Université de Montréal

Hierarchical Bayesian optimization of targeted motor outputs with spatiotemporal neurostimulation

par

Samuel Laferrière Cyr

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures et postdoctorales
en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

décembre 2019

Université de Montréal

Faculté des études supérieures et postdoctorales

Ce mémoire intitulé

Hierarchical Bayesian optimization of targeted motor outputs with spatiotemporal neurostimulation

présenté par

Samuel Laferrière Cyr

a été évalué par un jury composé des personnes suivantes :

Francois Major
(président-rapporteur)

Guillaume Lajoie
(directeur de recherche)

Pierre Louis Bellec
(membre du jury)

Mémoire accepté le :
9 décembre 2019

Sommaire

Ce mémoire par article part de la question suivante: pouvons-nous utiliser des prothèses neurales afin d'activer artificiellement certain muscles dans le but d'accélérer la guérison et le réapprentissage du contrôle moteur après un AVC ou un traumatisme cervical ? Cette question touche plus de 15 millions de personnes chaque année à travers le monde, et est au coeur de la recherche de Numa Dancause et Marco Bonizzato, nos collaborateurs dans le département de Neuroscience de l'Université de Montréal. Il est maintenant possible d'implanter des électrodes à grande capacité dans le cortex dans le but d'acheminer des signaux électriques, mais encore difficile de prédire l'effet de stimulations sur le cerveau et le reste du corps. Cependant, des résultats préliminaires prometteurs sur des rats et singes démontrent qu'une récupération motrice non-négligeable est observée après stimulation de régions encore fonctionnelles du cortex moteur. Les difficultés rattachées à l'implémentation optimale de stimulation motocorticale consistent donc à trouver une de ces régions, ainsi qu'un protocole de stimulation efficace à la récupération. Bien que cette optimisation a été jusqu'à présent faite à la main, l'émergence d'implants capables de livrer des signaux sur plusieurs sites et avec plusieurs patrons spatio-temporels rendent l'exploration manuelle et exhaustive impossible. Une approche prometteuse afin d'automatiser et optimiser ce processus est d'utiliser un algorithme d'exploration bayésienne. Mon travail a été de développer et de raffiner ces techniques avec comme objectif de répondre aux deux questions scientifiques importantes suivantes: (1) comment évoquer des mouvements complexes en enchainant des microstimulations corticales ?, et (2) peuvent-elles avoir des effets plus significatifs que des stimulations simples sur la récupération motrice? Nous présentons dans l'article de ce mémoire notre approche hiérarchique utilisant des processus gaussiens pour exploiter les propriétés connues du cerveau afin d'accélérer la recherche, ainsi que nos premiers résultats répondant à la question 1. Nous laissons pour des travaux futur une réponse définitive à la deuxième question.

***K*eywords** BCI · Stimulation Corticale · Processus Gaussien · Optimisation Bayesienne

Summary

The idea for this thesis by article sprung from the following question: can we use neural prostheses to stimulate specific muscles in order to help recovery of motor control after stroke or cervical injury? This question is of crucial importance to 15 million people each year around the globe, and is at the heart of Numa Dancause and Marco Bonizzato's research, our collaborators in the Neuroscience department at the University of Montreal. It is now possible to implant large capacity electrodes for electrical stimulation in cortex, but still difficult to predict their effect on the brain and the rest of the body. Nevertheless, preliminary but promising results on rats and monkeys have shown that a non-negligible motor recovery is obtained after stimulation of regions of motor cortex that are still functional. The difficulties related to optimal microcortical stimulation hence consist in finding both one of these regions, and a stimulation protocol with optimal recovery efficacy. This search has up to present day been performed by hand, but recent and upcoming large scale stimulation technologies permitting delivery of spatio-temporal signals are making such exhaustive searches impossible. A promising approach to automating and optimizing this discovery is the use of Bayesian optimization. My work has consisted in developing and refining such techniques with two scientific questions in mind: (1) how can we evoke complex movements by chaining cortical microstimulations?, and (2) can these outperform single channel stimulations in terms of recovery efficacy? We present in the main article of this thesis our hierarchical Bayesian optimization approach which uses gaussian processes to exploit known properties of the brain to speed up the search, as well as first results answering question 1. We leave to future work a definitive answer to the second question.

***K*eywords** BCI · Cortical Stimulation · Gaussian Processes · Bayesian Optimization

Table des matières

Sommaire iii

Summary iv

Liste des figures vii

Remerciements 1

Chapitre 1. Introduction 2

Chapitre 2. Studying the Motor System with Brain-Computer Interfaces (BCI) and Electromyograms (EMG) 4

Chapitre 3. Gaussian Processes and Bayesian Optimization 9

3.0.1. Bayesian Optimization using Gaussian Processes 12

Article. Hierarchical Bayesian optimization of targeted motor outputs with spatiotemporal neurostimulation 20

1. Introduction 23

2. Methods 24

2.1. Neural Stimulation: Setup and Experiment Description 24

2.2. Gaussian Processes for Bayesian Optimization 26

2.2.1. Gaussian Process Prediction 27

2.2.2. Practical example 28

2.2.3. Sequential Optimization 28

2.2.4. Hierarchical GP 28

3. Results 30

3.1. Single Event – Single Muscle 30
3.2. Two Event – Single Muscle (fixed Δt) 33
3.3. Two Event – Temporal Co-Activation (search Δt) 35
4. Conclusion 37
5. Discussion 37
Acknowledgments 39
Chapitre 4. Conclusion 40
Bibliography 42
Appendix A. Article1 Extra Material ..... A-i
A.1. Response Surface to double-event, temporal co-activation where we search Δt A-i
A.2. Different Two Event, Single Muscle (fixed Δt) ..... A-ii
A.3. Effect of UCB acquisition function's exploration parameter k ..... A-iii
Appendix B. Conditioning a Multivariate Gaussian Distribution ..... B-i
B.1. Conditioning ..... B-i
B.2. Block Matrix Inverse ..... B-ii
B.3. Back to Conditioning ..... B-iii

Liste des figures

2.1 Sensorimotor Cortex and its Divisions by Pancrat which is licensed under CC BY-SA 3.0 4

2.2 Neuroscience methods and their spatiotemporal resolution. Reprinted with permission from Springer Nature Customer Service Centre GmbH: Springer Nature, Nature Neuroscience. Putting big data to good use in neuroscience. Terrence J Sejnowski, Patricia S Churchland, J Anthony Movshon., Copyright © 2014, Springer Nature (2014) 5

2.3 Homunculus 7

2.4 (Rectified) EMG signal after a motor cortex stimulation (700 ms) 8

2.5 Experimental design on a Cebus Monkey 8

3.1 Stimulation optimization as a cartoon 9

3.2 Gaussian Process regression example. Input space is shown on the left, which consists of 10 channels. Output space is EMG responses, shown in middle. We take the max of these responses to get a scalar output, on which we can fit a Gaussian Process, as shown on the right. 10

3.3 Procedure for Bayesian Optimization using Gaussian Process 13

3.4 GP graphical model. 14

3.5 Comparing the RBF Kernel and the Matern Kernel. 17

0.6 Experiment Setup and Data Description. 25

0.7 GP graphical model. 26

0.8 Max response (mean±std) per channel for different muscles 28

0.9 Gaussian Process predictions with different number of randomly queried points (in green) 30

0.11 Metrics for single-event GP model on Muscle 4 (see text for distance definitions) 31

0.10 Full data GP fit on Muscle 0. 31

0.12 Search space consisting of Muscle 4 responses to double-event, $\Delta t = 60$ ms stimulations, with the single-event Muscle 4 responses plotted left (first stimulation) and on top (second stimulation). Average max responses are written on the plots, with the green cells denoting the best stimulations: both ([1,1],[1,0]) and ([0,3],[1,0]) have very similar max responses of 0.033 and 0.034, respectively. Independent linearly additive assumption would have predicted ([1,0],[1,0]) as the best stimulation. 34

0.13 Metrics for the double event with fixed $\Delta t = 60$ GP model on Muscle 4 34

0.14 Objective and Results 36

A.1 Double-event search space ($\Delta t = 20$) ..... A-i

A.2 Double-event search space ($\Delta t = 40$) ..... A-ii

A.3 Double-event search space ($\Delta t = 60$) ..... A-iii

A.4 Mean Prediction accuracy vs. # of queries for Muscle 0 $\Delta t0$, with and without prior. For $\Delta t = 0$ here and in Fig. A.8, we used a symmetric kernel, since the order of channel stimulation does not matter when they are done at the same time. This reduces to search space to only 50 channels, which makes it easier for the vanilla GP. We still show the results for completeness, and to show that our algorithm does find the best stimulation pattern despite not necessarily outperforming the vanilla approach by much. ..... A-iii

A.5 Muscle 0 $\Delta t0$ ..... A-iv

A.6 Mean Prediction accuracy vs. # of queries for Muscle 0 $\Delta t10$, with and without prior. ..... A-iv

A.7 Muscle 0 $\Delta t10$. Notice that we accept both ([1,1],[1,0]) and ([1,1],[1,1]) as best stimulation pattern. This is because of an outlier in the ([1,1],[1,0]) stimulation, without which ([1,1],[1,1]) is the best stimulation pattern. . . . . A-v

A.8 Mean Prediction accuracy vs. # of queries for Muscle 4 $\Delta t0$, with and without prior. . . . . A-v

A.9 Muscle 4 $\Delta t0$ . . . . . A-vi

A.10 Muscle 4 $\Delta t0$. We see that higher k values encourage exploration, whereas lower values of k encourage exploitation, at the cost of sometimes getting stuck in local minima (see upper left graph without prior and $k = 2$). Higher k value seems better for the vanilla GP, which needs to explore the space a lot, having started with no mean prior, whereas the hierarchical GP, which already has information about the space, performs worse with higher k values. For example, the hierarchical GP with $k = 6$ in the bottom right takes longer to converge because it is wasting time exploring regions that it does not need to. . . . . A-vii

Remerciements

Je tiens a remercier Guillaume Lajoie, pour m'avoir accepté dans son groupe à l'improviste, avoir eu l'idée originale pour ce superbe projet, et m'avoir guidé dans les moments difficiles; mes parents, pour m'avoir encouragé et rempli mon frigidaire chaque deux semaines; et mes colocs, pour m'avoir supporté et remplit l'appart de leur énergie contagieuse!

Chapitre 1

Introduction

As Michio Kaku likes to say, the brain is the most complicated object that we know of in the Universe [1]. Uncovering the code which it speaks and its learning mechanisms are two of the most fundamental challenges awaiting breakthroughs in the 21st century.

Unfortunately, its complexity still baffles us and leaves most of our analytical tools, and greatest minds, helpless. Paradoxically, we have great difficulty explaining the things that are most intuitive and unconscious to us. Moravec's Paradox, originally discovered in the 1980s when artificial intelligence researchers and roboticists were trying to develop human-like intelligence, describes this succinctly: "it is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility." This might explain why we have an utter fascination for displays of sensorimotor skills. According to Wikipedia, all of the 21 internationally broadcasted programs to have ever received an audience of more than 2 billion people were sports event, most of them being Olympic games or FIFA world cups. How the three pound piece of meat on our shoulder manages to control our body so effortlessly and efficiently, and in fact often optimally, has puzzled neurophysiologists for more than 200 years. Nikolai Bernstein, a self-taught pioneer in the field, coined the degrees of freedom problem: "It is clear that the basic difficulties for co-ordination consist precisely in the extreme abundance of degrees of freedom, with which the [nervous] centre is not at first in a position to deal." Although many solutions have been proposed over the years, a consensus is still far from having been reached.

Brain-Computer Interfaces (BCI) have become an indispensable tool towards this goal. Not only do they provide an ability to record from a large number of individual neurons in targeted areas of the brain, but they also permit electrical stimulation of neuronal regions,

promoting us scientists from a simple observer role to a much more involved and active role. We can finally test for causal relationships between brain regions and physiological and behavioral responses. However, with great power comes great responsibility, and in the case of BCI, this great responsibility is analyzing the massive deluge of data that we are now able to generate. Understanding this data will require the development of new mathematical and statistical techniques. And with this great goal in mind, I have tried to make a small dent in this problem and bring a modest contribution to our community. More specifically, I have developed an automatic and optimal stimulation algorithm, that is able to find which stimulation pattern to use to evoke a given target motor response. Up to now, most neurophysiologists would manually decide where to stimulate during data collection, or have a preprogrammed extensive stimulation protocol. Recent stimulation technologies permit exponentially large number of stimulation patterns, rendering this manual approach infeasible. **Our computer science contribution is solving the algorithmic challenges associated with automating this search, by developing a hierarchical version of Gaussian Processes which permits effective online optimization over the stimulation search space.** Our neuroscience hope is that this technique will be useful to a wide range of neurophysiologists, with potentially slightly different goals than those we used to demonstrate the effectiveness of our approach.

A great second motivating factor of this work is its clinical applications. Indeed, although we strongly believe that BCIs will be fundamental to understanding the brain's function and learning mechanisms, fundamental scientific progress will be slow and take some time. However, research in the field has already proven itself useful through diverse clinical applications such as stroke recovery [2], neuroprosthetic implants [3], and all sorts of motor system and cervical injuries [4, 5]. According to the World Health Organization, roughly 15 million people each year suffer from debilitating motor system injuries such as spinal cord trauma and strokes [6, 7]. But these people can be helped, and have been helped. People who have completely lost the ability to control limbs can regain some mobility through robotic limbs and exoskeletons [3], and others who have only lost partial function of limb control can regain a great amount of control through targeted spinal cord stimulation [8]. A lot more has been done and is known with non-human primates [9, 10, 11], and the general view is that this knowledge will eventually transfer to human species [12].

Chapitre 2

Studying the Motor System with Brain-Computer Interfaces (BCI) and Electromyograms (EMG)

Neuroscientists analyze the brain at different levels of resolution, from systems and networks, down to neurons, synapses, and even individual molecules. In this work, we are interested in the motor system, starting from the motor cortex (Fig. 2.1), and going all the way down to the limbs. More precisely, our interest lies in the mapping between electrical signals in primary motor cortex (M1) and forelimb muscle activity.

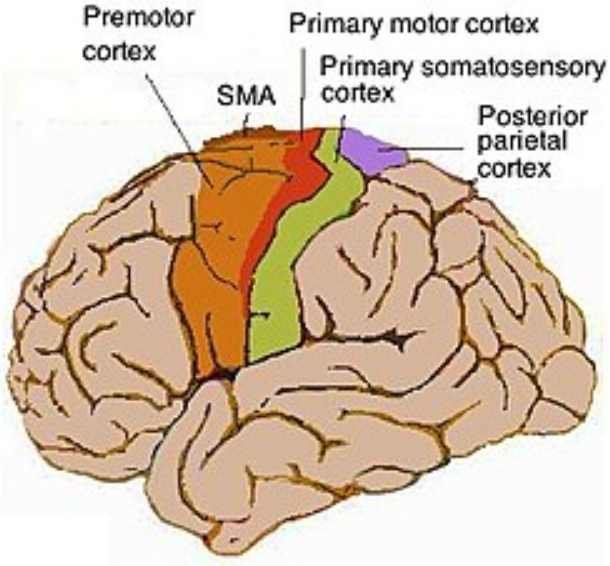


Fig. 2.1. Sensorimotor Cortex and its Divisions by Pancrat which is licensed under CC BY-SA 3.0

Before delving deeper into our specific setup, we need to explain the different hardware technologies and methods used to both stimulate and record from the brain. These methods are usually distinguished by spatiotemporal resolution, as in Fig. 2.2. The different techniques permit probing the central nervous system anatomically, at their respective spatial resolution. Temporal resolution on the other hand permits a functional analysis of circuits, providing a handle on the dynamics that happen over time.

Cognitive neuroscience has been making a lot of progress in recent years, thanks to general complementary techniques. Electroencephalography (EEG) gives a low spatial, high temporal window into the brain, and functional magnetic resonance imaging (fMRI) gives the exact opposite window, with a high spatial, low temporal resolution. Such techniques now allow us to ask questions about the brain as a whole, and interactions between different parts of the brain.

The focus of our research however, is on a much smaller scale. First of all, EEGs and fMRIs are purely recording devices that can read the neural code and attempt to decode it. We, on the other hand, are more interested in stimulating the brain, and hence probe its circuits causally. For this, we decided to work with a microstimulation paradigm in a monkey motor cortex.

