No-Pain-No-Gain or Low-Pain-With-Gain: Applying Time-Variant Constraints in Adaptive Sampling Methods Towards Reduced Recalled Pain in Neurostimulation Experiments

Kora S. Hughes[1] **Rabira Tusi**[1] **Ezekiel Ajayi**[1]
[1]Carnegie Mellon University
{kshughes,rtusi,eajayi}@andrew.cmu.edu

Abstract

Adaptive sampling and optimal control methods often overlook constraints needed to ensure that sampled parameters do not compromise system integrity. This is especially significant in the context of human trials, where reinforcement learning methods may incentivize actions that are effective for reward maximization but risk causing physical/psychological damage to patients. In neurostimulation experiments, this means that adaptive sampling techniques such as Bayesian Optimization may incentivize the injection of large current waveforms that inadvertently cause tissue damage or severe pain. Though hard limitations on waveform exploration are applied to ensure patient safety, they often still incentivize the most painful solutions and fail to take into account more complex, time-variant models of pain. To remedy this, we propose *PainOpt*; a pain-informed adaptive sampling method and data-driven model that leverages a time-variant understanding of patient pain to minimize recalled pain. When benchmarked against traditional sampling methods with no consideration of pain, we observe a 37+% reduction in recalled pain while preserving the original treatment objective.

1 Background

Neurostimulation has a wide range of applications for several neurological and neuropsychiatric conditions. A re-emerging technology that holds promise in tackling some of the weaknesses of previous techniques is pulsed transcranial electrical stimulation (pTES). The one drawback of pTES is the pain that it elicits [6]. Often, the required level of current to evoke motor evoked potential (MEPs) is also enough to excite scalp and dural nociceptors, causing intolerable pain. This pain often limits the space of acceptable waveforms applicable to neurostimulation trials and makes treatment options less accessible to patients. Thus, researchers have explored a series of techniques to mitigate this pain – the most successful of which incorporate adaptive sampling techniques.

1.1 Prior Adaptive Sampling In Neurostimulation

One technique employed to reduce pain is the use of "background hums" which are high-frequency, low amplitude pulse trains that are superimposed on the MEP-inducing pulse [2]. Waveform parameterization of these hums have often employed adaptive sampling techniques. Similarly, Safe model-based RL uses Lyapunov functions and Gaussian process models to guide safe exploration within an estimated region of attraction, cautiously expanding the explored space over time [1]. More recently, ECCBO also introduced a feedback controller that enforces constraint satisfaction by treating constraint values as trackable setpoints, enabling unconstrained Bayesian optimization over safe regions [4]. While Safe model-based RL and ECCBO have advanced safe exploration in adaptive sampling, they have key limitations when applied to neurostimulation. Both approaches aim to prevent catastrophic system failures by constraining the search space, but they do not account for the neuropsychological dynamics of pain perception. Specifically, while both methods can be applied to minimize the pain of a single stimulation event, neither method adapts to minimize perceived pain according to temporal-informed principles like memory.

1.2 Temporal Distortion in Pain

Pain is not simply a direct response to a noxious stimulus; it is a dynamic, psychologically constructed experience shaped by prior events, context, and memory. This gives the perception of pain a temporal weight of sorts. In the setting of neurostimulation, a patient's prior stimulation events or even experiments can distort the perception of pain. Yelle et. al. explores the influence of temporal contrast in a series of painful events and reveals that large, abrupt increases in painfulness across consecutive stimulation events can exacerbate perceived discomfort beyond what would be expected from the absolute intensity alone [7]. This means that reducing the amount of abrupt shifts in painful stimulation events can reduce the perceived discomfort. Similarly, the pain recalled from an experiment, a key indicator for patient retention and treatment adherence, is distorted by this temporal weight. In a 2011 study on chronic pain, Schneider et. al. showed that retrospective evaluations of painful experiences are disproportionately determined by the most intense (peak) pain and the pain experienced toward the end of the event sequence [5]. This means that, even if the majority of stimulation events are tolerable, a particularly painful event, especially if it occurs near the end of an experiment, can color the participant's memory of the entire session as highly unpleasant. We can leverage this principle in reverse to improve a patient's memory of a painful experiment. By incorporating these temporally-informed understandings of pain into neurostimulation optimization, particularly by designing sampling policies that minimize large pain jumps and ensure that sessions end on lower-pain events, we offer a new way to engineer not only safer, but more tolerable experiences.

