Analysis of Cortical Signals in Injured and Uninjured Rats

Kristen Drummey

Department of Physiology and Biophysics, University of Washington

Github: github.com/kldrummey/AMATH582

Abstract

Spinal cord injury (SCI) affects thousands of new patients every year, and although rehabilitative methods exist to improve functional outcomes, these methods come with significant limitations. One method, targeted-activity dependent spinal stimulation (TADSS), shows promise as a recovery method but is unusable in severely injured animals. In this study, the frequency components of neural activity in motor cortex were examined in an injured and uninjured rat. The eventual goal of this analysis is to develop a TADSS paradigm in which spinal stimulation is triggered off of cortical rather than muscle signals, and therefore usable across all injury levels.

1 Introduction and Overview

Spinal cord injury (SCI) impacts nearly 17000 new patients ever year in the US alone and causes debilitating losses of motor function that significantly impact the quality of life for affected patients. There are rehabilitative options available for patients, such as physical therapy or epidural electrical stimulation of the spine (EES), that can provide modest increases in functional recovery. However, these rehabilitative therapies come with drawbacks - for example, EES is only effective when it is turned on while the patient is performing a task[2].

Recent work has shown that delivering electrical stimulation to the spine in a targeted, activity-dependent manner can significantly improve motor functions and maintain stable recovery, even after stimulation is turned off. This targeted, activity-dependent spinal stimulation (TADSS) has been shown to be effective when triggered off of attempted muscle movements in a rat's injured forelimb. This method is effective in rats with mild to moderate injuries who retain some control over their injured limb, but it is not able to be used in severely injured animals where signals from the muscles are few and far between[3].

The goal of this project is to modify the existing TADSS paradigm to trigger spinal stimulation based off of activity in the forelimb area of motor cortex (MCtx), rather than off of muscle activity. Eight-channel electrocorticography arrays were implanted over the forelimb area of MCtx in two rats contralesional to their dominant forelimb. Recordings were taken while the rats were freely behaving in an arena. One rat had received a cervical-level 4 spinal cord contusion injury ('injured rat') while the other rat was uninjured ('uninjured rat').Frequency components were analyzed to https://www.overleaf.com/project/5e5444a51d04290001b572c1see if there was a difference in frequencies seen in the injured vs. uninjured rat, and spectrograms were attempted to show how these frequencies change over time. Eventually, these signals will serve as an activity-basis with which to trigger electrical stimulation in the spine to aid recovery in injured rats.

2 Theoretical Background

Fourier proposed that any function, f(x), can be portrayed by a series of sines and cosines, as shown below in Equation 1 and defined on the interval $x \in (-\pi, \pi]$.

$$f(x) = \frac{a_0}{2} + \sum (a_n \cos nx + b_n \sin nx) \tag{1}$$

This ability allows for functions collected outside of the frequency domain to be broken down into their component frequencies through the equation below:

$$F(k) = \frac{1}{\sqrt{2\pi}} \int e^{-ikx} f(x) dx \tag{2}$$

The beauty of the FT is that it permits any function, even those that are discontinuous, to be displayed as a series of the function's frequencies. By breaking a function down into its component frequencies, several methods can be applied to the data in order to denoise it and extract a signal of interest. For example, for a signal that is obscured by Gaussian white noise, transforming the signal and noise into the frequency domain allows for averaging of the signal. Since the mean of a Gaussian is equal to zero, averaging across enough measurements will force the noise to go to zero, allowing for a more clear picture of the signal of interest. Additionally, transforming a function into the frequency domain allows for filtering of the signal around the center frequency.

Transformed data is also able to be inverted back into its original domain, using the inverse FT shown in Equation 3. This invertibility means that functions can be filtered and processed in the frequency domain and transformed back to show how the data looks in the original domain where it was collected (e.g., the spatial or temporal domain).

$$f(x) = \frac{1}{\sqrt{2\pi}} \int e^{ikx} F(k) dk \tag{3}$$

A final key component of the FT's functionality is that it can perform transforms very quickly when used on an $O(N \log N)$ scale, rather than on a $O(N^2)$ scale. This difference in scaling allows for the operation count to increase linearly instead of exponentially, greatly decreasing the computational time and effort devoted to transforming a function.