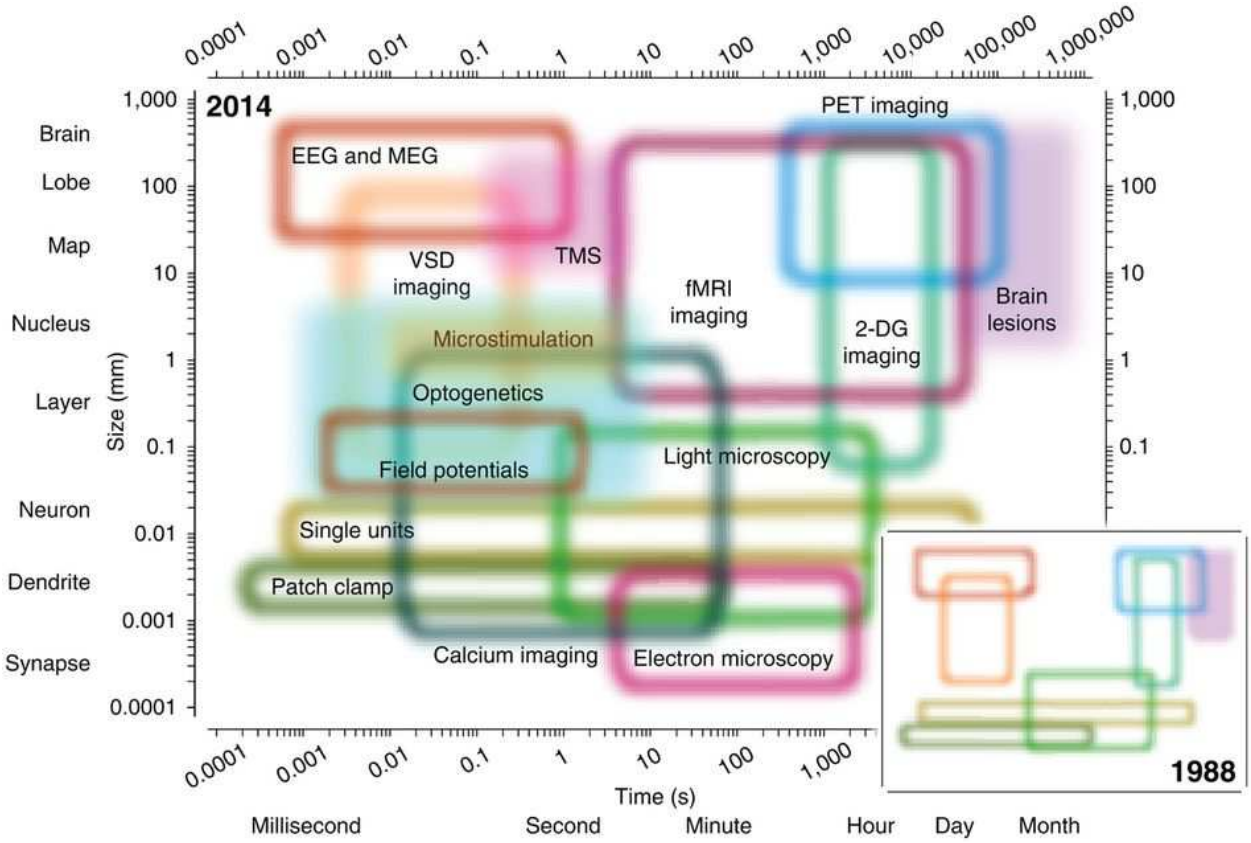


Fig. 2.2. Neuroscience methods and their spatiotemporal resolution. Reprinted with permission from Springer Nature Customer Service Centre GmbH: Springer Nature, Nature Neuroscience. Putting big data to good use in neuroscience. Terrence J Sejnowski, Patricia S Churchland, J Anthony Movshon., Copyright © 2014, Springer Nature (2014)

The set of experiments in the main article of this thesis were conducted in a male adult capuchin monkey. The experimental protocol followed the guidelines of the Canadian Council on Animal Care and was approved by the *Comité de Déontologie de l'Expérimentation sur les Animaux of the Université de Montréal*. The monkey was food restricted approximately 12 hours prior to each recording session. Between recording sessions, the monkey was group housed and supplied with food and water *ad libitum*.

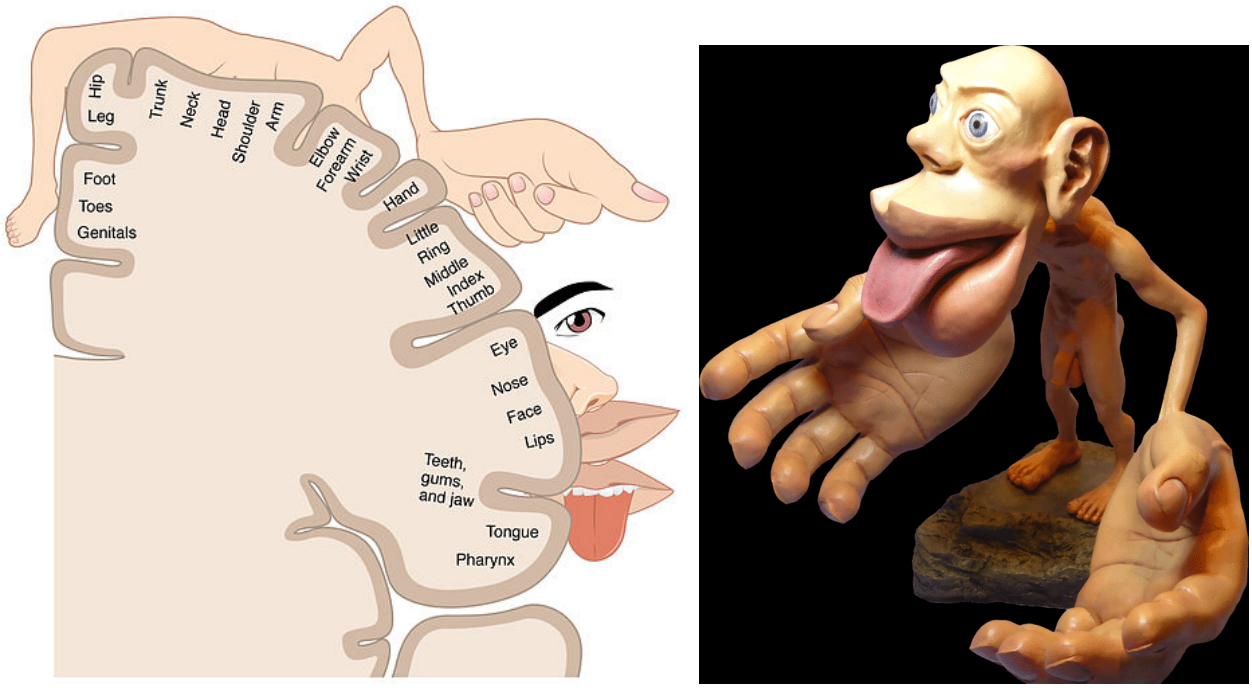
For stimulation, our hardware of choice was the Utah microelectrode array (see Fig. 2.5 on the right), so called because it was developed at the university of Utah. This array, consisting of 96 channels (10x10 but the corners are not usable), requires surgery in order to be implanted inside the skull. Once in place, it permits electrical stimulation of 96 small regions across a range of roughly 16mm^2. Depending on the density of nearby neurons, a single channel will stimulate somewhere between 10-10000 neurons. This is perfect for testing connections between different parts of the brain, and getting a coarse-grain view of the function of a specific brain region. A single moderate current stimulation permits a subtle jerk of the limbs of the monkey.

This has given us the so called cortical homunculus map of the somatosensory and motor regions of our cerebral cortex (Fig. 2.3a), originally discovered by Wilder Penfield [13] in Montreal, as well as the humoristic (yet scientifically accurate) depiction of our sensitivity to different body parts (Fig. 2.3b) that accompanies it. This is still a very coarse-grain depiction though, and is still contended by some as not being the entire picture [14]. All of this to say that new technologies, and new algorithms and analysis methods will be required to completely understand how the motor cortex exactly controls the body.

Up until recently, the 96 channels of the Utah array were the most available to us. However, Elon Musk's new company, Neuralink, has recently announced their work on a 1000 channel technology [15], which permits both stimulating and recording and which they plan to test on humans by the end of 2020. This massive increase in available data makes it that much harder for neuroscientists and surgeons to understand their experimental results. What interests us is refining and better understanding the role of motor cortex in controlling movements through cortical stimulations. With regards to this, the space of stimulation parameters has now grown exponentially large: amplitude of stimulation current, duration of stimulation, temporal pattern in stimulation, number of channels to stimulate synchronously, etc.

This is where our algorithm comes in. We want to automate the search process over the space of stimulation parameters. In essence, we want to answer questions such as "What is the best stimulation pattern to move the monkey's fourth finger?", "Given that the monkey has suffered a stroke, are there channel combinations that are still working and could help it regain some motor function?", "Is there a way to chain stimulations so as to make an anesthesized monkey walk again?", etc. The goal of our article is to formalize these questions and turn them into a working algorithm, but in order to understand our work, we need to explain one more hardware instrument.

In order to "optimize" a stimulation, we need a quantitative performance measure. In our case, we have decided to focus on the first question above, that of maximizing the hand movement. But in order to quantify this hand movement precisely, we would need a camera and advanced computer vision algorithms. Furthermore, hand displacement might actually depend on its original positioning, finger placement, muscle tension, etc., rendering the



(a) Somatosensory and motor cortical homunculus by OpenStax College, licensed under CC BY 3.0

(b) Sensitivity to-scale of our body parts by Mpj29, licensed from CC BY-SA 4.0

Fig. 2.3. Homunculus

optimization problem not well-defined. So instead, we have decided to use electromyograms (EMG) that were implanted in the monkey's forelimb muscles (see Fig. 2.5 on the left). Our collaborators, Marco Bonizzato and Numa Dancause, have instead directly implanted electromyograms in different arm and wrist muscles of the monkey (see the article, section Neural Stimulation: Setup and Experiment Description for more details). This gives us direct access to electrical activity, in volts, of specific muscles. Fig. 2.4 shows the rectified (filtered and absolute valued) signal of a forelimb muscle after a short pulse train stimulation of primary motor cortex.

With this information in hand, the reader should be able to understand the neuroscience details and experimental design mentioned in the article, which are related to the technologies depicted in Fig. 2.5.

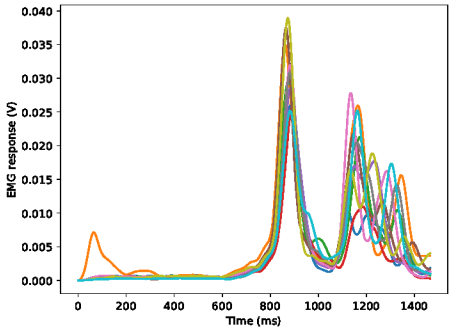


Fig. 2.4. (Rectified) EMG signal after a motor cortex stimulation (700 ms)

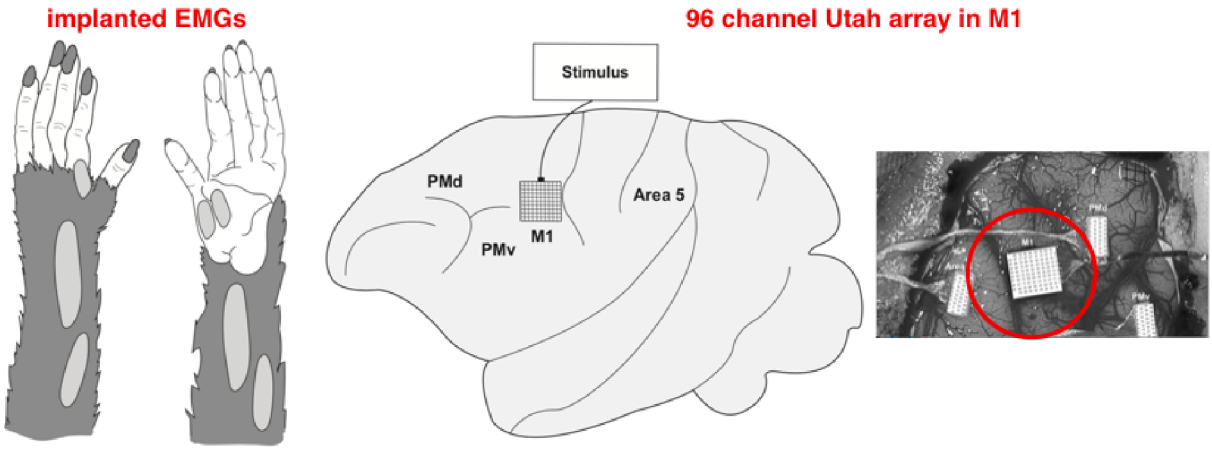


Fig. 2.5. Experimental design on a Cebus Monkey

Chapitre 3

Gaussian Processes and Bayesian Optimization

In this section we explain why we have decided to use Bayesian optimization (BO) and Gaussian Processes (GP). If we look more closely at the problem that we are trying to solve (Fig. 3.1), we see that at its heart all we are doing is learning a function, f, which maps between stimulation (input), and EMG response (output). Some amount of information is known about primary motor cortex (M1) [16, 13] and spinal cord [8], but definitely not enough to model the highly non-linear mapping between M1 stimulations and muscle outputs in any sort of sufficient detail. At least, we know where to place our array in M1 so that the stimulations will make the targeted limb move (see Fig. 3.2a). But beyond this we need to

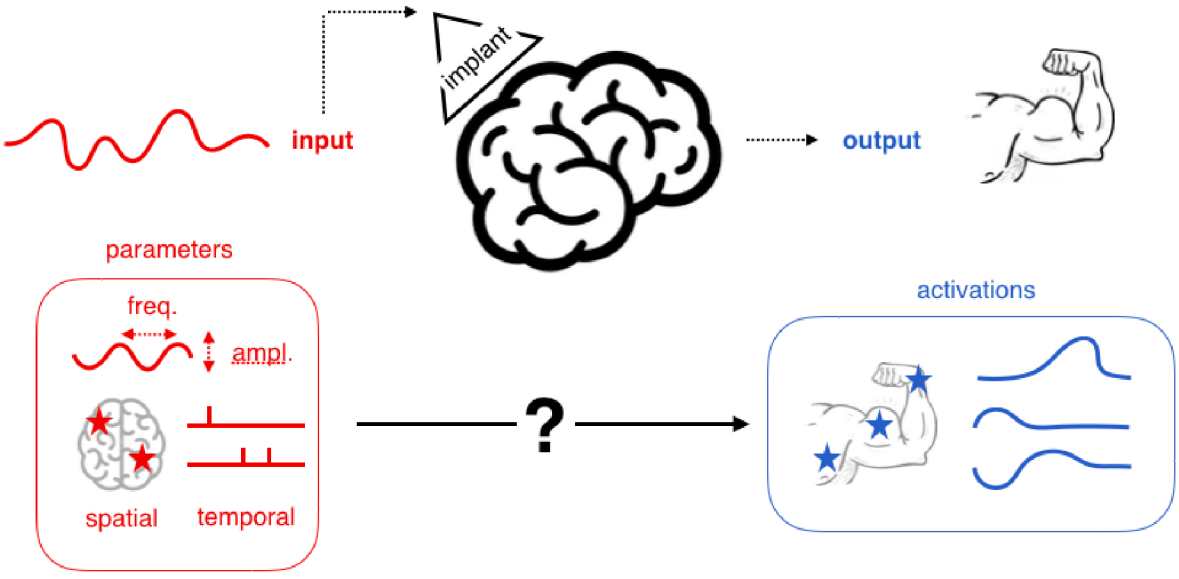
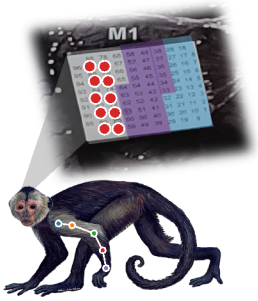


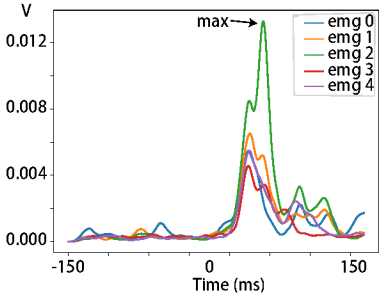
Fig. 3.1. Stimulation optimization as a cartoon

treat the function as a black box, and use statistical techniques to learn the mapping. Then, because our ultimate goal is to find an "optimal" stimulation, where optimal is defined with respect to some scalar function of the EMG responses (often, we will just want to get an EMG response with a high maximum voltage value. See Fig. 3.2b), we can bypass learning the EMG responses and simply learn the function from stimulation to this cost function.