2 Methods

The main objective of our project is to evaluate the performance of our pain-informed adaptive sampling method against a traditional adaptive sampling method with respect to its objective performance and the simulated participant's pain. In order to do this we defined an experimental paradigm where a simulated participant is given a noninvasive neurostimulation treatment. The goal of the experiment is to maximize the objective of the patient's treatment while minimizing the patient's overall pain. In this case, we arbitrarily treated the goodness of the treatment to be correlated with the firing rate f of a single layer 2/3 pyramidal neuron neuron within the participant's cortex. This firing rate was calculated via the NEURON [3] platform with the amplitude (nanoamps) and pulse width (microseconds) of the injected waveform as tunable parameters for each stimulation event. Between experiments, we treat the objective metric of success as the maximum firing rate, f_{max}. Similarly, we designed a set of functions to describe the simulated patient's pain levels in response to each injected waveform. We used this pain model to performed a systematic comparison of the patient's memory of their experiment's pain in addition to the optimal treatment outcome, f_{max}, returned by our respective adaptive sampling models. In order to do this, we define apriori **A**, a set of functions representing the patient's pain during an experiment, and **B**, a pain-informed adaptive sampling method leveraging our understanding of the patient's pain.

2.1 Modeling Pain (A)

We arbitrarily defined the patient's pain in response to the stimulated waveform as follows:

$$P_i = d_{amp} * \frac{A}{A_{max}} + d_{pw} * \frac{W}{W_{max}}$$

where P_i is the patient's pain on a continuous scale from 1 to 10, A and W are the amplitude and pulse width of the injected waveform respectively, and A_{max} and W_{max} are the maximum amplitude and pulse width we allowed our adaptive sampling models to investigate: set to 15μs and 980nA respectively. Additionally, d_{amp} and d_{pw} are tunable hyperparameters representing the maximum

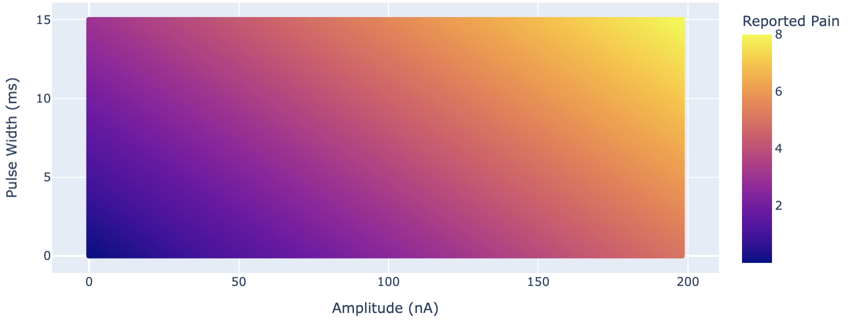


Figure 1: Hyperplane of the simulated patient's reported pain (instantaneous pain) after a single stimulation event with a specified amplitude and pulse with.

deviation in pain due to the amplitude and pulse width. During our trials we arbitrarily defined these to be $d_{amp} = 5$ and $d_{pw} = 3$ respectively. We used the aforementioned equation to describe each patients response to a single stimulation event, context-agnostic; from here on we will refer to this type of pain as instantaneous pain. Figure 1 shows the resulting instantaneous pain hyperplane as a function of the waveform parameters. In addition to our understanding of instantaneous pain, we defined two additional models of reported patient pain that account for time-variant stimulation events: relative pain distortion and total pain recalled.