The FFT is an excellent tool for examining broad fluctuations in neuronal signaling. Electrocorticography (EcoG) measures population activity of a group of neurons, typically through intracortically implanted wires or through subdural electrodes placed on the surface of the brain. This population activity is typically broken into frequency bands, which can be determined by using the FFT, and which provide an indication of underlying processes. For example, the delta and theta bands (0.1-3Hz and 4-7 Hz, respectively) are associated with resting states, while beta band activity (16-30Hz) and gamma band activity (31-100Hz) are associated with preparation to move and movement[4]. Analysis of underlying neuronal population dynamics can provide insight to how the cortex is functioning after spinal cord injury, and provide a readout for triggering electrical stimulation in the spine for rehabilitation.

3 Algorithm Implementation and Development

Data was acquired and read-in to the analysis script using a custom MATLAB GUI previously developed by Sung Q. Lee.

Algorithm 1: Load in data and experiment information

```
Use input to load file name for analysis.
Use loadfileemgedited to read data and convert from HEX code to structures.
Read in experiment information from .csv file.
```

After loading in data, the time period was determined and raw data was plotted. Raw data was then filtered, transformed, and the power spectrum was determined. The power spectrum determines how much power is in a given frequency for the entire course of the trial (i.e., temporal information is lost).

After filtering and transforming, data from each channel was also Gabor filtered and plotted as a spectrogram.

Algorithm 2: Filter, transform, and determine power spectrum of data

Use `bandpass` to filter out frequencies below 5 and above 250 Hz.
Use `fft` to transform filtered raw data.
Use `pwelch` to determine the power spectral density of the signal.

Algorithm 3: Gabor filter and spectrogram

Set timestep and Gabor filter width.
for $j = 1 : length(tslide)$ **do**
  Set Gabor filter
  Multiply Gabor filter by raw channel data
  Fourier transform filtered data
  Add data to spectrogram and plot using `pcolor`
end for

4 Computational Results

Figure 1A shows raw traces from two ECoG channels in an injured rat, while Figure 1B shows raw traces from two ECoG channels in an uninjured rat. There appears to be repetitive, low-oscillation noise in both samples, possibly attributable to heartbeat or respiration that is getting picked up by the ECoG array, but which should be filtered out using the bandpass filter.

Figure 2 shows the frequency components for the injured rat (Fig. 2A) and the uninjured rat (Fig. 2B). Both rats appear to have large amplitude deflections in the lower frequency ranges, suggesting the rats are at rest, although there are peaks that can be seen in the gamma range (30-100Hz). The power spectral densities show that both the injured and uninjured rat have large amounts of power in the lower frequency bands, aligning with the data seen in the FFT plots.

Ideally, this data would be analyzed using a spectrogram, so that the frequency changes in MCtx can be followed over the entire trial. Unfortunately, spectrograms created using Gabor filters with widths of 1, 10, and 25 did not appear to show anything. This could potentially because of undersampling - due to the large sizes of the datasets, they had to be significantly truncated in order to process. Acquiring a better computer with the capability of processing large amounts of data, or varying methods of truncating the data could potentially show better results in the spectrogram.

5 Summary and Conclusions

The FFT is a powerful tool for analyzing data, particularly for analyzing neural signals that have inherent oscillatory properties which provide information about an animal's current state. Although the raw data

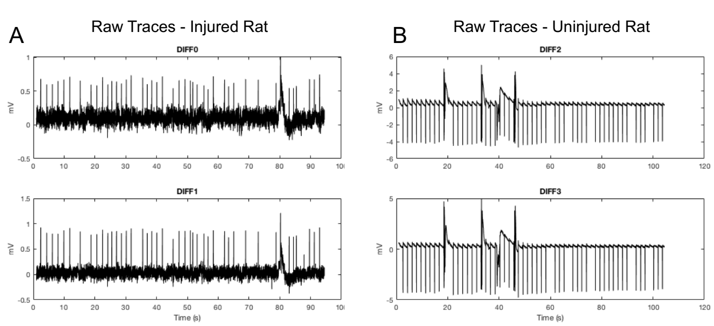
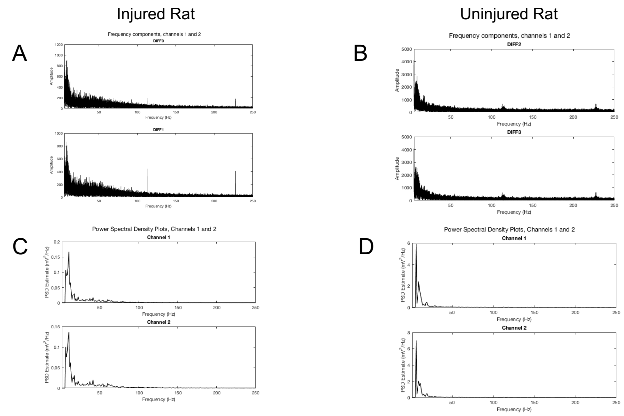


Figure 1: Raw data traces of injured rat (Fig. 1A) and uninjured rat (Fig. 1B), taken from ECoG array implanted over MCtx while rat was freely moving in an arena.