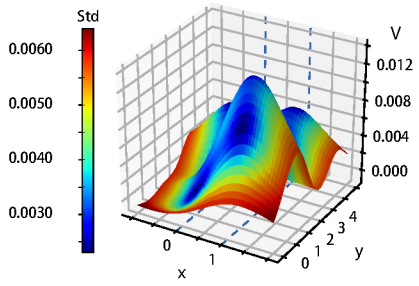
With this in mind, data considerations become important in determining which function approximation to use. Someone with access to a lot of data would and should consider using a neural network model [17]. In our case however, we want our algorithm to work online and to be data efficient: our clinical sessions are limited to a few hundred stimulations, which is not enough to train a neural network, let alone do it online. One important reason for this is clinical applications. It has been shown that stimulating pathways that are still functional helps restore movement after stroke or cortical injury [18]. But stimulation causes muscle fatigue, so we need to quickly find the pathways that are still functional. And if we actually want to optimize movement (as opposed to recovery for clinical applications), then we have to do so before the muscles get too tired. For this reason, we have decided to opt for Gaussian Process regression, which in essence is nothing but Bayesian non-linear regression, where the



(a) *Cebus apella* with Utah Array (M1) and implanted muscles (not real locations). Stimulation search space restricted to 2×5 grid in red.



(b) EMG responses (rectified) evoked from a single train delivered to a single-channel [1,1]. Arrow shows the max of EMG 2, thus $C(r_2([1,1])) \approx 0.013$.



(c) GP fit on EMG 2 for single-electrode stimulation. Surface shows GP's mean $\bar{f}$, and color shows standard deviation.

Fig. 3.2. Gaussian Process regression example. Input space is shown on the left, which consists of 10 channels. Output space is EMG responses, shown in middle. We take the max of these responses to get a scalar output, on which we can fit a Gaussian Process, as shown on the right.

non-linearity comes from a fixed basis function (also called kernel; see Sec. 3.0.1 for the formal mathematical definition). GP regression interpolates (and extrapolates) between observed data points according to this kernel, and furthermore gives a measure of uncertainty (standard deviation of a Gaussian distribution) at every point, as shown in Fig. 3.2c. Because our inputs are spatially distributed, we can use a simple Gaussian basis function, which works well for the single stimulation case. To generalize to spatiotemporal stimulation patterns (multiple channels with varying time delays in between) however requires modifications, which is where our main contribution lies. We will use this section to introduce the basic mathematical details of GPs and Bayesian Optimization, which should be enough to understand our article's contribution.

With the GP regressor described above, we now have an estimation of the loss surface on the stimulation parameter space. Furthermore, it is a probabilistic estimate, which means that we also have access to a measure of uncertainty. We can use this information to decide where to query next! This is the fundamental idea behind Bayesian optimization: we want to find the global optimal of a black-box function, and we do so through a sequential design process. There are many other global optimization techniques, evolutionary algorithms being a famous example, but Bayesian optimization has proven very effective on a number of problems lately [19, 20, 21], and it permits a sequential search, which is what we need for efficient online learning (evolutionary algorithms, on the other hand, require many parallel searches). Note that Fig. 3.2c is the result of a GP fit on many data points (we can tell from the very low uncertainty - blue color - at each of the 10 channels). Bayesian Optimization would then take this surface and decide on which of the 10 channels to stimulate next, so as to balance exploitation (high chance of having a high response) and exploration (gathering information about other channels).

We now delve into the mathematical details. We consider electrical stimulation signals that are composed of discrete events (e.g. single electrical pulses or short pulse trains) that can be delivered to one of N channels. A stimulation containing k events is a tuple $s_k = (n_1, \dots n_k, a_1, \dots, a_k, \Delta t_1, \dots, \Delta t_{k-1})$ where $n_i = 1 \dots N$ indicates the channel of the i^{th} event, a_i its amplitude, and Δt_i is the inter-event interval between events i and $i+1$. Each s_k generates a noisy pattern of EMG activity $g(s_k)$. Our goal is to optimize an objective

function $C(g(s_k))$. Here C is flexible; it can be extracting the maximum output of a single EMG, or measuring a distance between evoked pattern $g(s_k)$ and a target pattern g_{target}.

Hence, we want to find

$$\arg\max_{s_k} C(g(s_k))$$

where the argmax can be replaced by argmin if we want to minimize a distance function instead of maximizing an amplitude. This is a very well studied problem in optimization. However, a few considerations special to our problem naturally lead to using Bayesian optimization. These are

(1) We are optimizing over both discrete (n_i) and continuous (a_i, Δt_i) variables.
(2) The function $C(g(s_k))$ that we are optimizing is a black box (it is expensive to evaluate, and we do not have access to derivatives).
(3) The exploration needs to be as fast as possible (exhaustive search is to be avoided for clinical reasons, as mentioned above)
(4) Because of cortical plasticity, electrode displacement, and muscle fatigue, the EMG responses will change over time. We want to be able to track and adapt to these changes online.

3.0.1. Bayesian Optimization using Gaussian Processes

Bayesian optimization is a natural fit for this problem. It is a response surface (also called surrogate function) approach to global optimization, which means that at each iteration, it constructs a response surface that is meant to approximate the function being optimized (that we only have access to through a few datapoints), and then queries this function at the maximum of the response surface. The way it constructs the surrogate function is by treating the unknown function f it is trying to optimize as a random function and placing a prior over it. This prior dictates attributes of the function such as smoothness and speed of oscillation. The response of the function at the queried points so far is then treated as data, from which we get a posterior distribution over possible functions. We then use an acquisition function to turn this posterior distribution into a surrogate function, which we can maximize with deterministic optimization methods to find the next query point. Gaussian Processes are one way to model the random function, and the method that we use. Other function approximators such as trees [22] and random forests [23], but GPs often prove to be

better [20, 23, 21]. Furthermore, Bayesian optimization using Gaussian processes takes care of the four considerations above:

(1) Gaussian Processes are a continuous model, but after having built the surrogate function, we can use integer programming optimization to maximize it over integers only (for the discrete variables).

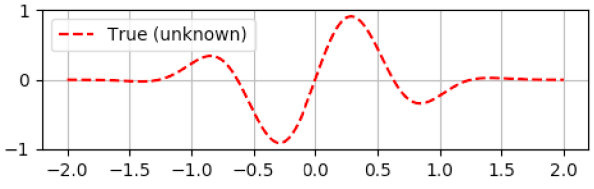
(2) Gaussian Processes do not use any information about the function other than its response to query points (blackbox).

(3) The acquisition function that we use to build the response surface has parameters to control the exploration vs. exploitation tradeoff (see below).

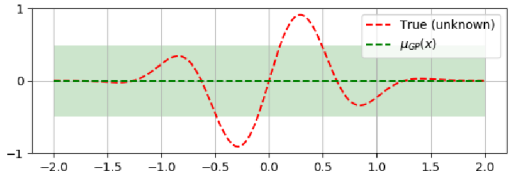
(4) The basic framework for Gaussian Processes assumes a fully-observed deterministic function. However, we can extend it to work with additive Gaussian noise observations $y = f(x) + \epsilon$ relatively easily (see below). It is also possible to extend it with an arbitrary stochastic function [21] (not just additive Gaussian noise), but it makes the analysis and numerics very non-trivial, and has not proven necessary for our dataset.

Our presentation will follow the procedure followed by all Bayesian optimization algorithms, as outlined in Fig. 3.3, which we will further specify to our case of using Gaussian processes.

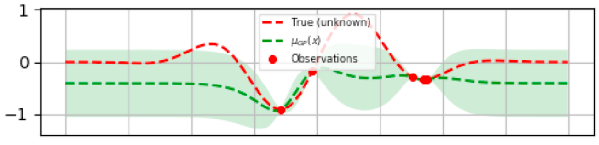
Gaussian processes have a long history, and have been used in fields as diverse as physics, signal processing, geostatistics, and machine learning. Hence, there are many different



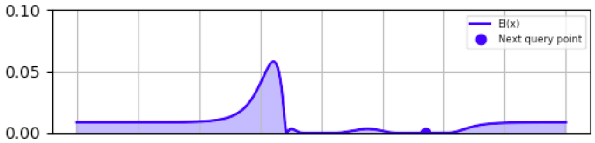
(a) Noise-free function to minimize.



(b) Prior over function (variance in green; covariance structure not visible from drawing)



(c) Posterior after 5 datapoints.



(d) Acquisition function, which we maximize to get next query point.

Fig. 3.3. Procedure for Bayesian Optimization using Gaussian Process

equivalent formal definitions for them, some of which are more useful for theoretical analyses and proofs, and others more beneficial to numerical computations and applications. Because of space concerns, we will only describe the computationally useful version, which is most often the only one known to machine learners, and which is sufficient to understand the language in our article.

Definition 3.0.1. A **Gaussian Process** is a stochastic process $\{X_t; t \in T\}$ such that for every finite set of indices $t_1, \ldots, t_n$ in the index set T,

$$X_{t_1,\ldots,t_n} = (X_{t_1}, \ldots, X_{t_n})$$

is a multivariate Gaussian random variable.

Remark. Note that the term *process* originally comes from the index set T being time, or $\mathbb{R}$. However, T can be much more general. For example, for the single stimulation case above, T is $\mathbb{R}^2$ because the function we are trying to model is over the utah array 2D grid.

Gaussian processes are essentially an infinite dimensional generalization of Gaussian random variables. Gaussian distributions are used to model random **variables**, multivariate Gaussian distributions are used to model random **vectors**, and Gaussian Processes are used to model random **functions**. Intuitively, the definition above just states that a random function is Gaussian if any finite sample from it is a multivariate Gaussian random vector.

We can turn this definition into a very effective computational tool, which is used throughout machine learning [24]. We do this by parameterizing the finite samples of the

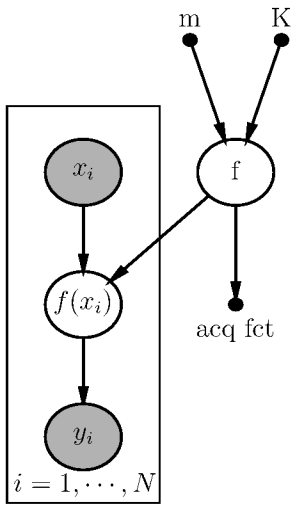


Fig. 3.4. GP graphical model.

Gaussian Process by two functions, $m : T \to X$ and $K : T \times T \to X$ (restricted to be positive definite), called the mean function and the kernel function, respectively. These play roles analogous to the mean and variance parameters of the Gaussian distribution. Indeed, just as we write $X \sim \mathcal{N}(\mu, \Sigma)$ for a random vector drawn from a multivariate Gaussian distribution, we write $f \sim \mathrm{GP}(m,K)$ for a random function drawn from a Gaussian Process (see Fig. 3.4).

In many practical cases when we don't have a priori information about the underlying function f, we assume $m \equiv 0$ (Fig. 3.3b). Then, by the above definition of Gaussian Processes, given a finite number of training data points $\boldsymbol{x} = (x_1, \ldots, x_n)$ and their response $\boldsymbol{f}$, plus a finite number of test data points $\boldsymbol{x_*}$ whose responses $\boldsymbol{f_*}$ we would like to predict (note here that $\boldsymbol{f}$ and $\boldsymbol{f_*}$ are not functions but vectors. We use the shorthand $\boldsymbol{f} = (f(x_1), \ldots, f(x_n))$, and similarly for $\boldsymbol{f_*}$), we get a Multivariate Gaussian

$$\begin{pmatrix} f \\ f_* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} K(x,x) & K(x,x_*) \\ K(x_*,x) & K(x_*,x_*) \end{pmatrix}\right)$$

and we can get our prediction (posterior, Fig. 3.3c) for $\boldsymbol{f_*}$ by simple conditioning of this MVN (see appendix B) distribution:

$$f_*|x_*, f, x \sim \mathcal{N}\left(K(x,x_*)K(x_*,x_*)^{-1}f, K(x,x) - K(x,x_*)K(x_*,x_*)^{-1}K(x_*,x)\right)$$

We can see that although the formal definition of the kernel function is intuitive, being the covariance of the underlying Gaussian process ($k(x,y) = \mathbb{E}_{f\sim GP(m,k)}[(f(x) - m(x))(f(y) - m(y))]$), its impact on the posterior, and hence future predictions, is much less intuitive. One way to reason about this prediction is to see it as a linear combination of previous observations f, where this linear map ($K(x,x_*)K(x_*,x_*)^{-1}$) is parameterized by the choice of kernel K and the relative position of new points x_* we are trying to predict. This view, which I believe is the most intuitive, has been called linear smoothing in the literature [25]. This view also permits relating Gaussian processes to other nonparametric estimation techniques from frequentist statistics, such as kernel ridge regression (KRR) and support vector machines (SVMs). Such techniques are closely related to Gaussian processes; for instance, the estimator of KRR is identical to the posterior mean of GP regression. Nonetheless, the theory and philosophy behind these approaches remains very different, although there are advances in bridging the gaps [26].

The last (very important) detail is the choice of kernel. There are really no restriction for a two-argument function to be a kernel, other than that it be positive semidefinite. Hence, a gazillion different kernels have been developed over the years in different fields, for different applications. Fortunately, there is some order to this zoo, and we leave the interested reader to a recent review on general classes of useful kernel functions [27]. The most often used (and often, unfortunately, for the wrong reasons) kernel is the Gaussian kernel, also called radial basis function (RBF) kernel:

$$K(x,x') = \sigma^2 \exp(-\frac{||x - x'||^2}{2l^2})$$

where l is the lengthscale parameter, which controls the covariance between points (a short lengthscale implies that a given observation of the underlying function will only affect other very nearby points, at an exponential rate controlled by l), and σ^2 is the prior variance parameter (note $\text{Var}[f(x)] = K(x,x) = \sigma^2$). Note that the "Gaussian" from Gaussian kernel has nothing to do with the Gaussian from Gaussian process, which explains why Neil Lawrence is advocating for renaming it exponentiated quadratic kernel.

In our case, because the motor cortex is spatially organized (Fig. 2.3a), with local regions having similar representations and effects, using an RBF kernel makes sense. However, Michael L. Stein, in his book Interpolation of Spatial Data [28], has argued that the infinite differentiability of the RBF kernel is a big problem for physical processes (such as geostatistics, his main field; we argue also the brain) since observing only a small continuous fraction of space is enough to infer the whole function. In simpler terms, if we could, in an imaginary world, stimulate a continuous region of 1mm^2 in the brain and know the EMG response at every point within this small region, we would also know the response of not only the whole Motor Cortex, but also the entire brain. He thus proposed the Matern kernel as a generalization of the RBF kernel [29]

$$C_\nu(x,x') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{||x - x'||}{l} \right)^\nu K_\nu \left(\sqrt{2\nu} \frac{||x - x'||}{l} \right)$$

where Γ is the gamma function, K_ν is the modified Bessel function of the second kind, l is the lengthscale parameter analogous to that of the RBF kernel, and ν is a non-negative parameter of the covariance. In this form, it is literally incomprehensible, but we note that the Matern kernel is actually a family of kernels, where each ν parameterizes a different

kernel, and as $\nu \to \infty$, it converges to the RBF kernel. There is also has a much simpler form when ν is a half integer, for which it can be expressed as a product of an exponential and a polynomial. In all of our experiments, we use $\nu = 5/2$, for which it simplifies to

$$C_{5/2}(x,x') = \sigma^2 \left(1 + \frac{\sqrt{5}||x - x'||}{l} + \frac{5||x - x'||^2}{3l^2}\right) \exp\left(-\frac{\sqrt{5}||x - x'||}{l}\right)$$

Fig. 3.5 gives a visual comparison of the RBF and the Matern kernels.