2.1.1 Relative Pain

Relative Pain Distortion (RPD) is a metric of pain used to describe how much a patient's reported pain score changes due to the context of a previous stimulation event [7]. We define this distortion P_r to be a function of the current instantenous pain P_i, and the previous instantaneous pain P_{i-1} as follows:

$$P_r = d_{rpd} * \alpha(\frac{P_i - P_{i-1}}{P_{max}})^2$$

where P_{max} is the highest pain value on our chosen scale (defaulted to 10), and α is -1 if $P_i < P_{i-1}$ and 1 otherwise. Similar to the previous d variables, $d_{rpd} \in [0, P_{max}]$ is a tunable hyperparameter representing the maximum amount of pain distortion that can occur due to offset analgesia; for our experiments this was arbitrarily set to 5. We can use this to calculate our patient's perceived pain P in the context of each trial as $P = P_i + P_r = P_i + d_{rpd} * \alpha(\frac{P_i - P_{i-1}}{P_{max}})^2$. The distortion due to offset analgesia is $[-d_{rpd}, d_{rpd}]$, scaling with the squared difference in normalized pain between the present and last trial. The relationship between P_i and P can be visualized as 'instantaneous' vs. 'perceived' pain. We can observe the difference, P_r, as the deviation of the red line from the blue line in Figure 2.

2.1.2 Memory of Pain

Total Pain Recalled (TPR) is a metric used to describe how the patient reports a set of painful experiences in retrospect, aka the memory of a painful experience. Schneider et. al. investigates the concept of peak-end and experimentally demonstrates that the mean, peak, and end in a set of painful experiences can estimate the end-of-experiment recall [5]. From their inter-participant results, we derive weights for the following equations to represent this phenomenon:

$$d_{mean} = \frac{b_{mean}}{b_{total}} * \frac{s_{eod}}{s_{mean}}, d_{peak} = \frac{b_{peak}}{b_{total}} * \frac{s_{eod}}{s_{peak} + s_{mean}}, d_{end} = \frac{b_{end}}{b_{total}} * \frac{s_{eod}}{s_{end} + s_{mean}}$$

where d represents the weighted distortion used to estimate the memory of a given pain (mean/peak/end), s_{eod} is the end-of-day pain measurement used as a proxy for the ground-truth memory of the participants' pain, s represents the Schneider's average pain observed for each (mean/peak/end) between participants ([5] table 1), b represents variance captured by a given pain (mean/peak/end) in estimating the EOD pain ([5] table 2), and $b_{total} = b_{mean} + b_{peak} + b_{end}$ is the sum of all variance weights. The resulting weights used were $d_{mean} = 0.81$, $d_{peak} = 0.17$, and $d_{end} = 0.07$. We then apply these to a weighted summation of our simulated patient's pain to create the final TPR equation, defined as follows:

$$P_{tpr} = d_{mean} * p_{mean} + d_{peak} * p_{peak} + d_{end} * p_{end}$$

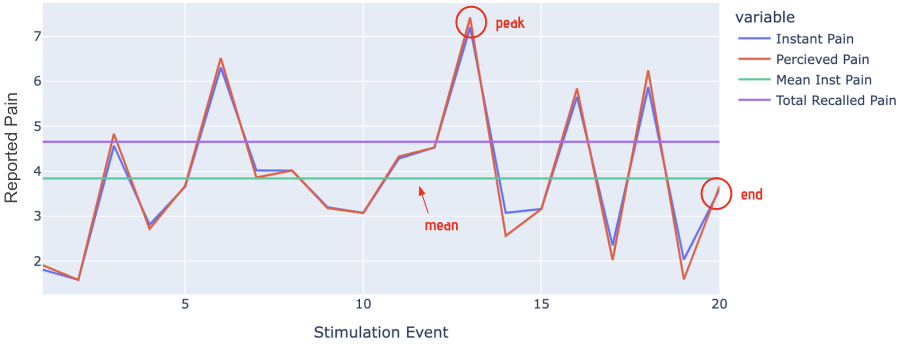


Figure 2: Comparison of the mean, peak, end, and TPR of a random set of stimulation events.

where P_{tpr} is the patient's remembered pain, $p_{mean} = \frac{\sum P}{N}$ is the average pain P reported across N patient trials, p_{peak} is the maximum pain reported, and p_{end} is the last pain reported in a trial. The main takeaway from this equation is that the memory of pain is calculated as a weighted sum of the expected memory of pain from each variable mean/peak/end with respect to the ratio of variance, $\frac{d_{pain}}{d_{total}}$, represented in Schneider's EOD estimation. The TPR of a random trial, shown with respect to the mean, peak, and end, is visualized in Figure 2.

