Adaptive EEG Decoding for a Brain-Computer Interface

Matthew Keeter
CISST-ERC REU at Johns Hopkins University
PI: Prof. Nitish Thakor
Mentors: Mayuresh Kothare, Geoffrey Newman
Summer 2009

Abstract

Electroencephalography (EEG) can be used as the basis for a non-invasive brain-computer interface (BCI). The BCI is controlled by μ-band brainwaves (8-15 Hz): imagined activity lowers the spectral power in this band by up to 100 dB compared to relaxation. The original BCI calculated the max or mean of the mu band to detect this difference; these may not be the best functions. This paper presents a system that generates a function to optimally detect this difference.

Mu-band data are collected during training sessions, in which a subject is either asked to relax or imagine activity. After a pair of training sessions, a linear combination of simpler functions (e.g. max, min) is found that maximizes the normalized difference between relaxed and active spectrums. This new function is used in the next training session and the process is repeated.

We compare six experimental sessions with the previous system to seven with the new system. Trials with the new system showed improved performance ($p < 10^{-8}$) at distinguishing between active and relaxed states. The subjects had similar performance before the adaptive system was turned on ($p = 0.12$), indicating that the system was responsible for the improvement, rather than differences in the subject set.

Introduction

Patients affected by paralysis or illness often have reduced control over their environment, which reduces their quality of life. Brain-computer interfaces (BCI) offer increased control, be it of a wheelchair or of a computer mouse. Such interfaces are useful because they do not rely on muscles or nerves, which may not be functional depending on the patient's situation. Electroencephalography (EEG) is a particularly interesting basis for a BCI, because it is non-invasive: it does not require surgery to set up, and can be used outside of a specialized medical facility. Furthermore, it has been shown to provide accurate control, after sufficient training [2].

To use EEG as a control system, a phenomenon known as Event-Related Desynchronization (ERD) is utilized. In a relaxed state, a subject's sensorimotor rhythms show a peak in the 8-15 Hz range, known as the Mu band. If the subject imagines activity, this peak shrinks in size, as seen in Figure 1. This difference is seen most clearly on a frequency-domain power spectrum from electrodes over the sensorimotor areas. Subjects can learn to control ERD, which allows us to use it as the basis for a one-dimensional BCI [1].

Problem Statement

The original EEG decoding pipeline uses a single data point to decide if the subject is relaxing or imagining movement: either the `mean` or the `max` of the mu-band spectrum, depending on which seems to give better results. This choice is made by the researcher during the course of the experiment. My research goal is to automate and improve that process, by developing an adaptive system. This system finds the function that best distinguishes between relaxation and imagined

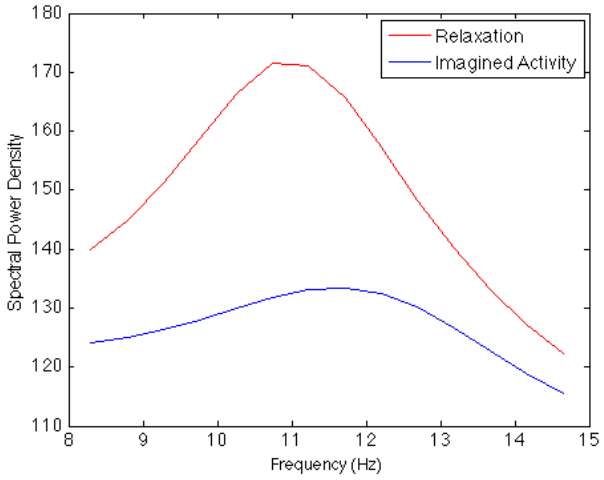

Figure 1: Spectrums showing event-related desynchronization in the Mu band

activity, based on data collected during the experiment. Such an adaptive system changes based on user characteristics, and should improve subject performance.

Description of Adaptive Algorithm

In the course of an experiment, the subject goes through a pair of training sessions separated by a brief pause, each lasting one minute. In the first training session, they are asked to relax; in the second, they are asked to imagine activity. The adaptive algorithm uses the data from training sessions to find a single function, which best distinguishes between relaxed and active μ-band spectrums.