If we come back to the posterior prediction equation, its underlying assumption is that the observed function values are noiseless. This assumption however will ever only be the case for example when we are using Gaussian Processes to approximate deterministic computer programs, but in our case, and most cases, our observations will be noisy. We model this by saying that the observed variables $\boldsymbol{y}$ are related to $\boldsymbol{f}$ by $\boldsymbol{y} = \boldsymbol{f} + \epsilon$, where $\epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbb{I})$ is drawn from some other independent multivariate Gaussian distribution. This is the simple case of additive Gaussian noise that we were referring to earlier. Although a simplification of reality, since noise is not often perfectly Gaussian (in fact, noise in biology often tends to be Poisson), it turns out to perform very well. In this noisy case, we get instead

$$\begin{pmatrix} f \\ y_* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} K(x,x) & K(x,x_*) \\ K(x_*,x) & K(x_*,x_*) + \sigma^2\mathbb{I} \end{pmatrix}\right)$$

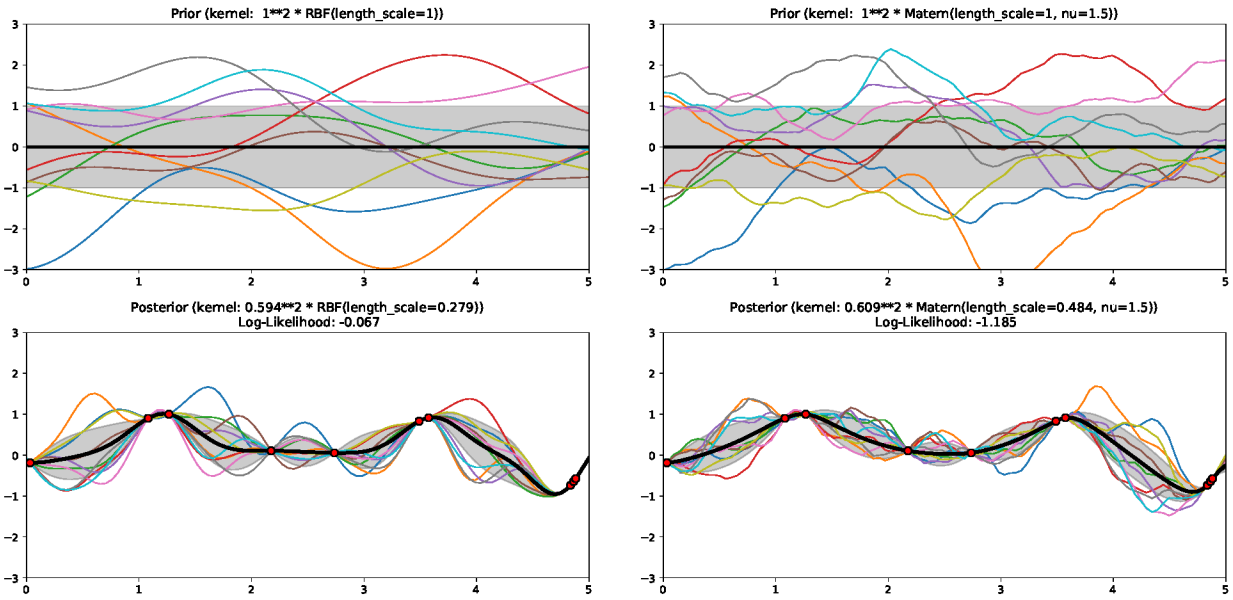


Fig. 3.5. Comparing the RBF Kernel and the Matern Kernel.

and everything else works as above.

We now have access to the machinery necessary for creating a probabilistic estimate, given some dataset, of the function we are trying to optimize. But our goal is to create an online algorithm which finds this optimal stimulation pattern efficiently. That is, how can we best make use of a given finite budget of N queries, so as to find the maximum value. That is, we need a procedure for, given a probabilistic estimate of the function (for example, the posterior after 5 datapoints in Fig. 3.3c), deciding where to next query the function, and hence augment our dataset. That is, we are in an active learning setting, where we can choose our dataset, as opposed to traditional machine learning where the dataset is given to us.

The traditional way of doing this is to choose N random points. This surprisingly actually works very well [30]. But in our case, we have another constraint, which is that every stimulation counts. As mentioned in section 2, for clinical applications, effective stimulations help restore function after cortical injury. Hence, if we spend half of our random queries querying channels that aren't functional anymore, we lose a lot of efficacy towards restoring patient movement. Hence, our objective, instead of simply being to find the optimal stimulation pattern **after** N queries, it is also to accumulate as many "as optimal as possible" queries on the way there (for those familiar with the jargon, this is a bandit problem rather than a Bayesian optimization problem, although the distinction between these is becoming blurred by recent theory [31]).

In choosing the next query point, we thus need to balance between exploitation (querying a point that we know, or expect, to have a high value), and exploration (gathering information about channels that might potentially have high value, even though we currently don't know). We thus need to define an acquisition function (Fig. 3.3d), which will take our stochastic Gaussian process posterior, and return a deterministic function, indicating for every point in the domain, it's "query value" (essentially balancing between exploration and exploitation). We can thus find the maximum of this function, and use this as our next query point. Many acquisition functions have been defined in the field of Bayesian optimization, and we leave the interested reader to go read about them in a proper tutorial [19]. Although there exist certain cases where one would prefer a certain acquisition function over another, in practice, they tend to perform very similarly. As the above tutorial mentions, the common saying is that proper modeling is more important than the choice of acquisition function. That is, if Bayesian

optimization is not working properly on some problem, it is in most cases more important to change the model (eg. change the kernel function, or noise model being used) rather than to change the acquisition function. Following this advice, we decided to use the very common Upper Confidence Bound (UCB) acquisition function [19]: $\text{UCB}(x) = \mu(x) + k\sigma(x)$. Notice that k is the parameter which explicitly modulates the trade-off between exploration (high k) and exploitation (low k).

This completes the machinery necessary for optimizing a blackbox function online. With this in hand, the reader will be able to understand the few mathematical definitions in the article.

Article.

Hierarchical Bayesian optimization of targeted motor outputs with spatiotemporal neurostimulation

par

Samuel Laferriere[1], Marco Bonizzato[2], Sandrine Cote[2], Numa Dancause[2], and Guillaume Lajoie[3]

([1]) Computer Science Dept., Université de Montréal
Mila – Quebec AI Institute

([2]) Neuroscience Dept., Université de Montréal

([3]) Math & Stats Dept., Université de Montréal
Mila – Quebec AI Institute

Cet article a été soumis à la revue Transactions on Neural Systems & Rehabilitation Engineering.

The great majority of the work in the following article was done by me. Numa Dancause and Marco Bonizzato from the Neuroscience department provided the data and tested the algorithms on animals, and my supervisor Guillaume Lajoie provided the initial idea for the project, as well as guidance and useful comments when I faced difficulties.

Résumé. Le développement de techniques de neurostimulation pour évoquer des outputs moteurs est un domaine de recherche actif. Ces dernières s'avèrent un outil expérimental crucial pour explorer la computation dans les circuits neuronaux et trouvent aussi des applications dans les neuroprothèses utilisées pour aider à la récupération motrice après AVC ou lésion cérébrale. Concevoir des algorithmes permettant de dévoiler et de contrôler les mappings neurostimulation-moteur pose deux défis importants, liant ainsi les patterns spatiotemporels de stimulation neuronale à leur activation musculaire: (1) l'exploration des cartes motrices doit être rapide et efficace (une recherche exhaustive doit être évité pour des raisons cliniques et expérimentales) (2) l'apprentissage en ligne doit être suffisamment flexible pour s'adapter aux changements sur ces cartes. Nous proposons un algorithme de recherche de pattern de stimulation pour résoudre ces problèmes et en démontrons l'efficacité avec des expériences sur des primates non humains. Notre solution est un nouveau processus itératif utilisant l'optimisation bayésienne via des processus gaussiens sur des espaces de signaux de plus en plus complexes. Nous montrons que notre algorithme peut apprendre avec succès et rapidement des correspondances entre des schémas de stimulation complexes et des schémas d'activation musculaire évoqués, lorsque les approches standard échouent. Fait important, nous découvrons dans M1 des calculs non linéaires au niveau du circuit qu'il n'aurait pas été possible d'identifier avec les techniques de mapping classiques.
Mots clés : BCI · Stimulation Corticale · Processus Gaussien · Optimisation Bayesienne

Abstract. The development of neurostimulation techniques to evoke motor patterns is an active area of research. It serves as a crucial experimental tool to probe computation in neural circuits, and has applications in neuroprostheses used to aid recovery of motor function after stroke or injury to the nervous system. There are two important challenges when designing algorithms to unveil and control neurostimulation-to-motor correspondences, thereby linking spatiotemporal patterns of neural stimulation to muscle activation: (1) the exploration of motor maps needs to be fast and efficient (exhaustive search is to be avoided for clinical and experimental reasons) (2) online learning needs to be flexible enough to deal with occasional spurious responses. We propose a stimulation search algorithm to address these issues, and demonstrate its efficacy with experiments in the motor cortex (M1) of a non-human primate model. Our solution is a novel iterative process using Bayesian Optimization via Gaussian Processes on increasingly complex signal spaces. We show that our algorithm can successfully and rapidly learn correspondences between complex stimulation patterns and evoked muscle activation patterns, where standard approaches fail. Importantly, we uncover nonlinear circuit-level computations in M1 that would not have been possible to identify using conventional mapping techniques.
Keywords: BCI · Cortical Stimulation and Gaussian Processes and Bayesian Optimization

1. Introduction

Each year, over 15 million people worldwide suffer major debilitating motor system injuries such as spinal cord trauma [6] or stroke [7]. A promising approach to help restore movement applies targeted, artificial stimulation to motor structures of the nervous system, e.g. motor cortex [4], spinal cord [2], or the periphery [5] using brain-computer interfaces (BCI). Despite years of research, it is still not fully understood how complex movements are generated [32, 33, 34], and even less so how to regain control of these movements after injury. Nevertheless, there are often local spatial correspondences between neurostimulation of specific sites and targeted muscle activation. We want to leverage these in an optimal way. Previous work has shown that long-train stimulations in motor cortex can activate entire circuits of neurons, thereby producing complex movements [32, 33]. The challenge we address here is to specifically identify optimal stimulation signals to evoke *targeted* muscle co-activations, where pre-selected muscles are required to be activated at specific times.

New implantable devices which are microfabricated with many electrodes hold potential for such targeted spatiotemporal stimulation, yet existing control algorithms do not fully take advantage of them, generally relying on incomplete and manual mapping, and often single electrode stimulation. Our goal is to develop Bayesian optimization methods to learn optimal multi-electrode stimulation patterns. Effectively searching the space of possible spatiotemporal stimulation patterns (which can include duration, intensity, spatial ordering, etc.) is a complex task because of its combinatorial explosion in size. Exhaustive search is therefore impossible in practice, especially if algorithms are to be used on-line in clinical settings. Moreover, relationships between stimulation and output are noisy, and may change over time due to plasticity of neural circuits [16, 34]. Any method to identify stimulation protocols must be robust, and flexible enough to track such changes.

We propose a Gaussian Process (GP) based Bayesian Optimization (BO) approach[1]. This leverages acquired knowledge of muscle responses for single channel stimulations to build priors for *stim-to-muscle* maps for multi-channel stimulation patterns, where only nonlinear correction terms to a linear prior need to be learned. We refer to this process as *hierarchical* GP-BO since it relies on GP models fitted in lower dimensional spaces to initialize

[1]We make the data and some example code available at https://github.com/samlaf/hierarchical-gaussian-process.

and constrain ones in higher dimensional spaces, where sampling would be prohibitively costly. The advantages of recursively learning correction terms, rather than a complete map, are threefold: **(1)** Convergence to optimal stimulation requires fewer exploratory stimuli than direct optimization on the space of all signals. **(2)** The algorithm can be used online and adapts quickly to changes in neural dynamics. **(3)** Our method precisely learns the nonlinearities introduced by network dynamics, and can track the evolution of population codes throughout recovery, thus uncovering circuit-level computations.

The main goal of this paper is to describe a novel algorithm to rapidly find optimal stimulation patterns of intracortical microstimulation (ICMS), for a targeted motor output. To complement this algorithmic contribution, we demonstrate its efficacy with a basic experiment in a non-human primate model where optimal multi-electrode stimulation patterns are identified to evoke temporal muscle coactivations. This experiment is intended as a proof-of-concept for our algorithm and as such, uses a single monkey but validates our findings with multiple combinations of electromyographic (EMG) output patterns. We record spatiotemporal stimulation patterns combinations and their responses exhaustively, on several trials. We then perform explicit validation of our approach, with offline tests that are run several times. However, this exhaustive dataset is limited in spatial and temporal resolution by experimental constraints, but our algorithm is designed to be scaled up towards our goal of evoking even more complex targeted movements.

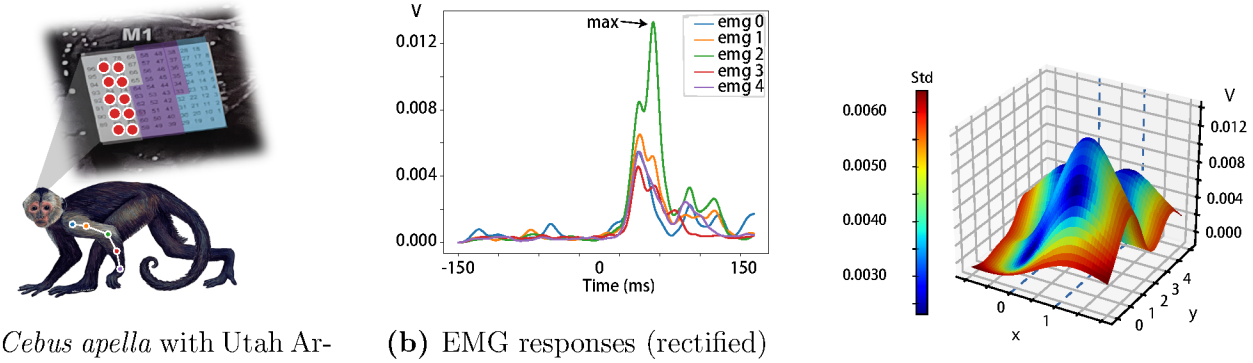
In the discussion, we outline the implementation and future use of our algorithm in online settings as well as circuit-level neural mechanisms present in M1 it uncovers.

2. Methods

2.1. Neural Stimulation: Setup and Experiment Description

The current set of experiments were conducted in a male adult capuchin monkey. The experimental protocol followed the guidelines of the Canadian Council on Animal Care and was approved by the *Comité de Déontologie de l'Expérimentation sur les Animaux of the Université de Montréal*. The monkey was food restricted approximately 12h prior to each recording session. Between recording sessions, the monkey was group housed and supplied with food and water *ad libitum*. Prior to the onset of data collection, a 96 channel Utah array was implanted in primary motor cortex (M1) and five different muscles of the forearm

and hand were chronically implanted to record EMG activity: *flexor carpi ulnaris, extensor digitorum communis, extensor carpi radialis, opponens pollicis and flexor pollicis brevis* (see monkey in Fig. 0.6a[2]). In each experimental session, we stimulated M1 by sending electrical pulse trains through one or many channels, and observed EMG responses (Fig. 0.6b). Our goal was to find the stimulation pattern, among a parametric family described below, that evoked a given target EMG response as best as possible.



(a) *Cebus apella* with Utah Array (M1) and implanted muscles (not real locations). Stimulation search space restricted to 2×5 grid in red.

(b) EMG responses (rectified) evoked from a single train delivered to a single-channel [1,1]. Arrow shows the max of EMG 2, thus $C(r_2([1,1])) \approx 0.013$.

(c) GP fit on EMG 2 for single-electrode stimulation. Surface shows GP's mean $\bar{f}$, and color shows standard deviation.

Fig. 0.6. Experiment Setup and Data Description.

Mathematically, we consider electrical stimulation signals that are composed of discrete events (single electrical pulses or short pulse trains) that we call *(stimulation) events* for generality. For our cortical stimulation experiment, we consider a two dimensional 2×5 grid that is circled in red in Fig. 0.6a and use stimulation events consisting of trains of 13 pulses of 30μA that last 0.2ms each, delivered at 330Hz (for a total of roughly 40 ms). Note the spatial configuration of channels is important for learning as Gaussian Processes make use of this information in their kernel distance function. In this article, we restrict our data collection to this small subgrid to allow extensive search of a two-electrode stimulation space formally defined in Sec. 3.3. With this current setup, the data collection protocol takes about an hour, during which the monkey needs to receive ketamine intravenously every 8 minutes. It would be hard to maintain the animal in an overall stable state for longer times, and this could change the data distribution.