2.1.3 Ideal Pain Schedule

Since we are handling larger series of trials, we created a continuous definition of the peak-end rule called the 'ideal patient schedule.' We defined this ideal curve, P_s, to be the a function of pain over the trial number that kept in mind both MOP and RPD by minimizing large jumps in pain and while keeping a low start/end.

$$P_s = \frac{4\varepsilon_{max}}{(d_{ideal} * P_{max})^2} * -\varepsilon(\varepsilon - (d_{ideal} * P_{max}))$$

where ε_{max} is the maximum number of trials in an experiment, ε is the current trial number, and d_{ideal} is a modifier dictating how much of the maximum pain should exist within the ideal schedule (defaulted to 0.9).

2.2 Pain-Informed Sampling (B)

In order to incorporate our understanding of pain into our adaptive sampling method, we leveraged a schema for weighted expected-improvement that we named *PainOpt*. *PainOpt* follows the traditional expected improvement equation but weights the means resulting from a radial basis function (RBF), μ_{rbf}, by the following equation:

$$\mu_{pain} = \mu_{rbf} \circ \hat{D}_{pain}$$

where μ_{pain} is a matrix of pain-informed means resulting from the element-wise multiplication of the RBF means, μ_{rbf}, and a matrix of pain weightings, $\hat{D}_{pain}$. In this case, we define each element $\hat{d}_{pain} \in \hat{D}_{pain}$ by the following equation:

$$d_{pain} = 1 - d_{pain} * P + d_s \frac{|P - P_s|}{P_{max}}$$

where d_{pain} and d_s are hyperparameters representing the distortion of the weights due to the perceived pain and deviation from the ideal pain schedule, which were set to around 0.4 and 0.8, respectively (trial dependent). To ensure that the lower bound of the schedule does not incentivize higher pain values to much during the middle trials, d_s was additionally down-weighted by another hyperparameter, $d_s = d_s * d_{lpm}$, when $P \leq P_s$ (also trial dependent). In all, this creates a matrix of weights $\hat{D}_{pain}$ that is dependent on the current trial number, ε, and the current waveform parameters, (A, W). The resulting hyperplane can be thought of as a combination of expected firing rate and expected pain that will be passed into the expected improvement function before sampling.

2.3 Additional Models

In an effort to test the effectiveness of our pain model, we opted to implement both a Thompson Sampling as well as a Q-Learning algorithm. In this section we shall discuss the efficacy of these models with regards to our pain model, and discuss the expectations and drawbacks of implementing them. For each model we attempted, they all conformed to variations of the following reward function:

$$R(x_i, t) = f_i(t) + (\epsilon_i \cdot f_i)$$

Where x_i represents a given arm or state-action pair, f_i is representative of the firing rate of the neuron at some given iteration t, and ϵ_i is the distribution associated with a particular arm designed to disincentivize high pain with respect to the firing rate. For these distributions, we opted to test an exponential distribution following the form: $f(x; \gamma, p_i) = \frac{1}{\gamma p_i} e^{\gamma p_i}$ Where $\gamma \in \Gamma = \{0.5, 1.0, 2.0, 2.5\}$, and p_i is the normalized pain at that time-step.

We also desired to leverage as a beta distribution of the form: $f(x; \alpha, \beta) = \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ Where $B(\alpha, \beta)$ is a normalization that follows the beta function: $B(\alpha, \beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1} dt$

In this method, α represents the number of iterations where the pain was lower than a certain threshold, and β represents all values where the pain was greater than that threshold. Furthermore, for this reward model, we desired to also test the following reward function alongside the aforementioned reward function:

$$R(x_i, p_i) = (f_i(t) \cdot \alpha) + (p_i(t) * \beta)$$

In this case, α and β are normalized with respect to the total number of iterations.

2.3.1 Thompson Sampling & Q-Learning

In order to test the breadth of our model, we opted to compare PainOpt to a Thompson Sampling and Q-Learning approach that incorporates pain as a factor in calculating the reward. The objective, as before, was to search for the optimal amplitude and pulse width that would yield a high firing rate, while decreasing pain.

