UNIVERSITY OF CALIFORNIA
SANTA CRUZ

MACHINE LEARNING AND CONTROL METHODS FOR BIOLOGICAL SYSTEMS: TOWARDS ADVANCING PRECISION MEDICINE

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

APPLIED MATHEMATICS

by

Giovanny Marquez

September 2023

The proposal of Giovanny Marquez
is approved:

Dr. Marcella Gomez, Chair

Dr. Hongyun Wang

Dr. Qi Gong

Dr. Mircea Teodorescu

Peter Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Machine learning and control methods for biological systems: towards advancing precision medicine

by

Giovanny Marquez

Integrating technology in bioelectronics into the medical field allows for the advancement of precision medicine to better care for patients. Precision medicine includes personalized treatment options that consider varying responses across patients to the same medical treatments. To better administer these medical strategies, closed-loop control is needed to make real-time changes in treatment strategies as new information is gathered. Closed-loop control of biological systems is difficult in practice due to model uncertainties and innate complexities. To cope with this issue, we need control methods that do not require full system information, can handle time-varying uncertainties, and work across various time scales. Here, we develop and apply feedback control algorithms interfaced with ion pumps and cell systems that exhibit similar challenges. A class of artificial neural networks called a Radial Basis Function (RBF) network is applied to control pH and cell migration. This approach is chosen because it can achieve real-time online control without prior knowledge of the dynamical model of the system. The controller is interfaced with an ion pump to control the pH of a solution. We then modify the update laws of the controller to handle a system where the input needs to be within a set range. This new controller is used to control cell migration. We also consider a sliding mode controller (SMC), where partial system information provides better performance and is used for the delivery of a therapeutic drug using an ion pump. To control more complex systems with constraints, we

need to be able to leverage data-driven models. Thus, finally, we present a switch controller, utilizing a model predictive controller and the sliding mode controller, to reduce the control effort needed to direct cell migration *in silico* and, thereby, reduce off-target effects.

To my family

Particularly my parents for being patient with me while I figured out my own path.

Acknowledgments

My advisor Marcella Gomez for giving me this opportunity to begin with and for the guidance and support necessary to complete this program. Mohammad Jafari for being a great mentor and being beyond helpful, even when moving on to his teaching position. Ksenia Zlobina for pointing out errors in my line of thinking. To all those I've had the opportunity to work with, past and present, due to the DARPA project. I have learned a lot along the way from everyone. Thanks to Bethany, Sara, and Martin for helping me navigate my years here. To my friends back home for being supportive throughout the years. My girlfriend Yuli for understanding how much time this program took.

Chapter 1

Introduction

1.1 Background and Motivation

Self-regulation is an important attribute for the health of many systems. It can be seen in ecological systems where the dynamics of the system can be modeled by a predator-prey model[6]. Biological systems achieve this using feedback control. In humans, once the body senses a change the system adjusts to keep everything in a dynamic stable state. One example being during exercise when vasodilation occurs allowing for increased blood flow, which allows for more oxygen delivery to the muscles[46]. Another example is the human body's response to the influenza virus. Once the body senses infection, the infected cells and antigen presenting cells stimulate immunity by secreting interferon molecules. These molecules interact with healthy cells, converting them to an infection-resistant state and limiting the spread of the virus [28].

To apply a self-regulation approach to medicine, bioelectronics have been in-

tegrated with biological systems important to the medical field. This has the potential to advance precision medicine where treatment can be specialized for the patient, as is the case with pacemakers[3]. The ability of such devices to specialize treatment comes from having sensors and actuators, which can help facilitate the closed-loop control seen naturally in biological systems[81]. The challenge being, how to properly control the signal/input added to the biological system to elicit a desired response. In terms of the pacemaker, this is regulating the heart from an irregular heartbeat back to a normal one.

The difficulty lies with the behavior of biological systems. Biological systems tend to be adaptive to changes in their environment. This adaptive property is a consequence of their ability to self-regulate in order to maintain stability for optimal survival [10]. This stability is dynamic in that it diverges as needed to better suit the ever-changing environment around it. This adaptive property, along with system uncertainties, makes mathematical modeling a challenging task. Under these conditions, control can be difficult since no mechanistic model might be available, and typical control strategies might falter.

1.2 Identification of Problem

The adaptive nature of biological systems requires an adaptive control strategy. Many traditional methods, such as proportional-integral-derivative (PID) control, can have issues with uncertainty in a system. This can cause the controller to fail to reach the desired trajectory. In order to address this issue, adaptive control is required. Two adaptive control methods are utilized: machine learning and sliding mode control.

Machine learning has been used to control biological systems in an online fashion where large datasets are not available for offline training in priori [38,

81]. This is due to the parameters being updated at each timestep using new information of the system output. Sliding mode control is another approach that is robust to system uncertainties. This is beneficial, considering we know no/partial information about the system.

When enough data is gathered, deep learning can be used to derive a model of the complex biological system [76]. This allows for an alternative control method, model predictive control. This allows for desired constraints, such as control effort, to be placed.

1.3 Contributions and Objectives

The proposed contributions of this work is to implement real-time feedback control of biological systems, in particular, to lay a framework for methods that can be applied towards expediting wound healing. Although biological systems are still not fully understood, we show that machine learning offers an adaptive control strategy that can allow us to drive responses toward a desired trajectory. As our understanding of biology grows, we can help guide these trajectories to yield beneficial results in wound healing. The contributions are as follows:

- Application of a machine learning algorithm to control pH of a solution using an ion pump *in vitro*.
- An improved version of the algorithm to control cell migration *in vitro*.
- Application of a sliding mode controller to control the amount of a therapeutic drug being delivered using an ion pump *in vitro*.
- Development of a switch controller that utilizes model predictive control and sliding mode control to control the migration of cells *in silico*.

These results show promise in the capability of feedback control to help expedite wound healing, and help lay a framework for methods that might be used *in vivo*.

We intentionally made each chapter in this dissertation independent by including all relevant background information and method descriptions within each chapter. Accordingly, there may be some repetition across chapters, but readers are welcome to approach the chapters out of order.

Chapter 2

Background

2.1 Modeling

Control theory uses mathematical models of systems to understand how a change in the control signal affects the systems' responses. In classic control theory, this is done by converting the ordinary differential equation (ODE) model of a system from the time domain to the frequency domain using a Laplace Transformation [21]. Once in the frequency domain, there are methods to analyze how a change in the control signal will affect the system, such as root locus and bode plots [97]. The limitation of these techniques was the inability to work for multiple-input multiple-output (MIMO) systems, which is why classical control theory deals with single-input single-output(SISO) systems [64]. This problem was solved, however, when state space representation of ODE models was introduced by Kalman [34]. Having models in state space form allows for multiple-input multiple-output (MIMO) systems to be analyzed and facilitates controllers design

for them.

In order to design more informed control laws, an accurate model representation of the system is required. A mathematical model can be derived using data gathered from the system or through first principles. Mathematical models can be of various types: linear, nonlinear, continuous, discrete, stochastic, deterministic, etc. The type of model used depends on how the represented system behaves over time [9].

When data is small, and there isn't a deep understanding of why a system behaves the way it is, a qualitative model might be used. A qualitative model's essence lies in capturing a system's essential dynamics based on a fundamental understanding of the interactions between its states. It can be further improved with system data. A qualitative model would not be used for making predictions but rather to see how certain dynamics change under different conditions. For control purposes, a model that shows saturation would allow for the development of a controller that is able to overcome the saturation bounds to achieve a desired steady state for the system. An advantage of a qualitative model is it can be a simple representation of a system, making them easier to work with and cheaper to develop.

A more predictive model can be developed when there is plentiful data for a system. Predictive models are of paramount importance to those control techniques, such as model predictive control (MPC), that rely on the predictive models of the systems. MPC utilizes predictive models to derive the control signal to send to the system [74]. Data improves the model's predictability by being used to properly tune parameters in the model. This is done by enforcing the model output to match that of the data. There are numerous methods for parameter tuning that are able to make the model closely match the given data, such as the

genetic algorithm, sequential quadratic programming (SQP), Newton's method, etc [8, 47, 60]. Although this helps make model outputs match data closer and used as predictive models, there are always complexities that can't be captured accurately. Examples can be the parameters of a model varying dynamically over time due to the performance of a system degrading over time.

As the models improve to address these complexities, they can grow difficult to work with in terms of control. Models that are of too high order can't be used for control design because it can result in a high-order control law that needs a reduction to be implemented [34]. The process of reduction could lead to poor performance of the resulting controller. An advantage to having a model is that you can perform a system stability analysis. However, if the model is too complex, it can prove difficult or impossible to provide proof of stability by the controller. Another disadvantage is the time and cost required to develop models that are detailed to match data better.

2.2 Model Based Control

Modeling is important for control since it allows for model-based control (MBC). Models allow for stability guarantees by the controller as long as the models are able to capture the dynamics of the system fully. MBC can be done in two ways: open-loop and closed-loop. The goal in control is to keep the state(s) of a system/model at a desired reference value. Both methods are used in practice and have their advantages and disadvantages.

Open-loop control is used with systems where unforeseen disturbances are a rarity. This means the system behaves in a predictable manner that is shown in the model of the system. Therefore, in open-loop control, there is no change in the control strategy based on the output of the system/model. This is also known

as a feedforward control. The system is generally simple to the model allowing for the parameters of the controller to be tuned to acceptable values that should keep the system near the reference without adjustments.

Closed-loop control uses the output of the model/system and updates the control strategy in real-time. This allows the controller to be able to react to changes in the dynamics of the model/system. In practical application, the model allows for the tuning of the control parameters before the controller is used in a real-life system. Even if the model doesn't perfectly match the system, the feedback aspect allows the controller to still keep the system around the desired reference point. Due to modern control theory techniques, verifying the stability of linear and nonlinear models is possible through different techniques. For linear models, feedback stabilization can be verified by looking at the eigenvalues of the system after the control law is applied. For nonlinear models, Lyapunov stability analysis allows us to verify that the energy of the system is decreasing, proving the stability of the system.

The problem is that the stability proofs fail in practice if the system is not captured accurately. This means that a controller that was appropriately tuned using a model might not be able to control a system leading to instability. Even if the model is close enough that the system remains stable, missed dynamics could lead to poor performance of the controller when applied in real life. One other challenge is that to formalize proofs, assumptions on the system are made. These assumptions might not be true about the actual system at all times leading to poor performance or instability.

2.2.1 Model Predictive Control (MPC)

Model predictive control is a control method based on an optimal control problem [77]. A model of the system is used to evaluate the next optimal control value to send to the system using a cost function J created by the user, as shown below

$$\begin{cases} x_{k+1} = f(x(k), u(k)) \\ y = x_{k+1} \\ min \;\; J(x(k), u(\cdot)). \end{cases} \tag{2.1}$$

The system dynamics is given by the function f. The system states are x, the control input is u, and the output seen is y. The problem can also handle constraints of the system that might be needed for specific problems, such as penalizing control efforts. The optimal control problem is repeated based on a receding horizon. The receding horizon is a prediction of how the system will behave during a preset amount of time. The horizon should be long enough to account for the change $u(k+1)$ will have on the system [30]. Figure 2.1 illustrates how the receding horizon works.