More specifically, the system finds a function that maximizes the normalized difference between relaxed and active training sessions. Normalized difference is defined as follows: given a function f and two sets of spectrums $r_{1...t}$ and $a_{1...t}$ for relaxed and active states respectively,

$$\mathrm{ND}(f) = \frac{\mathrm{mean}(f(r_{1...t})) - \mathrm{mean}(f(a_{1...t}))}{\frac{1}{2}(\mathrm{std}(f(r_{1...t})) + \mathrm{std}(f(a_{1...t})))}$$

where $f(s_{1...t}) = [f(s_1), f(s_2), f(s_3), ..., f(s_t)]$ and std is the standard deviation.

The function to be found is a linear combination of simpler functions. Each of these simpler constituent functions takes in a spectrum $\vec{s}$ and return a single point of information, e.g. the mean value. In other words, the generated function will be of the form

$$F(\vec{s}) = c_1 f_1(\vec{s}) + c_2 f_2(\vec{s}) + ... + c_n f_n(\vec{s})$$

with $c_{1...n}$ as the coefficients which must be found. To bound the search space, we limit the coefficients such that $c_1^2 + c_2^2 + ...c_n^2 = 1$. This makes the problem equivalent to searching across the surface of an n-dimensional unit sphere, where the Cartesian coordinates of a point are equivalent to the coefficients $c_{1...n}$.

The search process is iterative. Points are generated on the surface of the n-sphere; the best point is used as the center for the next set of points, which is more tightly spaced. An illustration of the search paradigm can be seen in Figure 2.

In each iteration of the search, the points are stored as columns in an $n \times m$ array C. This means that each column of C directly corresponds to a function which is a linear combination of the constituent function set. We will refer to these functions as $p_1(\vec{s})$ through $p_m(\vec{s})$.

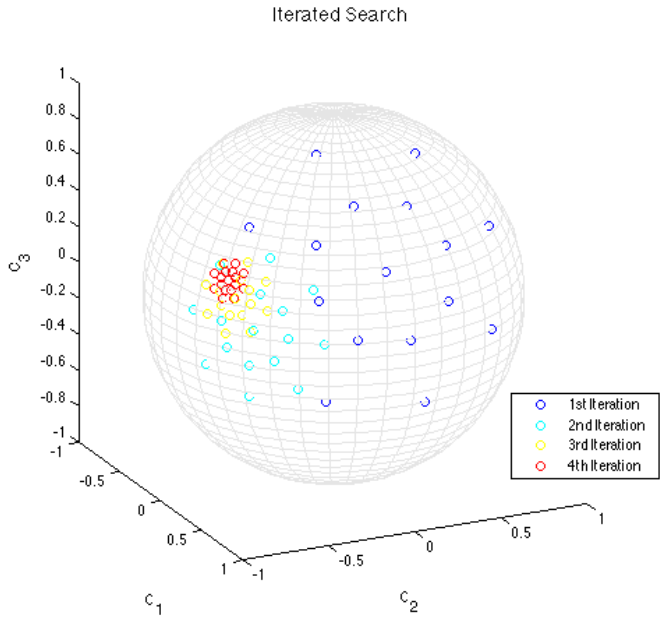

Figure 2: Example of iterated search across a three-dimensional sphere

Before the algorithm is run, data must be collected for both relaxed and active brain states. During a training session, the existing system produces a series of spectrums, which we designate here as s_1 through s_t. Each of the n constituent functions is applied to each spectrum as it is produced, and the results are saved into a matrix of the following form:

$$\begin{bmatrix} f_1(s_1) & f_2(s_1) & \dots & f_n(s_1) \\ f_1(s_2) & f_2(s_2) & \dots & f_n(s_2) \\ \dots & \dots & \dots & \dots \\ f_1(s_t) & f_2(s_t) & \dots & f_n(s_t) \end{bmatrix}$$

After a set of training sessions, two of these matrices exist, representing relaxed and active brain activity. We will refer to these matrices as V_{relaxed} and V_{active}.

