ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

The Information Bottleneck method in Neuroscience

Sreenivas Kartik Buddha

School of Computer and Communication Sciences, EPFL

Master Thesis

January 2013

Supervisor

Prof. Michael Gastpar
Laboratory for Information in Networked Systems - LINX

Contents

List of Figures

Abstract

Information measures have found widespread use in neuroscience. In this project, we extend the well-known information bottleneck method for certain uses in networks of neurons. The information bottleneck method is a technique for finding the best trade-off between accuracy and complexity. This is important for this project in the sense that it permits to infer facts about the structure of the signal and information processing in networks. The information bottleneck method is extended in this project for identifying linear relationships between random variables. This approach is then tested both on artificial data and real experimental data from a brain-interface experiment involving a monkey performing behavioral tasks. Using this technique we are able to identify neuron triplets in the data such that the spiking response of one of the neurons is a weighted *sum* of the spiking response of the other two neurons. Moreover, we observe that neurons which follow such a relationship in their spiking patterns during a particular experiment, also exhibit similar behavior in some of the different trials of the same experiment. An information bottleneck method based approach for clustering neurons in the network using their spike responses is also presented in this project.

Chapter 1

Introduction

1.1 Motivation

Information theory is well integrated into neuroscience research mainly for answering questions about neural coding. Neural coding is a fundamental aspect of neuroscience concerned with the representation of sensory and other information in the brain by networks of neurons. It characterizes the relationship between external sensory stimuli and the corresponding neural activity in the form of time-dependent sequences of discrete action potentials known as *spike trains* [10]. Information theory addresses issues similar to the ones posed in neural coding such as: how is information encoded and decoded? and what does a response (output) tell us about a stimulus (input)? It is therefore used as a general framework in neural coding for measuring how the neural responses vary with different stimuli ([2] and [6]). In classical neuroscience experiments, the responses of a single neuron to several stimuli are recorded and information-theoretic tools are used to quantify neural code reliability by measuring how much information about the stimuli is contained in neural responses.

However, new measurement techniques such as implanted tungsten micro-wire arrays and Electroencephalography (EEG) lead to larger datasets by being able to simultaneously measure the neural activity of multiple neurons. Consequently, on the datasets of this nature, additional questions pertaining to the network behavior of the neurons can also be asked. Statistical methods based on information measures such as mutual information and directed information have been used to estimate fundamental properties from the data. In [21], the concepts of mutual information were applied to quantify the redundancy of movement-related information encoded in the motor system neuron populations of a macaque and [20] provides a modified procedure for estimating directed information in order to obtain accurate and novel insights into the functional connectivity of neural ensembles that are applicable to data from neurophysiological studies in awake behaving animals.

In this project, another information measure known as the *information bottleneck* (IB) method is explored for identifying relationships between the spiking patterns of multiple neurons in the network. The information bottleneck method is a well known technique for finding the best trade-off between complexity and accuracy. The goal is to infer facts about the structure and information processing in networks of neurons. By analyzing the simultaneously recorded spike train responses of these neurons, we are interested in identifying neurons in the network which appear to behave in such a way that the spike train response of a neuron is dependent on the spike train responses of other neurons in the network. Furthermore, the neurons which we analyze for identifying such a behavior need not necessarily share common synaptic connections. We would like to find such relationships, if they exist, between any subset of neurons in the network. Thus, given three neurons and their spiking signals, we want to be able to answer the following questions: Does one of these neurons represent a *function* of the spiking signal of the other two neurons? If so, what kind of function would that be? At what time instances should we consider these three neurons and for how long, to infer such a dependency? In order to answer these questions, we treat the spiking signal of a neuron as a random variable by adopting a binning approach. The problem then amounts to determining whether these different random variables corresponding to different neural responses are functionally related or not.

Identifying such *functional relationships* between random variables is a very fundamental problem which has received the attention of several research efforts. One recent software tool is Eureqa [11] which identifies the simplest mathematical formulas to describe the underlying mechanisms that produced the data. In this project, instead of using the data directly, we look at the probability distribution of the data to make an inference about the relationships present. Accordingly, we mainly investigate the applicability of the information bottleneck method for achieving this task of recognizing functional relationships between random variables. Additionally, clustering approaches based on the information bottleneck method are also investigated to group the neurons into clusters.

The rest of the report is organized as follows. Section 1.2 states some important information theoretic definitions taken from [5] that are necessary to proceed further and mathematically formulate the information bottleneck method and the problem statement of this project. Following that, Section 1.3 gives the main motivation behind using the IB method. A detailed account of the IB method and algorithms along with some applications of the IB method in neuroscience is provided in Chapter 2. In the subsequent chapters, the different ways in which the IB method is applied in this project are described. The primary goal of this project which is the functional identification between random variables is discussed in Chapter 3 alongside results obtained on artificial data. Chapter 4 focuses on results obtained on real experimental data. Appendix A covers clustering of neurons using the information bottleneck method and finally, the report is concluded along with a brief mention of possible future work in Chapter 5.

1.2 Definitions

In what follows, uppercase letters denote the names of random variables; lowercase letters and calligraphic notations respectively denote the realizations and support of the corresponding random variables. The probability $P(X = x)$ that the random variable X takes on a value of x is denoted using the shorthand notation $p(x)$. The notation $\sum_x$ is used to denote the summation of x over all possible values in its support ($x \in \mathcal{X}$) and $|\mathcal{X}|$ denotes the cardinality of the random variable X.

1.2.1 Entropy

Let X be a random variable with support $\mathcal{X}$ and a probability distribution $p(x)$. The entropy of X is a measure of uncertainty associated with it and quantifies the expected value of its information content. It is defined as follows:

$$H(X) \equiv H[p(x)] = -\sum_{x} p(x) \log_2[p(x)] \tag{1.1}$$

If X and Y are two random variables given by their joint distribution $p(x, y)$, then the joint entropy of the two random variables is given by

$$H(X, Y) = -\sum_{x,y} p(x, y) \log_2[p(x, y)] \tag{1.2}$$

and the conditional entropy of Y given X is defined as

$$H(Y|X) = \sum_{x} p(x) H(Y|X = x) = -\sum_{x} \sum_{y} p(x, y) \log_2[p(y|x)] \tag{1.3}$$

$H(Y|X)$ is the expected uncertainty remaining on Y, once the value of the random variable X is known. These definitions of entropy can then be used for defining mutual information.

1.2.2 Mutual Information

Given two discrete random variables X and Y, the mutual information quantifies the amount of *information* X contains about Y and vice-versa. If $p(x, y)$ is the joint probability distribution of the two variables, the marginal distributions of X and Y are given by

$$p(x) = \sum_{y} p(x, y) \text{ and } p(y) = \sum_{x} p(x, y) \tag{1.4}$$

The mutual information $I(X; Y)$ between X and Y is then defined as follows:

$$I(X; Y) \equiv I[p(x, y)] = \sum_{x} \sum_{y} p(x, y) \log_2 \frac{p(x, y)}{p(x)p(y)} \tag{1.5}$$

From Equation 1.5, we see that mutual information is symmetric in X and Y and from the previous Equations 1.1, 1.2 and 1.3, it can be rewritten as follows:

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X, Y) \tag{1.6}$$

The concept of mutual information has several useful interpretations depending on which formulation we choose from Equation 1.6:

- $I(X; Y)$ is a measure of the number of bits gained (if the logarithm is in base 2) through a joint compression of X and Y, instead of compressing X and Y independently. This is due to the fact that the entropy of a random variable lower bounds its minimal achievable code length and encoding X and Y independently, ignores the possible correlations between the two variables.

- Alternatively, mutual information between X and Y can be seen as the decrease in the uncertainty of Y due to the knowledge of X, as $I(X;Y)$ is given by the difference of the entropy $H(Y)$ and the conditional entropy $H(Y|X)$.
- Suppose the random variable X is compressed using a quantized codebook Z, then the mutual information between the two variables $I(X;Z)$, gives the extent of compression and the amount of information Z preserves about X. Smaller the value of $I(X;Z)$, greater is the extent of compression (a more compact code). In the limiting case, if there is maximum compression by mapping all the elements of X to a single value of Z, then $I(X,Z) = 0$ as both $H(Z) = 0$ and $H(Z|X) = 0$. On the other hand, if there is no compression at all, when X and Z have a one-to-one mapping making $H(X|Z) = 0$, then $I(X;Z)$ takes its maximum value: $H(X)$.
- A more formal interpretation of mutual information characterizes $I(X;Y)$ to the expected maximal number of bits that can be reliably sent over a discrete memoryless channel with a probability transition matrix $p(y|x)$.

1.2.3 KL and JS Divergence Measures

The Kullback-Leibler (KL) divergence, also known as relative entropy between two probability distributions $p_1(x)$ and $p_2(x)$ gives a measure of *distance* between the two distributions and is defined as follows:

$$D_{KL}[p_1||p_2] = \sum_x p_1(x)\log_2\frac{p_1(x)}{p_2(x)} = \mathbb{E}_X\left[\log_2\frac{p_1(x)}{p_2(x)}\right] \tag{1.7}$$

The KL divergence is a non-negative quantity and is equal to zero if and only if $p_1(x) = p_2(x), \forall x$. It quantifies the coding inefficiency of assuming that the distribution is $p_2(x)$ when the true distribution is $p_1(x)$. Using this definition of KL divergence, the mutual information between two random variables X and Y can be written as:

$$I(X;Y) = D_{KL}\left[p(x,y)||p(x)p(y)\right] = D_{KL}\left[p(x|y)||p(x)\right] \tag{1.8}$$

The Jensen-Shannon (JS) divergence in an alternative divergence measure between distributions $p_1(x)$ and $p_2(x)$ defined as:

$$JS_{\Pi}[p_1,p_2] = \pi_1 D_{KL}[p_1||\bar{p}] + \pi_2 D_{KL}[p_2||\bar{p}] = H[\bar{p}] - \pi_1 H[p_1] - \pi_2 H[p_2] \tag{1.9}$$

where $\Pi = \pi_1, \pi_2, 0 < \pi_1, \pi_2 < 1, \pi_1 + \pi_2 = 1$ and $\bar{p} = \pi_1 p_1 + \pi_2 p_2$. JS divergence is related to mutual information. If the weights π_i in Π are taken as the prior probabilities $p(x)$, then the mutual information $I(X;Y)$ between X and Y is equal to the JS divergence between all the conditional distributions, $p(y|x)$.

1.3 Problem Statement

Consider the following problem which is the main motivation for this project: Given random variables X_1, X_2 and Y characterized by the joint distribution $p(x_1, x_2, y)$, we are interested in finding a functional relationship between the variables $\{X_1, X_2\}$ and Y, such that $Y \equiv f(X_1, X_2)$.

The mutual information $I(f(X_1, X_2); Y)$, between $f(X_1, X_2)$ and Y can be used as a quantitative measure that can be optimized for finding f, if indeed such a functional relationship exits.

Moreover, we want this function to be as compact as possible. One way of achieving this could be by imposing a constraint on the cardinality of $f(X_1, X_1)$ or on the entropy of $f(X_1, X_2)$ to be upper bounded by some parameter. This problem can be mathematically formulated as follows:

Problem 1

$$\max_{\substack{f: \\ |f(X_1, X_2)| \leq \Gamma}} \quad I\left(f(X_1, X_2); Y\right)$$

where $|f(X_1, X_2)|$ denotes the cardinality of $f(X_1, X_2)$.

A function $Z = f(X_1, X_2)$ can be generalized as a conditional probability $p(z|x_1, x_2)$. If the conditional probability distribution $p(z|x_1, x_2)$ for a given $\{x_1, x_2\}$ has a value of 1 for only one value of z and zeros for rest of the z's, then this constrained conditional probability represents a function. Therefore in the above problem, the function $f(X_1, X_2)$ can be replaced with another random variable Z, and instead of optimizing for f, we now optimize for the stochastic mapping between the pair of variables $\{X_1, X_2\}$ and Z, given by the conditional probability distribution $p(z|x_1, x_2)$.

As before, in order to make this mapping compact, an upper-bound constraint could be enforced on the the entropy $H(Z|X_1, X_2)$ of the mapping $p(z|x_1, x_2)$. However, this constraint alone is not sufficient as this leads to the following trivial solution: for $Z = \{X_1, X_2\}$, we have $H(Z|X_1, X_2) = 0$ and $I(Z; Y) = I(X_1, X_2; Y)$ which is its global maximum value. So, an additional constraint on the entropy $H(Z)$ of Z is necessary in order to avoid this trivial solution and make this a well defined optimization problem. Accordingly, the formulation in Problem 1 can be rewritten as:

Problem 2

$$\max_{\substack{p(z|x_1, x_2): \\ H(Z|X_1, X_2) \leq \Gamma_1, \\ H(Z) \leq \Gamma_2}} \quad I(Z; Y)$$

This alternative formulation in Problem 2 closely resembles the Information Bottleneck (IB) discussed in Chapter 2, which tries to find a compressed representation Z of the pair of variables $\{X_1, X_2\}$ that is as informative as possible about the variable Y.

Problem 3

$$\max_{\substack{p(z|x_1, x_2): \\ I(Z; X_1, X_2) \leq \Gamma}} \quad I(Z; Y)$$

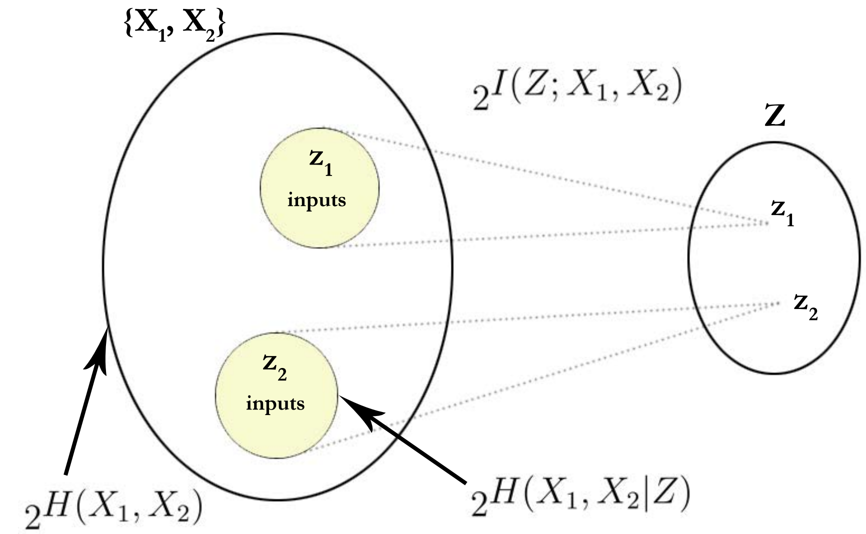


Figure 1.1: An illustration of the relation between the compression-information, $I(Z;X_1,X_2)$ and the average cardinality of the partition of $\{X_1,X_2\}$.

The only difference between Problem 2 and Problem 3 is the constraint being imposed for maximizing $I(Z;Y)$. Problem 2 imposes a constraint on the entropies $H(Z)$ and $H(Z|X_1,X_2)$ of the mapping $p(z|x_1,x_2)$, while the IB method in Problem 3 imposes a constraint on the *compression-information*: $I(Z;X_1,X_2) = H(Z) - H(Z|X_1,X_2)$.