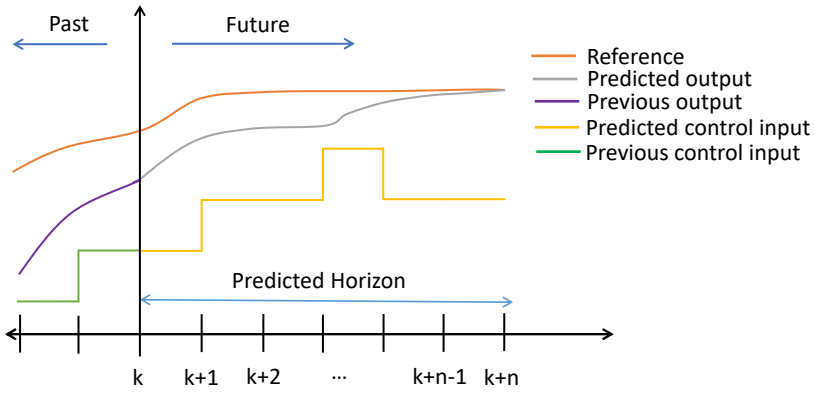

Figure 2.1: Example of the receding horizon/prediction

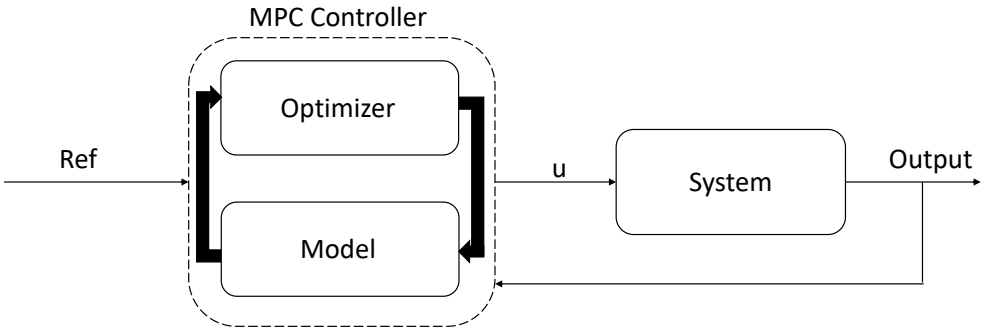

Figure 2.2: Block diagram depicting the MPC controller framework.

The diagram in 2.2 shows a simplified schematic of how the MPC controller works. A reference trajectory and output value are sent to the MPC controller. The optimizer decides the best control values over the receding horizon. Only the first input is used and applied to the system. The system responds, giving an output and closing the loop.

2.3 Data Driven Control

Due to difficulties in creating an accurate model of a complex system, a different approach to control has gained momentum. Data drive control (DDC) uses data, online or offline, to make adjustments to the control strategy without any model information of the system [34]. Some stability guarantees can still be made by making reasonable assumptions about the system and using a Lyapunov stability approach [34]. This is an advantage over requiring a model, where if the model doesn't capture all dynamics of the system, stability guarantees can diminish when a real-life application is done. Two methods will be discussed here, an online method where the control function is approximated and an offline method where parameters are tuned using offline data.

2.3.1 Offline - Proportional-Integral-Derivative (PID)

A PID controller is a control method that uses feedback to update the control signal. Each portion of the PID contributes to helping reach control objectives such as overshoot at a minimum, hitting steady state at a certain time, etc.

$$u(t) = k_P e(t) + k_I \int_{t_0}^{t} e(\tau)d\tau + k_D \frac{de}{dt} \tag{2.2}$$

The proportional control is related to the instantaneous system error and amplifies the control signal depending on the value of the proportional gain parameter k_P. The integral portion's goal is to integrate the error being received. Integrating the error allows integral feedback to minimize steady-state error and help deal with disturbances depending on the integral gain value k_I. Finally, the derivative feedback portion uses the derivative of the error. This helps speed up the transient response and reduce overshoot depending on the derivative gain value k_D [22].

Due to its simplicity and the rigorous study done on PIDs, it is the industry standard when control is required. Some techniques have been developed to tune the parameters k_P, k_I, k_D to achieve the control objectives desired. One such method is known as the Ziegler-Nichol method [42]. The method gives rules on how to tune each parameter based on finding consistent oscillations using k_P. This method of tuning is said to be appropriate for best rejecting disturbances.

2.3.2 Offline - Sliding Mode Control

Sliding mode control is a nonlinear control method known to be robust to uncertainties. This is due to the controller driving the system to a designed manifold/sliding surface [13]. Once the system is on this surface, the system 'slides' along the surface as the system gets pushed towards a desired reference.

Let the following system be a nonlinear affine system

$$\dot{x} = f(x) + g(x, u)$$
$$y = h(x). \tag{2.3}$$

First, a sliding surface needs to be created that the system will be driven to and used to get our desired outcome. A common surface chosen is

$$s = \dot{e} - ke \tag{2.4}$$

where $e = x - x_d$ with x_d being the desired value for the state x. Since the goal is to push the dynamics to the surface then $s = 0$ yields the wanted result being $\dot{e} = -ke$. This would mean that the error is converging to 0 and that the system is at a wanted set point. This leaves a second decision being the control law u, which needs to be designed in such a way as to make sure $s = 0$.

One major drawback to the sliding mode controller is the chattering phenomenon that can occur [101]. Due to the sliding mode controller being discontinuous, it can give control signals along the manifold that cause the system to oscillate around the reference. There have been many developments to counteract the chattering since quick changes to an actuator in a real-life system might cause harm (i.e., degrade performance).

2.3.3 Online - Radial Basis Functions Neural Network (RBF-NN)

An RBF-NN is used for function approximation due to its ability to universally approximate any function on a bounded compact set given enough neurons/centers [35]. An RBF is a three-layer neural network consisting of an input, hidden, and

output layer [29]. A radial basis function is a real-valued function defined by the distance between an input and its centers c_i, as shown below.

$$\Phi(x) = \Phi\|x - c_i\| \tag{2.5}$$

A typical radial basis function used in practice is the Gaussian function defined as

$$\Phi(x) = exp\frac{\|x - c_i\|^2}{2\beta^2} \tag{2.6}$$

where β is known as the width of the receptive field. The norm $\|\cdot\|$ is defined to be the two norm. This can be seen as a way of normalizing the data. This makes sense to do since, otherwise, the input values might be too large compared to the centers and cause the output of the Gaussian to be 0 at all times. The output of the RBF-NN is a linear combination of all radial basis functions $\Theta : \mathbb{R}^n \to \mathbb{R}$ defined as

$$\Theta(x) = \sum_{i=1}^{M} w_i \Phi(x) \tag{2.7}$$

where w_i are the weights associated with each radial basis function. The weights are updated through back-propagation. Figure 2.3 shows a diagram of the overall framework of the RBF-NN. For the control problem, no information about the system is given beforehand, and all parameters are randomized at the start. It takes in the output of the system to make real-time adjustments to the weights to approximate a control law to carry out the problem objective (i.e., a desired reference or trajectory).

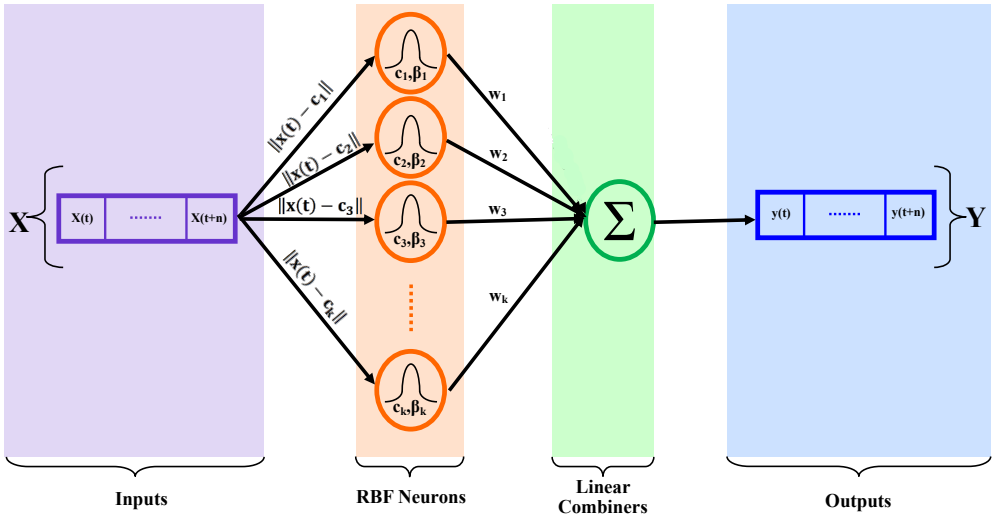

Figure 2.3: Diagram depicting how an RBF-NN works.

2.4 Bioelectronics

Galvani's experiments demonstrated the ability to stimulate a part of an organism by applying an electrical charge[ref]. This finding has lead to the field of bioelectronics. As advances in technology continue to grow, we see bioelectronics play more of a role in human lives. Devices have been able to shrink in size, allowing these devices to interface on a nano level to stimulate the cell membrane. Due to this, integrating bioelectronics with the medical field is possible and has already seen positive results through pacemakers, and drug delivery to stop seizures.

2.4.1 Proton pump

An ion pump is capable of delivering charged molecules. The ion pump is used and characterized as one that adds/removes protons. Figure 2.4 shows a schematic of the device. The ions travel through an ion bridge ((C) in Figure), which is connected to a reservoir ((A) in Figure) and a membrane that leads to the area

where pH change is being monitored ((B) in Figure), which is the target solution. It is capable of doing this by sending electrical signals to an electrode/electrodes ((D) in Figure), which allows for ions to go through a polymer membrane, which separates a reservoir containing a large number of ions and a target solution. A negative charge will remove ions from the place where pH is being monitored and back to the ion bridge/reservoir. A positive charge will have the reverse effect, pushing ions into the place where pH is being monitored. The amount of ions in the reservoir is considered to be infinite, meaning that moving ions to the target solution to reach a target pH value should not be limited by the supply of protons. It is important to note that the ion bridge has no fluids, strictly only allowing ion movement.