These two matrices, along with C, contain all of the information required to find the best function at each iteration of the search. Due to the properties of matrix multiplication, the multiplication VC (where V is either V_{relaxed} or V_{active}) will produce a matrix of the following form:

$$\begin{bmatrix} p_1(s_1) & p_2(s_1) & \dots & p_m(s_1) \\ p_1(s_2) & p_2(s_2) & \dots & p_m(s_2) \\ \dots & \dots & \dots & \dots \\ p_1(s_t) & p_2(s_t) & \dots & p_m(s_t) \end{bmatrix}$$

where s_x is either an active or relaxed spectrum, depending on which V was used. In other words, this multiplication is equivalent to evaluating each of the possible functions applied to each of the spectrums over the entire training session.

A matrix containing the normalized difference for each function p_1 through p_m can be generated using the following equation, with element-wise arithmetic operators and column-wise `mean` and `std` functions:

$$[\mathrm{ND}(p_1), \mathrm{ND}(p_2), \mathrm{ND}(p_3), ..., \mathrm{ND}(p_m)] = \frac{\texttt{mean}(V_{\text{relaxed}}C) - \texttt{mean}(V_{\text{active}}C)}{\frac{1}{2}(\texttt{std}(V_{\text{relaxed}}C) + \texttt{std}(V_{\text{active}}C))}$$

The highest normalized difference can be found from this array; the corresponding point in C represents the best function in this iteration. The process is then repeated with a tighter net of points, as illustrated in Figure 2. Once the improvement is negligible (less that 10^{-3} pp), the adaptive system returns the final function that has been found.

Implementation and Experimental Procedure

The implemented algorithm uses data from more than one training session, weighting the more recent sessions more heavily; however, this is a fairly minor extension of the core principles. The system uses five constituent functions (i.e. $n = 5$), as any more than five functions is inoperable on our hardware. The five functions used were $\texttt{max}(\vec{s})$, $\texttt{min}(\vec{s})$, $\texttt{mean}(\vec{s})$, $\texttt{first}(\vec{s})$ (returns the power of the lowest frequency in the spectrum), and $10 * \texttt{sum}(\texttt{diff}(\vec{s}, 2))$ (related to curvature of line, multiplied by 10 to normalize it with the other functions). These functions were chosen through experimentation, though later tests indicate that the set could be optimized: in particular, `max`, `mean`, and `first` are usually highly correlated.

The adaptive system was embodied in a GUI panel, shown in Figure 3

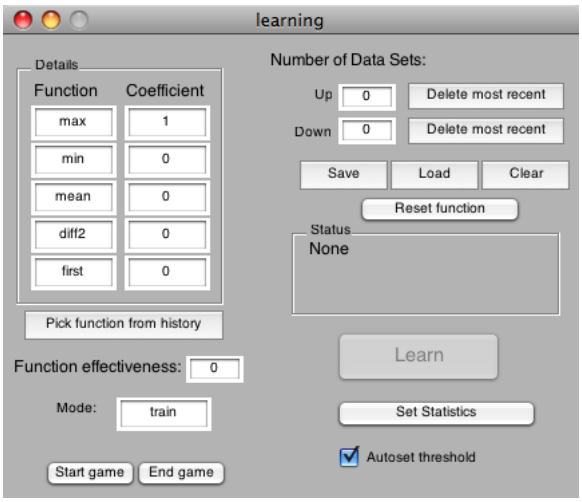

Figure 3: GUI panel to control adaptive system

The GUI panel shows the number of training data sets that are available, coefficients on the generated function, and the function's effectiveness. The effectiveness indicates how well the generated function worked on the stored data, and reflects how "confident" the system is about this function.

The adaptive system was tested 7 times, on 5 unique individuals (2 of them went through the experiment twice). Each subject went through 10+ pairs of training sessions, consisting of a minute of relaxation and a minute of imagined activity. Before the adaptive system was trained (e.g. for the initial training session), the `max` function was used. For the first three subjects, the adaptive

system was run on every training session after the first; due to issues with overfitting, later subjects began using the adaptive system after four training sessions.

Analysis of Data

To test the effectiveness of the adaptive system, we compared 7 experimental sessions using the adaptive system to 6 experimental sessions using the non-adaptive system. Within each data set, we examined normalized difference between active and relaxed training sessions. Results were split into three sets: Non-adaptive performance, adaptive performance, and "pre-adaptive" performance. The latter refers to data collected from training sessions before the adaptive system was trained, in an experimental session when it was eventually used. Theoretically, this set of data should be similar to the non-adaptive performance; we include it here to control for factors such as increased subject familiarity with the BCI.