O

Figure 2: Frequency components and power spectral densities. Figs. 2A and 2B show the frequency components from two channels recorded from ECoG arrays over MCtx in an injured and uninjured rat. Figs. 2C and 2D show the power spectral densities computed from those samples.

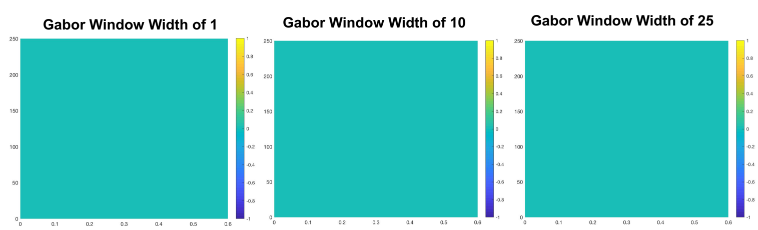


Figure 3: Spectrograms using different Gabor window widths. Raw data was downsampled and cut to only include the first 2500 datapoints. Changing the window widths did not impact the outputs of the spectrogram. It's likely that more datapoints are needed to show how frequency changes over time from these channels.

in this study suggests that there are some differences in MCtx signals in injured vs. uninjured rats, the transformed data suggests that these differences are not as large as originally thought.

Ideally, this data would be analyzed using spectrograms, although that proved to not be possible in the current study. Improved filtering or resampling of the recorded data could provide a better output of the spectrogram, which would give information about how the population activity is changing in MCtx over time, and could provide insight as to how these signals change during movement and movement preparation.

6 References

1. Kutz, J.N. (2013). Data-Driven Modeling Scientific Computation: Methods for Complex Systems and Big Data. Chapter 13. Oxford University Press. 1st. Ed.

2. Ahuja, C.S., Wilson, J.R., Nori, S., Kotter, M.R.N., Druschel, C., Cur, A., Fehlings, M.G. (2017) Traumatic spinal cord injury.*Nature Reviews Disease*, 27,3:17018.

3. McPherson, J.G., Miller, R.R., Perlmutter, S.I. (2015). Targeted, activity-dependent spinal stimulation produces long-lasting motor recovery in chronic cervical spinal cord injury. *Proceedings of the National Academy of Sciences*, 112,39,12193-12198.

4. Buzsaki, G., Anastassiou, C.A., Koch, C. (2016). The origin of extracellular fields and currents - EEG, ECoG, LFP, and spikes. *Nature Reviews Neuroscience*, 13(6):407-420.

Appendix A MATLAB Functions

Notable functions used in this study:

- `input` Request for user input.
- `csv2struct` Converts a .csv input into a structure array.
- `regexprep` Replaces text using a regular expression.
- `isfield` Determines if the input is a field in a structure array.
- `extractfield` Returns the field values of a structure array.
- `length(X)` Gives the length of X.
- `linspace(x1,x2,n)`Computes a linearly spaced vector from x1 to x2 in increments of n.
- `bandpass(X,fpass,fs)` Filters the input signal X using the inputted fpass (low and high band limits) and the sampling frequency, fs.
- `abs(X)` Returns the absolute values of values in X.
- `fft(X)` Performs the fast Fourier transform on X.
- `pcolor(X,Y,C)` Creates a pseudocolor plot along the axes X and Y using values from matrix C to determine color intensity.
- `shading interp` Sets shading of a color graph by interpolating colormap.
- `pwelch` Computes the power spectral density of an input signal using Welch's overlapped segment averaging estimator.
- `zeros(m,n)` Creates a matrix of zeros of size m,n.
- `exp(X)` Returns the exponential e for X.