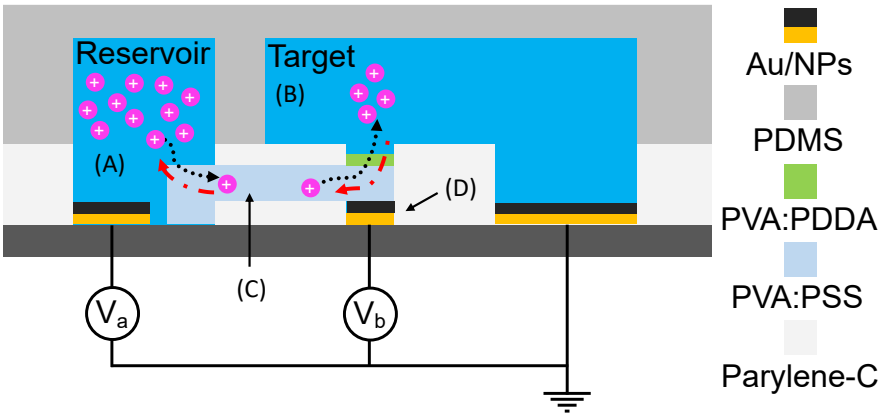

Figure 2.4: Diagram depicting how proton pump works.

Chapter 3

Machine learning for prediction

This chapter is from [57] as it appears in IEEE Symposium Series on Computational Intelligence (2019): pages 1120-125. The dissertation author was a co-first author of this paper.

3.1 Introduction

Synthetic biology has demonstrated the power of its approaches in many biomedical applications such as the synthesis of drug-delivery tools, the discovery of novel drugs, optogenetic systems based gene- and cell- therapies and so on [31, 96]. Modeling, identification, and prediction are key components of both systems and synthetic biology and, thus, necessary for their convergence [80]. They all are considered to be challenging tasks due to the complex nonlinear nature of biological systems [50]. In general, a model of a physical or biological system is a mathematical approximation (often in the form of differential equations and/or difference equations) of the underlying system which allows us to better

understand the rules governing a system and/or to predict the system behavior. For example, aiming at improving applications of optogenetic systems, Olsen and his colleagues developed an optogenetic 'function generator' method used to program custom time-varying gene expression levels in live bacterial cells [65]. Following their previous work, they developed a mathematical model of wavelength- and intensity-dependent photoconversion, signaling, and output gene expression for full spectral programming and multiplexing of optogenetic systems [66].

The precision of the model is the key feature required to accurately predict the system response. A clear advantage of a mechanistic model is its utility in engineering and modifying systems. However, a priori knowledge of the underlying chemical and genetic mechanisms governing the biological system is needed to build an accurate model. Furthermore, a lot of time might be spent determining the correct model parameter values only to have those values change under different conditions (e.g., changing environment or host strain for synthetic networks).

When a mechanistic model is not needed but accurate predictions are, researchers in systems biology can employ machine learning (ML)-based techniques, which are considered another way of doing system modeling, identification, and prediction using input/output data. Considering the necessity of identifying causal disease variants, Alipanahi et al. developed a deep learning-based algorithm, called DeepBind, to discover the sequence specificities of DNA- and RNA-binding proteins from large-scale experimental data [1]. Their proposed methods resulted in increased predictive power in comparison with the traditional techniques [71]. In a dynamical example, Ławryńczuk applied artificial neural networks to model predictive control (MPC) for temperature control in a yeast fermentation biochemical reactor [49]. Applications of neural networks have gained most popularity in chemical and biochemical processing industries due to

the complex nonlinear dynamics involved [53]. However, similar to the mechanistic modeling approach, one can only guarantee prediction under previously seen environmental conditions.

The availability of large datasets a priori is necessary for these methods and most of the ML-based techniques in order to successfully tackle modeling, identification, and prediction problems [5]. This makes them unsuitable for online implementation. Some Lyapunov-based online methods have been successfully employed in applications outside of biology [48, 103] in continuous-time. However, sampling times in biology are often sparse. In this paper, we extend an algorithm for a discrete-time approach using gradient descent for parameter updates on a neural network [51]. We demonstrate that the development of ML-based prediction strategies with less dependency on large datasets, and with low computational complexity is advantageous for future applications in controlling biology.

The main objective of this paper is to present an online ML-based technique that predicts the behavior of a biological system without a priori knowledge of the system dynamics or datasets. To accomplish this objective, we present an online ML-based predictor by leveraging the radial basis function artificial neural network (RBF-ANN). The presented approach predicts the output of the system one step ahead in an online manner. It has a low computational complexity, which makes it suitable for real-time implementation. The presented method is validated by using experimental data. Furthermore, the presented online predictor is shown to perform better than a commonly used offline neural network based approach.

Our main contributions are summarized as follows:

- Predicting the output of the system in an online manner (i.e., no offline training stage is needed).

- Low computational complexity (i.e., it can be implemented by using general

purpose computers with acceptable performance).

- Two algorithms are proposed to minimize the prediction error.

The rest of the paper is organized as follows. Section 3.2 presents details of the proposed methodology and the related preliminaries. The proposed method is utilized for predicting the response of optogenetic systems [66] in Section 3.3. The conclusion and future directions of our work are provided in Section 3.4.

3.2 Methodology

In this section, the methodology is highlighted in three stages. First, a brief description of the optogenetic two-component systems (TCS) model is given. Then, the required preliminaries of the RBF-ANN are provided. Finally, details of the proposed machine learning based predictor are elaborated.

3.2.1 Optogenetic Systems

In general, most optogenetic systems are based on a photoreceptor protein that generates a biological signal such as gene expression by externally applying light signals. They have been utilized in many biomedical applications such as the optogenetic systems based gene- and cell - therapies [96]. Aiming at improving the applications of optogenetic systems, an optogenetic 'function generator' method was developed and used to program custom time-varying gene expression levels in live bacterial cells [65]. Following that, a mathematical model of wavelength- and intensity-dependent photoconversion, signaling, and output gene expression systems was developed in [66].

Here, the experimental data from an in vivo optogenetic two-component systems (TCS) model is utilized (shown in Fig. 4.2). This system consists of a

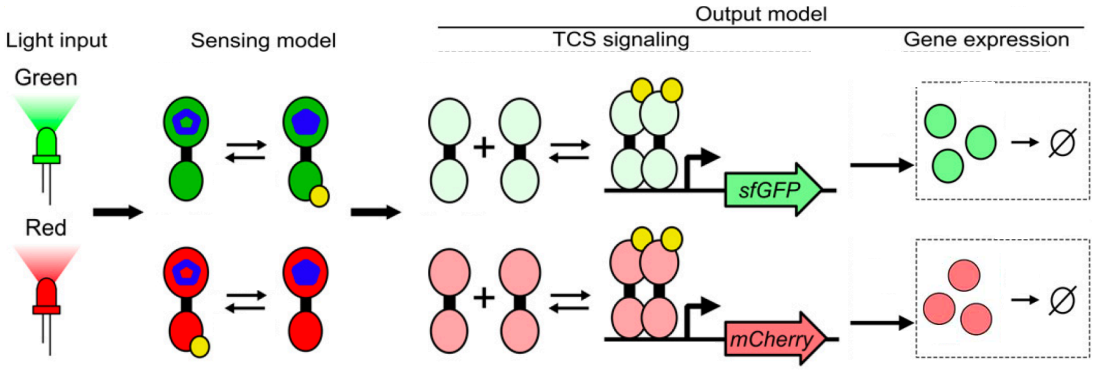

Figure 3.1: Model of optogenetic systems [66].

sensing and an output model. The sensing model converts light inputs into a ratio of photoreceptor populations, and the output model converts the photoreceptor populations into a gene expression signal. More detailed studies about the experimental setup can be found in [5], [66].

3.2.2 RBF Artificial Neural Network

Machine learning based approaches have been widely utilized by researchers from different disciplines aiming at improving the performance of diverse systems, including image processing, unmanned aircraft systems, pattern recognition, power systems, chemical processes, multi-agent systems, and so on [17, 25, 37, 39, 40, 41, 44, 49]. Among them, artificial neural network (ANN) based techniques have received wide interest by many researchers because of their inherent potential to deal with complex and unknown dynamical systems [27, 29, 62]. There are two major classes of ANN, (i) feedforward ANN (such as multi-layer perceptron (MLP), radial basis function (RBF) ANN, etc.), and (ii) feedback ANN (such as Hopfield ANN, etc.). Here the RBF-ANN is used, which is faster than the MLP-ANN and, therefore, more appropriate to be utilized for the online prediction and/or estimation of complex dynamical systems.

RBF-ANN, a feedforward ANN, has been shown to have the capabilities of global generalization and local specialization. This means that it can simultaneously capture both global and local attributes of the system, approximating input-to-output mapping. Furthermore, the capability of acting as a universal approximator makes RBF-ANN very promising for utilization in modeling, identification, and prediction of the unknown dynamical systems [70].

An RBF-ANN utilizes the radial basis functions as its activation functions. In general, the RBF-ANN consists of three different layers: (i) a hidden layer where its neurons have RBF activation functions, (ii) an output layer that is simply a linear combiner, and (iii) an input layer.

It has been shown that RBF-ANN has the capability of acting as the universal approximator on a compact subset of $\mathbb{R}^n$. In other words, an RBF-ANN (with enough number of neurons in its hidden layer) can approximate any continuous function on a closed bounded set. More detailed studies can be found in [70].

An RBF-ANN receives its input as a vector of real numbers $\mathbf{x}(t) \in \mathbb{R}^n$, and its output is defined by a scalar function of its input vector $\Phi : \mathbb{R}^n \to \mathbb{R}$. Many radial basis functions exist but here, a Gaussian function is considered with the norm (i.e., $\|\cdot\|$) defined by the Euclidean distance. The equation below shows the output of an RBF-ANN with the Gaussian function:

$$\Phi(\mathbf{x}(t)) = \sum_{i=1}^{M} \mathbf{w}_i(t)\Theta(||\mathbf{x}(t) - \mathbf{c}_i(t)||) + b(t) \tag{3.1}$$

$$\Theta\Big(\, \|\mathbf{x}(t) - \mathbf{c}_i(t)\| \,\Big) = e^{\frac{-\|\mathbf{x}(t)-\mathbf{c}_i(t)\|^2}{2\beta_i^2(t)}}, \tag{3.2}$$

where the number of neurons in the hidden layer is shown by M, the center vector for the i^{th} neuron is shown by $\mathbf{c}_i(t)$, the weight of the i^{th} neuron in the linear output neuron is shown by $\mathbf{w}_i(t)$, the bias term is shown by $b(t)$, and finally, $\beta_i(t)$ is a width of the receptive field (i.e., $\beta_i^2(t)$; variance of the Gaussian function). More detailed studies about neural networks can be found in [29].

3.2.3 Machine Learning based Predictor

Generally, the objective in prediction is to use the previous and current inputs and the past states/outputs to decide what the next states/outputs will be. It is known that modern machine learning techniques, such as deep learning, promise to leverage large-scale datasets for making accurate predictions [5]. However, the techniques are not suitable when there is insufficient data for offline training, unknown influences, such as intrinsic and extrinsic variability, not captured in the data, or limited access to high performance computing systems.