Initially, the approach selected was to define $K = 3$ arms such that $B = \{R_1, ..., R_K\}$, where B represents the set of bandit arms. In the $K = 3$ arms case, the goal was to have (1) one arm change only amplitude, (2) one arm change only pulse width, and (3) one arm changing both hyperparameters. As we began designing the ground truth and immediate reward functions, this approach became increasingly difficult, as it was not only counterintuitive to our objective, but also incredibly long to run. Therefore, we sought to redefine and discretize the search space by creating a set number of arms N that were unique combinations of amplitude and pulse widths. Due to the large range of potential amplitude and pulse width combinations, we were discouraged in continuing in this way. Q-Learning seemed a reasonable alternative given that we could define a Q-Table that would: (1) achieve a broad and discrete range of amplitude and pulse width combinations, (2) create an action space allowing us to (a) change either hyperparameter, (b) alter both hyperparameters, or (c) alter none, providing an action space of $K = 4$. Unfortunately, Q-Learning was difficult to implement for the same reasons as Thompson Sampling.

2.3.2 Continuous Arm Space (Bayesian Linear Regression + Thompson Sampling)

After attempting and finding minimal success with Q-learning and Thompson Sampling due to the necessary discretization of the hyperparameters, we instead opted to use a Bayesian Linear Regression Model in tandem with Thompson Sampling such that we may define a continuous search space, while still leveraging the adaptive sampling strengths of Thompson Sampling. Our approach here was to leverage a Bayesian Ridge Regressor, allow for brief exploration via random sampling across the full range of amplitude and pulse width combinations for $N = 10$ iterations, then update the model based on the selected best arm in each iteration.

While this technique still required the discretization of arms, it allows us to train the Bayesian Ridge model over numerous iterations wherein, the amplitude and pulse width of the best arm, as well as the associated firing rate was passed on to the model for fitting. Unfortunately, due to the time constraints, as well as numerous challenges inhibiting the compatibility of our pain-informed bandits and the NEURON module, we were unable to fully realize this solution.

3 Results

3.1 Ground truth sampling and bound determination

In order to determine the correct bounds for the pulse width and amplitudes in the Bayesian Optimization algorithms (and get an understanding of the ground-truth space), a grid of parameter values were sampled and their respective firing rates were recorded. A good set of bounds will (1) capture the maximum amount of variation in firing rates across its domain and (2) have the firing rates scale with either the amplitude or pulse width such that maximizing the firing rate will not simultaneously minimize the aforementioned pain function. Prior to any ground-truth sampling, the lower and upper bounds for amplitude and pulse width were set to [0nA, 40nA] and [0ms,30ms] respectively. An initial log-sampling scheme was employed to determine the minimum current and pulse width required to elicit a spike. Fig. 3 illustrates an increasing firing rate with amplitude and pulse width. Notably, this trend isn't perfectly monotonic illustrating the nonlinearity of the Pyramidal neuron model. More importantly, Fig. 3 demonstrates that this space is highly noncontinuous with large jumps in firing rate which poses problems for any form of Bayesian Optimization. Using this domain knowledge, the bounds for the pulse width were increased to [0ms, 980ms]. Moreover, the bounds for the amplitude were decreased to [0nA, 15nA] because firing rate did not increase beyond that point for all pulse widths within the sampled range. A heatmap was not generated for these new bounds as running a 1000 ms simulation would take an excessive amount of time.