[2]Royalty-free image from `unixtitan.net`

We denote each channel by discrete Cartesian coordinates $c = [x,y]$, $x \in \{0,1\}$, $y \in \{0,\ldots,4\}$. A stimulation containing k events is a tuple $s_k = (c_1,\ldots c_k,\Delta t_1,\ldots,\Delta t_{k-1})$ where c_i indicates the channel of the i^{th} event, and Δt_i is the inter-event interval between events i and $i+1$. In this experiment, we use trains of fixed intensity but power could be added to the stimulation parameters with the same formalism. Each s_k generates a noisy response pattern $r(s_k)$. In our case, we consider the (rectified) EMG responses of five muscles: $r(s_k) = (r_0(s_k),\ldots,r_4(s_k))$. Our goal is to optimize an objective function $C(r(s_k))$. Here C is flexible; it can be extracting the maximum output of a single EMG response (or combinations of EMG responses), or measuring a distance between the evoked pattern $r(s_k)$ and a target pattern r_{target}. In our case, $C(r(s_k))$ returns the maximum output of a single $r_i(s_k)$ or combinations of $r_i(s_k)$ for muscle synergies (see Results), in a window of 150ms following the first stimulus delivery. In general, $r(s_k)$ could also depend on time. We omit this for notation clarity. We want to find

$$\arg\max_{s_k} \mathbb{E}[C(r(s_k))]$$

where the expectation is needed because muscle responses are stochastic. The argmax can be replaced by argmin if we want to minimize a distance function instead of maximizing an amplitude.

In this article, we demonstrate and test our algorithm offline on the space of double-event stimulations s_2. To do so, we gathered an exhaustive dataset consisting of 10 trials per stimulation pattern (pairs of electrode) for each of these Δt: 0,10,20,40,60,80,100 ms, and sample from this data set to simulate online optimization. An online demonstration will be presented in a forthcoming publication.

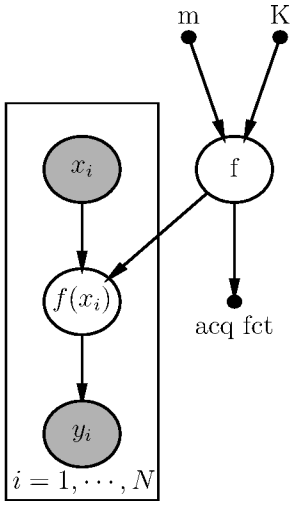


Fig. 0.7. GP graphical model.

2.2. Gaussian Processes for Bayesian Optimization

Given the constraints of our problem, namely that of black-box derivative-free global optimization under query constraints, Bayesian Optimization [19] is a natural fit. This provides uncertainty estimates that allow tracking and adapting online to

both signal delivery changes caused by the implant moving, and
structural changes in the underlying brain substrate.

BO constructs, at every iteration, a probabilistic surrogate to the function C being optimized, which is used to balance exploration and exploitation through the design of an acquisition function. It does so by treating the unknown function C as a random function and placing a prior over it. This prior dictates attributes of the function such as smoothness and frequency of oscillation. By conditioning on the so far observed responses of the function, a posterior distribution over possible functions is obtained, from which the algorithm can decide where to query next based on optimizing an acquisition function. Acquisition functions convert a probabilistic belief into a deterministic function that explicitly embodies the trade-off between exploration and exploitation. Following the current literature, we choose to model the random surrogate as a Gaussian Process [24], and use the *Upper Confidence Bound* [19] as acquisition function.

2.2.1. *Gaussian Process Prediction*

GPs are such that for a finite number of training data points $\boldsymbol{x}$ and their associated responses $\boldsymbol{y}$ (represented by the plate notation in Fig. 0.7), plus a finite number of test data points $\boldsymbol{x}_*$ whose response $\boldsymbol{f}_*$ we would like to predict, we get a Multivariate Gaussian

$$\begin{pmatrix} y \\ f_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(x) \\ m(x_*) \end{pmatrix}, \begin{pmatrix} K_y(x,x) & K(x,x_*) \\ K(x_*,x) & K(x_*,x_*) \end{pmatrix} \right)$$

where m and K are the mean and kernel functions associated to the GP, and

$$K_y(x_p,x_q) = K(x_p,x_q) + \sigma \mathbb{1}_{x_p=x_q} \tag{2.1}$$

where σ is the noise standard deviation parameter, which will be optimized along with K's parameters. We can get our prediction for $\boldsymbol{f}_*$ by simple conditioning on this Multivariate Normal distribution [24]:

$$\begin{aligned} &f_*|x_*, y, x \sim \mathcal{N}\left(\bar{f}_*, \mathrm{cov}(f_*)\right), \text{where} \\ &\bar{f}_* = m(x_*) + K(x_*,x)[K(x,x) + \sigma^2 \boldsymbol{I}]^{-1}(y - m(x)) \\ &\mathrm{cov}(f_*) = K(x_*,x_*) - K(x_*,x)[K(x,x) + \sigma^2 \boldsymbol{I}]^{-1} K(x,x_*). \end{aligned}$$

2.2.2. *Practical example*

Fig. 0.8 shows all of the single channel data. Note that different EMGs can have different optimal channels. EMG-4 is interesting because its max is found on channel [1,0], as opposed to all of the other EMGs whose max are found on channel [1,1]. Fig. 0.6c shows a GP fit on the data (20 responses per channel) from EMG-2, with its max found at [1,1] as required. An interesting point to notice is that the GP's standard deviation becomes larger (red) as we move away from the training data. This reflects the GP's uncertainty as to how the true function would respond at this point. Fig. 0.9 shows a GP being built with different number of initial random query points.

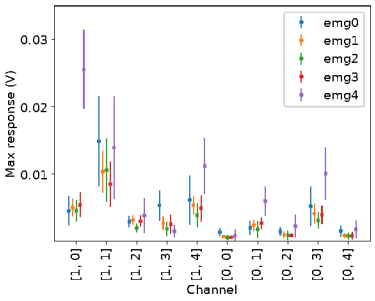


Fig. 0.8. Max response (mean±std) per channel for different muscles

2.2.3. *Sequential Optimization*

Given a GP and a set of possible points to explore, we use the Upper Confidence Bound (UCB) acquisition function [19]: $\text{UCB}(x) = \mu(x) + k\sigma(x)$ to identify the next query likely to maximize the objective. Notice that k is the parameter which explicitly modulates the trade-off between exploration (high k) and exploitation (low k). The performance of the algorithm is sensitive to this hyperparameter and tuning it by cross-validation or some other method will be necessary for portability across different kinds of neural interfaces. In most of our experiments, $k = 5$ performed best. However, when using an accurate prior mean (see sec. 2.2.4), sometimes reducing k to a value of 2 performed best, since less exploration is needed. We will discuss this more thoroughly in sec. 3.

The procedure for BO is as follows:

2.2.4. *Hierarchical GP*

We describe here the main (algorithmic) contribution of this paper. Our goal, as previously defined, is to find the multi-electrode stimulation pattern with the best response, where best is defined by the objective function. The challenge here is that the space of spatiotemporal multi-electrode patterns grows exponentially fast in the number of channels and stimulation events. For two-electrode stimulation on our 2×5 grid for example, there are 100 combinations

Algorithm 1: Bayesian Optimization

Result: Best Stimulation Pattern

Randomly pick m initial random pts and initialize Kernel hyperparameters;

while *haven't converged on single stimulation pattern* **do**

    Fit GP to current dataset;

    Compute Acquisition Function;

    next_stim = max(acq);

    Augment dataset with next_stim;

end

of channel pairs possible, without even considering different inter-event intervals Δt. The direct approach of training a GP on this space is not scalable, and does not take advantage of prior knowledge of motor circuit coding; namely, that motor outputs of spatiotemporal neural activations can often be decomposed (although not exactly) into individual neural-muscle correspondences [35]. We leverage this fact in a hierarchical approach where we use GPs fitted on lower dimensional stimuli spaces, to build priors for GPs in higher dimensional stimuli spaces.

More formally for the two-electrode space $s_2 = (c_1,c_2,\Delta t)$, if we write the single-electrode GP as $f_1(c) \sim \mathrm{GP}(0, K_1)$ then our prior on the two-electrode GP will be

$$f_2(c_1,c_2,\Delta t) \sim \mathrm{GP}(a_1 \bar{f_1^n}(c_1) + a_2 \bar{f_1^n}(c_2), K_2) \tag{2.2}$$

where $f_1^n := f_1|\text{Data}$ is the GP trained on the single-electrode data, $\bar{f_1^n}$ indicates its mean function, and K_2 is a standard Matern52 [24] multiplicative kernel which separates over time and space: $K_2\left((c_1^{(1)},c_2^{(1)},\Delta t^{(1)}),(c_1^{(2)},c_2^{(2)},\Delta t^{(2)})\right) = K_s\left((c_1^{(1)},c_2^{(1)}),(c_1^{(2)},c_2^{(2)})\right) K_t\left(\Delta t^{(1)},\Delta t^{(2)}\right).$

In short, our constructed prior is an independent, additive contribution from the two channels, factoring in the time delay Δt, which is also an explored parameter. We use the kernel in the two-electrode space to learn and correct the multiplicative, nonlinear difference from this prior. The weights a_1 and a_2 and the kernel hyper-parameters are optimized incrementally using BO after each new query. The same procedure can be recursively used to include more electrodes, although we present results only for the two-electrode case in

this paper. We show in the next section that important gains are obtained from using our method.

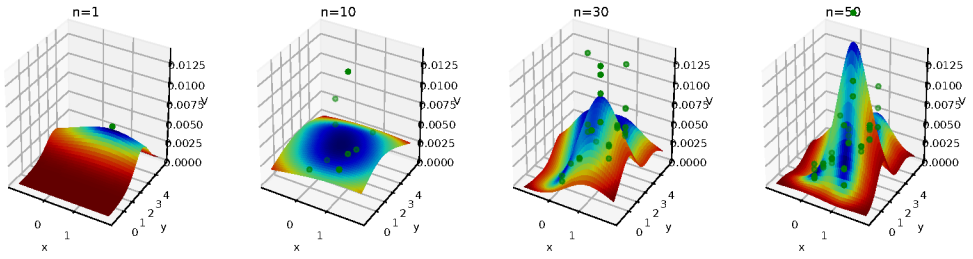


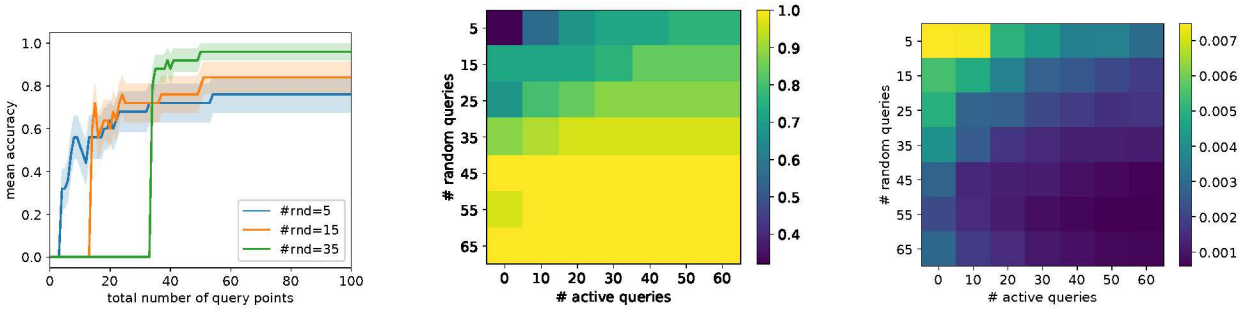
Fig. 0.9. Gaussian Process predictions with different number of randomly queried points (in green).

3. Results

The main goal of this paper is to provide a proof-of-concept, and show that Bayesian Optimization algorithms permit a viable approach to automate neurostimulation tuning. This is a desired improvement over an exhaustive parameter search, which might be infeasible in large search spaces, and often requires a human in the loop [20] to decide where to query. We propose a novel approach building on established GP optimization algorithms that relies on increasingly larger search spaces, leveraging learned structure from smaller ones. Here, we present basic experimental results on a single monkey, but several EMG readouts, thoroughly verified with exhaustive ground truth that demonstrates the efficacy of our method. In our view, the main and most interesting result, that of stimulating optimal muscle co-activations, is presented in section 3.3.

3.1. Single Event – Single Muscle

We start with a thorough analysis of the single-event stimulation space, to get an intuition for convergence times (in terms of number of initial random points and actively queried points) and introduce our metrics and visualizations. This step is crucial to our hierarchical method, as the GPs fitted here will serve as building blocks for the next level ones fitted on more complex spatiotemporal stimulation spaces.



(a) mean prediction accuracy (over 25 trials) as a function of total number of queries, for different initial random queries.

(b) mean prediction accuracy as a function of number of random queries and number of active queries.

(c) $l_{\max}$ distance to data as a function of number of random queries and number of active queries.

Fig. 0.11. Metrics for single-event GP model on Muscle 4 (see text for distance definitions)

We show results for an example Muscle (Muscle 0) in Fig. 0.10, and leave the results for other muscles to the appendix.

We can clearly see the difficulty with the standard GP likelihood model, which assumes i.i.d. noise (see eq. 2.1). As is common in biology, responses tend to follow poisson models, where the variance equals the mean, as we see in channel [1,1]. Thus, if the first responses to channel [1,1] are much below its mean, the GP will mistakenly optimize for a small σ, and might get stuck on a local minima, such as channel [1,4]. We could mitigate this by querying each of the randomly selected initial channels twice, but this amounts to a lot of extra queries to prevent a very rare event from happening. We leave further details to the discussion section.

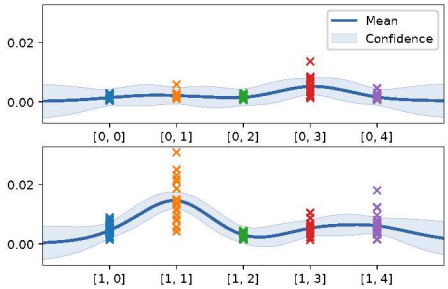


Fig. 0.10. Full data GP fit on Muscle 0.

Because in most cases (not only for the single-event case), there is a stimulation pattern which is a clear winner, we investigate how many query points are required by our algorithm to identify this optimum. We measure the prediction accuracy, which is 1 if the correct best channel was predicted and 0 otherwise, for GPs initialized with different number of initial random query points, and different number of active query points selected by the acquisition function. Fig. 0.11a shows the mean accuracy of these GPs over 25 independent trials for

three numbers of random query points. We also show the same mean accuracy measure but for many more different settings of initial random queries and active queries in Fig. 0.11b. For example, the blue line in Fig. 0.11a corresponds to the first row in Fig. 0.11b. We clearly see the trend; the more random points we use, the better the convergence. It is worth noting that the number of initial random points seems to be crucial. With certain "bad" initial random points (roughly 25% of the time for the blue run), we cannot compensate by having sequential query points. This is shown with the blue trace that plateaued at 75%. (It is possible that the GP hasn't actually plateaued yet, but nonetheless it would require many more queries before convergence to 1.0 accuracy.) In other words, for this muscle, having fewer than 35 query points implies that we do not converge for all of the runs. Fortunately, as we argue in the next paragraph, the hierarchical approach renders this less important than could be thought. But if necessary, this could be improved by having a more accurate (non i.i.d. likelihood model), as we further discuss in the discussion section. Another less elegant way to solve this problem is to set k higher. (Here, $k = 9$ was necessary, whereas for the easier Muscle 0 for example, $k = 5$ was amply sufficient. See Appendix.)

Because the point of these single event, single muscle GPs is to use them as a mean prior for higher dimensional GPs, correctly predicting the best channel is not as important as it might seem. For example, many channels could be close to the max (see for example Fig. 0.12), and as long as we can predict a high enough value for these, the higher dimensional GP will be able to test their combinations and find the best stimulation pattern. For this reason, in Fig. 0.11c, we show the convergence rate of our algorithm in terms of $l_{\max}$ distance (for a given number of events k), where

$$l_{\max}(f,\text{data}) = |\max_{s_k} f(s_k) - \max_{s_k} \mathbb{E}[C(r(s_k))]|$$

where f is the mean of our GP. We use this measure of distance because it is really the prediction at the max channel which interests us, and not how well the GP is modeling the true function at other (lower response) points. Of course we are making the assumption (Eq.2.2) that the most responsive stimulation pattern will contain the most responsive single channel stimulation. Nonetheless, any divergence from this prior will have to be found by the nonparametric part of the kernel, and hence the prior mean value will not be that important. For example, if two channels are not very responsive individually, but their

combined stimulation lead to a big response, accurately modeling their single event mean response (which is low anyway) is not that important.

