo, W., & Donoghue, J. P. (2011). Relationships among low-frequency local field potentials, spiking activity, and three-dimensional reach and grasp kinematics in primary motor and ventral premotor cortices. *Journal of Neurophysiology, 105*(4), 1603-1619. doi:10.1152/jn.00532.2010 [doi]

Berens, P., Keliris, G. A., Ecker, A. S., Logothetis, N. K., & Tolias, A. S. (2008). Comparing the feature selectivity of the gamma-band of the local field potential and the underlying spiking activity in primate visual cortex. *Frontiers in Systems Neuroscience, 2*, 2. doi:10.3389/neuro.06.002.2008 [doi]

Bishop, W., Chestek, C. C., Gilja, V., Nuyujukian, P., Foster, J. D., Ryu, S. I., . . . Byron, M. Y. (2014). Self-recalibrating classifiers for intracortical brain–computer interfaces. *Journal of Neural Engineering, 11*(2), 026001.

Bitzer, S., & van der Smagt, P. (2006). Learning EMG control of a robotic hand: Towards active prostheses. *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference On*, 2819-2823.

Björkman, A., Weibull, A., Olsrud, J., Henrik Ehrsson, H., Rosén, B., & Björkman-Burtscher, I. M. (2012). Phantom digit somatotopy: A functional magnetic resonance imaging study in forearm amputees. *European Journal of Neuroscience, 36*(1), 2098-2106.

Branner, A., Stein, R. B., & Normann, R. A. (2001). Selective stimulation of cat sciatic nerve using an array of varying-length microelectrodes. *Journal of Neurophysiology, 85*(4), 1585-1594.

Brown, G., Pocock, A., Zhao, M., & Luján, M. (2012). Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *The Journal of Machine Learning Research, 13*(1), 27-66.

Carpaneto, J., Cutrone, A., Bossi, S., Sergi, P., Citi, L., Rigosa, J., . . . Micera, S. (2011). Activities on PNS neural interfaces for the control of hand prostheses. *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, 4637-4640.

Carrozza, M. C., Cappiello, G., Micera, S., Edin, B. B., Beccai, L., & Cipriani, C. (2006). Design of a cybernetic hand for perception and action. *Biological Cybernetics, 95*(6), 629.

Chestek, C. A., Gilja, V., Blabe, C. H., Foster, B. L., Shenoy, K. V., Parvizi, J., & Henderson, J. M. (2013a). Hand posture classification using electrocorticography signals in the gamma band over human sensorimotor brain areas. *Journal of Neural Engineering, 10*(2), 026002.

Chestek, C. A., Gilja, V., Blabe, C. H., Foster, B. L., Shenoy, K. V., Parvizi, J., & Henderson, J. M. (2013b). Hand posture classification using electrocorticography signals in the gamma band over human sensorimotor brain areas. *Journal of Neural Engineering, 10*(2), 026002.

Choi, K., Hirose, H., Sakurai, Y., Iijima, T., & Koike, Y. (2009). Prediction of arm trajectory from the neural activities of the primary motor cortex with modular connectionist architecture. *Neural Networks, 22*(9), 1214-1223.

Christie, B. P., Tat, D. M., Irwin, Z. T., Gilja, V., Nuyujukian, P., Foster, J. D., . . . Chestek, C. A. (2014). Comparison of spike sorting and thresholding of voltage waveforms for intracortical brain–machine interface performance. *Journal of Neural Engineering, 12*(1), 016009.

Churchland, M. M., Byron, M. Y., Cunningham, J. P., Sugrue, L. P., Cohen, M. R., Corrado, G. S., . . . Scott, B. B. (2010). Stimulus onset quenches neural variability: A widespread cortical phenomenon. *Nature Neuroscience, 13*(3), 369-378.

Churchland, M. M., Yu, B. M., Ryu, S. I., Santhanam, G., & Shenoy, K. V. (2006). Neural variability in premotor cortex provides a signature of motor preparation. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience, 26*(14), 3697-3712. doi:26/14/3697 [pii]

Ciancio, A. L., Cordella, F., Barone, R., Romeo, R. A., Bellingegni, A. D., Sacchetti, R., . . . Di Lazzaro, V. (2016). Control of prosthetic hands via the peripheral nervous system. *Frontiers in Neuroscience, 10*

Cipriani, C., Controzzi, M., & Carrozza, M. C. (2011). The SmartHand transradial prosthesis. *Journal of Neuroengineering and Rehabilitation, 8*(1), 29.