The results are summarized in Figure 4.

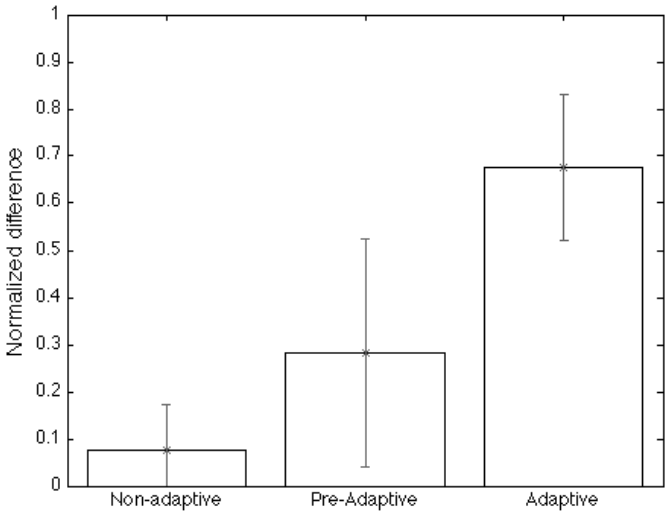

Figure 4: Summary of results. Error bars indicate 95% CI assuming normal distribution.

To analyze the data, each set of performances was treated as a normal distribution with unknown (but unique) variance. A Student's t-test (with unpooled variance) was used to test for significance. We found that the adaptive performance was significantly higher than both other sets of performance data ($p < 0.01$ in both cases), but non-adaptive and pre-adaptive performance were not significantly different ($p = 0.12$). This implies that the improved performance was due to the adaptive system, rather than any differences in the subject set.

Conclusions and Future Work

Based on these results, the adaptive system described in this paper has the potential to improve EEG decoding. It should speed up the learning process, by allowing the computer to make minor adjustments to improve performance. There are several limits to the system, which must be taken into account to make it as helpful as possible. One issue is overfitting: the first pair of training sessions for a brand new subject is likely to contain more noise then genuine information. Using the adaptive system on such data will not return a meaningful function, and feedback from that function

will further confuse the subject, making the training process more difficult. The effects of feedback on subject learning seems very powerful. In one situation, a subject was trained "backwards": when we asked them to relax, their μ-band flattened out. To avoid overfitting, the adaptive system should not be used until the subject has some degree of control. A second question is what to do with "bad trials": pairs of training sessions that show poor distinction. We are unsure as to whether they should be included in the training set, or discarded. The continued usage of the adaptive system should give us a better idea of how to address these issues.

Eventually, we hope to use this system to better understand the nature of ataxia, a neurological disorder that affects adaptation and correction to perceived error, leading to poor balance and coordination [3]. In particular, we hope to show differences in learning between subjects with ataxia and subjects without the disorder. The adaptive system provides more data for this research. For example, examining the coefficients over time could be informative: do they tend towards some final value? The data offered by the adaptive system could provide clues to the effects of ataxia on the learning process, eventually leading to better control for all of our subjects, with and without ataxia.

Acknowledgments

We would like to acknowledge the National Science Foundation for funding this program, and thank the following individuals and group for their support: Prof. Nitish Thakor and all of Thakor Lab, the EEG group (Youngseok Choi, Hyoung-Nam Kim, Mayuresh Kothare, Geoffrey Newman, Sarah Ying), Anita Sampath, Laura Libertini, Dr. Stephanie Stone.

References

[1] Birbaumer, Niels, Leonardo G. Cohen. "Brain-computer interfaces: communication and restoration of movement in paralysis." *The Journal of Physiology* 579.3(2007): 621-636. Print.

[2] Wolpaw, Jonathan and McFarland, Dennis. "Control of a two-dimensional movement signal by a noninvasive brain-computer interface." *Laboratory of Nervous System Disorders* 101(2004): 17849-17854. Print.

[3] Sarah Ying, personal communication with author