Similarly to the mean accuracy color plot, we notice that the distance metric in Fig. 0.11c gets progressively smaller as we actively query more sequential points (left to right), and also that the more initial random points we use, the better. With just 45 random and 40 sequential points, which is less than half the number of points in the dataset (200 in total), we get an $l_{\max}$ distance of 0.0009. Compared to channel [1,1]'s mean response of 0.015, this consists of a 6% error, which is negligible. Actually, we see that the diagonal consisting of a total of 55 query points already has a qualitatively (in terms of color) negligible error. Even more surprising is that we only need 25 total query points, 15 random and 10 actively queried, to build a good enough prior for the higher dimensional stimulation spaces. This is because it is the relative prediction of the channels (the shape of the GP), and not the absolute predicted value that matters, because we are learning the linear contributions $a1$ and $a2$ of the channels in the higher order stimulation patterns (see eq. 2.2). Thus, it will cost us 25 query points in the single-channel space to build a prior for the double-event space (see Fig. 0.13c).

To show the effectiveness of our algorithm, we decided to augment the search space in time, and show results in the following sections.

3.2. Two Event – Single Muscle (fixed Δt)

Toward our goal to evoke targeted muscle co-activations, we first show a minimal example where the hierarchical approach is useful. We tested our algorithm on the double-event stimulation space with fixed $\Delta t = 60$ms (i.e. $s_2 = (c_1,c_2,\Delta t_1 = 60)$) with target objective to maximize Muscle 4 response. We choose this search space because of its interesting nonlinearity, and show that our algorithm is able to find the best double-event stimulation pattern to elicit a response in a single muscle.

Fig. 0.12 show the search space for an example muscle (*opponens pollicis*, Muscle 4), with the mean (max of) responses displayed in the plots. We refer the reader to the appendix for several other target muscle, different search spaces, and target objective combinations. These all lead to similar outcomes. The linear additive prior (eq. 2.1) predicts stimulating channel [1,0] twice to give the highest response, however we see that nonlinear effects are

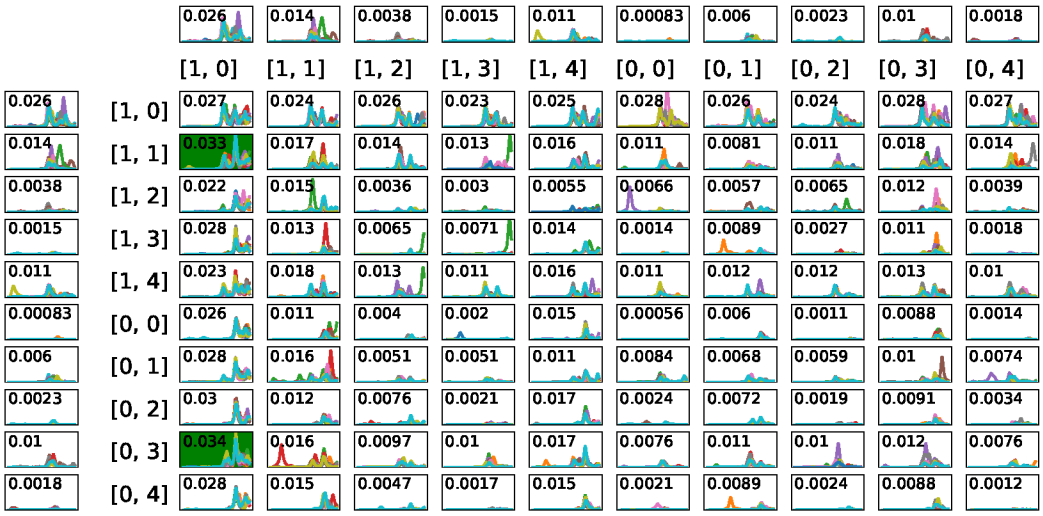
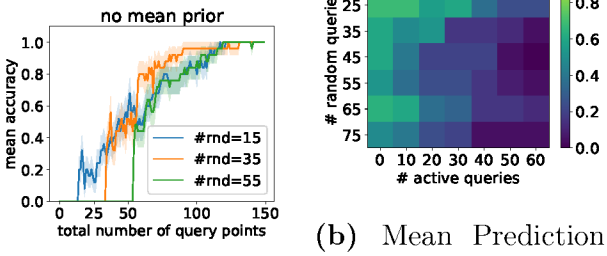
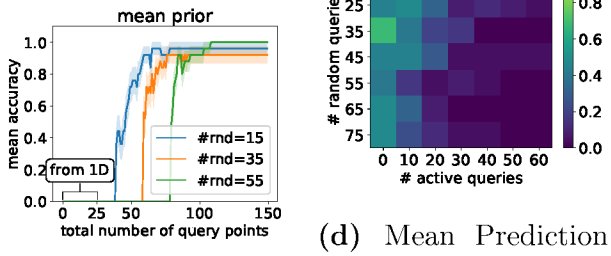


Fig. 0.12. Search space consisting of Muscle 4 responses to double-event, $\Delta t = 60$ ms stimulations, with the single-event Muscle 4 responses plotted left (first stimulation) and on top (second stimulation). Average max responses are written on the plots, with the green cells denoting the best stimulations: both ([1,1],[1,0]) and ([0,3],[1,0]) have very similar max responses of 0.033 and 0.034, respectively. Independent linearly additive assumption would have predicted ([1,0],[1,0]) as the best stimulation.



(a) Mean Prediction accuracy vs. total # of queries when not using a prior.

(b) Mean Prediction accuracy vs. # of random and active queries when not using a prior.



(c) Mean Prediction accuracy vs. total # of queries when using a prior.

(d) Mean Prediction accuracy vs. # of random and active queries when using a prior.

Fig. 0.13. Metrics for the double event with fixed $\Delta t = 60$ GP model on Muscle 4

such that priming the brain by first stimulating either channel [1,1] or channel [0,3] and then following with channel [1,0] gives the highest mean response. Because both of these patterns give extremely similar responses, we accept finding either of them as being accurate when

comparing algorithms in Fig. 0.13. Previous approaches would have stimulated the same channel more times, for a longer duration, or with a greater amplitude. We find that it is best to stimulate it with a spatiotemporal pattern.

Because this space is our final objective, and not being used to construct a prior for a more complex space, we are interested in performing well in terms of prediction accuracy, and so only plot performance in terms of this measure. Comparing the standard GP which starts with a mean prior of 0 (essentially no prior) in Fig. 0.13a to our hierarchical GP approach in Fig. 0.13c, we see that even for such a small search space of 100 possible stimulations, first spending 25 queries to build a mean prior from the single-channel stimulations is worth it. And the query savings will only get better as we increase the space of stimulation patterns.

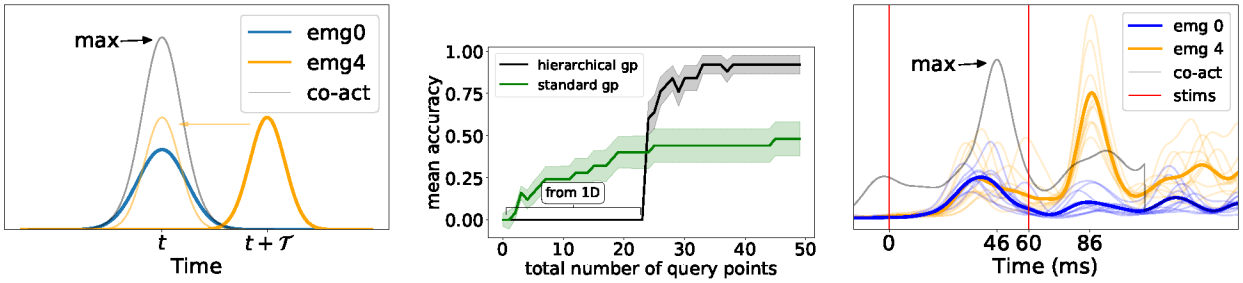
We also include the color plots in Fig. 0.13b and Fig. 0.13d to get a better overall qualitative comparison of both algorithms. An interesting (and intuitive) fact that emerges is that when we have a mean prior, initial random queries are essentially useless. This is because these random queries would make no use of that prior which, in a sense, was meant to curtail the need for initial queries. And already for this small search space, using a mean prior permits more rapid convergence.

3.3. Two Event – Temporal Co-Activation (search Δt)

To showcase our algorithm on a concrete example, we use a temporal co-activation of multiple muscles (see Fig. 0.14a) as a target in our objective function, which we define to act as a proxy for a complex sequential two-muscle movement. We target two muscles of the animal, the *flexor carpi ulnaris* (Muscle 0) and the *opponens pollicis* (Muscle 4), which we want to activate with a 40ms delay in between peaks. In order to formulate this problem using a similar objective function as described in the Methods section, we make the simplifying assumption that movement amplitude will correlate with the maximum amplitude of combined EMG responses, incorporating the desired delay. We define

$$C(r(s_2)) = max_t \left[r_0(s_2,t) + r_4(s_2,t+40) \right]. \tag{3.1}$$

In our search, we restrain the Δt dimension to the discrete set (20,40,60)ms due to data collection constraints (see Sec. 2.1). We found that having the spatial kernel dimensions share lengthscales gave the best results. Furthermore, we constrain this lengthscale to be between 1 and 2 so as to avoid spurious local minima where either the data is explained by noise



(a) Temporal co-activation target for objective function. To get a scalar objective, we translate r_4 by τ, sum the responses and take the max. Eq. (3.1)

(b) We plot the mean and std accuracy of our method compared to standard gp at finding the best stim pattern ([1,1],[1,0],60), in 25 repeated trials.

(c) Temporal co-activation found by our method. We set $\tau = 40$, and our algorithm finds this co-activation with stimulation delay $\Delta t = 60$.

Fig. 0.14. Objective and Results

only ($l = \infty$) or the responses become independent ($l = 0$) [24]. We also constrain the noise standard deviation to be between 5e−4 and 1e−3 (typical EMG response size are between 1e−2 and 3e−2), which encompasses the empirical standard deviation of every stimulation pattern. We compare our hierarchical approach, which uses priors built with a GP in the single-electrode space using 20 queries, to a GP which is directly trained on the two-electrode space.

The results in Fig. 0.14b show that our algorithm clearly outperforms the standard GP-BO procedure, which not only takes much longer to converge, but also is more sensitive to the UCB parameter k, and can more easily get stuck in local minima. We show that for the chosen muscle combination, the mean accuracy over 25 repeated trials is 1 after about 40 query points (including the initial ~ 25 queries used to train the GP in the single-electrode space). This means that our algorithm converges to the true best stimulation pattern, namely ($c1$,$c2$,Δt) = ([1,1],[1,0],60), out of a possible 300 (100 channel pairs and 3 time delays Δt) with only 40 total queries. Due to this stimulation pattern having a response much larger than other channels (0.041V, whereas other high-responding patterns are around 0.030V), we believe this metric (which focuses on the very best channel) to be a good measure of success. In other words, any other stimulation pattern would not have resulted in a movement as obvious and well-defined as this one. We show in Fig. 0.14c the resulting co-activation found.

We note that the true distance separating the two peaks (τ) is in fact a little more than 40, suggesting that the maximal stimulation pattern might have been with a Δt a little shorter than 60. Had the data been collected online, the GP, with its ability to interpolate, would have found and suggested this optimal pattern, which we remind, is not the one that would have been expected from additive combinations.

We also successfully tested our algorithm on different muscle combinations, which always required a very similar total number of query points to learn the optimal stimulation (see Appendix).

4. Conclusion

We showed that the hierarchical approach to build GPs on the space of multi-electrode stimulation patterns is a viable one to identify optimal inputs for a given target EMG output. Not only does it far outperform the standard GP approach (and random search), but it can also be used online to find the optimal stimulation strategy for a desired co-activation output. This is a step forward in linking brain activity and behavior by being able to control muscles directly [36], and for the use of neural prostheses to improve motor recovery after stroke or other motor system injuries.

The most novel and interesting part of our work is the ability of our algorithm to learn and elicit targeted complex movements online. As a proof of concept, we used a restricted stimulation search space for which we can exhaustively sample all stimulation combinations, and clearly demonstrated faster learning using our approach in thorough offline tests.

5. Discussion

We made a few simplifying modeling assumptions in our hierarchical kernel interactions, which could be improved upon, though we are unsure whether the performance gains would justify the significant level of complexity added. For one, we assumed homoscedastic additive Gaussian noise [24], whereas biology is often better described by Poisson noise (see Fig. 0.8 and 0.10), or even more complex models [37]. Second, we trained the GPs for different muscles (for each r_i) independently, whereas they are clearly correlated (see Fig. 0.6b), and could potentially share information through multi-output (also called co-Kriging) models. We actually tried this approach since it is relatively easy to implement (see [38] for a review),

but it proved unnecessary since for our data, many of the muscles have similar responses (eg. Fig. 0.8), which makes sharing of information much less effective. Third, using a non-stationary kernel could accelerate the search even more [21]. This would allow using a large lengthscale for most of the search space, yet have a smaller lengthscale near optimal stimulation patterns to permit finding the true maximum.

Nonetheless, we get reasonable results as is, and note that this method can easily be adapted to more complex objective functions such as incorporating both forelimb and hindlimb movements, and to different sensor modalities such as acceleration from an accelerometer or 3D position from a camera. This means that rather than optimizing for high EMG output, which here only correlates with movement amplitude, we could directly optimize for movement amplitude and direction using, for example, DeepLabCut [39] to get pose estimations. Furthermore, our approach is not confined to cortical microstimulation. Indeed, spinal and peripheral nerve stimulation are promising approaches that could be used to evoke targeted movements, and a hierarchical stimulation optimization is directly applicable to these settings.

Long train stimulations (500ms) have already been shown to evoke complex multi-joint movements [33]. However, we still lack a mechanistic explanation for the role of cortical dynamics in the generation of these movements. We believe that scaling our approach could provide answers by uncovering optimal spatiotemporal stimulation patterns that lead to complex movements. However our preliminary results already show that our algorithm is capable of revealing circuit-level computations, beyond the assumed linear and additive combination used to create priors. This makes it a good scientific tool that can not only be used for pure optimization of a BCI control signal, but also for asking hypothesis-driven questions about the brain.

Acknowledgments

We thank Andrew Bogaard, Eberhard Fetz, Chet Moritz, Maximilian Puelma Touzel and Olivier Caron-Grenier for useful discussions. We acknowledge the important contributions of Stephan Quessy for experimental implementations and data collection. Funding: MB [IVADO fellowship], ND [FRQNT group grant (2019-PR-253402)], GL [NSERC Discovery Grant (RGPIN-2018-04821), FRQNT Young Investigator grant (2019-NC-253251), FRQS Research Scholar Award, Junior 1 (LAJGU0401-253188)]

Chapitre 4

Conclusion

The presented article has introduced a hierarchical way to combine gaussian processes so as to effectively search the space of high-order spatiotemporal stimulation patterns to evoke forelimb muscles. We have shown its usefulness in previously collected monkey motor cortex neurostimulation data, where it was able to find a non-intuitive optimal stimulation pattern, where previous linear additive methods would have failed. Furthermore, it is being tested online in ongoing rat experiments by our collaborators, Marco Bonizzato and Numa Dancause, with promising results. Although neurostimulation has been proven to help restore motor function after injury [18] using simple stimulations, it remains to be shown whether even better results can be achieved using more complicated stimulations, such as those provided by our method. Nonetheless, such complicated stimulation patterns will definitely be useful for restoring complex movements such as gaiting [9]. We also see expanded use of this algorithm in other neurostimulation paradigms, such as peripheral nerve and cortico-cortical stimulations. It is also very possible to use other objective targets than EMG-based, such as accelerometer data or camera-based movement measurements.