Intuitively, the quantity $I(Z;X_1,X_2)$ can be seen as the compactness of Z as discussed previously in Section 1.2.2. Lower values of $I(Z;X_1,X_2)$ correspond to a more compact Z and higher values for $I(Z;X_1,X_2)$ correspond to higher cardinalities of the functional mapping Z. Using the Asymptotic Equipartition Property (AEP) [5], the probability $p(x_1,x_2)$ assigned to an observed input pair will be close to $2^{-H(X_1,X_2)}$ and the total number of (typical) input pairs is $\approx 2^{H(X_1,X_2)}$. In that sense, $2^{H(X_1,X_2)}$ can be seen as the *volume* of $\{X_1,X_2\}$. Also, for each (typical) value z of Z, there are $2^{H(X_1,X_2|Z)}$ possible $\{x_1,x_2\}$ pairs which map to this z, all of them equally likely. To ensure that no two input pairs map to the same z, the set of possible input pairs $\{x_1,x_2\}$ has to be divided into subsets of size $2^{H(X_1,X_2|Z)}$, where each subset corresponds to some different z. Thus, the average cardinality of the mapping (partition) of $\{X_1,X_2\}$ is given by the ratio of the volume of $\{X_1,X_2\}$ to that of the mean partition (Figure 1.1):

$$\frac{2^{H(X_1,X_2)}}{2^{H(X_1,X_2|Z)}} = 2^{I(Z;X_1,X_2)} \tag{1.10}$$

As a formal characterization of the optimal solution for the information bottleneck method (Problem 3) can be given and there exist several algorithms to solve for this mapping $p(z|x_1,x_2)$, we use this method for identifying functional relationships between random variables rather than using Problem 2 which does not have any known algorithms for solving it.

The next chapter gives an overview of the information bottleneck method, along with the solution characterization and different algorithms to obtain this solution.

Chapter 2

The Information Bottleneck method

2.1 Overview

The Information Bottleneck (IB) method, originally introduced by Tishby et al. [22] is an information theoretic technique for data analysis (compression). The basic idea of this approach is as follows: assuming that the joint probability distribution $p(x, y)$ of two random variables - X and Y is known, we are interested in finding a compressed representation (or quantized codebook) for X, say Z, which is as informative as possible about the random variable Y. This code Z of X is a random variable characterized through a conditional probability distribution $p(z|x)$ which effects a soft partitioning of the values of X. This means that each value of X is associated with all the codebook elements (Z values), with a normalized probability. Intuitively, this approach can be viewed as squeezing the information that the random variable X provides about the random variable Y through a *bottleneck* formed by a limited set of codewords Z. The IB method offers a fundamental trade-off between the complexity of a model and its precision which are respectively reflected by the extent of compression of X and the amount of information the compressed variable Z preserves about Y. Section 2.2 formally defines the IB method, Section 2.3 derives a closed form solution to this problem and Section 2.4 outlines some of the algorithms for achieving this solution.

2.2 Problem Formulation

Formulating the information bottleneck method as an optimization problem can be done along similar lines to the well known rate distortion function [5]. Both these methods seek to find a compressed representation of a random variable X using a quantized codebook Z by minimizing the information rate $I(Z; X)$, which measures the compactness of the new representation Z and characterizes the quality of the quantization. However, this quantity alone is not sufficient to do any meaningful optimization as the compression-information can always be reduced by throwing away details in X. Therefore, some additional constraints are required. It is in these additional constraints that are imposed for performing the optimization, that the rate distortion function and the information bottleneck method differ.

In rate distortion theory, the constraint is imposed by means of a distortion measure $d : X \times Z :\rightarrow \mathbb{R}^+$, which measures the *distance* between the random variable and its new representation. The formulation of the rate distortion function involves the monotonic trade-off between the extent of compression (compactness of the code) and the expected distortion measure. The greater the value of the information rate $I(Z;X)$, the smaller the achievable distortion $\langle d(x,z)\rangle_{p(x,z)}$ and vice-versa. Accordingly, the rate distortion function is defined as the minimal achievable rate under a given upper bounding constraint on the expected distortion. The minimization is performed over all the normalized conditional distributions, $p(z|x)$ for which the distortion constraint is satisfied. However, choosing an *appropriate* distortion function is not trivial, as an arbitrary choice of the distortion function leads to an arbitrary compression.

In contrast to defining a non-trivial distortion measure to compress X, the IB method looks at a *target* variable Y (which is not independent from X) in order to address the same quantization problem, by preserving the *relevant information* about Y. In this case, the distortion upper bound constraint is replaced by a lower bound constraint over the relevant information, given by $I(Z;Y)$. In other words, we wish to minimize $I(Z;X)$; while preserving $I(Z;Y)$ above some minimal level. Equivalently, the same problem can be formulated as maximizing the relevant information $I(Z;Y)$ while constraining the compression-information $I(Z;X)$ below some maximal level. The IB method can thus be formulated in the following two ways:

$$\max_{\substack{p(z|x):\\ I(Z;X)\leq\Gamma_1}} \quad I(Z;Y) \tag{2.1}$$

$$\text{or} \quad \min_{\substack{p(z|x):\\ I(Z;Y)\geq\Gamma_2}} \quad I(Z;X) \tag{2.2}$$

where Γ_1 is a parameter which upper bounds the compression-information $I(Z;X)$; while maximizing the relevant information $I(Z;Y)$ and Γ_2 is a parameter which lower bounds the relevant information $I(Z;Y)$ while minimizing the compression-information $I(Z;X)$.

The lossy compression Z cannot convey more information about Y than the original data X as Z depends only on X. This comes from the Data Processing inequality [5] which states that $I(Z;Y) \leq I(X;Y)$. In effect, we pass the information that X provides about Y through a *bottleneck* formed by the compact summaries in Z. Similar to the rate distortion theory, there is a trade-off between compressing the representation and preserving meaningful information. The only assumption of the IB method is that the input is given in the form of the joint distribution $p(x,y)$.

2.2.1 Some Equalities

As Z is a compressed representation of X, it should be completely defined by X alone. This means that Z, X and Y form the following Markovian relation:

$$Z \leftrightarrow X \leftrightarrow Y \tag{2.3}$$

This Markovian relation implies that

$$p(z|x,y) = p(z|x) \tag{2.4}$$

Subsequently,

$$p(x,y,z) = p(x,y)p(z|x,y) = p(x,y)p(z|x) \tag{2.5}$$

$$p(z|y) = \frac{1}{p(y)}\sum_x p(x,y,z) = \frac{1}{p(y)}\sum_x p(x,y)p(z|x) = \sum_x p(x|y)p(z|x) \tag{2.6}$$

$$p(y|z) = \frac{1}{p(z)}\sum_x p(x,y,z) = \frac{1}{p(z)}\sum_x p(x,y)p(z|x) = \sum_x p(y|x)p(x|z) \tag{2.7}$$

Also, from Bayes' rule we have:

$$p(z) = \sum_x p(x)p(z|x) \tag{2.8}$$

Differentiating Equations 2.8 and 2.6 w.r.t. $p(z|x)$ leads to:

$$\frac{\delta p(z)}{p(z|x)} = p(x) \tag{2.9}$$

$$\frac{\delta p(z|y)}{p(z|x)} = p(x|y) \tag{2.10}$$

2.3 Solution Characterization

The IB problem of minimizing $I(Z;X)$ is a concave function of $p(x)$ for fixed $p(z|x)$, and a convex function of $p(z|x)$ for a fixed $p(x)$. Therefore, this is a constrained minimization problem of a convex function over the convex set of all $p(z|x)$ which satisfy the lower bound constraint on the relevant information $I(Z;Y)$. This is a variational problem that can be solved by introducing Lagrange multipliers, β for the relevant information constraint and $\lambda(x)$ for the normalization of the conditional distributions $p(z|x)$ at each x. Accordingly, the functional to be minimized is given by:

$$\begin{aligned}\mathcal{L}[p(z|x)] =& I(Z;X) - \beta I(Z,Y) - \sum_{x,z}\lambda(x)p(z|x) \qquad (2.11)\\ =& \sum_{x,z} p(z,x)\log\left[\frac{p(z|x)}{p(z)}\right] - \beta\sum_{x,z} p(z,y)\log\left[\frac{p(z|y)}{p(z)}\right] - \sum_{x,z}\lambda(x)p(z|x)\end{aligned}$$

The solution to this variational problem can then be obtained by taking the derivative of $\mathcal{L}[p(z|x)]$ w.r.t. $p(z|x)$ and setting it to zero for given x and z. This gives:

$$\begin{aligned}\frac{\delta\mathcal{L}[p(z|x)]}{\delta p(z|x)} &= p(x)\left[1+\log(p(z|x))\right] - \sum_x p(x|z)\frac{\delta p(z)}{\delta p(z|x)}\left[1+\log(p(z))\right]\\ &\quad -\beta\sum_y p(y)\frac{\delta p(z|y)}{\delta p(z|x)}\left[1+\log(p(z|y))\right] + \beta\sum_y p(y|z)\frac{\delta p(z)}{\delta p(z|x)}\left[1+\log(p(z))\right] - \lambda(x)\\ &= p(x)\left[1+\log(p(z|x))\right] - p(x)\left[1+\log(p(z))\right] - \beta\sum_y p(y)p(x|y)\left[1+\log(p(z|y))\right]\\ &\quad +\beta p(x)\left[1+\log(p(z))\right] - \lambda(x) \quad \text{[from Eq. 2.9, 2.10]}\\ &= p(x)\log\left[\frac{p(z|x)}{p(z)}\right] - \beta\sum_y p(x,y)\left[1+\log(p(z|y))\right] + \beta\sum_y p(x,y)\left[1+\log(p(z))\right] - \lambda(x)\end{aligned}$$

$$
\begin{aligned}
&= p(x)\log\left[\frac{p(z|x)}{p(z)}\right] - \beta\sum_y p(x,y)\log\left[\frac{p(z|y)}{p(z)}\right] - \lambda(x) \\
&= p(x)\left\{\log\left[\frac{p(z|x)}{p(z)}\right] - \beta\sum_y p(y|x)\log\left[\frac{p(y|z)}{p(y)}\right] - \frac{\lambda(x)}{p(x)}\right\}
\end{aligned}
$$

The quantity $I(x;Y) \equiv \sum_y p(y|x)\log[p(y|x)/p(y)]$ which is the contribution of x towards the total information $I(X;Y)$ depends only on x and thus can be absorbed into the multiplier $\lambda(x)$. By setting

$$
\tilde{\lambda}(x) = \frac{\lambda(x)}{p(x)} - \beta\sum_y p(y|x)\log\left[\frac{p(y|x)}{p(y)}\right],
$$

the final variational condition is given according to:

$$
\frac{\delta\mathcal{L}[p(z|x)]}{\delta p(z|x)} = p(x)\left\{\log\left[\frac{p(z|x)}{p(z)}\right] + \beta\sum_y p(y|x)\log\left[\frac{p(y|x)}{p(y|z)}\right] - \tilde{\lambda}(x)\right\} = 0 \qquad (2.12)
$$

By setting $\log Z(x,\beta) = \beta\tilde{\lambda}(x)$, we obtain:

$$
p(z|x) = \frac{p(z)}{Z(x,\beta)}e^{-\beta D_{KL}[p(y|x)||p(y|z)]}, \quad \forall x, \forall z \qquad (2.13)
$$

This is a formal solution since $p(z)$ and $p(y|z)$ on the right hand side of the equation are implicitly determined using $p(z|x)$ (Equations 2.8 and 2.7). The final solution in Equation 2.13 along with these two equations, self-consistently determine the optimal solution. Moreover, the KL divergence $D_{KL}[p(y|x)||p(y|z)]$, emerges as the relevant distortion measure from the IB principle, rather than having to assume it in advance. Therefore, $d(x,z) = D_{KL}[p(y|x)||p(y|z)]$, in this sense, is the correct compression distortion measure for the IB method.

The parameter β controls the trade-off between the extent of compression of the variable X and the amount of information retained in Z about Y. As $\beta \to \infty$, we achieve arbitrarily detailed quantization, i.e., minimal compression and we are focused only on preserving the relevant information. One solution in this case where Z copies X and we have $I(Z;Y) = I(X;Y)$. In this case, as there is no compression, $I(Z,X) = H(X)$ is maximized as well. On the other hand, as $\beta \to 0$, we achieve maximum compression and all the values of X are mapped to a single value of Z. In this case, the compression is optimal, $I(Z;X) = 0$, but all the relevant information is lost as well, $I(Z;Y) = 0$. Thus, by varying β, this trade-off between compression and preservation of meaningful information can be explored at different resolutions.

Figure 2.1 plots this *relevance-compression* curve obtained by plotting $I(Z;Y)/I(X;Y)$ versus $I(Z;X)/H(X)$ computed at different β values ranging from 0 to ∞. This is a normalized curve as $I(Z;Y)$ and $I(Z;X)$ are upper bounded by $I(X;Y)$ and $H(X)$ respectively. This curve separates the plane into two regions: the region below the curve is the achievable region, i.e., any point below this curve denotes a compression level and relevant information that can be achieved. On the other hand, the region above this curve is non-achievable.

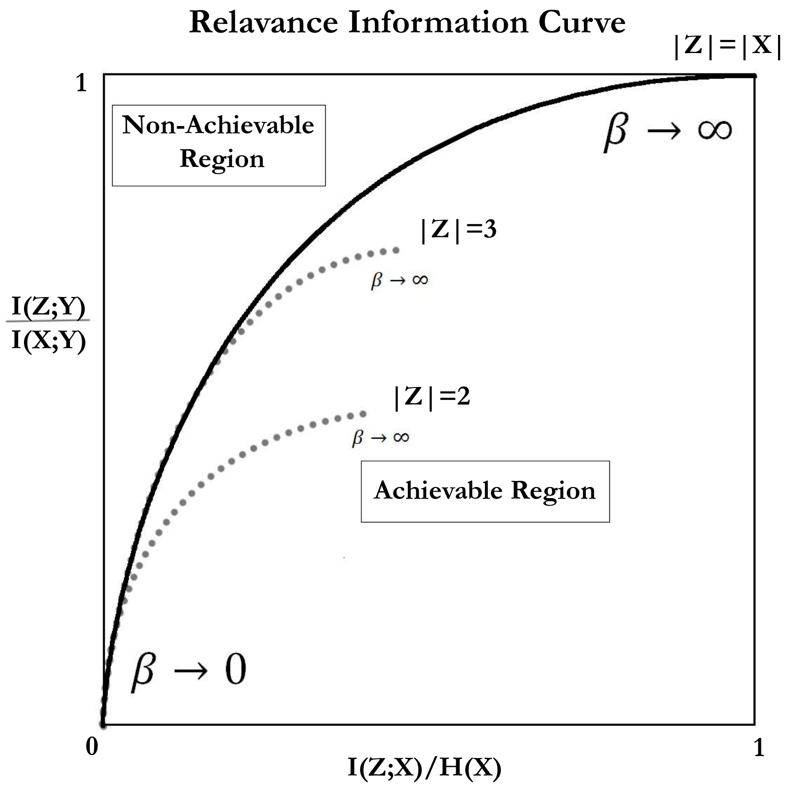


Figure 2.1: The information bottleneck relevance-information curve. The dotted curves are a family of sub-optimal curves obtained by constraining the cardinality of Z.

Another important issue to look at is the set of representatives of the compressed variable Z. The rate distortion function implicitly defines the set of representative of Z through the chosen distortion measure. The question of how to choose an optimal set of representatives is disregarded in rate distortion theory. On the other hand, the information bottleneck method does not make any assumption on the set of representatives of Z. It simultaneously optimizes for the soft partitioning probabilities $p(z|x)$, the probability distribution $p(z)$ of the compressed variable and also over the cluster representatives $p(y|z)$. However, the optimization results only in the above mentioned probability quantities, and it does not present a way for us to obtain the support $\mathcal{Z}$ of the compressed variable Z. The values z which the compressed variable takes are not part of the IB solution. We will discuss more on this aspect of the IB method in Chapter 3 which focuses on using the IB method for functional identification.