Clark, G. A., Wendelken, S., Page, D. M., Davis, T., Wark, H. A., Normann, R. A., . . . Hutchinson, D. T. (2014). Using multiple high-count electrode arrays in human median and ulnar nerves to restore sensorimotor function after previous transradial amputation of the hand. *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE,* 1977-1980.

Collinger, J. L., Kryger, M. A., Barbara, R., Betler, T., Bowsher, K., Brown, E. H., . . . Gaunt, R. A. (2014). Collaborative approach in the development of High-Performance Brain–Computer interfaces for a neuroprosthetic arm: Translation from animal models to human control. *Clinical and Translational Science, 7*(1), 52-59.

Collinger, J. L., Wodlinger, B., Downey, J. E., Wang, W., Tyler-Kabara, E. C., Weber, D. J., . . . Schwartz, A. B. (2013). High-performance neuroprosthetic control by an individual with tetraplegia. *The Lancet, 381*(9866), 557-564.

Cunningham, J. P., Gilja, V., Ryu, S. I., & Shenoy, K. V. (2009). Methods for estimating neural firing rates, and their application to brain–machine interfaces. *Neural Networks, 22*(9), 1235-1246.

Davis, T., Wark, H., Hutchinson, D., Warren, D., O'Neill, K., Scheinblum, T., . . . Greger, B. (2016). Restoring motor control and sensory feedback in people with upper extremity amputations using arrays of 96 microelectrodes implanted in the median and ulnar nerves. *Journal of Neural Engineering, 13*(3), 036001.

Davis, T., Wark, H., Hutchinson, D., Warren, D., O'Neill, K., Scheinblum, T., . . . Greger, B. (2016a). Restoring motor control and sensory feedback in people with upper extremity amputations using arrays of 96 microelectrodes implanted in the median and ulnar nerves. *Journal of Neural Engineering, 13*(3), 036001.

Davis, T., Wark, H., Hutchinson, D., Warren, D., O'Neill, K., Scheinblum, T., . . . Greger, B. (2016b). Restoring motor control and sensory feedback in people with upper extremity amputations using arrays of 96 microelectrodes implanted in the median and ulnar nerves. *Journal of Neural Engineering, 13*(3), 036001.

Demirer, R. M., Ozerdem, M. S., & Bayrak, C. (2009). Classification of imaginary movements in ECoG with a hybrid approach based on multi-dimensional hilbert-SVM solution. *Journal of Neuroscience Methods, 178*(1), 214-218.

Dhillon, G. S., Lawrence, S. M., Hutchinson, D. T., & Horch, K. W. (2004). Residual function in peripheral nerve stumps of amputees: Implications for neural control of artificial limbs. *The Journal of Hand Surgery, 29*(4), 605-615.

Dhillon, G. S., & Horch, K. W. (2005a). Direct neural sensory feedback and control of a prosthetic arm. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, 13*(4), 468-472.

Dhillon, G. S., & Horch, K. W. (2005b). Direct neural sensory feedback and control of a prosthetic arm. *Neural Systems and Rehabilitation Engineering, IEEE Transactions On, 13*(4), 468-472.

Di Pino, G., Porcaro, C., Tombini, M., Assenza, G., Pellegrino, G., Tecchio, F., & Rossini, P. M. (2012). A neurally-interfaced hand prosthesis tuned inter-hemispheric communication. *Restorative Neurology and Neuroscience, 30*(5), 407-418.

Dietterich, T. G. (2000). Ensemble methods in machine learning. *International Workshop on Multiple Classifier Systems,* 1-15.

Drucker, H., Cortes, C., Jackel, L. D., LeCun, Y., & Vapnik, V. (1994). Boosting and other ensemble methods. *Neural Computation, 6*(6), 1289-1301.

Duan, K., & Keerthi, S. S. (2005). Which is the best multiclass SVM method? an empirical study. *Multiple classifier systems* (pp. 278-285) Springer.

Flint, R. D., Lindberg, E. W., Jordan, L. R., Miller, L. E., & Slutzky, M. W. (2012). Accurate decoding of reaching movements from field potentials in the absence of spikes. *Journal of Neural Engineering, 9*(4), 046006.