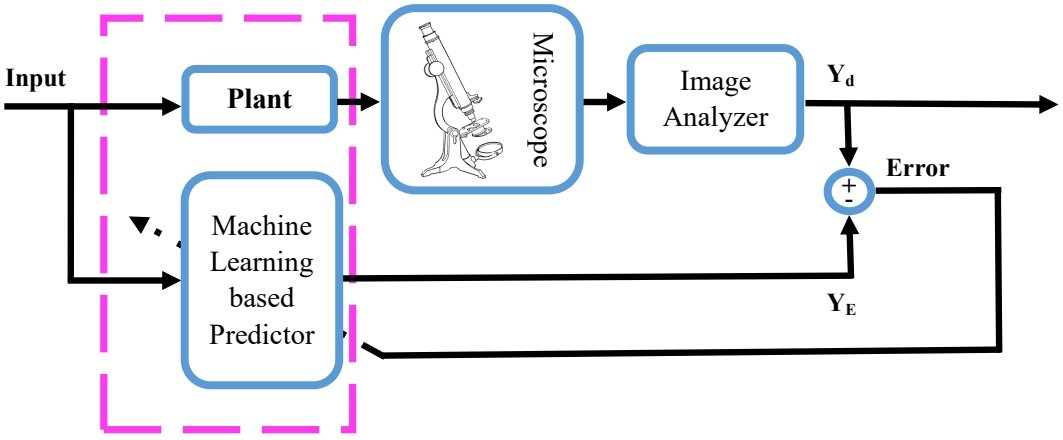

Figure 3.2: Online machine learning-based predictor architecture.

In order to address these limitations, we present an online machine learning-based methodology (shown in Fig. 3.2, where Y_d is the output of the system,

and Y_E is the predicted output of the system.) to predict the output of biological systems without a priori knowledge of their dynamical models and with no dependency on the large-scale datasets, while it can be implemented on general purpose computing systems. By leveraging the RBF-ANN introduced in Section 3.2.2, the details of the proposed algorithm is presented below.

In order to improve predictions, whenever new information is presented to the system, adjustments are made to the ML-based predictor's parameters. In other words, prediction happens in real-time, which simply means that the model is only learning from the information provided up until the current time-step and does not require seeing the whole dataset. Therefore, the cost function to be minimized is the total instantaneous error energy

$$\mathbf{E}(t) = \frac{1}{2}e^2(t), \tag{3.3}$$

where $e(t)$ is the prediction error (i.e., $e(t) = Y(t) - \Phi(\mathbf{x}(t))$) and $Y(t)$ is the output of the system. The updates to the ML-based predictor's parameters are made by computing the appropriate gradients (i.e., $\frac{\partial \mathbf{E}}{\partial \mathbf{W}}, \frac{\partial \mathbf{E}}{\partial \mathbf{C}}, \frac{\partial \mathbf{E}}{\partial b}, \frac{\partial \mathbf{E}}{\partial \beta}$). Here, only the centers $\mathbf{c}_i(t)$, the weights $\mathbf{w}_i(t)$, and the bias term $b(t)$ are updated by using equations

$$\begin{aligned}\mathbf{c}_i(t+1) &= \mathbf{c}_i(t) \\ &+ \eta_c \left(e(t)\mathbf{w}_i(t)\Theta\Big(\|\mathbf{x}(t) - \mathbf{c}_i(t)\| \Big) \frac{-\|\mathbf{x}(t) - \mathbf{c}_i(t)\|^2}{2\beta_i^2(t)} \right),\end{aligned} \tag{3.4}$$

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta_w \Big(e(t)\Theta\Big(\|\mathbf{x}(t) - \mathbf{c}_i(t)\| \Big) \Big), \tag{3.5}$$

and

$$b(t+1) = b(t) + \eta_b e(t), \tag{3.6}$$

respectively. Here, η_c, η_w, and η_b are the learning rates (η_c, η_w, $\eta_b > 0$). However, the width of the receptive fields $\beta_i(t)$ could also be updated if needed.

Algorithm 1 : The Method 1 for Minimization.

Read the current outputs/states $Y(t)$ for updating the NN parameters
while $e(t) > \epsilon$ **do**
  Compute $\Phi(\mathbf{x}(t)) = \sum_{i=1}^{M} \mathbf{w}_i(t)\Theta(||\mathbf{x}(t) - \mathbf{c}_i(t)||) + b(t)$
  Compute $e(t) = Y(t) - \Phi(\mathbf{x}(t))$
  Update $\mathbf{w}_i(t)$
  Update $\mathbf{c}_i(t)$
  Update $b(t)$
end while

Algorithm 2 : The Method 2 for Minimization.

$\alpha - - > scaling\ factor$
Read the current outputs/states $Y(t)$ for updating the NN parameters
Compute $\frac{dY(t)}{dt} = (Y(t) - Y(t-1))/t_s$
Compute $Y_{new}(t) = (\alpha \frac{dY(t)}{dt} + Y(t))$
while $e(t) > \epsilon$ **do**
  Compute $\Phi(\mathbf{x}(t)) = \sum_{i=1}^{M} \mathbf{w}_i(t)\Theta(||\mathbf{x}(t) - \mathbf{c}_i(t)||) + b(t)$
  Compute $e(t) = Y_{new}(t) - \Phi(\mathbf{x}(t))$
  Update $\mathbf{w}_i(t)$
  Update $\mathbf{c}_i(t)$
  Update $b(t)$
end while

Two algorithms are proposed: **Algorithm** 1 and **Algorithm** 2, for minimizing the prediction error before the next prediction. They provide the capability of real-time error correction for improving the performance of the proposed ML-based predictor.

Algorithm 3 : The Machine Learning Based Methodology for Online Prediction.

Initialization:
Set $\mathbf{w}_i = rand$, and $b = rand$, for $i = 1, 2, ..., M$.
Define $\mathbf{c}_i = rand$, and $\beta_i^2 = predefined\ value$ for $i = 1, 2, ..., M$.
Define $\epsilon = minimization\ criteria$.

for each iteration $t = t_s$ **do**
  Read the current inputs and past outputs/states $\mathbf{x}(t)$
  Return the predicted output
  Minimize $e(t)$ Using **Algorithm** 1 or **Algorithm** 2
end for

The idea of using **Algorithm** 1 is simply to minimize the prediction error based on the current input/output data in order to better estimate the next output. We found that this method can sometimes lead to a lag in accurate prediction of system behavior. In a preliminary attempt to deal with such a delay in prediction, we developed **Algorithm** 2, which initially computes the derivative (the instantaneous rate of change) of the output with respect to time to define an initial guess to the next output. Current input information is not yet given. Then, the minimization is applied with respect to this guess and the updated parameters, accordingly, are used to predict the next output. The results of utilization of both minimization algorithms are provided in Section 3.3. **Algorithm** 3 is summarizing the overall online machine learning based predictor methodology proposed in this paper as pseudo-code.

3.3 Results and Discussion

This section presents the results obtained by applying the proposed online ML-based predictor to the experimental data of optogenetic systems provided in (**Supplementary Materials, Dataset EV8**) [66]. This data consists of the multiple sets of inputs (two different light intensities ($\mu mol\ m^{-2}s^{-1}$)) and their

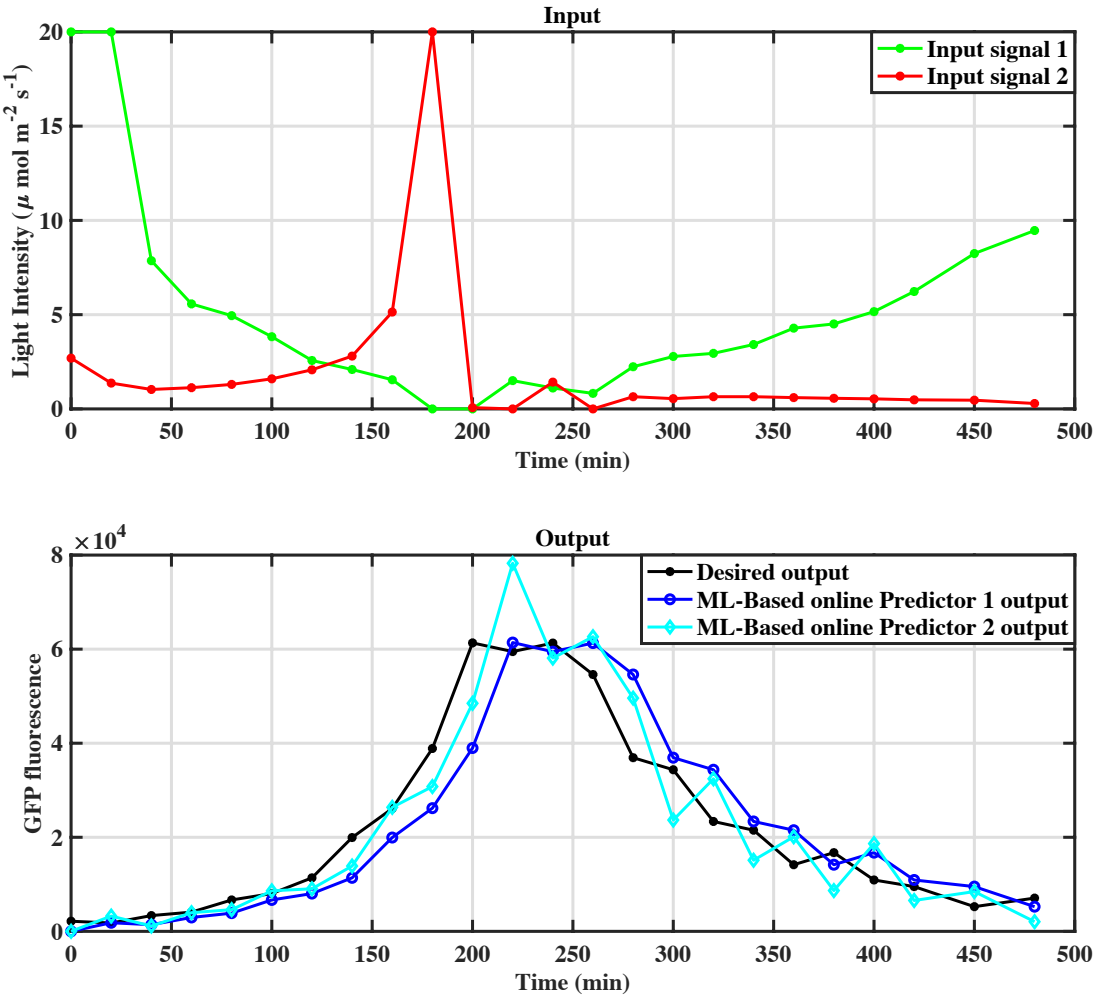

Figure 3.3: The online prediction of the sfGFP fluorescence using Algorithm 1 and Algorithm 2.
(Top) Two different light intensities ($\mu mol\ m^{-2}s^{-1}$) are in Green and Red colors. (Bottom) The desired output is in Black color, and the predicted responses of applying **Algorithm** 1 and **Algorithm** 2 are in Blue and Cyan colors respectively.

corresponding outputs (measured fluorescence of sfGFP and mCherry). The results are presented under three different scenarios: i) online prediction using **Algorithm** 1, ii) online prediction using **Algorithm** 2, and iii) prediction using an offline training methodology. In all scenarios, there are a total number of 24 data points that are sampled during the 480 minutes of the experiment. All simula-