2.4 IB Algorithms

There are four different complementary algorithms for solving the IB variational principle in an exact or an approximate way. The original paper [22] proposed an iterative algorithm and a deterministic annealing based algorithm. Other greedy algorithms exist, such as the agglomerative algorithm [15] and the sequential algorithm [16], that are more suited for clustering applications. A comparison of these four algorithms and their applications is discussed in detail in [14]. In this project we use the iterative algorithm for functional identification and the sequential algorithm for clustering.

2.4.1 Iterative Algorithm

Algorithm 1 IB Iterative Algorithm: Pseudo-code

```
**Inputs :** joint distribution $p(x,y)$, cardinality parameter $|\mathcal{Z}|$ of $Z$, trade-off parameter $\beta$ and convergence parameter $\epsilon$.
**Output:** probability distribution $p(z)$ of $Z$, soft partitioning $p(z|x)$ into $K$ clusters, representative distributions $p(y|z)$.
**Algorithm:**
Randomly initialize $p^0(z|x)$.
$p^0(z) \leftarrow \sum_x p(x)p^0(z|x)$.
$p^0(y|z) \leftarrow \frac{1}{p^0(z)} \sum_x p(x,y)p^0(z|x)$.
while true do
    $p^{k+1}(z|x) \leftarrow \frac{p^k(z)}{Z^{k+1}(x,\beta)} e^{-\beta D_{KL}[p(y|x)||p^k(y|z)]}, \quad \forall x, z$
    $p^{k+1}(z) \leftarrow \sum_x p(x)p^{k+1}(z|x), \quad \forall z$
    $p^{k+1}(y|z) \leftarrow \frac{1}{p^{k+1}(z)} \sum_x p(x,y)p^{k+1}(z|x), \quad \forall y, z$
    if $\forall x \in \mathcal{X}, JS_{\{0.5,0.5\}}[p_{k+1}(\tilde{x}|x), p_k(\tilde{x}|x)] < \epsilon$ then Break.
```

The self-consistent equations of the IB method derived previously can be used for finding the unknown distributions at different values of β. These self-consistent equations 2.7, 2.8 and 2.13, are satisfied simultaneously at the minima of the functional,

$$\mathcal{F}[p(z|x); p(z); p(y|z)] = -\langle \log Z(x,\beta) \rangle_{p(x)} = I(X;Z) + \beta \langle D_{KL}[p(y|x)||p(y|z)] \rangle_{p(x,z)}$$

where the minimization is performed independently over the convex sets of the normalized distributions, $\{p(z)\}$, $\{p(z|x)\}$, $\{p(y|z)\}$. Namely,

$$\min_{[p(z|x)} \min_{p(z)} \min_{p(y|z)} \mathcal{F}[p(z|x); p(z); p(y|z)] \tag{2.14}$$

The minimization is performed by the converging alternating iterations. Algorithm 1 outlines the pseudo-code of this algorithm for a fixed value of β. The updates defined by these equations can only decrease [22] this functional in Equation 2.14 which is lower bounded by zero and thus, the algorithm converges to a locally optimum solution. However, this algorithm does not yield a unique solution, as the functional $\mathcal{F}[p(z|x); p(z); p(y|z)]$ is not jointly convex in the three distributions, but is only convex in each of the distributions independently.

The iterative algorithm requires the cardinality of the compression variable Z to be specified as an input. The relevance-compression curve of Figure 2.1 can be alternatively be interpreted in terms of the cardinality variable which increases monotonically along the curve [14]. At the maximal compression, Z is at its most compact representation ($|\mathcal{Z}| = 1$). By gradually increasing β, the constraint over $I(Z;Y)$ become more and more demanding until the single value of Z bifurcates into two values in order to fulfill the relevant information constraint. This process continues resulting in additional splits by successive increases in β. Eventually, at the limit $\beta \to \infty$, we look only at retaining all the relevant information and not at compression, thus setting the cardinality $|\mathcal{Z}|$ to its maximum value of $|\mathcal{X}|$.

This means that for practically applying the iterative algorithm, instead of choosing the right value of β, we could fix the cardinality of Z to some value less than $|\mathcal{X}|$, and choose a very high value for β. This behavior is depicted in the relevance compression curves (Figure 2.1) plotted with a constraint on $|\mathcal{Z}|$, resulting in a family of sub-optimal characteristic curves. Essentially, we are enforcing the constraint on compression by restricting the cardinality of Z to a value below $|\mathcal{X}|$ and can thus set high value for β while applying this iterative algorithm.

2.4.2 Sequential Algorithm

The sequential algorithm proposed in [16] performs *hard* mapping between the input variable X and the compressed variable Z by looking at the following equivalent IB maximization problem:

$$\mathcal{L}' = I(Z;Y) - \beta^{-1} I(Z;X) \tag{2.15}$$

The algorithm begins by randomly partitioning $\mathcal{X}$ into K classes. At each step, every $x \in \mathcal{X}$ is drawn from its current cluster z and represented as a singleton cluster. It is then merged into z^{new} that minimizes the merging criterion $\Delta\mathcal{L}'(z, \{x\})$ which is the difference in $\mathcal{L}'$ before and after merging x into its new cluster. Algorithm 2 gives a pseudo-code of this method.

Algorithm 2 IB Sequential Algorithm: Pseudo-code

```
Inputs : joint distribution $p(x,y)$, trade-off parameter $\beta$ and cardinality $|\mathcal{Z}|$ of $Z$
Output: A hard partition $Z$ of $\mathcal{X}$ into K clusters.
Algorithm:
while true do
    for all $x \in \mathcal{X}$ do
        Remove $x$ from its cluster $z$
        $z^{new} = \min_z \Delta\mathcal{L}'(z, \{x\})$
        Merge $x$ into $z^{new}$
    if $\forall x \in X, z^{new} = z$ then Break.
```

2.5 IB Applications

Most applications of the IB method proposed over the years have been in the domain of clustering. A few of them include using word-clusters for supervised and unsupervised text classification [18], [19], gene expression data analysis [23], galaxy spectra analysis [17] and image clustering [7] among others.

Following are a few works in computational neuroscience which use the IB method. All these works assign particular entities to the input variable X and the target variable Y, and then infer insights from the data by analyzing the compressed variable Z obtained by the IB method.

2.5.1 Other uses of the IB method in Neuroscience

In [13], the variable in consideration is the stimulus presented to a H1 neuron in the visual system of a fly. This stimulus is in the form of a long movie, with s_t being the stimulus portion preceding time t. The target variable is the neural response in the form of spike trains which are discretized into time bins of size Δt. Using the the IB method, they extract stimuli features which are essentially clusters of times along the stimulus movie, that maximize the information regarding the resulting spike trains. Similar work is done in [12] on the neural population of a retinal ganglion celles from a salamander where instead of clustering the stimuli, the neurons are clustered. They group the N neurons in the population into classes by mapping: neuron $i \to$ class C using the IB method. To do so, they use the neurons in the population as the original variable and the neural responses from each neuron in the population as the target variable. Visual stimulus of the spatially uniform flicker is presented to the population using a multi-electrode array. The mapping of the neurons into classes is done to capture as much information as possible about the stimulus-response relation while constraining the amount of information that class labels provide directly about the identity of the neuron.

The temporal aspect of the signals is dealt with in [1] where they suggest that the internal representations that an organism maintains about the outside world are constructed so that the information about the future of sensory inputs is maximized at a fixed value of the information about its past. As an example, if X_{past} are the past sensory inputs to a single neuron, for times $-T < t \leq 0$ and X_{future} are its future sensory inputs, then the the goal is to perform neural coding of predictive information by finding the internal representation X_{int} of the neuron (spike train response) using the IB method. The mapping $X_{past} \to X_{int}$ minimizes the information $I(X_{int}; X_{past})$ about the past while maintaining information $I(X_{int}; X_{future})$ about the future.

IB algorithm is used to propose an online learning rule for the synaptic weights of neurons in [3]. They consider the synapses, and subsequently the input spike trains from M different subgroups G_l, each of size N/M to a linear Poisson neuron with N synaptic weights $w_j, j = 1, ..., N$ as the original variable. The spike trains, G_l are generated from Poisson processes which can be of a constant or a modulated rate. The target signal Y_T is chosen as the sum of two Poisson trains (two of the G_ls), one with a constant rate and the other with a modulated rate. The goal is to predict the spike output of the neuron which receives the N input spike trains as inputs. The output spike train depends on the input synaptic weights and accordingly, they propose an online learning rule which updates the weights $w_j(t)$ at time t by performing gradient ascent on the IB objective function. In [8], [9], the compressed variable being sought is the output spike train of the learning neuron, which depends on the synaptic weights. An update rule for the synaptic weights of the learning neuron is then derived using IB optimization.

[4] modifies the IB learning rule from [9] to make it simpler and more transparent. A neuron that has N synaptic weights $w_1, ..., w_N$ taking in an input consisting of N spike trains is considered as a bottleneck since it compresses its high dimensional input history to a single output spike train. This mapping is parameterized by the weights for which a learning rule is to be found. The main assumption in this work is that the bottleneck neuron has access to a rich preprocessing of a relevance signal. An estimation of the gradient of the relevant information w.r.t. to the weights is required for the optimization and is parameterized with some parameters of the preprocessing of the relevance signal. The compressed representation, as before is the output of the bottleneck neuron. The weights are learned such that the relevant information contained in the neuron output is maximal under some constraints.

Here we attempt to apply the IB method in computational neuroscience in an entirely different way by using the output Z of the IB algorithms for identifying functional relationships between different random variables. This is the topic of discussion in the next chapter.

Chapter 3

The IB method for Functional Identification

3.1 Problem Formulation

Let X_i $(i = 1, ..., n)$ be n mutually independent random variables with support $\mathcal{X}_i$ and a joint probability distribution $p(x_1, ..., x_n)$. Suppose Y is an observed random variable with support $\mathcal{Y}$ that is assumed to be a function $\mathcal{F}$ of these n input variables, corrupted by an independent additive noise W with probability distribution $p(w)$.

$$Y = \mathcal{F}(X_1, ..., X_n) + W \tag{3.1}$$

The goal is to recover the function $\mathcal{F}$ using the joint probability distribution $p(x_1, ..., x_n, y)$ of the input and the observed variables, if indeed such a function exists. Moreover, we also want to be able to identify scenarios when such a functional relationship is not possible based on the given joint distribution. Specifically, if the function $\mathcal{F}$ is linear, then this problem amounts to estimating the coefficients α_i $(i = 1, ..., n)$ in the below equation:

$$Y = \sum_{i=1}^{n} \alpha_i X_i + W \tag{3.2}$$

The rest of this chapter deals with this scenario when the function is assumed to be linear and proposes an algorithm for estimating these linear coefficients, if a functional relationship exists. Let the estimated values of the coefficients α_i be denoted by $\hat{\alpha}_i, i = 1, ..., n$. Therefore, given the joint probability distribution $p(x_1, ..., x_n, y)$, we would like recover a Z of the form

$$Z = \sum_{i=1}^{n} \hat{\alpha}_i X_i \tag{3.3}$$

which captures the function $\mathcal{F}$.

In order to solve this problem, the information bottleneck (IB) method is a good candidate because the goal of the IB method is to find a compact representation Z of the input variable X that is as informative as possible about the observed variable Y. If we can force the Z thus obtained from the IB method to be of the form given in Equation 3.3, we can find the estimates $\hat{\alpha}_i, i = 1, ..., n$.

However, using only information measures for solving this problem leads to an inherent limitation that prevents us from uniquely estimating these coefficients $\hat{\alpha}_i$. The next section addresses this issue.

3.2 Limitation of Information Measures

Define Z' as follows:

$$Z' = \mathcal{G}\left(\sum_{i=1}^{n} \gamma \hat{\alpha}_i X_i\right) = \mathcal{G}(\gamma Z)$$

where $\mathcal{G}$ is a uniquely invertible one-to-one mapping and $\gamma \in \mathbb{R}$ is a constant.

For a Z' defined in such way, $I(Z;Y) = I(Z';Y)$. This result is trivial for the case where all the involved variables are discrete because the computed probabilities $p(z')$, $p(z'|x_i, ..., x_n)$ and $p(y|z')$ would respectively be equal to $p(z)$, $p(z|x_1, ..., x_n)$ and $p(y|z)$ as there would be a one-to-one mapping between $\mathcal{Z}$ and $\mathcal{Z}'$, and thus $I(Z;Y) = I(Z';Y)$.

This result also holds true for the continuous case due to the following argument: If Z' is a *homeomorphism* (smooth and uniquely invertible map) of Z and $J_Z = ||\partial Z/\partial Z'||$ is the Jacobian determinant of the transformation, then

$$p(z') = J_Z(z')p(z) \text{ and } p(z', y) = J_Z(z')p(z, y) \tag{3.4}$$

which gives

$$\begin{aligned} I(Z';Y) &= \int\int dz'dyp(z', y) \log \frac{p(z', y)}{p(z')p(y)} \\ &= \int\int dzdyp(z, y) \log \frac{p(z, y)}{p(z)p(y)} \\ &= I(Z;Y) \end{aligned} \tag{3.5}$$

Therefore, the estimates $\hat{\alpha}_i$ of the coefficients α_i cannot be uniquely determined and can only be estimated up to a scale factor γ. As a result, there will be ambiguity in the scale γ of these estimates. It should be noted that this is not a limitation caused by using the IB method, but is an inherent limitation of using only information measures for solving this problem. One can only expect to estimate the ratios - $\frac{\alpha_1}{\alpha_k}, \frac{\alpha_2}{\alpha_k}, ..., \frac{\alpha_n}{\alpha_k}$, for any $k \in \{1, ..., n\}$, $\alpha_k \neq 0$. However, this limitation is not a serious one as in most real life applications, it is these ratios that are more intuitive to interpret rather than the actual values themselves. For instance, in the neuroscience setting, we would be more interested in the ratios of contributions from different neurons rather than the absolute contribution of each particular neuron which may not add much meaning.

Before we go further and attempt to use the IB method to identify this linear functional relationship between Y and $\{X_1, ..., X_n\}$, let us consider the case when the inputs are Gaussian random variables so that we can analytically compute the various information quantities which we deal with in the IB method.