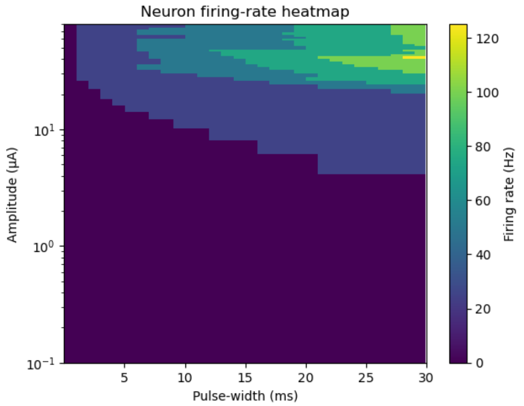
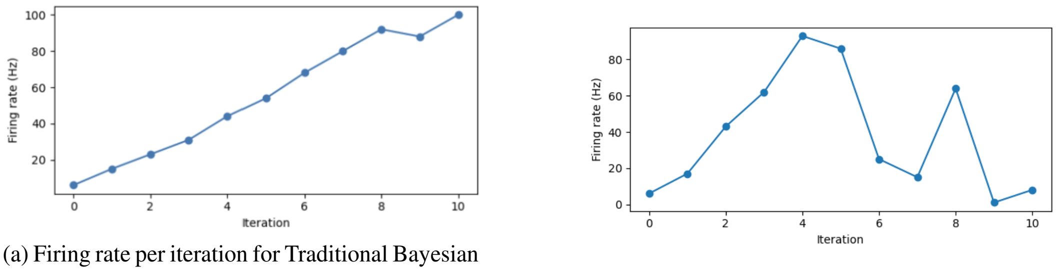


Figure 3: Ground truth heatmap of Pyramidal Neuron model firing rate with varying pulse widths and amplitudes. Uses finer sampling at smaller pulse widths and is accordingly log-scaled.

3.2 Maximum firing rate acquired by PainOpt is similar to Bayes Opt

Both Bayesian Optimization and PainOpt (with the best model parameters) were run for 10 iterations with the same base model parameters (RBF kernel, length constant of [2.75, 250], no hyperparameter optimization). The firing rate at each iteration was recorded and is illustrated in Fig. 4. Notably, the peak firing rate for traditional Bayesian Optimization was 109 Hz while that of PainOpt (d_{pain} = 3 d_{lpm} = 0.1) was 101 Hz (7% reduction). Moreover, we see that the firing rate in traditional Bayesian optimization monotonically increases (except for iteration 9) while PainOpt follows a curve analogous to pain schedule. For 20 iterations, traditional Bayesian optimization shows a similar initial monotonic increase up until iteration 11. This is followed by a sporadic sampling of the space and is likely caused by the fact that the model has already explored the best possible point. Notably

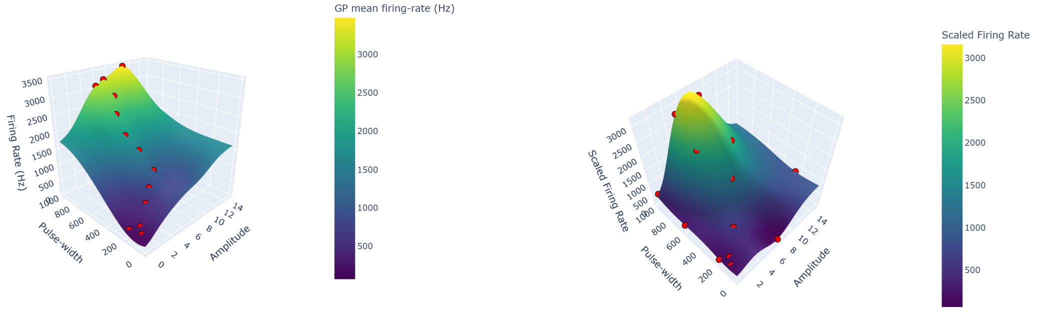


(a) Firing rate per iteration for Traditional Bayesian Optimization

(b) Firing rate per iteration for PainOpt

Figure 4: Comparison of traditional Bayesian optimization with and PainOpt over the course of 10 iterations with metric being the firing rate sampled at each iteration

both PainOpt and traditional Bayesian optimization were able to sample the same maximum firing rate. Fig. 5 shows the final predicted gaussian process firing rate means after the 10th iteration of traditional Bayesian optimization and PainOpt respectively. Akin to 4, we see that traditional Bayesian Optimization explores the space incrementally. Meanwhile, PainOpt tries to balance this incremental ascent with the weighting of the pain function. The result of this is more "exploration" of the domain driven by the variable pain weightings across iterations.