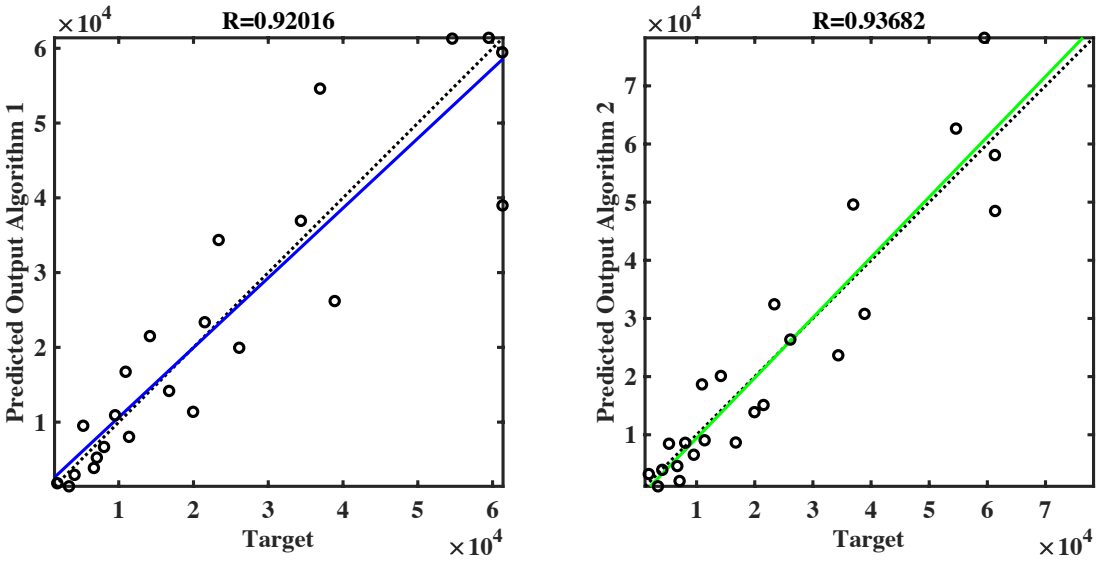

Figure 3.4: Prediction accuracy for the online prediction of the sfGFP fluorescence using Algorithm 1 (left) and Algorithm 2 (right).
Target is output is along x-axis and predicted output is along the y-axis. Data points are plotted with open circles, and colored lines are a linear regression fit to the data. The dotted black line is the one-to-one line, which is ideal accuracy. Note that **Algorithm** 2 outperforms **Algorithm** 1 with a regression line closer to the one-to-one line and a correlation coefficient, R, closer to 1.

tions are carried out on a platform with the following specifications: MacBook Pro (macOS 10.14.1), Processor: 2.3 GHz Intel Core i5, RAM: 16.00 GB.

Generally, the configuration of the RBF-ANN differs for different problems [29]. In this paper, in order to ensure a fair comparison, all predictors (online/offline) are designed with the same number of nodes in their input, hidden, and output layers. All weights and biases are initialized with random values, and all predictors are initialized with the same centers and receptive fields values. In the offline scenario, 75% of the dataset (i.e., first 18 points) is used for training and 25% (i.e., last 6 points) for testing (i.e., prediction). It should be noted that the offline training phase is not needed or used for the proposed method. The input vectors for all predictors consist of the same 9 components, the current and two previous

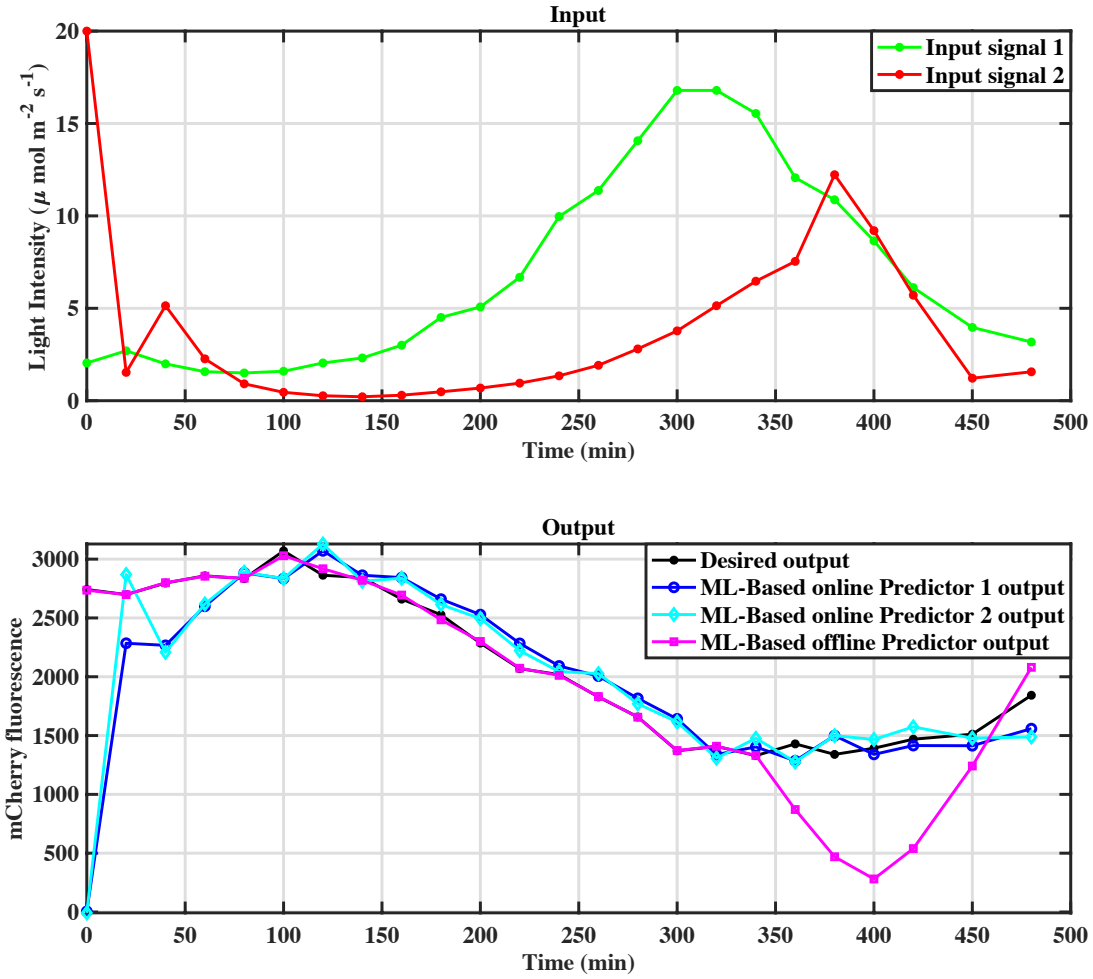

Figure 3.5: The online prediction of the mCherry fluorescence using Algorithm 1 and Algorithm 2.
(Top) Two different light intensities ($\mu mol\ m^{-2}s^{-1}$) are in Green and Red colors. (Bottom) The desired output is in Black color and the predicted responses of applying **Algorithm** 1, **Algorithm** 2, and offline prediction are in Blue, Cyan, and Magenta colors respectively. Note that in the offline method, 75% of datasets (i.e., first 18 points) is used for training and 25% (i.e., last 6 points) for testing (i.e., prediction).

inputs (i.e., light intensities 1 and 2) and three previous outputs.

Fig. 3.3 shows the online prediction of the sfGFP fluorescence by using **Algorithm** 1 and **Algorithm** 2. Fig. 3.5 shows the online prediction of the mCherry fluorescence for a different set of experiments by using **Algorithm** 1, **Algo-**

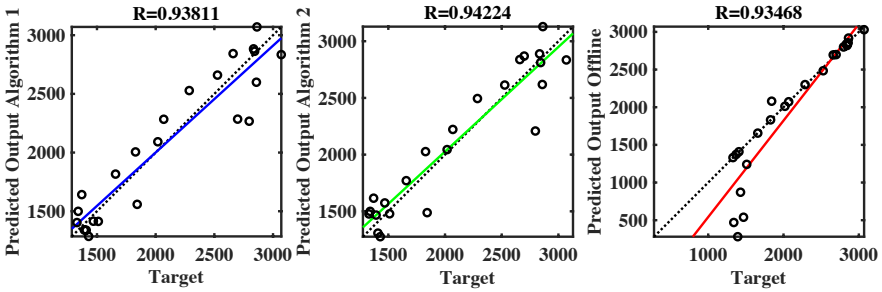

Figure 3.6: Prediction accuracy for the prediction of the mCherry fluorescence using online Algorithm 1 (left), online Algorithm 2 (middle), and the offline prediction (right).
Target output is along x-axis and predicted output is along the y-axis. Data points are plotted with open circles, and colored lines are a linear regression fit to the data. The dotted black line is the one-to-one line, which is ideal accuracy. Note that **Algorithm** 2 outperforms **Algorithm** 1 and the offline prediction with a regression line that is closer to the one-to-one line and a correlation coefficient, R, that is closest to 1.

rithm 2 and also the offline prediction method.

To rigorously evaluate the performance of the proposed algorithms, the linear regression between target and predicted values are plotted in Figs. 3.4 and 3.6. These figures demonstrate that the proposed method has successfully predicted the outputs of the optogenetic systems and the predictions are good given sparse data. It should be mentioned that, although the prediction of the offline method is very good for the previously observed data points, it performs poorly for the new data points (i.e., the data points which are not within the training domain). It is simply due to the fact that there is no real-time error correction in the offline method. Furthermore, comparing the results of applying **Algorithm** 1 and **Algorithm** 2, it is shown that, while **Algorithm** 1 might predict the output with a slight lag whenever the variations of the outputs are not smooth, **Algorithm** 2 has a reduced lag in this instance (see Fig. 3.3). However, **Algorithm** 2 can result in many under and over shoots in prediction. We will aim to improve on

this method.

Moreover, the accuracy and effectiveness of proposed algorithms are demonstrated using three different numerical metrics. The Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) are fairly common methods used when discussing accuracy. In addition, the Accuracy Ratio (AR) is a different way of calculating accuracy that has been shown to perform better than MAPE [92].

Tables 3.1, 3.2, and 8.2 show the
$\text{MAE} = \dfrac{\sum_{j=1}^{l} |Y_j(t) - \Phi_j(\mathbf{x}(t))|}{l} = \dfrac{\sum_{j=1}^{l} |e_j(t)|}{l}$,
$\text{MAPE} = \dfrac{100\%}{l} \sum_{j=1}^{l} \left| \dfrac{Y_j(t) - \Phi_j(\mathbf{x}(t))}{Y_j(t)} \right|$, and $\text{AR} = \sum_{j=1}^{l} \log \left(\dfrac{\text{predicted}}{\text{actual}} \right)^2$
values for all three methods in experiment 2 (i.e., Fig. 3.5) respectively.