3.3 Gaussian Case

Consider the case where the inputs are mutually independent jointly Gaussian distributed random variables with $X_i \sim \mathcal{N}(0, P), i = 1, ..., n$. Let Y be a linear combination of these input X_i's with an additive independent noise which is also a Gaussian variable with $W \sim \mathcal{N}(0, N)$ and Z be the variable to be estimated:

$$Y = \sum_{i=1}^{n} \alpha_i X_i + W \quad \text{and} \quad Z = \sum_{i=1}^{n} \hat{\alpha}_i X_i \tag{3.6}$$

Additionally, let us artificially restrict to a scalar Z which is jointly Gaussian with X_i's and Y. Then, define a quantity $\rho(Z)$ as follows:

$$\rho(Z) = \frac{I(Z;Y)}{I(X_1, ..., X_n; Y)} \tag{3.7}$$

As mentioned previously in Chapter 2, because of the data processing inequality, this quantity $\rho(Z)$ is upper bounded by 1 which is attained when $Z = \{X_1, ..., X_n\}$. If we denote the vectors $[\alpha_1 ... \alpha_2]^T$ and $[\hat{\alpha}_1 ... \hat{\alpha}_2]^T$ by $\boldsymbol{\alpha}$ and $\hat{\boldsymbol{\alpha}}$ respectively, then we have:

$$I(X_1, ..., X_n; Y) = \frac{1}{2} \ln \left[1 + ||\boldsymbol{\alpha}||^2 \frac{P}{N} \right] \tag{3.8}$$

$$I(Z;Y) = \frac{1}{2} \ln \left[\frac{1 + ||\boldsymbol{\alpha}||^2 \frac{P}{N}}{1 + \frac{||\boldsymbol{\alpha}||^2 ||\hat{\boldsymbol{\alpha}}||^2 - \langle \hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha} \rangle^2}{||\hat{\boldsymbol{\alpha}}||^2} \frac{P}{N}} \right] \tag{3.9}$$

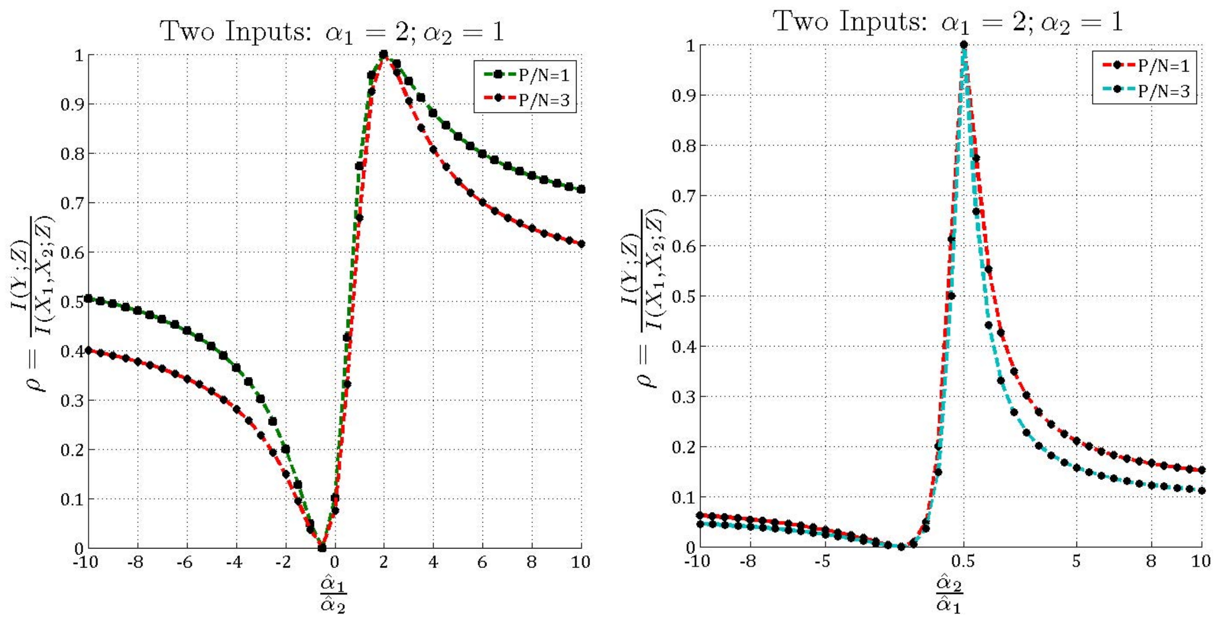
From the above two equations we get,

$$\rho(Z) = 1 - \frac{\ln \left[1 + \frac{||\boldsymbol{\alpha}||^2 ||\hat{\boldsymbol{\alpha}}||^2 - \langle \hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha} \rangle^2}{||\hat{\boldsymbol{\alpha}}||^2} \frac{P}{N} \right]}{\ln \left[1 + ||\boldsymbol{\alpha}||^2 \frac{P}{N} \right]} \tag{3.10}$$

This expression attains its maximum value of 1, when

$$\begin{aligned} ||\boldsymbol{\alpha}||^2 ||\hat{\boldsymbol{\alpha}}||^2 - \langle \hat{\boldsymbol{\alpha}}, \boldsymbol{\alpha} \rangle^2 &= 0 \\ \text{i.e. } \hat{\boldsymbol{\alpha}} &= \gamma \boldsymbol{\alpha}, \text{ for some } \gamma \end{aligned} \tag{3.11}$$

From this equation we see that $I(Z;Y)$ becomes equal to $I(X_1, ..., X_n; Y)$ for all estimates $\hat{\boldsymbol{\alpha}}$ that are multiples of the original coefficients $\boldsymbol{\alpha}$. This result is consistent with the discussion in the preceding section (Section 3.2). For the two input Gaussian case ($n = 2$), Figure 3.1 depicts $\rho(Z)$ computed according to Equation 3.10 as a function of the ratios of the estimated coefficients $\hat{\alpha}_1/\hat{\alpha}_2$ and $\hat{\alpha}_2/\hat{\alpha}_1$. From these plots we see that if the computed $\hat{\boldsymbol{\alpha}}$ are such that, $I(Z;Y)$ is *close* to $I(X_1, ..., X_n; Y)$, then these estimated coefficients are also *close* to the original coefficients $\boldsymbol{\alpha}$ up to a scale factor, due to the sharp peaks in the plots at these points.



(a) $\rho\left(\frac{\hat{\alpha}_1}{\hat{\alpha}_2}\right)$

(b) $\rho\left(\frac{\hat{\alpha}_2}{\hat{\alpha}_1}\right)$

Figure 3.1: $\rho\left(\frac{\hat{\alpha}_1}{\hat{\alpha}_2}\right)$ and $\rho\left(\frac{\hat{\alpha}_2}{\hat{\alpha}_1}\right)$ for 2 Gaussian inputs at different SNR levels. We see sharp peaks where the coefficients of Z are equal to the actual coefficients of Y up to a scale factor.

Therefore, by using the information bottleneck if we are able to find a compact Z such that $I(Z;Y)$ is as close a possible to $I(X_1,...,X_n;Y)$, then the computed coefficients from this Z reflect the original functional relationship between $\{X_1,...,X_n\}$ and Y up to a scale factor. The next section discusses different methods for estimating these coefficients once the IB algorithm outputs a compact Z.

3.4 Estimating coefficients from Z

The information bottleneck method finds a compressed representation Z of $(X_1,...,X_n)$ which is as informative as possible about a target variable Y. The iterative IB algorithm takes in the joint probability distribution $p(x_1,...,x_n,y)$, the cardinality $|\mathcal{Z}|$ of Z and the Lagrange multiplier β which controls the extent to which the information about Y is preserved in Z . The estimates $\hat{\alpha}_i$ of the coefficients α_i, can then be obtained by casting the obtained Z into the form:

$$Z = \sum_{i=1}^{n} \hat{\alpha}_i X_i$$

However, by using the IB method as mentioned earlier in Chapter 2, the actual support $\mathcal{Z}$ of Z cannot be determined; instead, Z is characterized only through the distributions $p(z), p(z|x_1,...,x_n)$ and $p(y|z)$ which are iteratively solved for by the algorithm. The values z which the compressed variable takes are not part of the IB solution returned by this iterative algorithm. Therefore, certain heuristics need to be used in order to find the z values so that the $\hat{\alpha}_i$'s can be estimated. In this project three methods are proposed to estimate these coefficients using the output variable Z from the IB method.

The first method does not make any assumptions on the support of Z and just uses the *labels* of Z for which each input value $(x_1, ..., x_n)$ has a mapping, in order to compute the coefficients. This method is computationally expensive because we analyze each pair of inputs which lead to the same mapping to the compressed variable and solve the resulting system of linear equations. The other two methods try to associate the support $\mathcal{Z}$ to some values in the support of $\mathcal{Y}$ and then solve the resulting over-determined system of equations in a least squares sense. These three approaches for estimate these coefficients are elaborated below:

3.4.1 Method 1

Solve the below system of equations for $\left(\frac{\hat{\alpha}_1}{\hat{\alpha}_n}, \frac{\hat{\alpha}_2}{\hat{\alpha}_n}, ..., \frac{\hat{\alpha}_n}{\hat{\alpha}_n}\right)$:

$$\begin{gathered}\sum_{i=1}^{n} \frac{\hat{\alpha}_i}{\hat{\alpha}_n} x_i - \sum_{i=1}^{n} \frac{\hat{\alpha}_i}{\hat{\alpha}_n} x'_i = 0, \\ \forall (x_1, ..., x_n) \in (\mathcal{X}_1 \times ... \times \mathcal{X}_n), \text{ and } (x'_1, ..., x'_n) \in (\mathcal{X}_1 \times ... \times \mathcal{X}_n), \\ \text{such that } \max_z p(z|x_1, ..., x_n) = \max_z p(z|x'_1, ..., x'_n) \end{gathered} \tag{3.12}$$

3.4.2 Method 2

Solve the below system of equations for $(\hat{\alpha}_1, ..., \hat{\alpha}_n)$:

$$\begin{aligned}&\sum_{i=1}^{n} \hat{\alpha}_i x_i = \mathbb{E}[p(y|z^*)] \;\; \forall (x_1, ..., x_n) \in (\mathcal{X}_1 \times ... \times \mathcal{X}_n) \\ &\text{where } z^* = \max_z \{p(z|x_1, ..., x_n)\}\end{aligned} \tag{3.13}$$

3.4.3 Method 3

Solve the below system of equations for $(\hat{\alpha}_1, ..., \hat{\alpha}_n)$:

$$\begin{aligned}&\sum_{i=1}^{n} \hat{\alpha}_i x_i = \max_y \{p(y|z^*)\} \;\; \forall (x_1, ..., x_n) \in (\mathcal{X}_1 \times ... \times \mathcal{X}_n) \\ &\text{where } z^* = \max_z \{p(z|x_1, ..., x_n)\}\end{aligned} \tag{3.14}$$

Although, methods 2 and 3 solve explicitly for $(\hat{\alpha}_1, ..., \hat{\alpha}_n)$, it is only the ratios $\left(\frac{\hat{\alpha}_1}{\hat{\alpha}_k}, \frac{\hat{\alpha}_2}{\hat{\alpha}_k}, ..., \frac{\hat{\alpha}_n}{\hat{\alpha}_k}\right)$ which have to be considered as those are the best one could expect to be able to retrieve in this setup. These two methods only give an appropriate scaling for the possible values of Z.

The nature of these three methods is such that they will always output *some* coefficients irrespective of whether a functional relationship exists between the input and observed random variables or not. Therefore, an additional final check needs to be performed to ensure that the coefficients obtained from these three methods actually correspond to a compact function. The next section describes how this final test can be performed.

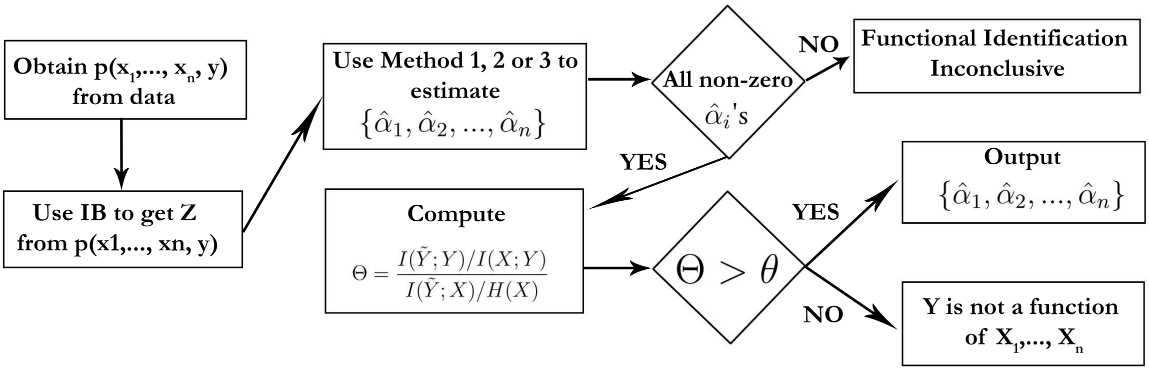

Obtain p(x$_1$,..., x$_n$, y) from data
Use IB to get Z from p(x1,..., xn, y)
Use Method 1, 2 or 3 to estimate $\{\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_n\}$
All non-zero $\hat{\alpha}_i$'s
NO
Functional Identification Inconclusive
YES
Compute $\Theta = \frac{I(\tilde{Y};Y)/I(X;Y)}{I(\tilde{Y};X)/H(X)}$
$\Theta > \theta$
YES
Output $\{\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_n\}$
NO
Y is not a function of X$_1$,..., X$_n$

Figure 3.2: Overall Functional Identification Algorithm.

3.4.4 Function or not ?

Given the joint distribution between the input and observed random variables $p(x_1, ..., x_n, y)$, the preceding subsections give algorithms to always estimate some linear coefficients $\{\hat{\alpha}_1, \hat{\alpha}_2, ..., \hat{\alpha}_n\}$, up to a scale factor. From now on let us assume that these coefficients are normalized with respect to one of the non-zero coefficients $\hat{\alpha}_k$ and rounded to the nearest integer (denoted by [.]) as follows:

$$\{\hat{\alpha}_1, ..., \hat{\alpha}_k, ..., \hat{\alpha}_n\} \triangleq \left\{ \left[\frac{\hat{\alpha}_1}{\hat{\alpha}_k}\right], ..., \left[\frac{\hat{\alpha}_k}{\hat{\alpha}_k}\right], ..., \left[\frac{\hat{\alpha}_n}{\hat{\alpha}_k}\right] \right\} \tag{3.15}$$

Once we have these normalized coefficients $\{\hat{\alpha}_1, ..., \hat{\alpha}_n\}$, we need to decide whether these coefficients should be accepted or discarded, i.e., decide the validity of our obtained results. In order to do this, consider the random variable $\tilde{Y}$ defined using these normalized coefficients as follows:

$$\tilde{Y} = \sum_{i=1}^{n} \hat{\alpha}_i X_i \tag{3.16}$$

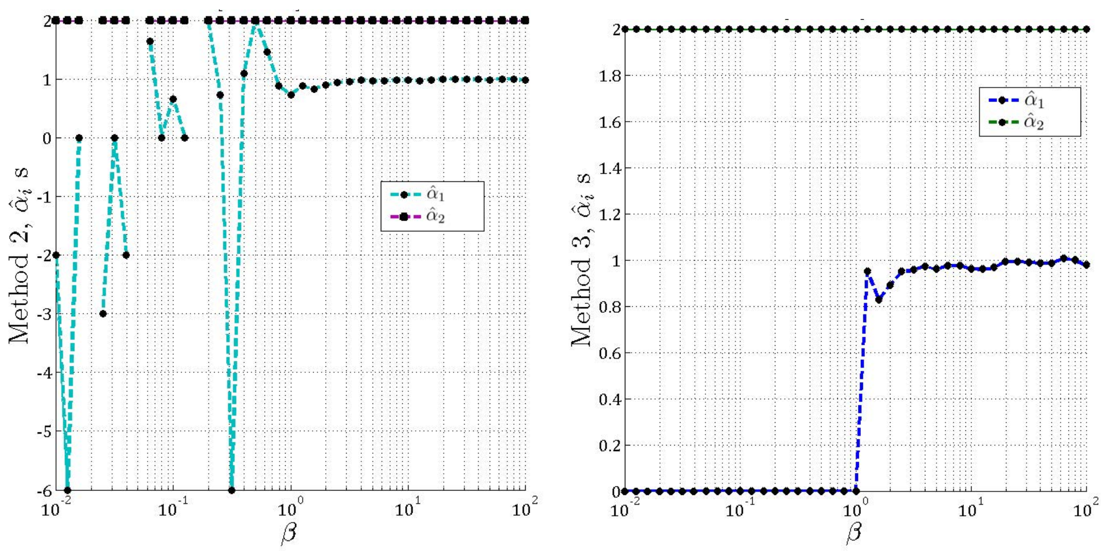
Compute $p(\tilde{y}, y)$ according to the below equation:

$$p(\tilde{y}, y) = \sum_{\substack{x_1,...,x_n: \\ \tilde{y}=\hat{\alpha}_1 x_1 + ... + \hat{\alpha}_n x_n}} p(x_1, ..., x_n, y, \tilde{y}) \tag{3.17}$$

Subsequently compute the quantity Θ defined by:

$$\Theta = \frac{I(\tilde{Y};Y)/I(X;Y)}{I(\tilde{Y};X)/H(X)} \tag{3.18}$$

Accept the normalized coefficients if this so computed Θ is greater than some threshold θ ($\Theta > \theta$). A typical value for this threshold could be 1, as in this indicates that the coefficients represent a compact random variable which has more normalized information about Y than the normalized information about X. The greater the value of this Θ, the greater the confidence with which we can accept the estimated coefficients. The flowchart of the entire procedure is given in Figure 3.2.