Frien, A., & Eckhorn, R. (2000). Functional coupling shows stronger stimulus dependency for fast oscillations than for low-frequency components in striate cortex of awake monkey. *European Journal of Neuroscience, 12*(4), 1466-1478.

Ganguly, K., & Carmena, J. M. (2009). Emergence of a stable cortical map for neuroprosthetic control. *PLoS Biology, 7*(7), 1508.

Ganguly, K., & Carmena, J. M. (2010). Neural correlates of skill acquisition with a cortical brain–machine interface. *Journal of Motor Behavior, 42*(6), 355-360.

Garde, K., Keefer, E., Botterman, B., Galvan, P., & Romero-Ortega, M. I. (2009). Early interfaced neural activity from chronic amputated nerves. *Frontiers in Neuroengineering, 2*, 5.

Gilja, V., Nuyujukian, P., Chestek, C., Cunningham, J., Yu, B., Ryu, S., & Shenoy, K. (2010). High-performance continuous neural cursor control enabled by a feedback control perspective. *Conference Abstract: Computational and Systems Neuroscience,*

Gilja, V., Nuyujukian, P., Chestek, C. A., Cunningham, J. P., Byron, M. Y., Fan, J. M., . . . Ryu, S. I. (2012). A high-performance neural prosthesis enabled by control algorithm design. *Nature Neuroscience, 15*(12), 1752-1757.

Gilja, V., Nuyujukian, P., Chestek, C. A., Cunningham, J. P., Byron, M. Y., Fan, J. M., . . . Shenoy, K. V. (2012). A brain machine interface control algorithm designed from a feedback control perspective. *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE,* 1318-1322.

Gilja, V., Pandarinath, C., Blabe, C. H., Nuyujukian, P., Simeral, J. D., Sarma, A. A., . . . Hochberg, L. R. (2015). Clinical translation of a high-performance neural prosthesis. *Nature Medicine*,

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *The Journal of Machine Learning Research, 3*, 1157-1182.

Guyon, I., & Elisseeff, A. (2006). An introduction to feature extraction. *Feature extraction* (pp. 1-25) Springer.

Hao, Y., Zhang, Q., Controzzi, M., Cipriani, C., Li, Y., Li, J., . . . Carrozza, M. C. (2014a). Distinct neural patterns enable grasp types decoding in monkey dorsal premotor cortex. *Journal of Neural Engineering, 11*(6), 066011.

Hao, Y., Zhang, Q., Controzzi, M., Cipriani, C., Li, Y., Li, J., . . . Carrozza, M. C. (2014b). Distinct neural patterns enable grasp types decoding in monkey dorsal premotor cortex. *Journal of Neural Engineering, 11*(6), 066011.

Hebb, D. O. (2005). *The organization of behavior: A neuropsychological theory* Psychology Press.

Horch, K., Meek, S., Taylor, T. G., & Hutchinson, D. T. (2011). Object discrimination with an artificial hand using electrical stimulation of peripheral tactile and proprioceptive pathways with intrafascicular electrodes. *Neural Systems and Rehabilitation Engineering, IEEE Transactions On, 19*(5), 483-489.

House, P. A., MacDonald, J. D., Tresco, P. A., & Normann, R. A. (2006). Acute microelectrode array implantation into human neocortex: Preliminary technique and histological considerations. *Neurosurgical Focus, 20*(5), 1-4.

Hsu, C., & Lin, C. (2002). A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions On, 13*(2), 415-425.

Hwang, E. J., & Andersen, R. A. (2013). The utility of multichannel local field potentials for brain–machine interfaces. *Journal of Neural Engineering, 10*(4), 046005.

Ison, M., & Artemiadis, P. (2014). The role of muscle synergies in myoelectric control: Trends and challenges for simultaneous multifunction control. *Journal of Neural Engineering, 11*(5), 051001.

Jarosiewicz, B., Sarma, A. A., Bacher, D., Masse, N. Y., Simeral, J. D., Sorice, B., . . . Hochberg, L. R. (2015). Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. *Science Translational Medicine, 7*(313), 313ra179. doi:10.1126/scitranslmed.aac7328 [doi]

Jia, X., Smith, M. A., & Kohn, A. (2011). Stimulus selectivity and spatial coherence of gamma components of the local field potential. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience, 31*(25), 9390-9403. doi:10.1523/JNEUROSCI.0645-11.2011 [doi]

Johansson, R. S., & Flanagan, J. R. (2009). Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience, 10*(5), 345-359.