From an algorithmic point of view, using the mean function of a gaussian process model to inject prior knowledge is not a new idea, and is used for example by NASA to approximate computer simulation results [40]. Although the majority of machine learning applications set the mean function of the GP to zero, most applications in physics, chemistry, biology, etc., where information about the underlying function is known, can set the mean to some reasonable a priori value. **The main contribution of this work comes from applying this prior mean information recursively, forming a hierarchical gaussian process**, circumventing the curse of dimensionality in cases such as ours where this hierarchical

construction is valid. Although complicated and high-dimensional parameter spaces have been explored previously using bayesian optimization over treed structures [41, 22], our approach of searching through smaller spaces first, and then moving on to larger spaces recursively using a combination scheme based on knowledge of the system (in this case, the brain), is novel. We do note similarities with curriculum learning [42] in the machine learning literature, where classifiers are trained by being presented examples "in a meaningful order which illustrates gradually more concepts, and gradually more complex ones". In our case, where we are interested in optimizing a function rather than training a classifier, it is the search space that is gradually complexified.

Although we have only applied our approach to neurostimulation, the idea seems general enough to find applications in other domains. For example, a natural domain that comes to mind where hierarchy is important is control and reinforcement learning [43]. In normal (non-hierarchical) reinforcement learning, gaussian process models have already been used successfully to get state of the art data efficient results on low dimensional problems like cartpole [44]. Perhaps making the approach hierarchical could make it scale to larger dimensional spaces. However, it is not clear which kinds of problems would have similar structure as that found in motor cortex, where our approach shines; that is, a spatial hierarchy. Otherwise, if we give up the requirement of data efficiency, then perhaps we could instead use deep gaussian processes [45] or scalable sparse gaussian processes [46], though here it is unclear whether such models could rival deep learning approaches.

Bibliography

[1] The Future of the Mind (The Scientific Quest to Understand, Enhance, and Empower the Mind). *Mens Sana Monographs*, 14(1):214–220, 2016.

[2] Nikolaus Wenger et al. Spatiotemporal neuromodulation therapies engaging muscle synergies improve motor control after spinal cord injury. *Nature medicine*, 22(2):138–145, 02 2016.

[3] Ana R C Donati, Solaiman Shokur, Edgard Morya, Debora S F Campos, Renan C Moioli, Claudia M Gitti, Patricia B Augusto, Sandra Tripodi, Cristhiane G Pires, Gislaine A Pereira, Fabricio L Brasil, Simone Gallo, Anthony A Lin, Angelo K Takigami, Maria A Aratanha, Sanjay Joshi, Hannes Bleuler, Gordon Cheng, Alan Rudolph, and Miguel A L Nicolelis. Long-Term Training with a Brain-Machine Interface-Based Gait Protocol Induces Partial Neurological Recovery in Paraplegic Patients. *Scientific Reports*, 6:30383, aug 2016.

[4] Cioni B, Tufo T, Bentivoglio A, Trevisi G, and Piano C. Motor cortex stimulation for movement disorders. *J Neurosurg Sci*, pages 230–41, 2016.

[5] Aurelie Selfslagh et al. Non-invasive, brain-controlled functional electrical stimulation for locomotion rehabilitation in individuals with paraplegia. *Scientific Reports*, 9(1):6782, 2019.

[6] Spinal cord injury. `http://www.who.int/news-room/fact-sheets/detail/spinal-cord-injury`, 2013. Accessed: 2018-11-15.

[7] Amanda G Thrift et al. Global stroke statistics. *International Journal of Stroke*, 12(1):13–32, 2018/11/15 2016.

[8] Fabien B Wagner, Jean-Baptiste Mignardot, Camille G Le Goff-Mignardot, Robin Demesmaeker, Salif Komi, Marco Capogrosso, Andreas Rowald, Ismael Seáñez, Miroslav Caban, Elvira Pirondini, Molywan Vat, Laura A McCracken, Roman Heimgartner,

Isabelle Fodor, Anne Watrin, Perrine Seguin, Edoardo Paoles, Katrien Van Den Keybus, Grégoire Eberle, Brigitte Schurch, Etienne Pralong, Fabio Becce, John Prior, Nicholas Buse, Rik Buschman, Esra Neufeld, Niels Kuster, Stefano Carda, Joachim von Zitzewitz, Vincent Delattre, Tim Denison, Hendrik Lambert, Karen Minassian, Jocelyne Bloch, and Grégoire Courtine. Targeted neurotechnology restores walking in humans with spinal cord injury. *Nature*, 563(7729):65–71, 2018.

[9] Marco Capogrosso et al. A brain-spine interface alleviating gait deficits after spinal cord injury in primates. *Nature*, 539(7628):284–288, 11 2016.

[10] Simon A Overduin, Andrea d'Avella, Jose M Carmena, and Emilio Bizzi. Microstimulation activates a handful of muscle synergies. *Neuron*, 76(6):1071–1077, 2012.

[11] S. Raspopovic et al. Experimental validation of a hybrid computational model for selective stimulation using transverse intrafascicular multichannel electrodes. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 20(3):395–404, 2012.

[12] Grégoire Courtine, Mary Bartlett Bunge, James W Fawcett, Robert G Grossman, Jon H Kaas, Roger Lemon, Irin Maier, John Martin, Randolph J Nudo, Almudena Ramon-Cueto, Eric M Rouiller, Lisa Schnell, Thierry Wannier, Martin E Schwab, and V Reggie Edgerton. Can experiments in nonhuman primates expedite the translation of treatments for spinal cord injury in humans? *Nature medicine*, 13(5):561–566, may 2007.

[13] WILDER PENFIELD and EDWIN BOLDREY. SOMATIC MOTOR AND SENSORY REPRESENTATION IN THE CEREBRAL CORTEX OF MAN AS STUDIED BY ELECTRICAL STIMULATION1. *Brain*, 60(4):389–443, 12 1937.

[14] Michael SA Graziano. *The Intelligent Movement Machine: An Ethological Perspective on the Primate Motor System*. Oxford University Press, 2008.

[15] Elon and Musk. An integrated brain-machine interface platform with thousands of channels. *bioRxiv*, 2019.

[16] J H Kaas. Plasticity of sensory and motor maps in adult mammals. *Annual Review of Neuroscience*, 14(1):137–167, 1991.

[17] Rajesh PN Rao. Towards neural co-processors for the brain: combining decoding and encoding in brain–computer interfaces. *Current Opinion in Neurobiology*, 55:142 – 151, 2019. Machine Learning, Big Data, and Neuroscience.

[18] David J. Guggenmos, Meysam Azin, Scott Barbay, Jonathan D. Mahnken, Caleb Dunham, Pedram Mohseni, and Randolph J. Nudo. Restoration of function after brain damage using a neural prosthesis. *Proceedings of the National Academy of Sciences*, 110(52):21177–21182, 2013.

[19] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599, 2010.

[20] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, Jan 2016.

[21] Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for bayesian optimization of non-stationary functions. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1674–1682, Bejing, China, 22–24 Jun 2014. PMLR.

[22] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, pages 2546–2554, USA, 2011. Curran Associates Inc.

[23] Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. An evaluation of sequential model-based optimization for expensive blackbox functions. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '13 Companion, pages 1209–1216, New York, NY, USA, 2013. ACM.

[24] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[25] Andreas Buja, Trevor Hastie, Robert Tibshirani, et al. Linear smoothers and additive models. *The Annals of Statistics*, 17(2):453–510, 1989.

[26] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K. Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences. *ArXiv*, abs/1807.02582, 2018.

[27] Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *J. Mach. Learn. Res.*, 2:299–312, March 2002.

[28] Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012.

[29] Dahn (https://stats.stackexchange.com/users/45474/dahn). What is the rationale of the matern covariance function? Cross Validated. URL:https://stats.stackexchange.com/q/325027 (version: 2018-01-26).

[30] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, February 2012.

[31] Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process bandits without regret: An experimental design approach. *CoRR*, abs/0912.3995, 2009.

[32] PD Cheney, DM Griffin, and GM Van Acker III. Neural hijacking: action of high-frequency electrical stimulation on cortical circuits. *The Neuroscientist*, 19(5):434–441, 2013.

[33] Michael SA Graziano, Charlotte SR Taylor, and Tirin Moore. Complex movements evoked by microstimulation of precentral cortex. *Neuron*, 34(5):841–851, 2002.

[34] Adjia Hamadjida, Melvin Dea, Joan Deffeyes, Stephan Quessy, and Numa Dancause. Parallel cortical networks formed by modular organization of primary motor cortex outputs. *Current Biology*, 26(13):1737–1743, Jul 2016.

[35] AP Georgopoulos, AB Schwartz, and RE Kettner. Neuronal population coding of movement direction. *Science*, 233(4771):1416–1419, 1986.

[36] A Paul Alivisatos et al. Neuroscience. the brain activity map. *Science (New York, N.Y.)*, pages 1284–1285, 2013.

[37] Lev S Tsimring. Noise in biology. *Reports on progress in physics. Physical Society (Great Britain)*, 77(2):026601–026601, 02 2014.

[38] Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.

[39] Alexander Mathis, Pranav Mamidanna, Kevin M. Cury, Taiga Abe, Venkatesh N. Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. Deeplabcut: markerless

pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281–1289, 2018.

[40] Robert B. Gramacy, Herbert K. H. Lee, and William G. Macready. Parameter space exploration with gaussian process trees. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 45–, New York, NY, USA, 2004. ACM.

[41] Robert B Gramacy and Herbert K. H Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.

[42] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[43] Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

[44] Marc Peter Deisenroth and Carl Edward Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 465–472, USA, 2011. Omnipress.

[45] Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

[46] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *arXiv preprint arXiv:1807.01065*, 2018.

[47] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[48] Schur complement. Schur complement, 2003. [Online; accessed 23-January-2019].

Appendix A

Article1 Extra Material

A.1. Response Surface to double-event, temporal co-activation where we search Δt

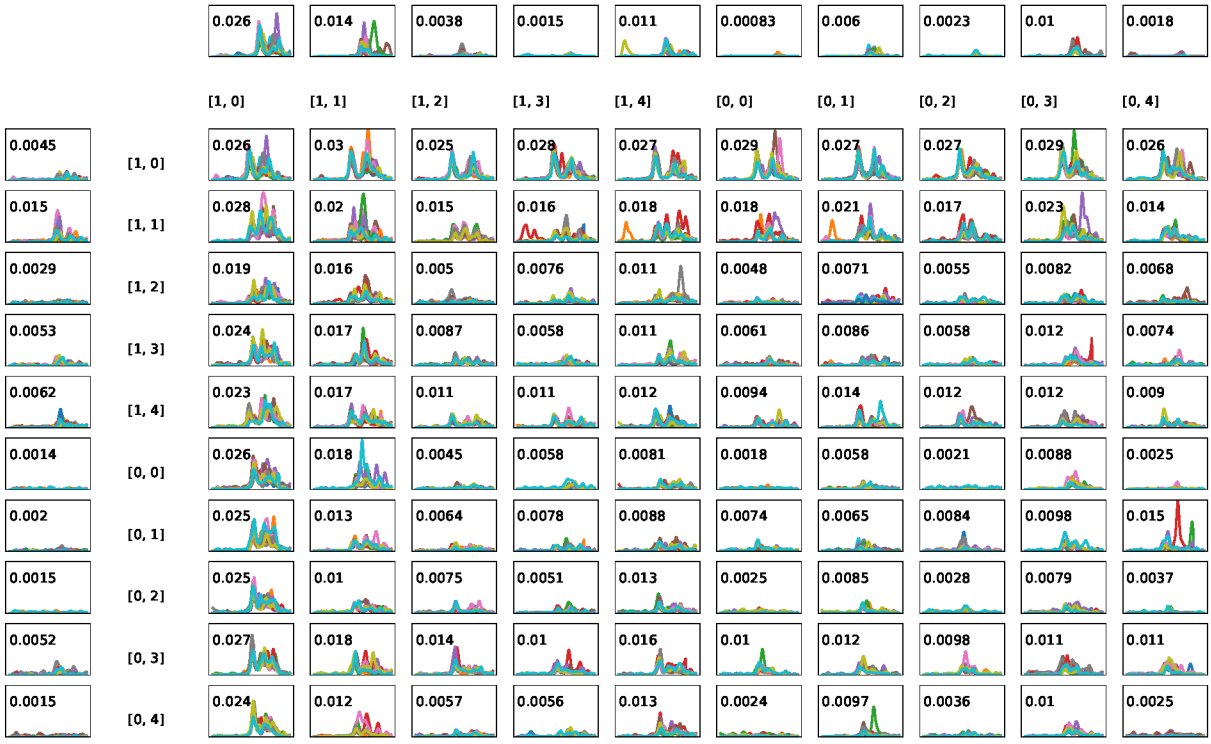


Fig. A.1. Double-event search space ($\Delta t = 20$)

Fig. A.2. Double-event search space ($\Delta t = 40$)

A.2. Different Two Event, Single Muscle (fixed Δt)

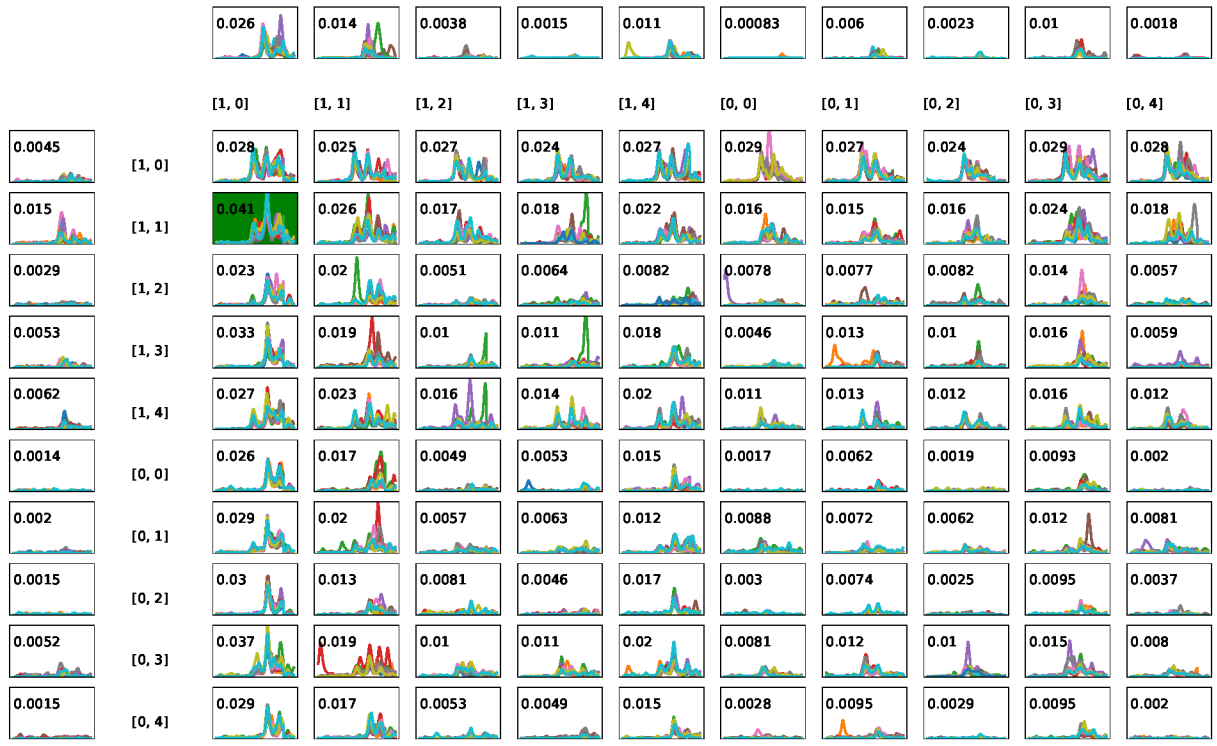


Fig. A.3. Double-event search space ($\Delta t = 60$)

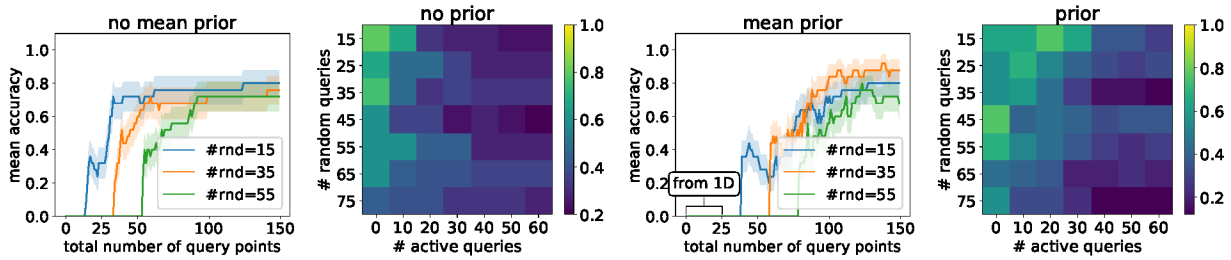


Fig. A.4. Mean Prediction accuracy vs. # of queries for Muscle 0 $\Delta t0$, with and without prior. For $\Delta t = 0$ here and in Fig. A.8, we used a symmetric kernel, since the order of channel stimulation does not matter when they are done at the same time. This reduces to search space to only 50 channels, which makes it easier for the vanilla GP. We still show the results for completeness, and to show that our algorithm does find the best stimulation pattern despite not necessarily outperforming the vanilla approach by much.