(a) Estimating $\hat{\alpha}_1$ and $\hat{\alpha}_2$ using Method 2 when $\alpha_1 = 1, \alpha_2 = 2$.

(b) Estimating $\hat{\alpha}_1$ and $\hat{\alpha}_2$ using Method 3 when $\alpha_1 = 1, \alpha_2 = 2$.

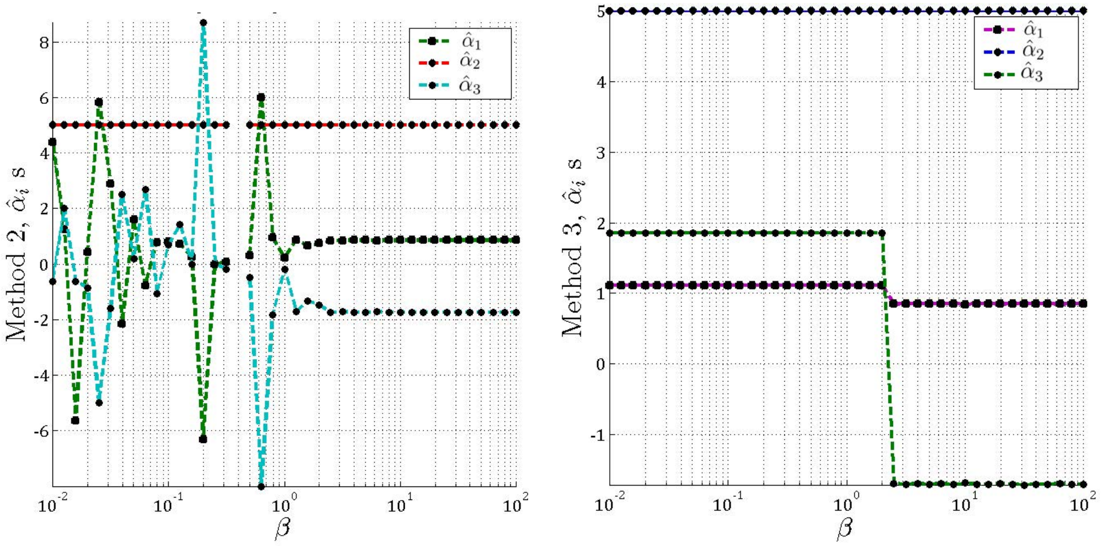
Figure 3.3: Estimating coefficients using Methods 2 and 3 on artificial data with 2 inputs of support $\{-5, ..., 5\}$. Here $[\alpha_1 = 1; \alpha_2 = 2]$, $|\mathcal{Y}| = 31$ and $|\mathcal{Z}| = 5$.

A special case in this setup is when *some* of the $\hat{\alpha}_i$'s are rounded to 0's which seems to suggest that the corresponding X_i's do not contribute to the function Y. For instance, in the two input case, if one of $\hat{\alpha}_1$ or $\hat{\alpha}_2$ is 0, then we could be tempted to conclude that only one of X_1 or X_2 dominates in the relationship with Y, thus not making it a function at all. However, it could also happen that these variables are related in a more complex way and our algorithm fails to identify it. It is difficult to make a judgment either way with our method in such cases and it is best to pursue other approaches to analyze those cases where some variables seem to dominate the rest. On the other hand, in the normal scenarios where we obtain all non zero $\hat{\alpha}_i$'s, we can conclude definitively that a linear functional relationship exists between X_i's and Y using the method proposed in this project.

3.5 Test on Artificial Data

Consider each input X_i and also the noise W to be randomly distributed with support $\mathcal{X}_i = \{-M, ..., 0, ..., M\}$, where $M \in \mathbb{Z}$. Assume all X_i's and W to be mutually independent. Let the coefficients $\alpha_i \in \mathbb{Z}$ be randomly selected from the interval $\{-M, ..., 0, ..., M\}$ as well. Then Y is a random variable defined as before according to Equation 3.2. The algorithms outlined in the previous section for estimating linear functional relationship between input and observed random variables are then applied on this artificially generated data. Here we set the cardinality $|\mathcal{Z}|$ of the compressed variable required for the iterative IB algorithm, to be much smaller than the true cardinality $|\mathcal{Y}|$ of Y.

For example, consider the case when we have two inputs ($n = 2$) with $M = 5$ and the actual coefficients $\alpha_1 = 1$ and $\alpha_2 = 2$. Then the support of X_1 and X_2 becomes $\{-5, ..., 0, ..., 5\}$. In this scenario the true cardinality $|\mathcal{Y}| = 31$. We then run our algorithm for estimating the coefficients



(a) Estimating $\hat{\alpha}_1, \hat{\alpha}_2$ and $\hat{\alpha}_3$ using Method 2 when $\alpha_1 = 1, \alpha_2 = 5, \alpha_3 = -2$.

(b) Estimating $\hat{\alpha}_1, \hat{\alpha}_2$ and $\hat{\alpha}_3$ using Method 3 when $\alpha_1 = 1, \alpha_2 = 5, \alpha_3 = -2$.

Figure 3.4: Estimating coefficients using Methods 2 and 3 on artificial data with 3 inputs of support $\{-5, ..., 5\}$. Here $[\alpha_1 = 1; \alpha_2 = 5; \alpha_3 = -2]$, $|\mathcal{Y}| = 81$ and $|\mathcal{Z}| = 10$.

by setting $|\mathcal{Z}| = 5$. As Methods 2 and 3 are more computationally efficient than Method 1, we focus on these two methods in the rest of the report. Figure 3.3 plots the estimated normalized coefficients $\hat{\alpha}_1$ and $\hat{\alpha}_2$ for two inputs using both Method 2 and Method 3 at different values of the trade-off parameter β. Similar plots are also depicted in Figure 3.4 for three inputs with the same support and the actual coefficients set as $\alpha_1 = 1$, $\alpha_2 = 5$ and $\alpha_3 = -2$. In this case, the true cardinality $|\mathcal{Y}| = 81$ and the cardinality set in the IB algorithm $|\mathcal{Z}| = 10$.

From these plots (Figures 3.3 and 3.4) we observe that at relatively small β values, the estimated $\hat{\alpha}_i$'s converge to the actual coefficients α_i's even when the cardinality is set such that $|\mathcal{Z}| \ll |\mathcal{Y}|$. Moreover, Method 2 and Method 3 converge to the actual coefficients in different ways. Method 2 fluctuates greatly at very small β values before it converges to the actual coefficient values. On the other hand, Method 3 is stable at small β values and at a particular β value, it converges to the actual coefficients.

Therefore, this approach is able to recover a compact function to explain the linear function dependence of Y on X_i's for this artificial data. Next, we need to try and see if we can obtain some meaningful results on real experimental data. While working with experimental data, we assume that the data follows a linear model, and look at the confidence parameter Θ obtained by applying the algorithm proposed in this project. By setting a suitable threshold θ, we can apply this algorithm even on real data, which is the focus of the next chapter.

Chapter 4

Experimental Results

In this project, the proposed algorithm is tested on experimental data obtained from a brain interface experiment performed on a monkey. The data is courtesy of Prof. J. M. Carmena, University of California, Berkeley. The next subsection briefly outlines the experimental setup used for obtaining the neural data.

4.1 Data Description

In this experiment, a monkey performs a behavioral task for a duration of about 15 minutes (1080353 milliseconds, to be precise) while the resulting voltage traces are measured simultaneously across 64 sites in the brain using a multi-electrode array. The task consists of holding the hand in the center of a switchboard with eight light bulbs arranged in a circle around the center. When one of the eight light bulbs lit up, the monkey has to move the hand to the light bulb, and then back to the center of the board. This is often referred to as a reaching task. During the entire 15 minutes, the voltage traces thus measured are not fully stored and instead, we have access to a low-pass filtered version of the voltage of each of the 64 electrodes, filtered up to 500Hz. These signals are called *LFP* signals, for local field potentials. Subsequently, an intricate algorithm searches through all 64 electrodes to identify spike times of individual neurons. This spike sorting algorithm located 184 individual neurons, and for each neuron, spike times are recorded in seconds.

Additionally, we also have access to the precise timings of occurrences of all the corresponding *actions*. An action can be a light bulb that is lighting up, or the moment when the hand leaves the center of the switchboard, or the moment when the hand reaches the correct light bulb, or the moment when the hand is back in the center. Over the course of the entire experiment, the reaching task is performed in different directions: $0^\circ, 45^\circ, 90^\circ, 180^\circ$, etc. Moreover, each experiment is also repeated several times; for example, the 180° experiment is repeated 36 times at different starting points in the entire duration of 15 minutes.

The functional identification algorithm outlined in Section 3.4.4 is applied on this dataset to infer some structure present in the data. Before doing that, we first need to decide how to estimate the required probability distributions $p(x_1, ..., x_n, y)$ from the data. Additionally, we also need to decide a way to deal with the temporal aspect of the neural spike trains from different neurons. The following sections discuss these issues.

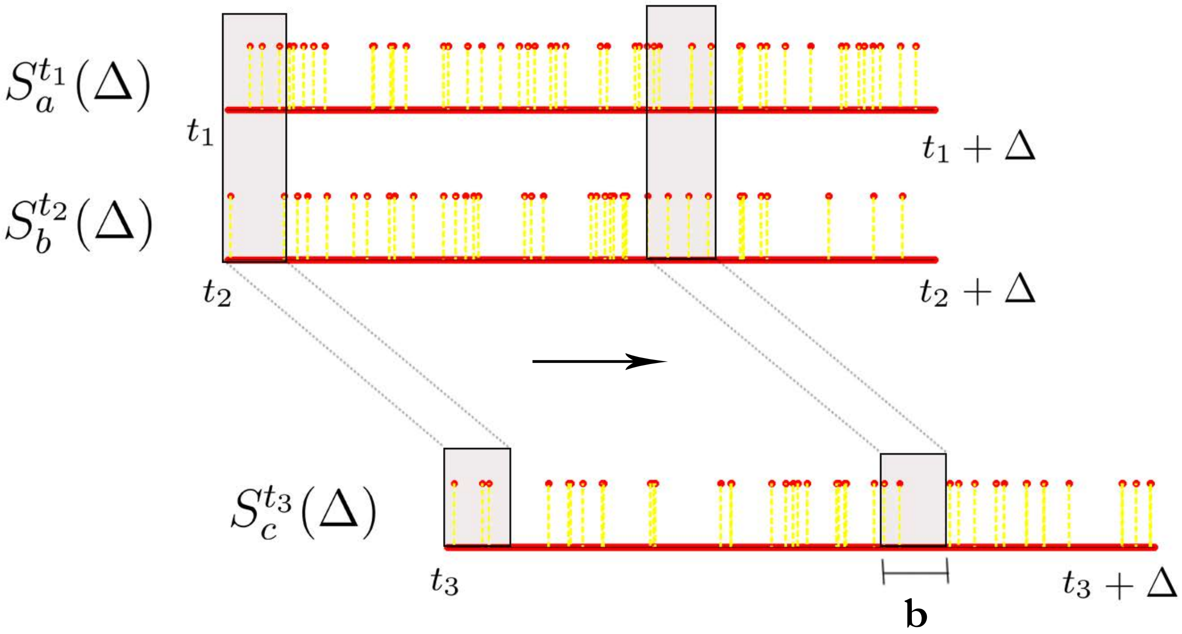


Figure 4.1: Estimating joint histograms from spike trains, where we consider overlapping bins using a sliding window.

4.2 Setup

Let $S_i^t(\Delta)$ denote the spike train of neuron i starting from time t and lasting for Δ milliseconds, i.e, we are looking at the neural response of neuron with id i from time t ms to $(t + \Delta)$ ms with a millisecond precision. $S_i^t(\Delta)$ can be seen as a vector of length Δ comprising of 0's and 1's where 0 represents no spike and 1 represents a spike. The number of spikes we have in this time window is denoted by $|S_i^t(\Delta)|$.

Then a random variable denoted by $R_i^t(\Delta, b)$, is estimated from $S_i^t(\Delta)$ in the following way (b here, is a binning parameter): Compute the histogram of the realizations $r_i^t(\Delta, b)$ given by:

$$r_i^t(\Delta, b) = |S_i^{t'}(b)|, \forall t' \in \{t, ..., \Delta - b\} \tag{4.1}$$

and normalize this histogram to get the probability distribution $p(R_i^t(\Delta, b) = r_i^t(\Delta, b))$. In other words, this procedure maintains a sliding window of length b ms starting from the beginning of the spike train $S_i^t(\Delta)$, counts the number of spikes in this window while stepping this window to the right until we reach the end of the spike train $S_i^t(\Delta)$ and normalizes this binned histogram to obtain the probability distribution of the random variable $R_i^t(\Delta, b)$ (Figure 4.1). Accordingly, the support of this random variable is the number of spikes observed in any contiguous segment of length b ms of the spike train $S_i^t(\Delta)$.