Jolliffe, I. (2002). *Principal component analysis* Wiley Online Library.

Kantardzic, M. (2011). *Data mining: Concepts, models, methods, and algorithms* John Wiley & Sons.

Karl, A., Birbaumer, N., Lutzenberger, W., Cohen, L. G., & Flor, H. (2001). Reorganization of motor and somatosensory cortex in upper extremity amputees with phantom limb pain. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience, 21*(10), 3609-3618. doi:21/10/3609 [pii]

Kellis, S., Miller, K., Thomson, K., Brown, R., House, P., & Greger, B. (2010). Decoding spoken words using local field potentials recorded from the cortical surface. *Journal of Neural Engineering, 7*(5), 056007.

Kemere, C., Santhanam, G., Yu, B. M., Afshar, A., Ryu, S. I., Meng, T. H., & Shenoy, K. V. (2008). Detecting neural-state transitions using hidden markov models for motor cortical prostheses. *Journal of Neurophysiology, 100*(4), 2441-2452. doi:10.1152/jn.00924.2007 [doi]

Kim, H., Kim, Y., Shin, H., Aggarwal, V., Schieber, M. H., & Thakor, N. V. (2012). Neuron selection by relative importance for neural decoding of dexterous finger prosthesis control application. *Biomedical Signal Processing and Control, 7*(6), 632-639.

Kim, S., Sanchez, J. C., Rao, Y. N., Erdogmus, D., Carmena, J. M., Lebedev, M. A., . . . Principe, J. (2006a). A comparison of optimal MIMO linear and nonlinear models for brain–machine interfaces. *Journal of Neural Engineering, 3*(2), 145.

Kim, S., Sanchez, J. C., Rao, Y. N., Erdogmus, D., Carmena, J. M., Lebedev, M. A., . . . Principe, J. (2006b). A comparison of optimal MIMO linear and nonlinear models for brain–machine interfaces. *Journal of Neural Engineering, 3*(2), 145.

Kim, S., Simeral, J. D., Hochberg, L. R., Donoghue, J. P., & Black, M. J. (2008). Neural control of computer cursor velocity by decoding motor cortical spiking activity in humans with tetraplegia. *Journal of Neural Engineering, 5*(4), 455.

Kirsch, E., Rivlis, G., & Schieber, M. H. (2014). Primary motor cortex neurons during individuated finger and wrist movements: Correlation of spike firing rates with the motion of individual digits versus their principal components.

Klaes, C., Kellis, S., Aflalo, T., Lee, B., Pejsa, K., Shanfield, K., . . . Andersen, R. A. (2015). Hand shape representations in the human posterior parietal cortex. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience, 35*(46), 15466-15476. doi:10.1523/JNEUROSCI.2747-15.2015 [doi]

Kleim, J. A. (2011). Neural plasticity and neurorehabilitation: Teaching the new brain old tricks. *Journal of Communication Disorders, 44*(5), 521-528.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Ijcai, , 14*(2) 1137-1145.

Kohavi, R., & John, G. H. (1997a). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1), 273-324.

Kohavi, R., & John, G. H. (1997b). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1), 273-324.

Kuiken, T. A., Li, G., Lock, B. A., Lipschutz, R. D., Miller, L. A., Stubblefield, K. A., & Englehart, K. B. (2009). Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms. *Jama, 301*(6), 619-628.

Leopold, D. A., & Logothetis, N. K. (2003). Spatial patterns of spontaneous local field activity in the monkey visual cortex. *Reviews in the Neurosciences, 14*(1-2), 195-205.

Letelier, J. C., & Weber, P. P. (2000). Spike sorting based on discrete wavelet transform coefficients. *Journal of Neuroscience Methods, 101*(2), 93-106.

Lewicki, M. S. (1998a). A review of methods for spike sorting: The detection and classification of neural action potentials. *Network: Computation in Neural Systems, 9*(4), R53-R78.

Lewicki, M. S. (1998b). A review of methods for spike sorting: The detection and classification of neural action potentials. *Network: Computation in Neural Systems, 9*(4), R53-R78.