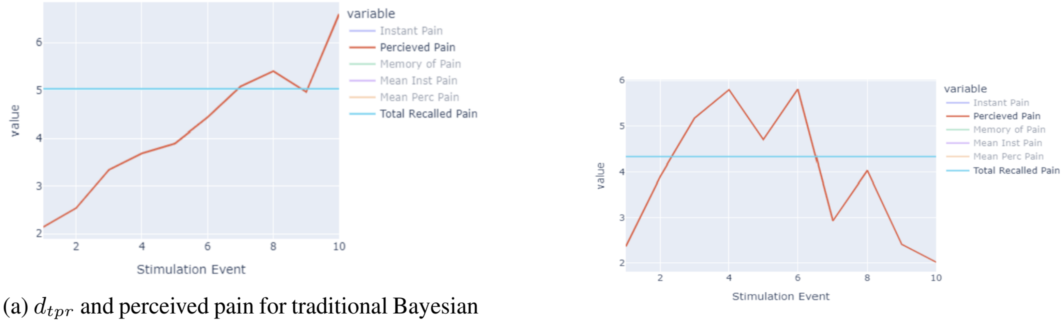
(a) Predicted firing rate means for traditional Bayesian Optimization

(b) Predicted firing rate means for PainOpt

Figure 5: Comparison of scaled predicted gaussian process means for traditional Bayesian Optimization and PainOpt. Predicted GP is at 10 iteration.

3.3 d_{tpr} is substantially lowered with PainOpt

To evaluate the impact of incorporating pain constraints into adaptive sampling, we compared traditional Bayesian Optimization to our PainOpt model (with the best model parameters) over 10 stimulation events and recorded the perceived pain at each iteration as well as the d_{tpr} throughout all iterations (Fig. 6). In the traditional Bayesian Optimization condition, perceived pain increased steadily across stimulation events, reaching a peak value near 7 by the final stimulation. This persistent escalation of perceived pain corresponded with a higher d_{tpr} score. The overall d_{tpr} with traditional Bayesian Optimization here was 5.02. In contrast, PainOpt actively modulated stimulation parameters to manage perceived pain during the experiment. Consistent with the pain schedule, we see a curve in the perceived pain across iterations with a peak pain of 6.74 at iteration 6. The d_{tpr} with PainOpt was 4.23 (37.3% reduction in d_{tpr}). A less consistent adherence to the schedule is seen with PainOpt across 20 iterations. Still the d_{tpr} here was lower than that of traditional Bayesian Optimization (6.02 vs. 4.83 points for a 20% reduction in d_{tpr}).



(a) d_{tpr} and perceived pain for traditional Bayesian Optimization

(b) d_{tpr} and perceived pain for PainOpt

Figure 6: Comparison of d_{tpr} and perceived pain per iteration for traditional Bayesian Optimization and PainOpt for 10 iterations.

3.4 d_{tpr} scales proportionally with peak sampled firing rate with varying model hyperparameters

PainOpt model hyperparameters (d_{lpm} and d_{pain}) were varied and each model was run for 20 iterations, after which d_{tpr} and the f_{max} were obtained. Fig. 7 illustrates the trend between d_{tpr} and f_{max}, suggesting that models that try to maximize the firing rate will also increase the d_{tpr} on average. The trend approximately follows the formula $d_{\text{tpr}} = 0.1 \times d_{\text{fr}} - 5$. Several PainOpt models (as well as the traditional Bayesian Optimization model illustrated as the red dot in Fig. 7) were able to sample a f_{max} of 109 Hz given the simple structure of the ground-truth function. Notably, however, some models were able to elicit a lower d_{tpr} for the same given f_{max}. This is especially true for a f_{max} of 109 Hz where traditional Bayesian Optimization only outperformed one PainOpt model. A table of PainOpt model hyperparameters and the corresponding model performance is provided in Table 1. Two general trends are observed: (1) as d_{pain} increases, d_{tpr} tends to decrease, and (2) at a fixed d_{pain}, increasing the d_{lpm} tends to decrease the d_{tpr}.