The above procedure can be extended for estimating the joint probability distribution from multiple spike trains. In this project, we restrict ourselves to the case where given two spike train segments $S_a^{t_1}(\Delta)$ and $S_b^{t_2}(\Delta)$, we want to know if there exists a linear functional relationship between these two spike train segments in order to explain a third spike train segment $S_c^{t_3}(\Delta)$. Therefore, we need to estimate the joint probability distribution of the three random variables $R_a^{t_1}(\Delta, b), R_a^{t_1}(\Delta, b)$ and $R_c^{t_1}(\Delta, b)$ associated with these three spike train segments. To do this we ensure that the sliding window is appropriately aligned across all these three spike trains while obtaining the joint histogram. This procedure is illustrated in Figure 4.1. Once we have this joint histogram we can use the procedure outlined earlier in this chapter to estimate α_1 and α_2 such that the below functional relationship holds:

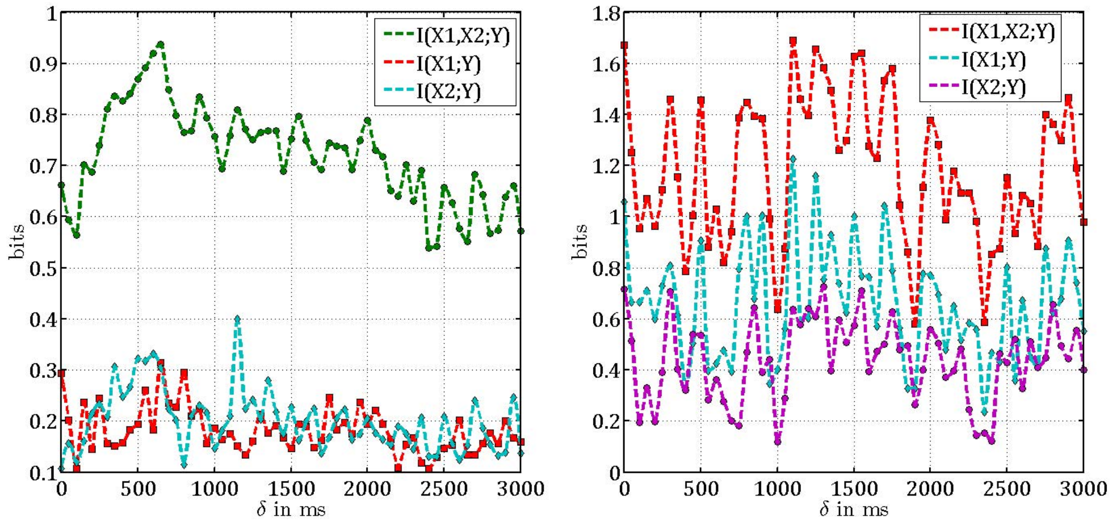
$$\alpha_1 R_a^{t_1}(\Delta, b) + \alpha_2 R_b^{t_2}(\Delta, b) = R_c^{t_3}(\Delta, b) \tag{4.2}$$

$R_a^{t_1}(\Delta, b)$ and $R_b^{t_2}(\Delta, b)$ are the input random variables (X_1, X_2, as in the notation used in Section 3.1) and $R_c^{t_3}(\Delta, b)$ is the output random variable (Y). It should be noted that we should expect to be able to identify such a relationship only occasionally from the data, as neurons generally do not behave in a predictable and deterministic way. We need to perform an exhaustive search to find the *right* neurons (a, b, c), the time frames (t_1, t_2, t_3) when these neurons have interesting behaviors and also the suitable parameters Δ and b for which such relationships exist and can be identified by our method. Accordingly, in order to reduce the search space, we assume that the two inputs neuron spike trains are aligned and start at the same time, i.e., $t_1 = t_2$. We then try different delays δ in the output spike neuron spike, i.e., $t_3 > t_1 = t_2$ and $\delta = t_3 - t_1$.

4.3 Example

Consider a concrete example where we observe neurons 141, 63 and 139 in an experiment involving a manual reaching task at a 180° angle lasting for about 3 seconds (let us call this experiment E_{180}). Overall, there are 36 such trial of this 180° experiment over the course of the entire 15 minute experimental time. One particular trial of this experiment E_{180} starts at time 40718 ms and ends at 43457 ms. The various times when events occur during this trail of E_{180} are listed below:

Time (ms)	Event
40718	center appears
40719	manual target 180 degrees
⋮	⋮
41080	hand enters center
⋮	⋮
42130	go cue
42362	hand leaves center
⋮	⋮
42705	hand enters target
⋮	⋮
43206	force code 51
43207	reward on
43457	successful trial end



(a) $\Delta = 2000ms, b = 50ms$

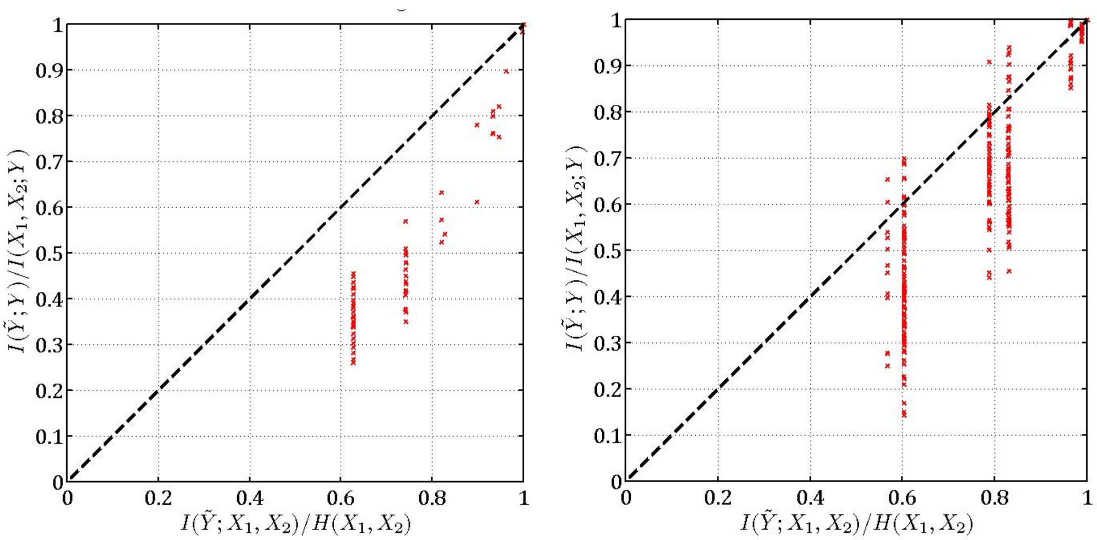
(b) $\Delta = 200ms, b = 10ms$

Figure 4.2: Plotting $I(X_1, X_2; Y(\delta))$, $I(X_1; Y(\delta))$ and $I(X_2; Y(\delta))$ versus the delay δ in Y for different values of Δ and b. (Here, $X_1 \equiv R_{141}^{40718}(\Delta, b)$, $X_2 \equiv R_{63}^{40718}(\Delta, b)$ and $Y(\delta) \equiv R_{139}^{40718+\delta}(\Delta, b)$).

If we take the input neuron spike trains to start at the beginning of the experiment, we have the following spike trains $S_{141}^{40718}(\Delta)$, $S_{63}^{40718}(\Delta)$, $S_{139}^{40718+\delta}(\Delta)$ and the resulting random variables $R_{141}^{40718}(\Delta, b)$, $R_{63}^{40718}(\Delta, b)$, $R_{139}^{40718+\delta}(\Delta, b)$. We then need to choose the parameters Δ and b depending on whether we want to observe longer spike trains or shorter ones and at what resolution. It should be noted that choosing Δ too large (compared to δ) would not be such a good idea as there would be a large overlap between output spike trains which do not differ much in their delays with respect to the input neurons. For convenience let us denote $X_1 \equiv R_{141}^{40718}(\Delta, b)$, $X_2 \equiv R_{63}^{40718}(\Delta, b)$ and $Y(\delta) \equiv R_{139}^{40718+\delta}(\Delta, b)$.

Figure 4.2 plots the quantities $I(X_1, X_2; Y(\delta))$, $I(X_1; Y(\delta))$ and $I(X_2; Y(\delta))$ as a function of the delay δ of the spike train of neuron 139, for parameter values of $\{\Delta = 200 \text{ ms}, b = 10 \text{ ms}\}$ and $\{\Delta = 2000 \text{ ms}, b = 50 \text{ ms}\}$. From these plots we notice that $I(X_1, X_2; Y(\delta))$ is significantly larger than $I(X_1; Y(\delta))$ and $I(X_2; Y(\delta))$ meaning that the two input neurons contribute more together towards the information with respect to the output neuron than separately on their own. Also, we observe that by taking longer Δ, we see more structure in the plots than by taking shorter Δ. The information plots are more smoother because as mentioned before, due to the large value of Δ, the output neurons which do not differ much in delay look similar and so $I(X_1, X_2; Y(\delta))$ and $I(X_1, X_2; Y(\delta + 1))$ are close for large Δ values. We see that there is a peak in $I(X_1, X_2; Y)$ around the time when the monkey starts to perform the moving action at a delay of 0.5 seconds when we choose $\{\Delta = 2000 \text{ ms}, b = 50 \text{ ms}\}$, and this is not so obvious in the other configuration $\{\Delta = 200 \text{ ms}, b = 10 \text{ ms}\}$.

However, this structure in the information plots does not necessarily lead to identifying functional relationships between the involved variables as can be seen from Figure 4.3. This figure plots $I(\tilde{Y}(\delta); Y(\delta))/I(X1, X2; Y(\delta))$ versus $(\tilde{Y}(\delta); X_1, X_2)/H(X_1, X_2)$ for both these parameter configurations. $\tilde{Y}(\delta)$ is defined in a similar way as before using the coefficients $\hat{\alpha}_1(\delta)$ and $\hat{\alpha}_2(\delta)$



(a) $\Delta = 2000$ ms , $b = 50$ ms. No functions identified at a threshold of $\theta = 1$.

(b) $\Delta = 200$ ms , $b = 10$ ms. A few functions identified at a threshold of $\theta = 1$.

Figure 4.3: Plotting $I(\tilde{Y}(\delta);X)/I(X1,X2;Y(\delta))$ versus $I(\tilde{Y}(\delta);X)/H(X_1,X_2)$. Points which lie above the $45°(\Theta > 1)$ are candidates for functions.

estimated from $Y(\delta)$ with X_1 and X_2 as the inputs. In this case, $\tilde{Y}(\delta)$ becomes:

$$\tilde{Y}(\delta) = \hat{\alpha}_1(\delta)R_{141}^{40718}(\Delta,b) + \hat{\alpha}_2(\delta)R_{63}^{40718}(\Delta,b) \tag{4.3}$$

From this plot we see that all the points for $\{\Delta = 2000$ ms , $b = 50$ ms$\}$, lie below the 45° line indicating the absence of any functions. On the other hand, for the parameter configuration of $\{\Delta = 200$ ms , $b = 10$ ms$\}$, a few points lie above the 45° line (i.e., $\Theta > 1$ for these points) which are candidates for functional relationships between the input neurons neurons and the appropriately delayed output neuron. Larger Δ values seem to *average* the behavior of the neurons over a big window, while smaller Δ values seem to capture local fluctuations occurring in the neural responses as the experiment progresses.

As described earlier in Section 4.1, we have access to the neural activity of 184 neurons for a duration of about 15 minutes. Performing an exhaustive search across all neurons triplets $\{a, b, c\}$ at all time frames $\{t_1, t_2, t_3\}$ and at different resolutions $\{\Delta, b\}$ for obtaining functional relationships would be an extremely task difficult due to the sheer magnitude of the possibilities. Here we focus on different trials of experiments E_{180} and E_{90} and look at neurons which have *reasonable* spiking activities for the duration of these different trials to cut down on the number of possibilities. We set $t_1 = t_2$ at the start of different events in the experiment and set $t_3 = t_1 + \delta$ with the resolution parameter set as $\Delta = 200$ ms and $b = 10$ ms. Also, we limit the delay parameter δ to be less than 1 seconds as it is hard to justify the occurrences of functional relationships between neurons after a delay of more than 1 second, given that the whole duration of the experiment is less than 3 seconds.

The next section discusses a few case studies obtained by running our algorithm on different scenarios and searching for functions between neurons.

4.4 Case Study I: E_{180}

4.4.1 A Particular Trial

Consider a particular trial of experiment E_{180} which lasts from $t = 40718$ ms to $t = 43457$ ms. The two input neuron spike trains are set to start at the advent of different events (like: center appears, hand enters center, go cue, hand enters target, etc.) and the output neuron spike train is set to start at different delay values with respect to the input neurons with a maximum delay of 800 ms. We set the parameters for obtaining the joint histograms as $\Delta = 200$ ms, $b = 10$ ms and the threshold θ is set to 1.25. Below are a few functions obtained at different events of this experimental trial. As it turned out, *all* the functions we found were direct, unweighted sums. The functions are sorted in the descending order of the confidence parameter Θ.

Center appears: $t = 40719$ **ms**

- $R_{141}^{40719} + R_{63}^{40719} = R_{114}^{40719+440}$ with $\Theta = 1.477$.
- $R_{139}^{40719} + R_{141}^{40719} = R_{28}^{40719+750}$ with $\Theta = 1.338$.
- $R_{141}^{40719} + R_{63}^{40719} = R_{98}^{40719+270}$ with $\Theta = 1.272$.
- $R_{63}^{40719} + R_{28}^{40719} = R_{98}^{40719+230}$ with $\Theta = 1.256$.

Hand enters center: $t = 41080$ **ms**

- $R_{139}^{41080} + R_{114}^{41080} = R_{98}^{41080+250}$ with $\Theta = 1.456$.
- $R_{28}^{41080} + R_{114}^{41080} = R_{98}^{41080+370}$ with $\Theta = 1.321$. $\leftarrow$
- $R_{139}^{41080} + R_{98}^{41080} = R_{28}^{41080+310}$ with $\Theta = 1.278$.
- $R_{139}^{41080} + R_{28}^{41080} = R_{114}^{41080+750}$ with $\Theta = 1.273$.
- $R_{114}^{41080} + R_{28}^{41080} = R_{63}^{41080+450}$ with $\Theta = 1.267$.

Go cue: $t = 42130$ **ms**

- $R_{141}^{42130} + R_{98}^{42130} = R_{114}^{42130+700}$ with $\Theta = 1.261$.
- $R_{139}^{42130} + R_{98}^{42130} = R_{114}^{42130+510}$ with $\Theta = 1.256$.

Hand enters target: $t = 42707$ **ms**

- $R_{98}^{42705} + R_{114}^{42705} = R_{63}^{42705+670}$ with $\Theta = 1.493$.
- $R_{114}^{42705} + R_{28}^{42705} = R_{141}^{42705+360}$ with $\Theta = 1.356$.
- $R_{114}^{42705} + R_{63}^{42705} = R_{139}^{42705+170}$ with $\Theta = 1.332$.
- $R_{141}^{42705} + R_{63}^{42705} = R_{28}^{42705+540}$ with $\Theta = 1.308$.
- $R_{63}^{42705} + 2R_{114}^{42705} = R_{28}^{42705+570}$ with $\Theta = 1.303$.
- $R_{63}^{42705} + R_{114}^{42705} = R_{98}^{42705+110}$ with $\Theta = 1.271$.

Reward on: $t = 43206$ **ms**

- $R_{141}^{43206} + R_{114}^{43206} = R_{28}^{43206+660}$ with $\Theta = 1.306$.
- $R_{139}^{43206} + R_{114}^{43206} = R_{28}^{43206+130}$ with $\Theta = 1.281$.
- $R_{114}^{43206} + R_{141}^{43206} = R_{139}^{43206+370}$ with $\Theta = 1.231$.

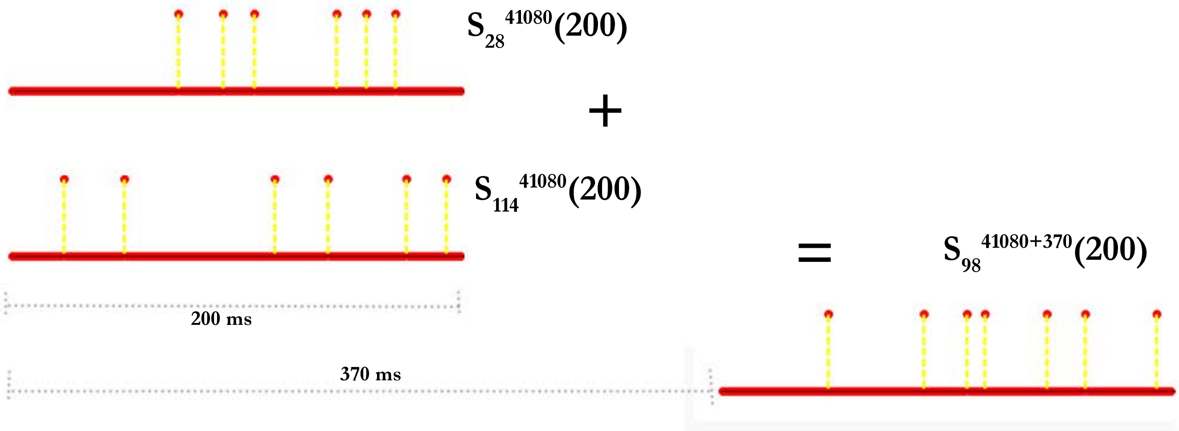


Figure 4.4: The *sum* of neuron 28 and neuron 114 at time 41080 (which corresponds to the action of the hand entering center) is *equal* to neuron 98 after a delay of 370 ms.