Table 3.1: Mean Absolute Error (MAE).

Method	MAE (Excluding the initial point)	MAE (Last 6 data points)
Algorithm 1	170.9599	131.8767
Algorithm 2	163.4260	146.6755
Offline	175.8645	662.8856

Table 3.2: Mean Absolute Percentage Error (MAPE).

Method	MAPE (Excluding the initial point)	MAPE (Last 6 data points)
Algorithm 1	8.2621	8.5456
Algorithm 2	8.0827	9.4283
Offline	11.9670	46.3159

Table 3.3: log of the Accuracy Ratio (AR).

Method	AR (Excluding the initial point)	AR (Last 6 data points)
Algorithm 1	0.2349	0.0592
Algorithm 2	0.2379	0.0791
Offline	4.9666	4.9655

In all tables, in the second column, the error associated with the initial point is eliminated. This is due to the fact that the initial point is randomized by the values given for weights and biases from the random number generator, and, therefore, the algorithms have not begun “learning". In the third column, only the last 6 data points are chosen due to the fact that this is when the offline algorithm is finally predicting the unseen data points (the first 18 points were used to train the offline algorithm).

Our proposed online ML-based methodology is advantageous to current alternatives by the following key criteria:

- No offline training phase is needed.
- Implementable on general purpose computers with a promising performance.
- Minimization of the prediction error.
- Real-time error correction.

3.4 Conclusion and Future Work

An online machine learning-based technique is presented in this paper to predict the output of biological systems without a priori knowledge of their dynamical models and with no dependency on large-scale datasets. Considering its learning capability and low computational complexity, the proposed technique is a promising tool for practical and real-time implementation for biological systems. The prediction results demonstrate the effectiveness of the presented method.

Future work will consider improvements on the presented methodology for predicting the output of biological systems with better accuracy and more than one step ahead. In addition, the proposed algorithm will be integrated with

a machine learning based control strategy for real-time control of a biological system.

Acknowledgment

Research was sponsored by the Army Research Office and the DARPA Biotechnologies Office (DARPA/BTO) and was accomplished under Cooperative Agreement Number W911NF-18-2-0104. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office and the DARPA Biotechnologies Office (DARPA/BTO) or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

The authors would like to thank Evan J Olson for his guidance on finding experimental data to test our algorithm on.

Chapter 4

pH control on a bioelectronic device using machine learning

This chapter uses portions of [38] as it appears in IEEE Control Systems Letters 5, no. 4, August (2020): pages 1133-1138. Mohammad Jafari was the primary author, and Marcella Gomez is a co-author of this paper. The dissertation author was a contributor to this paper.

4.1 Introduction

Advancements in technology has allowed for further integration between man made devices and biological systems. Bioelectronics devices have been created that allow for monitoring blood sugar levels, controlling stem cell fate, applying electrical stimulation and deliverying therapeutic drugs [67][26][84]. Ion pumps are bioeletronic devices that allow for the movement of ions and charged drugs to move towards a targeted area using electrical signals. They are capable of helping

in the case of precision medicine where applying a concentration of a drug at a specific region and time is more effective than current drug delivery methods such as digested pills [90].

Ion movement is important in a biological system because a system in general has a healthy range of pH values it stays between. Being outside of the healthy range of pH could be harmful and cause sickness or even death [87]. Delivery of ion H+ is capable of helping to maintain a healthy pH balance. Maintaining these healthy pH values can help with gene expression and maintaining a healthy neuronal function [80].

One area where bioelectronics come to play is feedback control. Biological systems tend to have a form of feedback where it self regulates to stay in a healthy state. Feedback control is when after sensing a change, the system adjusts itself to push itself back to its steady state. An example of this is during blood clotting where a signal goes out once the body senses a cut that activates blood platelets which in turn stop bleeding and form blood clots [54]. Using bioelectronics it is possible to reproduce this type of feedback. Such an application was demonstrated in the control of the resting potential of cells [38].

As technology becomes more advanced more time is needed to thoroughly test devices. Though well monitored experiments are necessary to make conclusive results *in silco* experiments are a means of helping bridge the gap when proper experimentation is slow/expensive to iterate. This could lead to slowing down the process of tuning parameters in a feedback control algorithm. This requires a predictive model.

Trying to model these devices that interact with biology can be a difficult task. Being unable to fully understand the system can lead to a poor performance of the model when compared to the actual data. The devices have unforeseen hidden

dynamics that aren't understood. Manufacturing differences between devices lead to differing responses requiring a more specific model per device.

Due to these difficulties, a machine learning approach leveraging a Radial Basis Function (RBF) Neural Network. An RBF is used for control here since priori knowledge of the dynamical system is not required and due to its low complexity. This makes it feasible as a controller where a model isn't available and real-time input values for the system are needed every few seconds. To verify that the controller will successful on the experimental setup, a pump model capturing similar challenges is used.

4.2 Methods

In this section, we highlight the methodology in three parts describing the plant, sensor, and controller. First, a brief description of the bioelectronic proton (H^+) pump system is provided. Second, we introduce the fluorescent dye used for imaging. Finally, the details of the proposed machine learning-based feedback controller are elaborated.

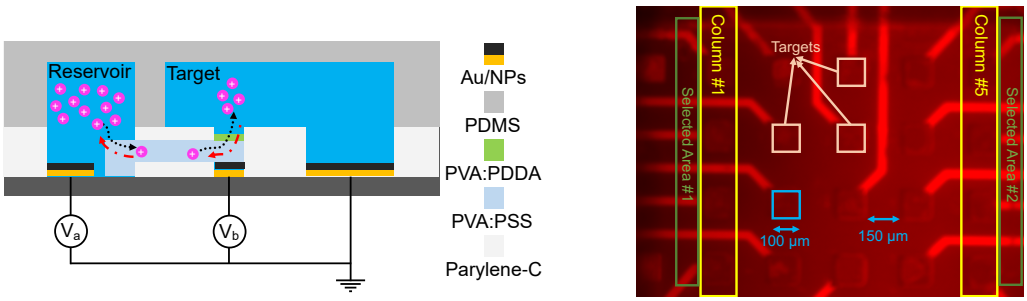

Figure 4.1: Bioelectronic proton (H^+) pump system. Top: Schematic of the bioelectronic proton (H^+) pump with applied voltages. Bottom: Top view of an array of targets for the proton pump system. Actuated columns and selected areas for measurements are indicated.

Ion (H^+) pump system

A proton (H^+) pump (shown in Fig. 4.1–Top) is a bioelectronic device that can be used to repetitively control and maintain the pH level in biological processes [90]. Here, the bioelectronic platform consists of multiple columns of proton (H^+) pumps, which are devices capable of modulating the pH of solution, and work by transferring H^+ between polymer membranes within the device and a target solution (labeled as "target" in Fig. 4.1–Top) upon an applied voltage (i.e., V_b in Fig. 4.1–Top). A positive value for V_b drives the release of H^+ from the H^+ cation exchange membrane into the solution (see black dotted arrow in Fig. 4.1–Top) thus increasing H^+ of the solution and a negative value for V_b will drive H^+ from the solution into the cation exchange membrane (see red dashed-dotted arrow in Fig. 4.1–Top) thus decreasing H^+ of the solution. The proton pumps are positioned in an array with $100\mu m^2$ pixel size and are individually accessible through a custom-designed expansion board using a microcontroller. Figure 4.1–Bottom, shows the top view of proton pumps distribution in the target demonstrating different columns, selected areas for measurements, and the corresponding dimensions. Microscope-based real-time imaging is used for monitoring the H^+ changes over the target. More detailed studies about the bioelectronic proton (H^+) pump systems (such as the details of all the layers in Fig. 4.1–Top, fabrication process, etc.) can be found in [90].

Sensing pH

Measurement of media pH on the surface of the ion pump is a challenging task due to the small volume and requires specialized instrumentation. In this work, we instead read out changes in pH using Carboxy SNARF-1 dye, a fluorescent pH indicator developed by Molecular Probes, useful for measuring pH changes

between pH 7 and pH 8. Common applications of Carboxy SNARF-1 dye are in biology, where intracellular pH is generally between 6.8 and 7.4 in the cytosol. Additionally, pH inside a cell varies by only fractions of a pH unit, and such changes may be quite slow. Since small changes in pH can have significant impact on cellular response, there is a need for tight control within this small range of pH for applications in synthetic biology. Roughly speaking, the relevant range of pH for cells corresponds to an overall 10% change in fluorescence [56]. In this work, we demonstrate control with less than 5% error and active control over a 10% change in fluorescence.

Machine Learning based Controller

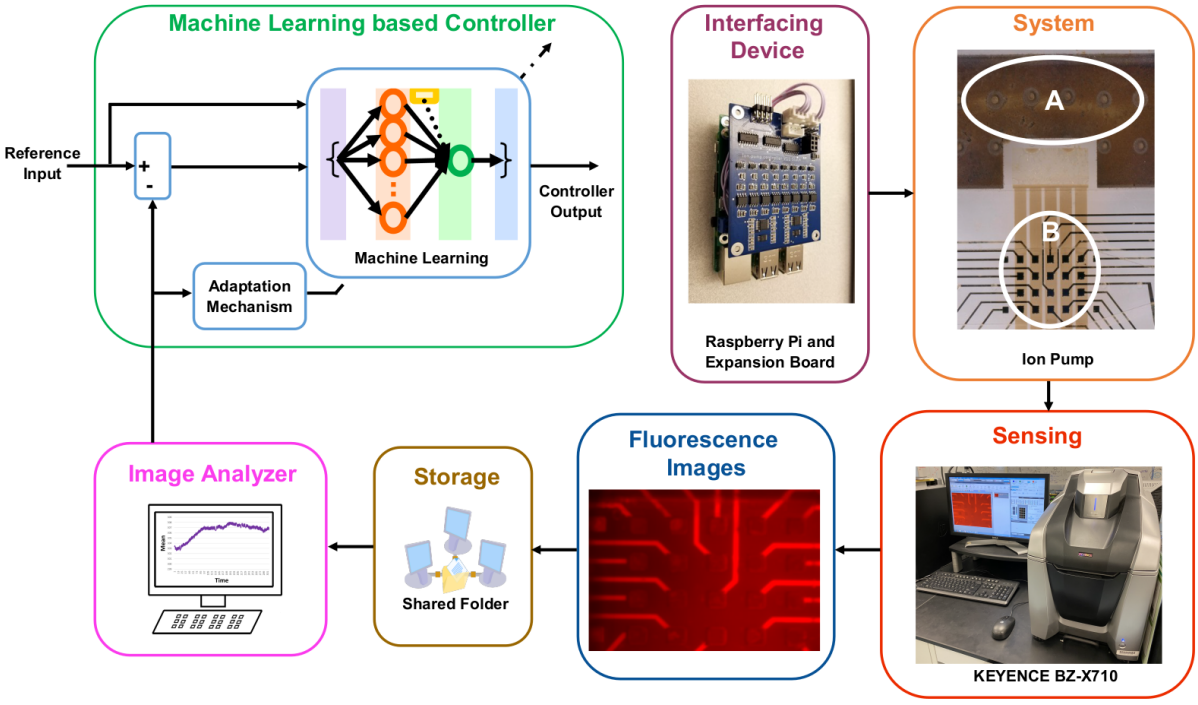

Figure 4.2: Architecture of the implemented online machine learning based feedback controller designed for the bioelectronic proton (H^+) pump system.