A.3. Effect of UCB acquisition function's exploration parameter k

Fig. A.5. Muscle 0 $\Delta t0$

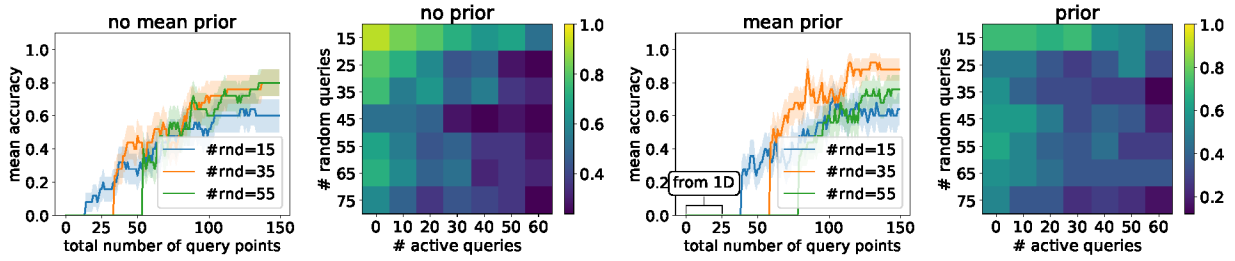


Fig. A.6. Mean Prediction accuracy vs. # of queries for Muscle 0 $\Delta t10$, with and without prior.

Fig. A.7. Muscle 0 $\Delta t10$. Notice that we accept both ([1,1],[1,0]) and ([1,1],[1,1]) as best stimulation pattern. This is because of an outlier in the ([1,1],[1,0]) stimulation, without which ([1,1],[1,1]) is the best stimulation pattern.

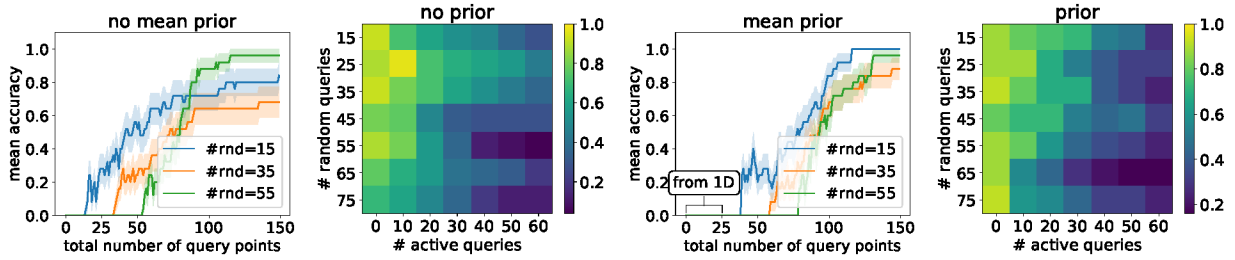


Fig. A.8. Mean Prediction accuracy vs. # of queries for Muscle 4 $\Delta t0$, with and without prior.

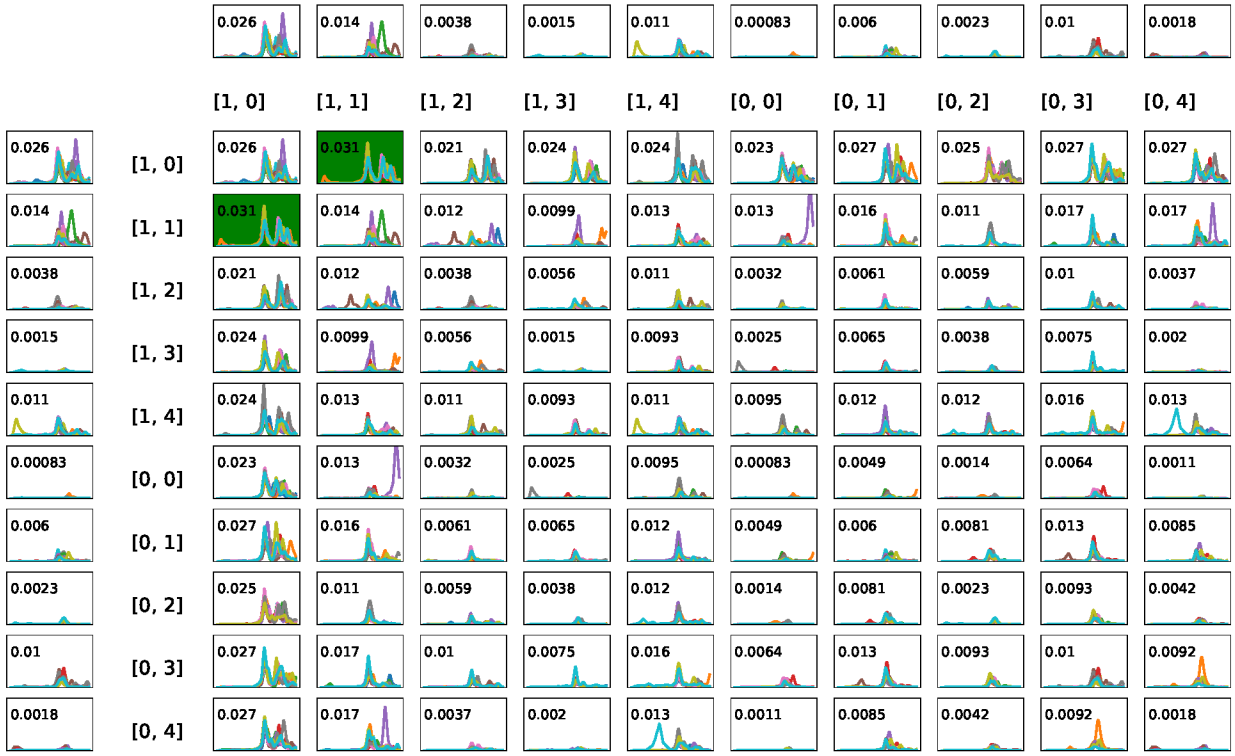


Fig. A.9. Muscle 4 $\Delta t0$

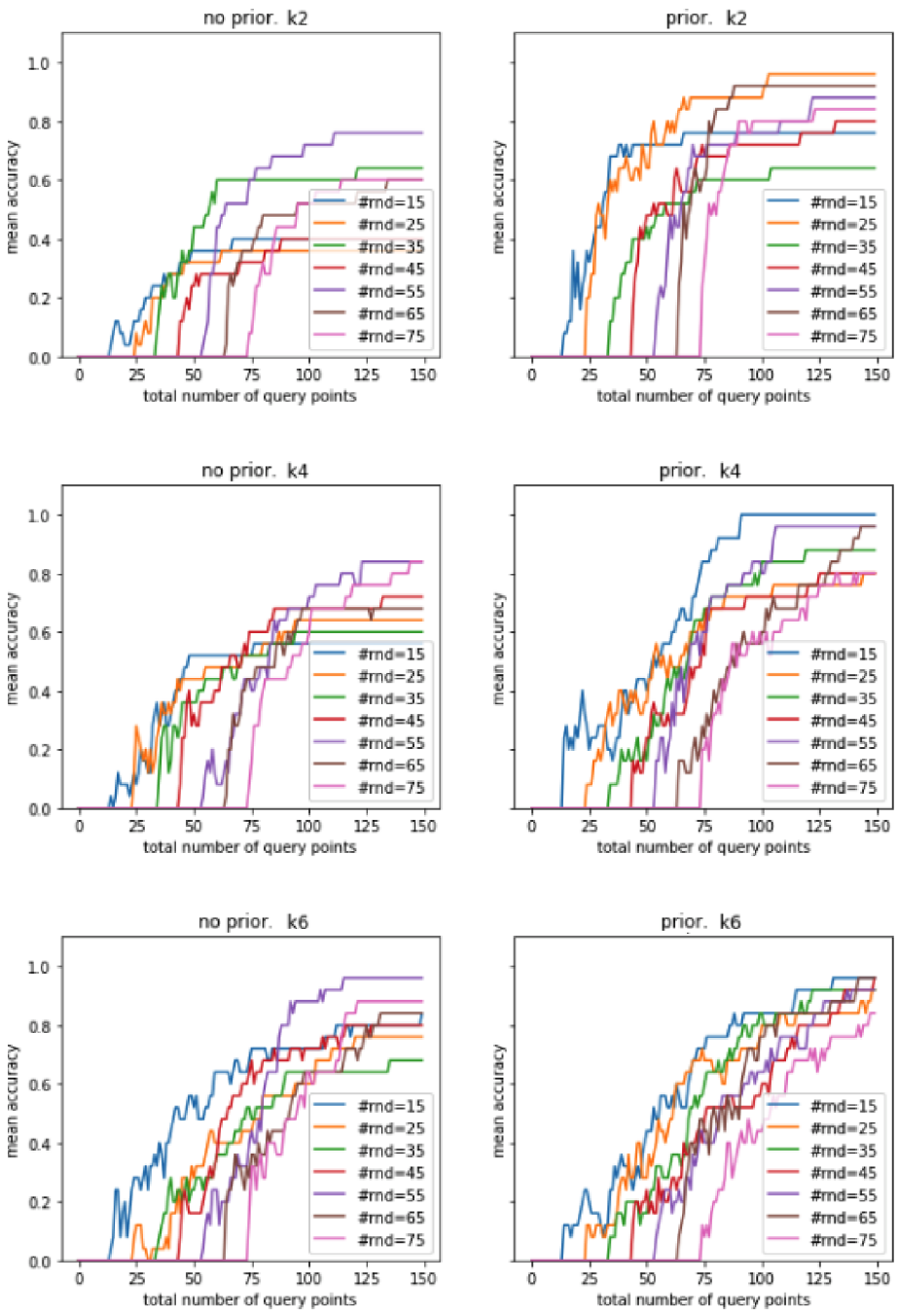


Fig. A.10. Muscle 4 $\Delta t0$. We see that higher k values encourage exploration, whereas lower values of k encourage exploitation, at the cost of sometimes getting stuck in local minima (see upper left graph without prior and $k = 2$). Higher k value seems better for the vanilla GP, which needs to explore the space a lot, having started with no mean prior, whereas the hierarchical GP, which already has information about the space, performs worse with higher k values. For example, the hierarchical GP with $k = 6$ in the bottom right takes longer to converge because it is wasting time exploring regions that it does not need to.

Appendix B

Conditioning a Multivariate Gaussian Distribution

Here we develop the necessary details for understanding the distribution that results from conditioning a multivariate gaussian distribution. We aim for the crux of the matter, intending on giving a good intuition and understanding of what is necessary for Gaussian Processes, while leaving a general theory to proper references [47, 48].

B.1. Conditioning

We start with a block MVN

$$\begin{pmatrix} X_a \\ X_b \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}\right)$$

where we only deal with the $\mu = 0$ since this is all that we need for Gaussian Processes. We are interested in calculating $X_a|X_b = x_b$. We do so by doing syntactic manipulations on the shape of the exponential distribution, showing that after conditioning, it remains of the same form: $exp(-1/2x^T\Sigma^{-1}x)$. To simplify the calculations, we will work with the inverse of the covariance matrix, the precision matrix $\Lambda := \Sigma^{-1}$, which we write as

$$\Lambda := \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

From here, we find

$$
\begin{aligned}
p(x_a|x_b) \propto p(x_a,x_b) &\propto \exp\left(-1/2\begin{pmatrix} x_a & x_b \end{pmatrix}\begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}\begin{pmatrix} x_a \\ x_b \end{pmatrix}\right) \\
&\propto \exp\left(x_a^T\Lambda_{aa}x_a + x_a^T\Lambda_{ab}x_b + x_b^T\Lambda_{ba}x_a\right) \\
&= \exp\left(x_a^T\Lambda_{aa}x_a + 2x_a^T\Lambda_{ab}x_b\right) \\
&\propto \exp\left((x_a - \underbrace{(-\Lambda_{aa}^{-1}\Lambda_{ab}x_b)}_{\mu_{a|b}})^T \underbrace{\Lambda_{aa}}_{\Sigma_{a|b}^{-1}}(x_a - (-\Lambda_{aa}^{-1}\Lambda_{ab}x_b))\right)
\end{aligned}
$$

which shows that conditioning a MVN gives back another MVN

$$X_a|(X_b = x_b) \sim \mathcal{N}\left(\mu_{a|b}, \Sigma_{a|b}\right)$$

where $\mu_{a|b} = -\Lambda_{aa}^{-1}\Lambda_{ab}x_b$ and $\Sigma_{a|b} = \Lambda_{aa}^{-1}$. Now the only concern is that our Gaussian Process is specified by a covariance function (the Kernel), and not a precision matrix. So we would need to write all of the Λ in terms of Σ. We do so in the following section.

B.2. Block Matrix Inverse

We have

$$\begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix}$$

and we need to express Λ_{ab} and Λ_{aa} in terms of Σ's. Note that Λ_{aa} is NOT equal to Σ_{aa}^{-1}. We perform the composition by taking the LDU decomposition of Σ, which we can then invert easily and multiply back to get Λ. We can do this by performing a Gaussian Elimination on Σ. To make the syntax easier to follow, we will work with a generic (not necessarily symmetric) block matrix

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

First, we write

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}\begin{pmatrix} I & 0 \\ -D^{-1}C & I \end{pmatrix} = \begin{pmatrix} A - BD^{-1}C & B \\ 0 & D \end{pmatrix}$$

By continuing the Gaussian elimination process to now eliminate the B, we find

$$\begin{pmatrix} I & -BD^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} A - BD^{-1}C & B \\ 0 & D \end{pmatrix} = \begin{pmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{pmatrix}$$

And so putting these together we get

$$\begin{pmatrix} I & -BD^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} I & 0 \\ -D^{-1}C & I \end{pmatrix} = \begin{pmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{pmatrix}$$

and by inverting the diagonal matrices (easily done by replacing the negative sign by a positive sign, as can be checked)

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} I & BD^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{pmatrix} \begin{pmatrix} I & 0 \\ D^{-1}C & I \end{pmatrix}$$

From this we can invert both sides

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} I & 0 \\ -D^{-1}C & I \end{pmatrix} \begin{pmatrix} (A - BD^{-1}C)^{-1} & 0 \\ 0 & D^{-1} \end{pmatrix} \begin{pmatrix} I & -BD^{-1} \\ 0 & I \end{pmatrix}$$

We define $M := (A - BD^{-1}C)^{-1}$ to simplify the notation and multiply the three matrices

$$\begin{aligned} \begin{pmatrix} I & 0 \\ -D^{-1}C & I \end{pmatrix} \begin{pmatrix} M & 0 \\ 0 & D^{-1} \end{pmatrix} \begin{pmatrix} I & -BD^{-1} \\ 0 & I \end{pmatrix} &= \begin{pmatrix} M & 0 \\ -D^{-1}CM & D^{-1} \end{pmatrix} \begin{pmatrix} I & -BD^{-1} \\ 0 & I \end{pmatrix} \\ &= \begin{pmatrix} M & -MBD^{-1} \\ -D^{-1}CM & D^{-1} + D^{-1}CMBD^{-1} \end{pmatrix} \end{aligned}$$

B.3. Back to Conditioning

Now that we have a formula for the inverse of a block matrix, this gives us

$$\Lambda_{aa} = (\Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba})^{-1}$$

and

$$\Lambda_{ab} = -(\Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba})^{-1}\Sigma_{ab}\Sigma_{bb}^{-1}$$

and so we find

$$\mu_{a|b} = -\Lambda_{aa}^{-1}\Lambda_{ab}x_b = \Sigma_{ab}\Sigma_{bb}^{-1}x_b$$

and

$$\Sigma_{a|b} = \Lambda_{aa}^{-1} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba}$$