Liang, N., & Bougrain, L. (2012). Decoding finger flexion from band-specific ECoG signals in humans. *Frontiers in Neuroscience, 6*, 91.

Lu, Y., Cohen, I., Zhou, X. S., & Tian, Q. (2007). Feature selection using principal feature analysis. *Proceedings of the 15th International Conference on Multimedia*, 301-304.

Marathe, A., & Taylor, D. M. (2011). Decoding position, velocity, or goal: Does it matter for brain–machine interfaces? *Journal of Neural Engineering, 8*(2), 025016.

Marathe, A., & Taylor, D. (2013). Decoding continuous limb movements from high-density epidural electrode arrays using custom spatial filters. *Journal of Neural Engineering, 10*(3), 036015.

Maynard, E. M., Nordhausen, C. T., & Normann, R. A. (1997). The utah intracortical electrode array: A recording structure for potential brain-computer interfaces. *Electroencephalography and Clinical Neurophysiology, 102*(3), 228-239.

McMullen, D. P., Hotson, G., Katyal, K. D., Wester, B. A., Fifer, M. S., McGee, T. G., . . . Ravitz, A. D. (2014). Demonstration of a semi-autonomous hybrid brain–machine interface using human intracranial EEG, eye tracking, and computer vision to control a robotic upper limb prosthetic. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, 22*(4), 784-796.

Mercier, C., Reilly, K. T., Vargas, C. D., Aballea, A., & Sirigu, A. (2006). Mapping phantom movement representations in the motor cortex of amputees. *Brain : A Journal of Neurology, 129*(Pt 8), 2202-2210. doi:awl180 [pii]

Miall, R. C., & Wolpert, D. M. (1996). Forward models for physiological motor control. *Neural Networks, 9*(8), 1265-1279.

Micera, S., Carpaneto, J., & Raspopovic, S. (2010). Control of hand prostheses using peripheral information. *Biomedical Engineering, IEEE Reviews In, 3*, 48-68.

Micera, S., Rossini, P. M., Rigosa, J., Citi, L., Carpaneto, J., Raspopovic, S., . . . Carrozza, M. C. (2011). Decoding of grasping information from neural signals recorded using peripheral intrafascicular interfaces. *Journal of Neuroengineering and Rehabilitation, 8*(1), 53.

Mollazadeh, M., Aggarwal, V., Thakor, N., Law, A., Davidson, A., & Schieber, M. (2009). Coherency between spike and LFP activity in M1 during hand movements.

Neural Engineering, 2009. NER'09. 4th International IEEE/EMBS Conference On, 506-509.

Musallam, S., Corneil, B. D., Greger, B., Scherberger, H., & Andersen, R. A. (2004). Cognitive control signals for neural prosthetics. *Science (New York, N.Y.), 305*(5681), 258-262. doi:10.1126/science.1097938 [doi]

Navarro, X., Krueger, T. B., Lago, N., Micera, S., Stieglitz, T., & Dario, P. (2005). A critical review of interfaces with the peripheral nervous system for the control of neuroprostheses and hybrid bionic systems. *Journal of the Peripheral Nervous System, 10*(3), 229-258.

Ohara, S., Ikeda, A., Kunieda, T., Yazawa, S., Baba, K., Nagamine, T., . . . Shibasaki, H. (2000). Movement-related change of electrocorticographic activity in human supplementary motor area proper. *Brain : A Journal of Neurology, 123 (Pt 6)*(Pt 6), 1203-1215.

O'Leary, J. G., & Hatsopoulos, N. G. (2006). Early visuomotor representations revealed from evoked local field potentials in motor and premotor cortical areas. *Journal of Neurophysiology, 96*(3), 1492-1506. doi:00106.2006 [pii]

Opitz, D., & Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research, 11*, 169-198.

Padmanaban, S. (2015). *Comparison of Feature Selection Methods for Robust Dexterous Decoding of Finger Movements from the Primary Motor Cortex of a Non-Human Primate using Support Vector Machine,*

Perge, J. A., Zhang, S., Malik, W. Q., Homer, M. L., Cash, S., Friehs, G., . . . Hochberg, L. R. (2014). Reliability of directional information in unsorted spikes and local field potentials recorded in human motor cortex. *Journal of Neural Engineering, 11*(4), 046007.