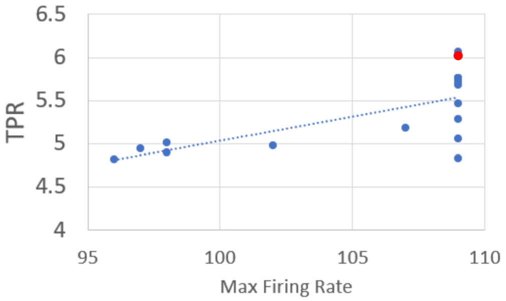


Figure 7: Scatterplot of recorded d_{tpr} and f_{max} for different PainOpt models (with varying lpm and d_{pain} values). The red dot is the d_{tpr} and f_{max} achieved for traditional Bayesian optimization.

4 Conclusion

Within this work, we have shown that PainOpt is capable of achieving a pain reduction of 37% in d_{tpr} while maintaining strong adjacency to the max firing rate of traditional Bayesian Optimization. An observed tradeoff for this performance occurs between the firing rate objective f_{max} and the pain-recall d_{tpr}: as pain savings of PainOpt increase, the model objective diverges farther from traditional sampling techniques. Similarly, as the number of interactions decreases and model complexity increases, this tradeoff is exacerbated.

Although PainOpt does provide a stronger foundation for optimizing with complex time-variant pain constraints with neural data, PainOpt cannot outperform Bayesian Optimization at estimating a pain-uninformed objective. As such, algorithms like PainOpt are best leveraged in conditions where pain is a significant limiting factor in patient retention.

Table 1: Simulated results (f_{max}, d_{tpr}) for Δ (d_{pain}, d_{lpm}) at $\varepsilon = 20$.

d_{pain}	d_{lpm}	f_{max}	d_{tpr}
0	0.1	109	5.76
0	0.3	109	5.76
0	0.6	109	5.76
0	0.9	109	5.76
0.5	0.1	109	5.06
0.5	0.3	107	5.177
0.5	0.6	109	5.47
0.5	0.9	102	4.98
1	0.1	109	5.68
1	0.3	98	5.00
1	0.6	109	5.28
1	0.9	109	5.71
3	0.1	109	4.83
3	0.3	96	4.82
3	0.6	97	4.95
3	0.9	98	4.90

4.1 Future Work

While these results prove promising for simulating pain, it is unlikely that *every* patient adheres to our initial pain model assumptions of instantaneous and recalled pain. As such, proper experimentation with human trials is necessary to fully assess model efficacy and tune hyperparameters d. One potential source of improvement is implemented a multi-objective implementation of PainOpt that estimates patient pain parameters in real-time to minimize the sim-real gap between model assumptions. Additional parameter tuning like hums could allow PainOpt to work alongside existing methods of instantaneous pain reduction to create a more robust system.

References

[1] Felix Berkenkamp, Matteo Turchetta, Angela P. Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees, 2017.

[2] Mats Forssell, Rabira Tusi, Jeehyun Kim, et al. Reducing scalp pain for ptes of motor cortex using background hums. Preprint, Research Square, April 2025. Version 1.

[3] Michael L. Hines and Nicholas T. Carnevale. Neuron: a tool for neuroscientists. *The Neuroscientist*, 7(2):123–135, April 2001.

[4] Dinesh Krishnamoorthy. Eccbo: An inherently safe bayesian optimization with embedded constraint control for real-time optimization, 2024.

[5] Silvia Schneider, Arthur A. Stone, Joseph E. Schwartz, and Joan E. Broderick. Peak and end effects in patients' daily recall of pain and fatigue: a within-subjects analysis. *The Journal of Pain*, 12(2):228–235, February 2011.

[6] Andrea Szelényi, Karl F. Kothbauer, and Vedran Deletis. Transcranial electric stimulation for intraoperative motor evoked potential monitoring: Stimulation parameters and electrode montages. *Clinical Neurophysiology*, 118(7):1586–1595, July 2007.

[7] Marc D. Yelle, June M. Rogers, and Robert C. Coghill. Offset analgesia: A temporal contrast mechanism for nociceptive information. *Pain*, 134(1):174–186, January 2008.

Author's Note: Kora primarily implemented the pain models and weighted sampling method. Rabira performed the PainOpt testing, parameter tuning, and results compiling. Ezekiel created the Thomson sampling, q learning, and bandit implementations. Code available at https://github.com/KoraSHughes/Pain-Opt/.