Figure 4.2 shows the architecture of the implemented online machine learning-

based feedback controller designed for the bioelectronic proton (H^+) pump system. The system consists of the bioelectronic proton pump where “A" is a reservoir for H^+ ions (e.g., water (H_2O)) and “B" is a medium with appropriate dye (e.g., water with SNARF-1 dye). The sensing system consists of the KEYENCE BZ-X710 microscope imaging system, which takes fluorescent images in real-time and sends them to a shared folder. The image analyzer reads images from the shared folder using MATLAB scripts and computes the mean value of the selected area of the proton pumps distribution in the target. This value then is used as feedback of current states of the system to update the ML-based control output. Machine learning (ML) based controller consists of an ML-based control algorithm that is running on a PC. The interfacing device consists of a Raspberry Pi and an expansion board which applies the commands received from the ML-based controller to the bioelectronic proton pump.

To design a feedback controller for the pH level control in a bioelectronic proton (H^+) pump system, we formulate the problem as a tracking control problem. In a tracking control problem, our interest is in a certain output variable rather than the states [86].

Let us consider the following representative nonlinear dynamical system:

$$\begin{cases} \dot{x} = f(x,u) + \xi(t) \\ y = x, \end{cases} \tag{4.1}$$

where x is the system state, u is the system input, y is the system output, $||\xi(t)|| < \xi_{max}$ is a continuous uniformly bounded time-dependent disturbance, and f is defined within a bounded domain $V_{min} < u < V_{max}$ ($V_{min} < 0$, $V_{max} > 0$) to be a continuously differentiable and strictly monotonic function in u such that

$f(x,0) = 0$ and $a_0 < f(x,u) < a_1$ $(a_0 < 0, a_1 > 0)$. As mentioned previously, positive voltages deliver ions, while negative voltages remove ions. It follows that the system response $x(t) = g(x_0, t; [u], [\xi])$ is also continuous and monotonic in u in the sense that for constant u, $u_1 > u_2 \implies g(x_0, t; u_1, [\xi]) > g(x_0, t; u_2, [\xi]) \forall t$ [4]. Furthermore, the biolectronic device is inherently internally stable. The equilibrium point x^* for the noiseless system satisfying $f(x^*, u) = 0$ is a monotonic function of u and is a stable equilibrium point. However, changes in f can be largely sensitive or insensitive to changes in u depending on the range of u, making control difficult. We are interested in having system (4.1) track a reference trajectory $r(t)$. Given the reference trajectory $r(t)$, there exists $u^*(t)$ such that

$$\dot{r} = f(x, u^*) + \xi(t). \tag{4.2}$$

for $(\dot{r} - \xi(t)) \in [a_0, a_1]$.

Let us assume the existence of the above-mentioned control law. The objective is to design u_{nn} that can approximate u^* which drives system (4.1) towards the desired reference trajectory assuming it is a reachable state. The error dynamics are given by

$$\begin{aligned} \dot{e} &= \dot{r} - \dot{x} \\ &= \dot{r} - f(x, u_{nn}) - \xi(t) \\ &= f(x, u^*) - f(x, u_{nn}) \end{aligned} \tag{4.3}$$

and since f is an invertible function, we have $\dot{e} \to 0$ implies $u_{nn} \to u^*$.

The universal approximation capability of the RBF network is leveraged to design u_{nn} [70]. The output of an RBF network with the Gaussian function is shown in the following equation:

$$\Phi(\mathbf{z}(t)) = \sum_{i=1}^{M} w_i(t)\Theta(||\mathbf{z}(t) - \mathbf{c}_i(t)||) + b(t) \tag{4.4}$$

$$\Theta\Big(\|\mathbf{z}(t) - \mathbf{c}_i(t)\| \Big) = e^{\frac{-\|\mathbf{z}(t)-\mathbf{c}_i(t)\|^2}{2\beta_i^2(t)}}, \tag{4.5}$$

where M is the number of neurons in the hidden layer, $\mathbf{z}(t)$ is the RBF network input vector, $\mathbf{c}_i(t)$ is the center vector for the i^{th} neuron, $\mathbf{w}_i(t)$ is the weight of the i^{th} neuron in the linear output layer, $\beta_i(t)$ is a width of the receptive field (i.e., $\beta_i^2(t)$, variance of the Gaussian function), and $b(t)$ is the bias term. More detailed studies about neural networks can be found in [29].

Let us define u_{nn} by using an RBF network as follows:

$$u_{nn}(t) = \Phi(\mathbf{z}(t)) = \sum_{i=1}^{M} w_i(t)\Theta(||\mathbf{z}(t) - \mathbf{c}_i(t)||) + b(t). \tag{4.6}$$

Let us choose $\dot{\mathbf{c}}_i$, $\dot{w}_i$, and $\dot{b}$, respectively, as

$$\dot{\mathbf{c}}_i(t) = e(t)w_i(t)\Theta\Big(\|\mathbf{z}(t) - \mathbf{c}_i(t)\| \Big)\frac{(\mathbf{z}(t) - \mathbf{c}_i(t))}{\beta_i^2(t)}, \tag{4.7}$$

$$\dot{w}_i(t) = e(t)\Theta\Big(\|\mathbf{z}(t) - \mathbf{c}_i(t)\| \Big), \tag{4.8}$$

$$\dot{b}(t) = e(t). \tag{4.9}$$

For implementation purposes, we use the update laws

$$\dot{\mathbf{c}}_i(t) \approx \frac{\mathbf{c}_i(t+T_s) - \mathbf{c}_i(t)}{\eta_c}, \tag{4.10}$$

$$\dot{w}_i(t) \approx \frac{w_i(t+T_s) - w_i(t)}{\eta_w}, \tag{4.11}$$

and

$$\dot{b}(t) \approx \frac{b(t+T_s) - b(t)}{\eta_b}, \tag{4.12}$$

where $\eta_{c,w,b}$ are the learning rates ($\eta_{c,w,b} > 0$) and T_s is the sampling time. Note that updates 4.7, 4.8, and 4.9 are partial derivatives of the cost function $J = \frac{1}{2}e^2$ with respect to parameters $\mathbf{c}, w$, and b. That is, with 4.10, 4.11, and 4.12, the discrete-time parameter updates are given by

$$\mathbf{c}_i(n+1) = \mathbf{c}_i(n) - \eta_c \nabla_{\mathbf{c}_i} J(\mathbf{c}_i(n), w_i(n), b(n)), \tag{4.13}$$

$$w_i(n+1) = w_i(n) - \eta_w \nabla_{w_i} J(\mathbf{c}_i(n), w_i(n), b(n)), \tag{4.14}$$

$$b(n+1) = b(n) - \eta_b \nabla_b J(\mathbf{c}_i(n), w_i(n), b(n)). \tag{4.15}$$

Here $\nabla_{[]} J = e\frac{\partial(r-y)}{\partial[]} = -e\frac{\partial y}{\partial u_{nn}}\frac{\partial u_{nn}}{[]}$. Since $\frac{\partial y}{\partial u_{nn}}$ is unknown, we set $\frac{\partial y}{\partial u_{nn}} = 1$. Note that for finite M in 4.6, J is continuously differentiable with $\nabla J < \infty$, and, hence, is Lipschitz continuous with respect to parameters c, w, and b. This means that

for small enough learning rates η_c, η_w, and η_b, gradient descent decreases J.

4.3 Results

4.3.1 *In Silico*

Model

To be able to verify that the RBF controller should be successful in the pH control *in vitro* experiment, a water pump model is used for the system shown below for *in silico* simulations..

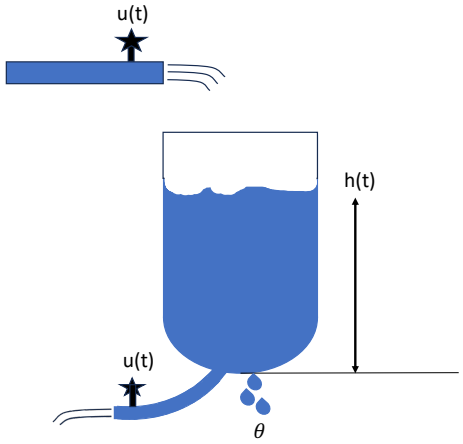

Figure 4.3: Figure of pump system.

$$\frac{dh(t)}{dt} = \frac{-k\sqrt{(h(t))}}{A(h(t))} + \frac{\xi}{A(h(t))}u(t) - \theta \tag{4.16}$$

Here, $h(t)$ is the height of the water in the tank, ξ is the parameter if there is clogging in the actuator pump, and θ is a constant term to account for a leak in the tank. The function $A(h(t))$ is the cross-sectional area of the tank and

$k = \rho a\sqrt{2g}$ where ρ is the parameter that deals with the output of the pipe, a is the cross-sectional of the pipe and $g = 9.8$ for gravity. The control input into the tank system is denoted by $u(t)$.

This model is seen as having similar challenges we will see in the *in vitro* experiments. This is due to the controller having to pick different values for adding water into the tank and taking water out of the tank. This is similar to how it is easier to add ions into the media than it is to remove them. There is also a leakage term that can be similar to the natural diffusion of the ions after being added to the area we are monitoring for mean fluorescence intensity.

Results

For the *in silico* study, three different references are chosen: square wave, sine wave, and constant reference. This is to show that we can control a wide array of control strategies that might be needed for *in vitro* studies. The control effort is kept between [0,4] as the ion pump does have bounded constraints on the allowed control effort.

Figure 4.4 shows the simulation results of tracking the three different references. The stop plots are the system response, and the bottom is the control effort. The left plots are with a square wave reference with an amplitude of 1.25 around 16.125, the middle plots are with a sinewave reference with an amplitude of 2.5 around 17.5, and the right plots are a constant reference held at 17.5. After the transient response, we see that the system is kept at the wanted reference value throughout the simulation. There is a slight reappearance of transient response in the square wave plot due to the student change in reference value requiring a quick change in control effort.