Pisani, A., Centonze, D., Bernardi, G., & Calabresi, P. (2005). Striatal synaptic plasticity: Implications for motor learning and parkinson's disease. *Movement Disorders, 20*(4), 395-402.

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods—support Vector Learning, 3*

Polasek, K. H., Hoyen, H. A., Keith, M. W., & Tyler, D. J. (2007). Human nerve stimulation thresholds and selectivity using a multi-contact nerve cuff electrode. *Neural Systems and Rehabilitation Engineering, IEEE Transactions On, 15*(1), 76-82.

Polasek, K., Schiefer, M., Pinault, G., Triolo, R., & Tyler, D. (2009). Intraoperative evaluation of the spiral nerve cuff electrode on the femoral nerve trunk. *Journal of Neural Engineering, 6*(6), 066005.

Quandt, F., Reichert, C., Hinrichs, H., Heinze, H., Knight, R., & Rieger, J. W. (2012). Single trial discrimination of individual finger movements on one hand: A combined MEG and EEG study. *NeuroImage, 59*(4), 3316-3324.

Quinlan, J. R. (1996). Bagging, boosting, and C4. 5. *AAAI/IAAI, Vol. 1,* 725-730.

Randles, R. H. (1988). Wilcoxon signed rank test. *Encyclopedia of Statistical Sciences,*

Reilly, K. T., Mercier, C., Schieber, M. H., & Sirigu, A. (2006). Persistent hand motor commands in the amputees' brain. *Brain : A Journal of Neurology, 129*(Pt 8), 2211-2223. doi:awl154 [pii]

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386.

Rule, M. E., Vargas-Irwin, C., Donoghue, J. P., & Truccolo, W. (2015). Contribution of LFP dynamics to single-neuron spiking variability in motor cortex during movement execution. *Frontiers in Systems Neuroscience, 9*, 89.

Ryan, M. (2001). *Narrative as virtual reality: Immersion and interactivity in literature and electronic media* Johns Hopkins University Press.

Santhanam, G., Ryu, S. I., Byron, M. Y., Afshar, A., & Shenoy, K. V. (2006). A high-performance brain–computer interface. *Nature, 442*(7099), 195-198.

Schwartz, A. B. (2007). Useful signals from motor cortex. *The Journal of Physiology, 579*(3), 581-601.

Shenoy, K. V., Meeker, D., Cao, S., Kureshi, S. A., Pesaran, B., Buneo, C. A., . . . Andersen, R. A. (2003). Neural prosthetic control signals from plan activity. *Neuroreport, 14*(4), 591-596.

Shenoy, P., Miller, K. J., Ojemann, J. G., & Rao, R. P. (2008). Generalized features for electrocorticographic BCIs. *IEEE Transactions on Biomedical Engineering, 55*(1), 273-280.

Shenoy, K. V., Kaufman, M. T., Sahani, M., & Churchland, M. M. (2011). A dynamical systems view of motor preparation: Implications for neural prosthetic system design. *Progress in Brain Research, 192,* 33-58. doi:10.1016/B978-0-444-53355-5.00003-8 [doi]

Shimoda, K., Nagasaka, Y., Chao, Z. C., & Fujii, N. (2012). Decoding continuous three-dimensional hand trajectories from epidural electrocorticographic signals in japanese macaques. *Journal of Neural Engineering, 9*(3), 036015.

Shoham, S., Fellows, M. R., & Normann, R. A. (2003). Robust, automatic spike sorting using mixtures of multivariate t-distributions. *Journal of Neuroscience Methods, 127*(2), 111-122.

Stam, C. J., Nolte, G., & Daffertshofer, A. (2007). Phase lag index: Assessment of functional connectivity from multi channel EEG and MEG with diminished bias from common sources. *Human Brain Mapping, 28*(11), 1178-1193.

Stark, E., & Abeles, M. (2007). Predicting movement from multiunit activity. *The Journal of Neuroscience : The Official Journal of the Society for Neuroscience, 27*(31), 8387-8394. doi:27/31/8387 [pii]

Suminski, A. J., Tkach, D. C., & Hatsopoulos, N. G. (2009). Exploiting multiple sensory modalities in brain-machine interfaces. *Neural Networks, 22*(9), 1224-1234.