We observe that most of the normalized linear coefficients estimated by our method on this dataset are equal to 1, with rare occurrences of 2, and no values greater than 2. This can be explained by looking at the support of the different random variables estimated from the spike trains. These supports are compact and concentrated in a particular range for most of the spike trains. Therefore, we do not observe higher normalized coefficient estimates such as 3 or 4.

In order to better validate the results obtained using our proposed algorithm, we need to verify whether the functional relationships listed above are replicated in different trials of the same experiment. The next section goes through a particular case study where the functional relationships between a particular triplet of neurons are analyzed over different trails of the same experiment.

4.4.2 Behavior across trials

Consider the following three neurons: 28, 114 and 98, with neurons 28 and 114 as the input neurons and neuron 98 as the output neuron. We want to observe the behavior of these three neurons across all 36 trials of the E_{180} experiments. Out of these 36 trials, only 26 are as successful and the rest are considered unsuccessful as the monkey's hand leaves the center before the go cue is given. The functional relationships obtained from one of the 26 successful trials that lasts from $t = 40718$ ms to $t = 43457$ ms were listed in the previous subsection. Let us look at one particular function which is marked in the previous list:

$$R_{28}^{41080} + R_{114}^{41080} = R_{98}^{41080+370} \tag{4.4}$$

The functional relationship in this equation implies that the *sum* of neuron 28 and neuron 114 at time 41080 (which corresponds to the action of the hand entering center) is *equal* to neuron 98 after a delay of 370 ms (Figure 4.4). In order to check if similar relationship exists between these neurons at identical stages of different trials of the same E_{180} experiment, we plot $I(\tilde{Y}^k(370); Y^k)/I(X_1^k, X_2^k; Y^k(370))$ versus $(I(\tilde{Y}^k(370); X_1^k, X_2^k)/H(X_1^k, X_2^k))$ as before, for all these scenarios (Figure 4.5). Here $X_1^k \equiv R_{28}^t(200, 10)$, $X_2^k \equiv R_{114}^t(200, 10)$ and $Y^k \equiv R_{98}^{t+370}(200, 10)$, where t is the time where the hand enters the center for trial k.

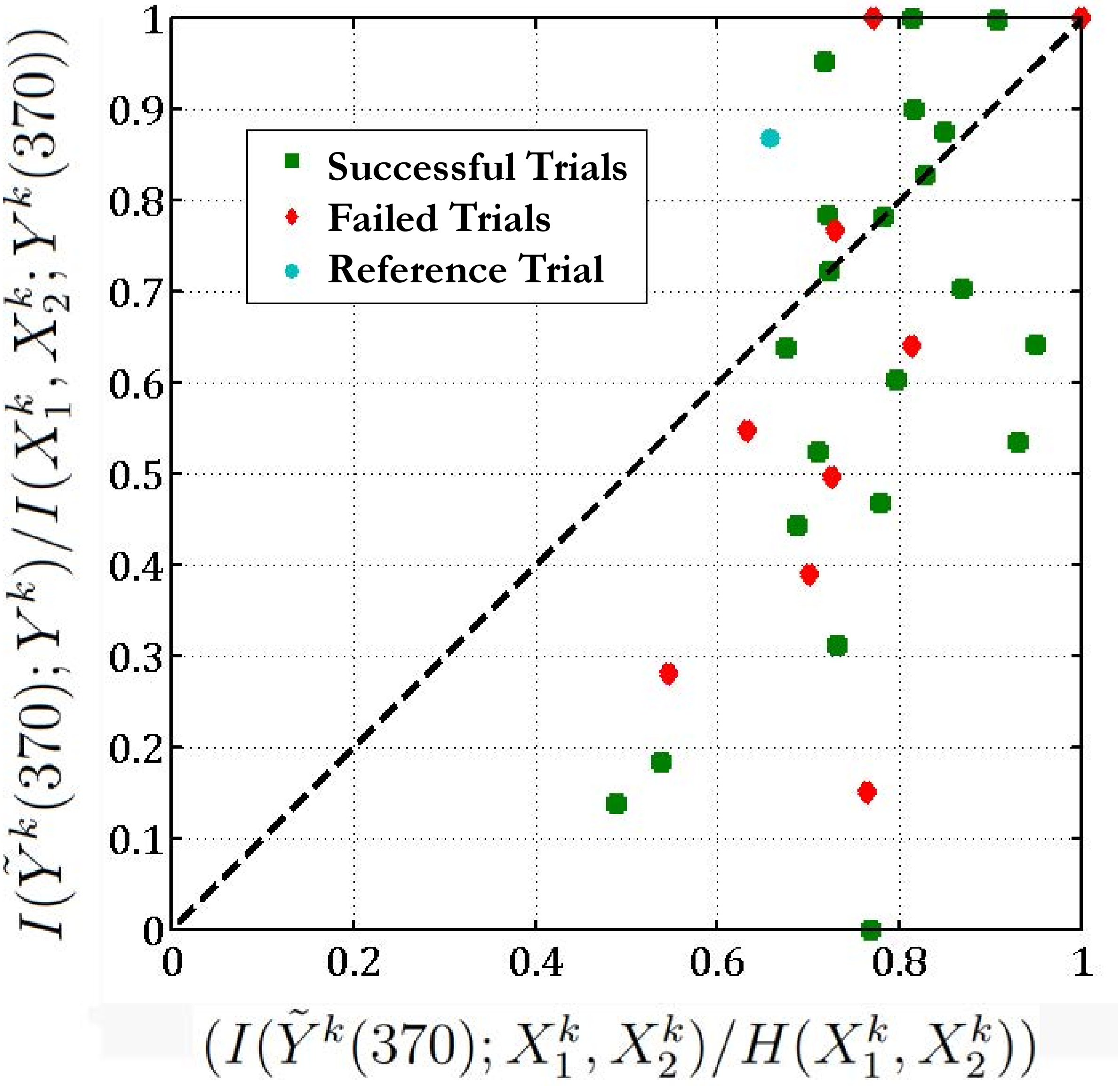


Figure 4.5: We apply the function (Equation 4.4) obtained between neurons 28, 114 and 98 in the reference trial (t = 40718) across all 36 trials and plot $(I(\tilde{Y}^k(370); X_1^k, X_2^k)/H(X_1^k, X_2^k))$ versus $I(\tilde{Y}^k(370); Y^k)/I(X_1^k, X_2^k; Y^k(370))$ for these different trials of E_{180}. Here $X_1^k \equiv R_{28}^t(200, 10)$, $X_2^k \equiv R_{114}^t(200, 10)$ and $Y^k \equiv R_{98}^{t+370}(200, 10)$, where t is the time where the hand enters the center for trial k. We observe that 11 out of the 26 successful trials satisfy the function obtained from the reference trial and most of the unsuccessful trials do not follow this relationship.

Figure 4.5 implies that in 13 out of the 36 different trials of E_{180}, the *sum* of neurons 28 and 114 is equal to the response of neuron 98 as the points corresponding to these trials lie above the 45° line (here we set the threshold $\theta = 1$). In one-third of the experimental trials, the functional relationship given in Equation 4.4 holds. Moreover, if we exclude the unsuccessful trials, then 10 out of 26 of the successful trials follow the above functional relationship between neurons 28, 114 and 98. Most of the unsuccessful trials lie below the 45° line ($\Theta < 1$) which indicates that these neurons behave in a different manner during an unsuccessful trial. The below table lists the Θ values corresponding to all the trials (we exclude 5 trials which give Θ of the form $0/0$ and $1/0$):

Successful Trials		Unsuccessful Trials	
$\Theta \geq 1$	$\Theta < 1$	$\Theta \geq 1$	$\Theta < 1$
1.331	0.947	1.000	0.197
1.321	0.811	1.052	0.513
1.229	0.758	1.293	0.555
1.114	0.739		0.685
1.109	0.676		0.787
1.102	0.647		0.865
1.091	0.601		
1.000	0.574		
1.000	0.427		
1.000	0.341		
	0.283		
	0.000		

For successful trials, we can consider the cases when $\Theta \geq \theta$ as *true positives* (TP) and the cases when $\Theta < \theta$ as *false negatives* (FN). Similarly for the unsuccessful trials, we can consider the cases when $\Theta \geq \theta$ as *false positives* (FP) and the case when $\Theta < \theta$ as *true negatives* (TN). Then, at this value of the threshold $\theta = 1$ we can compute the following quantities:

- True Positive Rate (TPR) = TP/(TP+FP) = 76.92%
- True Negative Rate (TNR) = TN/((TN+FN) = 33.33%
- Sensitivity = TP/(TP+FN) = 45.45%
- Specificity = TN/(TN+FP) = 66.67%

We achieve high TPR but not such a high TNR as there are many false negatives. It should be noted that these values depend on the value of the threshold θ.

4.5 Case Study II: E_{90}

Below are the functions identified for a particular trial of experiment E_{90} starting from $t = 80599$ ms to $t = 83242$ ms, at the advent of different events. All these results are obtained by setting $\Delta = 200$ ms, $b = 10$ ms and setting the threshold θ to decide whether we have a function or not to be equal to 1.25. We again found mostly direct, unweighted sums, like in the previous case except for two occurrences of weighted sums.

4.5.1 A particular trial

Center appears: $t = 80600$ **ms**

- $2R_{65}^{80600} + R_{161}^{80600} = R_{141}^{80600+590}$ with $\Theta = 1.408$. ←
- $R_{65}^{80600} + R_{161}^{80600} = R_{163}^{80600+320}$ with $\Theta = 1.387$.
- $R_{63}^{80600} + R_{161}^{80600} = R_{65}^{80600+50}$ with $\Theta = 1.341$.
- $R_{65}^{80600} + R_{163}^{80600} = R_{161}^{80600+290}$ with $\Theta = 1.323$.
- $2R_{65}^{80600} + R_{161}^{80600} = R_{139}^{80600+620}$ with $\Theta = 1.293$.

Hand enters center: $t = 80912$ **ms**

- $R_{65}^{80912} + R_{139}^{80912} = R_{163}^{80912+130}$ with $\Theta = 1.479$.
- $R_{65}^{80912} + R_{63}^{80912} = R_{161}^{80912+300}$ with $\Theta = 1.356$.
- $R_{65}^{80912} + R_{139}^{80912} = R_{161}^{80912+290}$ with $\Theta = 1.353$.
- $R_{163}^{80912} + R_{63}^{80912} = R_{161}^{80912+350}$ with $\Theta = 1.321$.
- $R_{63}^{80912} + R_{161}^{80912} = R_{65}^{80912+400}$ with $\Theta = 1.312$.

Hand enters target: $t = 82491$ **ms**

- $R_{65}^{82491} + R_{63}^{82491} = R_{163}^{82491+600}$ with $\Theta = 1.319$.
- $R_{65}^{82491} + R_{163}^{82491} = R_{141}^{82491+490}$ with $\Theta = 1.286$.
- $R_{161}^{82491} + R_{139}^{82491} = R_{65}^{82491+350}$ with $\Theta = 1.261$.
- $R_{139}^{82491} + R_{141}^{82491} = R_{161}^{82491+130}$ with $\Theta = 1.256$.
- $R_{63}^{82491} + R_{139}^{82491} = R_{65}^{82491+720}$ with $\Theta = 1.254$.

Reward on: $t = 82991$ **ms**

- $R_{63}^{82991} + R_{141}^{82991} = R_{65}^{82991+340}$ with $\Theta = 1.524$.
- $R_{65}^{82991} + R_{63}^{82991} = R_{163}^{82991+380}$ with $\Theta = 1.304$.
- $R_{163}^{82991} + R_{63}^{82991} = R_{161}^{82991+480}$ with $\Theta = 1.251$.

4.5.2 Behavior across trials

Results similar to the E_{180} experiments are obtained even for these E_{90} experiments if we analyze the functional behavior of a particular neuron triplet across different trials. Consider the function marked in the above list where one of the coefficients obtained is 2. By applying this function to all the 76 E_{90} experiments (out of which only 26 are successful) we have the following results with the threshold $\theta = 0.9$:

- True Positive Rate (TPR) = 38.46%
- True Negative Rate (TNR) = 75%
- Sensitivity = 76.92%
- Specificity = 36%

These numbers seem to indicate that successful trials and unsuccessful trials result in different relationships between the spiking patterns of the neurons.

Chapter 5

Conclusion

This project explores the applicability of the Information Bottleneck (IB) method in the context of neuroscience. While most direct practical applications of the IB method are in the domain of supervised and unsupervised clustering, we use the IB method in an entirely different way for identifying compact linear functional relationships between different random variables. This project tries to answer the following questions: When can we say that a functional relationship exists between random variables? How can we estimate these coefficients that explain linear dependencies between random variables? How reliable are these estimates? This approach is then tested on artificial data to investigate the performance of the proposed algorithm. We then also run it on experimental data involving neural activity of several neurons recorded during a brain-interface experiment on a monkey while it performs some behavioral tasks. As the huge amount of possibilities prevented us from performing exhaustive simulations on the data, we confined ourselves to run this algorithm on a small subset of the data. In particular, we were able to identify a few neurons which seem to exhibit linear relationships towards other neurons. Additionally, we also investigated if the relationships obtained from a particular trial of an experiment are replicated in other trials of the same experiment. It was observed that in one-third of the trails, the relationships are consistent. Finally, we also explored approaches to cluster neurons using the IB method, based on their neural responses at different stages of an experiment. Some directions for future work could be as follows:

- Extend the algorithm for estimating coefficients with more than one output random variable. That would lead to solving the problem of the following structure:

$$\sum_{i=1}^{n} \alpha_i X_i = \sum_{i=1}^{m} \beta_i Y_i \tag{5.1}$$

- In this project, while applying our algorithm on experimental data, we confined ourselves to just two inputs (neurons). It would be interesting to identify linear relationships between n input neurons and m output neurons.
- To reduce the search space for identifying the function, we assumed that the two input neuron spike trains are aligned. But this need not be the case. Additional experiments could be performed to identify functions of this nature as well.
- Analysis of the heuristics employed in this project (method 2 and method 3) for estimating the coefficients.

Bibliography

[1] W. Bialek, R.Rd.R. van Steveninck, and N. Tishby. Efficient representation as a design principle for neural coding and computation. pages 659 --663, July 2006.

[2] A Borst and F E Theunissen. Information theory and neural coding. *Nature neuroscience*, 2(11):947--957, November 1999. PMID: 10526332.

[3] Lars Buesing and Wolfgang Maass. Simplified rules and theoretical analysis for information bottleneck optimization and PCA with spiking neurons. In *NIPS*, 2007.

[4] Lars Buesing and Wolfgang Maass. A spiking neuron as information bottleneck. *Neural Computation*, 22(8):1961--1992, August 2010.

[5] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 99th edition, August 1991.

[6] Peter Dayan and L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press, 1st edition, December 2001.