Appendix B MATLAB Code

```
    %% AMATH582 Final Project - Analysis of ECoG Signals
clear all; close all; clc;
%% Clean up inputs and assign file names etc.
%Input file name for analysis; enter filename surrounded by quotes
fileName=input('Enter name of file for analysis')

%Call load_file_emg.m to convert hex files to integers
load_file_emg_edited

%Call in structure of channels recorded on day/time
dataFiles=csv2struct('datafiles_channels.xlsx');

%Rename fileName to match file name input into struct
fileName=regexprep(fileName,'.txt','_txt');

%Pull out information about which channels were recorded from and date/time.
fldnm=fileName;

if isfield(dataFiles,fileName)==1
    expInfo=extractfield(dataFiles,fldnm)
end

info=expInfo{1,1};
channel1Name=info{3,1};
channel2Name=info{4,1};
experimentDate=info{1,1};
experimentStartTime=info{2,1};

%% Raw data graphs
%convert sample rate to time
recording_length=(length(ch1_s))/fs;
time=linspace(1,recording_length,length(ch1_s));

figure(1)
subplot(2,1,1)
plot(time,ch1_s,'k')
ylabel('mV')
title(sprintf(channel1Name))
subplot(2,1,2)
plot(time,ch2_s,'k')
ylabel('mV')
title(sprintf(channel2Name))
xlabel('Time (s)')

%% FFT and power spectral density of data
n1=length(ch1_s); n2=length(ch2_s);
L1=n1/fs; L2=n2/fs; t1=(1:n1)/fs; t2=(1:n2)/fs;
k1=[0:(fs/2)/(n1/2-1):fs/2]; k2=[0:(fs/2)/(n2/2-1):fs/2];
ch1_bp=bandpass(ch1_s,[5 250],fs); ch2_bp=bandpass(ch2_s,[5 250], fs);

%FFT
ch1_bpt=abs(fft(ch1_bp)); ch2_bpt=abs(fft(ch2_bp));
```

```
figure(1)
subplot(2,1,1); plot(k1,ch1_bpt(1:length(ch1_bpt)/2),'k'); xlim([5 250]);
ylabel('Amplitude'); xlabel('Frequency (Hz)'); title(sprintf(channel1Name));
subplot(2,1,2); plot(k2,ch2_bpt(1:length(ch2_bpt)/2),'k'); xlim([5 250]);
ylabel('Amplitude'); xlabel('Frequency (Hz)'); title(sprintf(channel2Name));
sgtitle('Frequency components, channels 1 and 2','FontSize',[14])

%PSD
[Pxx1,f1]=pwelch((ch1_bp)-mean(ch2_bp),[],[],fs);
[Pxx2,f2]=pwelch((ch2_bp)-mean(ch2_bp),[],[],fs);

figure(2)
subplot(2,1,1);plot(Pxx1,'k','LineWidth',1.25); xlim([1 250]); title('Channel 1');
xlabel('Frequency (Hz)'); ylabel('PSD Estimate (mV^2/Hz)');
subplot(2,1,2);plot(Pxx2,'k','LineWidth',1.25); xlim([1 250]); title('Channel 2');
xlabel('Frequency (Hz)'); ylabel('PSD Estimate (mV^2/Hz)');
sgtitle('Power Spectral Density Plots, Channels 1 and 2')
%% Gabor filtering and spectrogram
ch1_cut=ch1_bp(1:2500); ch2_cut=ch2_bp(1:2500); %Take segment of data so memory doesn't overload

%Set parameters
n1=length(ch1_cut); n2=length(ch2_cut);
L1=n1/fs; L2=n2/fs; t1=(1:n1)/fs; t2=(1:n2)/fs;
k1=[0:(fs/2)/(n1/2-1):fs/2]; k2=[0:(fs/2)/(n2/2-1):fs/2];

%Gabor filter and spectrogram
tslide=0:0.1:L1;
ch1_spec=zeros(length(tslide),length(ch1_cut));

for j=1:length(tslide)
    g=exp((-25*(t1-tslide(j)).^2)).*cos((t1-tslide(j))*pi);
    ch1g=g.*ch1_cut;
    ch1gt=fft(ch1g);
    ch1_spec=[ch1_spec; abs(ch1gt)];
end
ch1_spec=ch1_spec(1:length(tslide),1:length(ch1_cut)/2);

f=figure('Visible',false);
pcolor(tslide,k1,ch1_spec.'); ylim([0 250])
shading interp; colorbar;
print('DIFF0_Width25_11052019.jpeg','-djpeg')
close(f)
```