Sussillo, D., Nuyujukian, P., Fan, J. M., Kao, J. C., Stavisky, S. D., Ryu, S., & Shenoy, K. (2012). A recurrent neural network for closed-loop intracortical brain–machine interface decoders. *Journal of Neural Engineering, 9*(2), 026027.

Tan, D. W., Schiefer, M. A., Keith, M. W., Anderson, J. R., Tyler, J., & Tyler, D. J. (2014a). A neural interface provides long-term stable natural touch perception. *Science Translational Medicine, 6*(257), 257ra138. doi:10.1126/scitranslmed.3008669 [doi]

Tan, D. W., Schiefer, M. A., Keith, M. W., Anderson, J. R., Tyler, J., & Tyler, D. J. (2014b). A neural interface provides long-term stable natural touch perception. *Science Translational Medicine, 6*(257), 257ra138. doi:10.1126/scitranslmed.3008669 [doi]

Tehovnik, E., Woods, L., & Slocum, W. (2013). Transfer of information by BMI. *Neuroscience, 255*, 134-146.

Tenore, F., Aggarwal, V., White, J. R., Schieber, M. H., & Thakor, N. V. (2009). Computational complexity versus accuracy in classification of cortical neural signals. *Neural Engineering, 2009. NER'09. 4th International IEEE/EMBS Conference On*, 750-753.

Torkkola, K. (2003). Feature extraction by non parametric mutual information maximization. *The Journal of Machine Learning Research, 3*, 1415-1438.

Traversa, R., Cicinelli, P., Bassi, A., Rossini, P. M., & Bernardi, G. (1997). Mapping of motor cortical reorganization after stroke. A brain stimulation study with focal magnetic pulses. *Stroke, 28*(1), 110-117.

Truccolo, W., Eden, U. T., Fellows, M. R., Donoghue, J. P., & Brown, E. N. (2005). A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology, 93*(2), 1074-1089. doi:10.1152/jn.00697.2004 [doi]

van Gerven, M. A., Chao, Z. C., & Heskes, T. (2012). On the decoding of intracranial data using sparse orthonormalized partial least squares. *Journal of Neural Engineering, 9*(2), 026017.

Velliste, M., McMorland, A., Diril, E., Clanton, S., & Schwartz, A. (2012). State-space control of prosthetic hand shape. *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, 964-967.

Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S., & Schwartz, A. B. (2008). Cortical control of a prosthetic arm for self-feeding. *Nature, 453*(7198), 1098-1101.

Vetter, R. J., Williams, J. C., Hetke, J. F., Nunamaker, E., & Kipke, D. R. (2004). Chronic neural recording using silicon-substrate microelectrode arrays implanted in cerebral cortex. *Biomedical Engineering, IEEE Transactions On, 51*(6), 896-904.

Vinjamuri, R., Weber, D., Degenhart, A., Collinger, J., Sudre, G., Adelson, P., . . . Crammond, D. (2009). A fuzzy logic model for hand posture control using human cortical activity recorded by micro-ECoG electrodes. *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE,* 4339-4342.

Wang, B., Shi, W., & Miao, Z. (2015). Confidence analysis of standard deviational ellipse and its extension into higher dimensional euclidean space. *PloS One, 10*(3), e0118537. doi:10.1371/journal.pone.0118537 [doi]

Ward, N. S., & Cohen, L. G. (2004). Mechanisms underlying recovery of motor function after stroke. *Archives of Neurology, 61*(12), 1844-1848.

Wickens, J. R., Reynolds, J. N., & Hyland, B. I. (2003). Neural mechanisms of reward-related motor learning. *Current Opinion in Neurobiology, 13*(6), 685-690.

Wissel, T., Pfeiffer, T., Frysch, R., Knight, R. T., Chang, E. F., Hinrichs, H., . . . Rose, G. (2013a). Hidden markov model and support vector machine based decoding of finger movements using electrocorticography. *Journal of Neural Engineering, 10*(5), 056020.

Wissel, T., Pfeiffer, T., Frysch, R., Knight, R. T., Chang, E. F., Hinrichs, H., . . . Rose, G. (2013b). Hidden markov model and support vector machine based decoding of finger movements using electrocorticography. *Journal of Neural Engineering, 10*(5), 056020.

Zhang, Y., & Chase, S. M. (2013). A stabilized dual kalman filter for adaptive tracking of brain-computer interface decoding parameters. *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, 7100-7103.