[7] Jacob Goldberger, Hayit Greenspan, and Shiri Gordon. Unsupervised image clustering using the information bottleneck method. In *Proceedings of the 24th DAGM Symposium on Pattern Recognition*, pages 158--165, London, UK, UK, 2002. Springer-Verlag.

[8] Stefan Klampfl, Robert Legenstein, and Wolfgang Maass. Information bottleneck optimization and independent component extraction with spiking neurons. In *In Proc. of NIPS 2006, Advances in Neural Information Processing Systems*. MIT Press, 2007.

[9] Stefan Klampfl, Robert Legenstein, and Wolfgang Maass. Spiking neurons can learn to solve information bottleneck problems and extract independent components. *Neural Computation*, 21(4):911--959, April 2009.

[10] Fred Rieke, David Warland, Rob, and William Bialek. *Spikes: Exploring the Neural Code*. MIT Press, Cambridge, MA, 1st edition, 1997.

[11] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81--85, April 2009.

[12] Elad Schneidman, William Bialek, and Michael J. Berry II. An information theoretic approach to the functional classification of neurons. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, pages 197--204. MIT Press, 2002.

[13] Elad Schneidman, Noam Slonim, Naftali Tishby, Rob R. de Ruyter van Steveninck, and William Bialek. Analyzing neural codes using the information bottleneck method. 2001.

[14] N. Slonim. *The Information Bottleneck: Theory And Applications*. PhD thesis, The Hebrew University, 2003.

[15] Noam Slonim, Nir Friedman, and Naftali Tishby. Agglomerative multivariate information bottleneck. In *NIPS*, pages 929--936, 2001.

[16] Noam Slonim, Nir Friedman, and Naftali Tishby. Unsupervised document classification using sequential information maximization. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, page 129–136, New York, NY, USA, 2002. ACM.

[17] Noam Slonim, Rachel Somerville, Naftali Tishby, and Ofer Lahav. Objective classification of galaxy spectra using the information bottleneck method, 2001.

[18] Noam Slonim and Naftali Tishby. Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, pages 208--215, New York, NY, USA, 2000. ACM.

[19] Noam Slonim and Naftali Tishby. The power of word clusters for text classification. In *In 23rd European Colloquium on Information Retrieval Research*, 2001.

[20] Kelvin So, Karunesh Ganguly, Jessica Jimenez, Michael C. Gastpar, and Jose M. Carmena. Redundant information encoding in primary motor cortex during natural and prosthetic motor control. *Journal of Computational Neuroscience*, 32(3):555--561, November 2011.

[21] Kelvin So, Aaron C Koralek, Karunesh Ganguly, Michael C Gastpar, and Jose M Carmena. Assessing functional connectivity of neural ensembles using directed information. *Journal of Neural Engineering*, 9(2):026004, April 2012.

[22] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *37th Annual Allerton Conference on Communication, Control and Computing*, page 368–377, Piscataway, NJ, 1999. IEEE Press.

[23] Naftali Tishby and Noam Slonim. Data clustering by markovian relaxation and the information bottleneck method. pages 640--646. MIT Press, 2000.

[24] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1073--1080, New York, NY, USA, 2009. ACM.

Appendix A

The IB method for Clustering

In this chapter we discuss another way of applying the Information Bottleneck (IB) method in neuroscience. We attempt to perform clustering of the neurons based on their spike trains responses.

A.1 Problem Statement

The IB method discussed in Chapter 2 solves the following minimization problem:

$$\min_{\substack{p(z|x):\\ I(Z;Y)\geq\Gamma_2}} \quad I(Z;X) \tag{A.1}$$

Now in the clustering context, we set the random variable X to be the neuron id and Y to be the random variable obtained by binning the spike trains like before. $\mathcal{Z}$ then is the number of clusters we want to partition the values of X (i.e, the neurons). The IB method assigns each neuron to a cluster z so that these clusters contain as much information as possible about the neural responses Y. If we are interested in performing a *hard* clustering ($H(Z|X) = 0$), then the set of self-consistent equations derived as a solution for the IB method in Section 2.3 can be rewritten as follows:

$$p(z|x) = \begin{cases} 1 & \text{if } x \in z \\ 0 & \text{otherwise} \end{cases}$$

$$p(y|z) = \frac{1}{p(z)} \sum_{x\in z} p(x)p(y|x) \text{ and } p(z) = \sum_{x\in z} p(x)$$

In this context, the IB problem reduces to the following problem:

$$\min_{\substack{p(z|x):\\ I(Z;Y)\geq\Gamma_2}} \quad H(Z) \tag{A.2}$$

If we assign all the neurons (support of X) to the same cluster, then $H(Z) = 0$. In this case, $I(Z;Y) = H(Y) - H(Y|Z)$ also become 0 because when there is only one cluster, $H(Y|Z) = H(Y)$. Hence, there is a trade-off between $H(Z)$ and $I(Z;Y)$ depending on how we cluster the neurons. The IB method clusters these neurons in such a way that for a fixed $H(Z)$, we retain

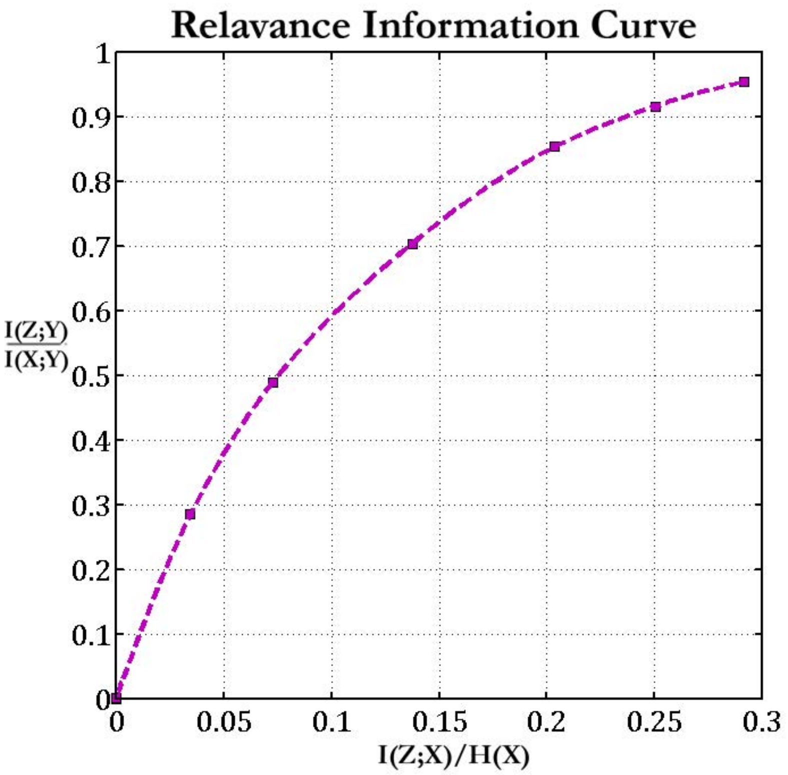


Figure A.1: The information bottleneck relevance-information curve, where X are the neuron ids and Y are the spike counts in a 10 ms window.

as much information $I(Z;Y)$ these clusters contain about the neural responses of all the neurons. The sequential algorithm outlined in Section 2.4.2 is better suited than the iterative algorithm for obtaining these hard partitioning of X values into clusters. The next section gives some results obtained by applying the IB method for clustering neurons in the above mentioned way on the brain-machine interface experiment data.

A.2 Experimental Results

Consider the neural responses of all 184 neurons from the brain-interface experiment dataset measured during a particular trial of E_{180} (say the one beginning at time $t = 40718$ ms). We compute the joint histogram of the neural responses to obtain the random variables $R_i^{40718}(200, 10), (i = 1, ..., 184)$ with parameters $\Delta = 200$ ms and $b = 10$ ms. In this example following the IB notation we have, $\mathcal{X} = \{1, ..., 184\}$, $\mathcal{Y} = \{0, 1, 2, 3\}$, i.e., these 184 neurons have 0, 1, 2 or 3 spikes in a window of 10 ms over the time interval $[40718, 40718 + 200]$. Also, we assume uniform probability distribution of the neurons: $p(x) = 1/184$.

Once we have this joint distribution $p(x, y)$, we can then apply the sequential IB algorithm with 4 clusters. Figure A.1 plots the relevance-compression curve obtained by applying the sequential IB algorithm on this $p(x, y)$ at different β values. This plot shows that by using just 4 clusters, we can achieve $I(Z;Y)$ (which is the information all the clusters contain about the neural responses) close to $I(X;Y)$ (which is the information all the neurons contain about the neural responses) at a high compression of X indicated by the ratio $I(Z;X)/H(X) \approx 0.3$. The below table indicates the cardinalities of these 4 clusters at different values of β^{-1}.

β^{-1}	$\lvert z_1 \rvert$	$\lvert z_2 \rvert$	$\lvert z_3 \rvert$	$\lvert z_4 \rvert$
1	0	0	184	0
0.1	182	0	0	2
0.01	8	23	47	106
0.001	12	75	68	29
0	27	68	42	47

As expected from the IB method, at lower values for β^{-1} (i.e., high β) we obtain clusters which are more balanced and at higher values for β^{-1} (i.e., low β) we obtain very dominant clusters. Let the cluster mapping $i \to z$ between neuron $i \in \{1, ..., 184\}$ and cluster $z \in \{z_1, z_2, z_3, z_4\}$ obtained by considering spike responses starting from time t, be denoted as C_t. In what follows, we use $\Delta = 200$ ms, $b = 10$ ms and $\beta = 0$ for obtaining cluster C_t. Once we have these clusters, we can then try to compare clusters obtained at different points of time during the experiment. This is discussed in the next section.

A.3 Comparing Clusters

Let the 184 neurons be clustered into two different clusters C_a and C_b by applying the above IB algorithm for clustering at different times a and b. The extent of similarity between these two clusters C_a and C_b can be obtained by computing the normalized mutual information adjusted to chance between these two clusters [24]. Let this quantity be denoted by $I(C_a, C_b)$. This is a normalized quantity which lies between 0 and 1. If the clusters C_a and C_b are the same, irrespective of the order of the labels, $I(C_a, C_b) = 1$.

We then compare clusters obtained at different times during the E_{180} experimental trial discussed in the previous section. If C_{40718} is the cluster obtained at the start of experiment and $C_{40718+\delta}$ is the cluster obtained at after a delay of δ ms, we look at the following quantities:

$$F_1(\delta) = I(C_{40718}, C_{40718+\delta}) \text{ and} \tag{A.3}$$

$$F_2(\delta) = I(C_t, C_{t+1}),\ \forall t \in [40718, 43457] \tag{A.4}$$

The first quantity $F_1(\delta)$ measures how the clusters evolve as the experiment progresses with respect to the initial cluster and the second quantity $F_2(\delta)$ measures how the clusters evolve with respect to the preceding cluster. Figure A.2 plots both these quantities as a function of the delay δ. By looking at the $F_1(\delta)$ trace we see that the clusters keep gradually moving away from the initial cluster and after the monkey performs an action ($\delta \approx 1200$), the clusters start to slightly resemble the initial cluster as $F_1(\delta)$ increases from this point onwards. Additionally, by looking at the $F_2(\delta)$ trace, we notice that in general, there is very high fluctuation in successive clusters as indicated by the high

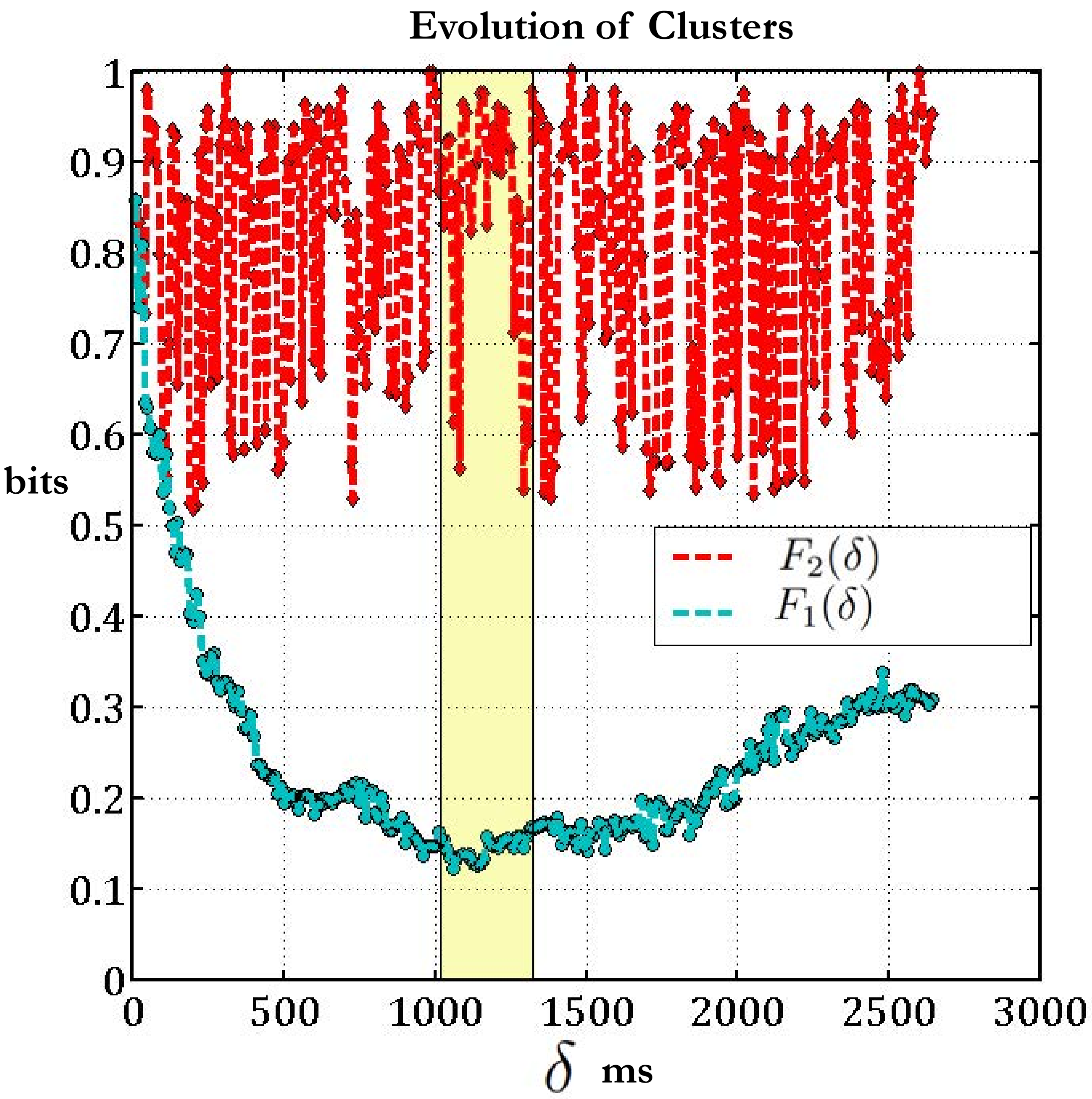


Figure A.2: $F_1(\delta)$ and $F_2(\delta)$ as a function of the delay δ. $F_1(\delta)$ starts to increase around the time ($\delta \approx 1200$ ms) $F_2(\delta)$ remains constant for a while, indicating the time when the clusters stabilize momentarily.

frequency of the signal. However, around the time $F_1(\delta)$ starts to increase, $F_2(\delta)$ does not fluctuate much indicating that the clusters do not change much during this period.