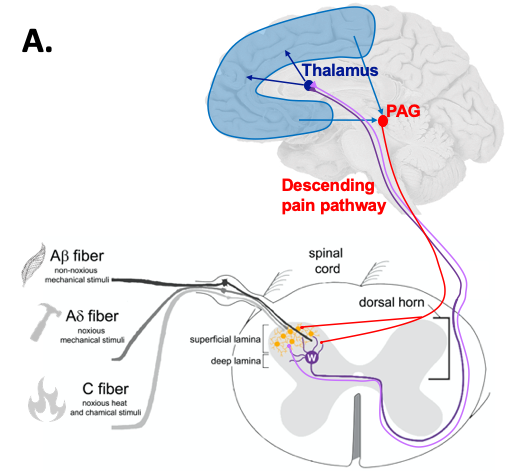
**Background**

Chronic pain is a significant burden to public health, and has proved elusive to effective long-term treatment. Neuromodulation by electrically stimulating nerve fibers of the periphery or the spinal cord have shown promise as a replacement to opioids with fewer side effect. Major medical device companies have spinal cord stimulators on the market, though the programming of the therapy parameters is performed without a thorough understanding of the pain system’s response to stimulation. Peripheral nerve stimulation (PNS) approaches have shown promise in early trials, but none have received FDA approval to our knowledge. Most programming approaches are open-loop, in that they involve finding a combination of pulse width and frequency that provides pain reduction and holding that combination constant, rather than adapting to the state of the patient over time. Some closed-loop strategies have been proposed to adapt to some aspect of the patient state, with the goal to suppress *all* pain once it is detected. This acts like a local anesthetic, dangerous in situations of tissue-damaging pain where a patient needs to quickly be aware of the damaging stimuli. To instead only suppress pathological pain and retain nociceptive pain, it is necessary to understand the difference between the pain response in health and injury or disease. **The primary goal of this project is to create a closed-loop PNS therapy that uses knowledge of the response of the pain system to stimulation, via linear time-invariant (LTI) models, to *restore* an injured pain system to a healthy state, thereby maintaining the benefits of nociceptive pain.**

 Success with stimulation therapies to this point has been elusive because the pain system is complex and builds on a tightly regulated dynamical crosstalk between the peripheral nervous system and the brain via the spinal cord (Figure A). When you damage tissue, such as a paper cut, the ascending pain pathway carries information from the periphery to the brain. A primary sensory neuron specific to pain (nociceptor) fires a series of action potentials down its axon and synapses in the deep lamina of the dorsal horn on a wide dynamic range (WDR) neuron and other neurons in the more superficial lamina of the dorsal horn. The WDR neuron integrates information from primary sensory and interneurons and acts as the first major relay station, sending its axon, along with others, to the ventral posterolateral nucleus (VPL) of the thalamus in the brain. Finally, the thalamus relays the information that you have been cut to numerous cortical areas to sense, think about, and emotionally respond to the pain. To add to this complexity, there is a descending pathway that begins in the brain and ends with synapses in the dorsal horn, modulating the ascending transmission.

In the periphery, there are fast-transmitting fibers (A fibers) that generally transmit touch information like form, movement, and texture, as well as slow-transmitting fibers (C fibers) that transmit burning, aching pain sensations. The WDR neuron is special in that it receives inputs from both types of fibers and in response to a large, painful stimulus fires in two bursts. The first is the touch burst, then shortly after is the pain burst. You will see this clearly in the data provided to you.

This data was collected by stimulating with electricity directly on the sciatic nerve of rats and recording from individual WDR neurons in the spinal cord. Some subjects are naïve, healthy rats while others have undergone a spinal cord crush injury to create a pain model. We isolated action potentials from the WDR neurons, then created normalized firing rate curves over time with a smoothing filter. You are getting firing rates in response to a particular stimulus pattern called wind-up, where we stimulated once each second at 1 mA. As you can imagine, if someone pinched you over and over in the same place, it would get a little more painful each time. That’s the wind-up effect. You will get an input time-course, a naïve firing rate response to that input, and an injured firing rate response to the same. Your task is to fit an LTI model for the naïve response and another for the injured response. You then will design a closed-loop controller to make the injured response act like the naïve response (i.e. restoring the system to health).

Diagram, schematic

Description automatically generated

Figure B. Conceptual diagram of your goal, to minimize the error between the healthy response and closed loop injured response.

Dataset of WDR recordings:

Within the .mat file are 5 variables:

Fs = sampling time of the recordings (seconds)

t = time vector (seconds)

r = input vector (mA)

y\_i = injured rat WDR response

y\_n = naïve rat WDR response

Skelton Code and Simulink Model:

We have provided a skeleton code to show you how to fit a LTI transfer function in Matlab using the tfest function. You may modify this code however you want. Using the tfest function, you will fit two separate LTI transfer function models: naïve and injured. Adjust the order of the transfer function model.

Additionally, we have provided a Simulink model to help you design your controller for Part 2. We define several variables in the Matlab skeleton code to use in Simulink.

**Part 1: Fitting the Models**

Plot the naïve and injured data over time on the same plot. What differences do you see between the responses?

Run the first section of the Matlab skeleton code to see the variables it produces. Plot the model fit alongside the data it is representing (whether that is naïve or injured). See how well the fit matches. How could you quantify this fit?

Modify the lines noted in the comments to test a variety of combinations of poles and zeros. Select the one that fits the data the best. Describe and show your rationale for selecting this model fit. You should show 1) some kind of plot showing a qualitative metric and 2) describe additional metrics used to decide on a best model.

Repeat the process for the second rat data.

Create a figure that shows your data and model fits all on one plot (2 lines of data, 2 lines of model fits). ***We should be able to easily see how well your fitted model matches the data.***

**Part 2: Closing the Loop**

Using your TF models from Part 1, we want to create a control signal via closed loop design to make the injured response look like that in a healthy system. To do this, we will use a Proportional Integral Derivative (PID) controller. We have set up the architecture for you on Simulink, but it is your job to tune the parameters (Kp, Ki, and Kd) of the PID controller.

Define your fitted models into the Simulink architecture using Matlab (Part 2 of Skeleton Code). Double check in Simulink that your naïve model appears where it says “Naïve” and the injured model is in two places, where it says “Injured” and “CL Injured.” Also, double check your model is using a fixed-step integrator (follow the steps in the blue box in the Simulink model).

Run the Simulink model as is to get an idea of what comes up. Look at the results of each of the scopes.

Tune the parameters of the PID controller until you are satisfied with your closed loop injured model matching your naïve model. You can do this any way you want.

Describe your tuning process. What metrics did you choose to be important as you are tuning your controller? Why did you decide on this as your final controller?

Create a figure that shows the recorded naïve WDR response, injured WDR response, and your closed loop injured simulated response on one plot. ***We should be able to easily see that your controller moves the injured response toward the healthy response.***

**Extra Credit (Optional):** Can you construct a different controller (not PID) that works better? If so, show us.

**Presentation**

The team will have 5 minutes to present their results to the class. Shoot for no more than 5 slides. Each person should speak during the presentation.

* Intro/Problem statement (30 seconds)
* Approach to solution (~ 1 min)
* Description of data (30 seconds)
* Models fits (~ 1 min)
* Control design and architecture (~ 1 min)
* CL results (~ 1 min)

**Submission: Due Noon on March, Thursday 11th**

Please submit both the presentation and the final transfer function models and PID controller gains (using the final part of the Matlab code). If you did the extra credit and you found a better controller, then also submit your updated Simulink code.

Please name your file “Team\_[insert #]\_Results.mat”