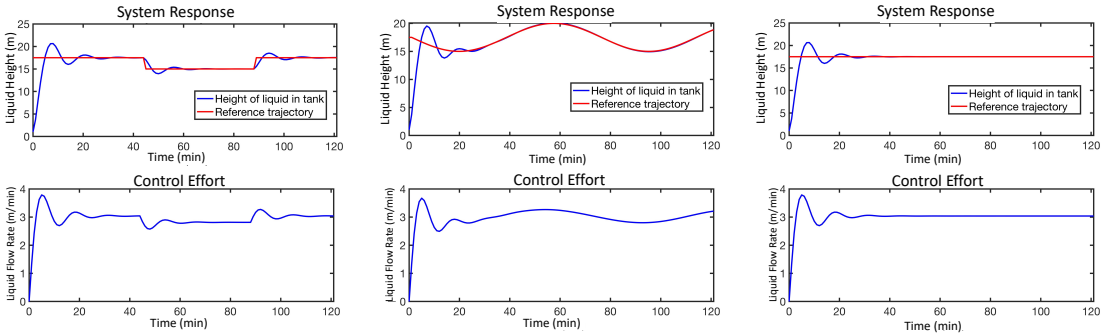

Figure 4.4: *In silico* results of different references being tracked by the RBF-NN controller.

4.3.2 *In Vitro*

Some of the data gathered is shown in Fig.3-8, wherein each figure the top panel shows a reference trajectory (blue) and the actual output of the system (red). The lower panel shows the voltage/input being applied to the system.

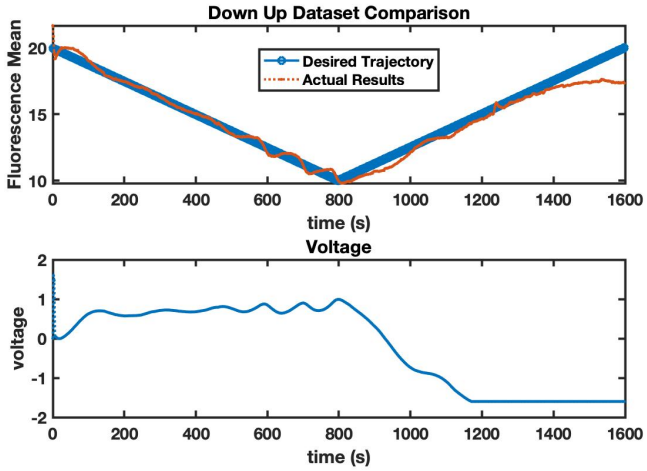

Figure 4.5: Closed loop experiment of controller tracking a reference signal linearly decreasing followed by a linearly increasing trend. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

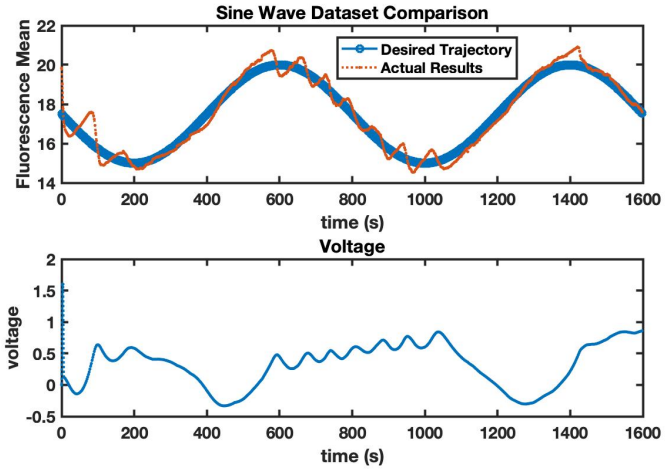

Figure 4.6: Closed loop experiment of controller trying tracking a sine wave. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

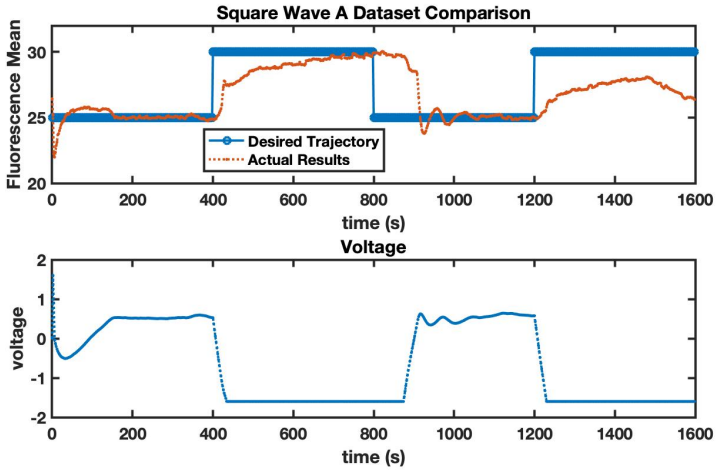

Figure 4.7: Closed loop experiment of controller tracking a square wave. This run is called Square Wave A. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

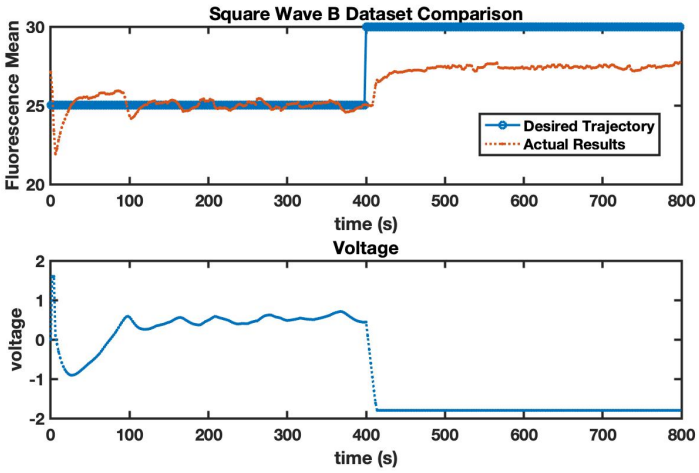

Figure 4.8: Closed loop experiment of controller tracking a square wave. This run is called Square Wave B. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

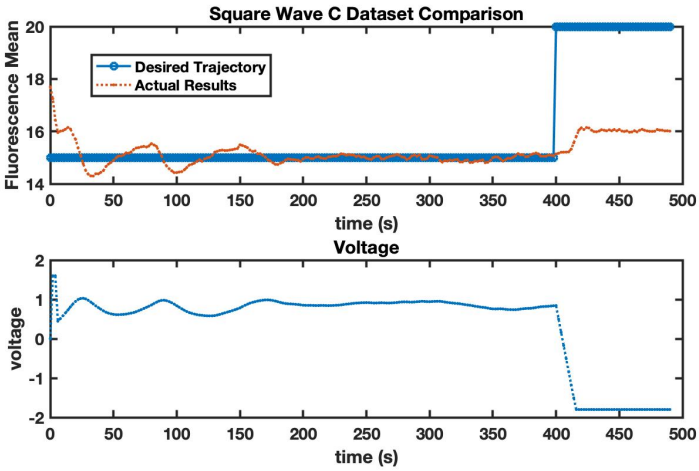

Figure 4.9: Closed loop experiment of controller tracking a square wave. This run is called Square Wave C. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

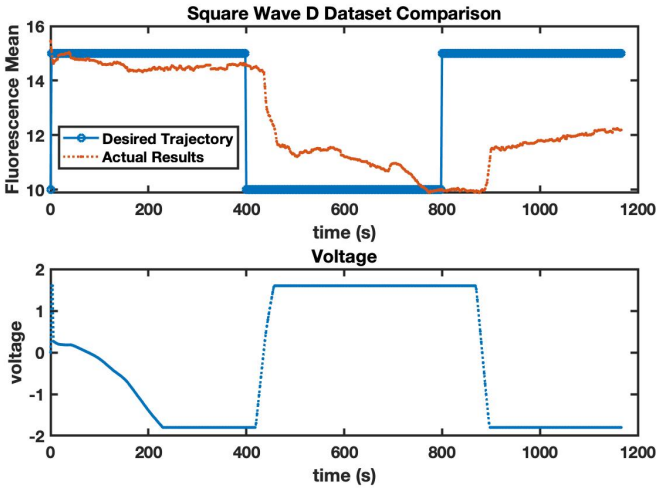

Figure 4.10: Closed loop experiment of controller tracking a square wave. This run is called Square Wave D. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

4.4 Discussion

Looking through the data gathered, we notice variability in the response of the fluorescence mean value compared to the target trajectory. One thing noticed is that when the control value doesn't hit the voltage limits, the system remains fairly responsive and tracks the trajectory. When the saturation values are hit, however, we see that the system tends to stop responding.

A saturation of the system seems to occur when the minimum voltage is applied for too long as shown in Square Wave A and Down Up datasets. The Sine Wave and Down Up datasets tend to suggest it is easier to lower the fluorescence mean value compared to increasing it. This implication can be seen from the oscillations that occur when the fluorescence value is being decreased. The controller has a difficult time keeping the decrease in value smooth while the increase is capable of being tracked smoothly.

Chapter 5

Control of cell migration using machine learning

This chapter is currently under review. The dissertation author was the primary investigator and author of this paper.

5.1 Introduction

Regulation of cell migration plays a critical role in many biological processes essential to life. The role of cell migration is most prominent and studied in development, homeostasis, and disease [95]. Deregulation of cell migration can lead to autoimmune diseases and cancer[95]. Recent work suggests that such processes, when malfunctioning, can be externally regulated through feedback control facilitated by the integration of biological sensors and actuators [105]. Examples of sensors and actuators include bioelectronic devices, which have paved the way for humans to control the responses of biological systems in favorable

ways. Bioelectronic devices allow for monitoring of blood sugar levels, controlling stem cell fate, applying electrical stimulation and delivering therapeutic drugs [26, 67, 84, 89]. An effective signaling cue for cell migration that can be achieved and regulated automatically with current technology is the electric field. However, the presence of an EF can also result in undesired system responses, including phenotype changes[99]. Thus, it is important to be able to regulate EF carefully to maximize target response while minimizing off-target effects. This can be achieved with feedback control.

Successful application of feedback control to biological systems faces many challenges that primarily stem from time-varying unmodeled dynamics. The dynamics of biological systems are inherently stochastic and non-linear[50], but more detrimental to feedback systems is their adaptive nature[18]. That is, the system response to the same stimuli can change over time. Feedback controllers must also work under different environmental conditions and account for disturbances that arise in those conditions. Finally, the fabrication process of sensors and actuators inevitably results in variable performance and response across devices. This implies an inability to derive a single predictive model for all like devices. Furthermore, device response can change over time through repeated use.

Previous work has demonstrated the potential of neural network-based controllers that are updated in real-time to effectively control biological systems without a working model and adapt to changing system conditions [38]. One drawback that has received less attention is the effects of saturation on controller performance. Actuation devices always have a finite range of operation. If suggested controller actions exceed safe operating regions, the control output is clipped, while controller updates resume normally. This deteriorates the performance of the controller. Initial conditions and update parameters must be carefully set to

avoid such problems. Previous work towards controlling systems when the model is unknown has dealt with saturation head-on using sliding mode control [33]. This controller was shown to be effective in controlling bioelectronic devices with no biological system in the loop. However, the controller can be easily challenged in the presence of large sampling times and delays, which are expected in biological experiments.

Previous work in controlling cell systems has been successful in eliciting desired responses using different methods. Directed cell migration through electrical cues occurs naturally during wound healing [104]. The authors in [102] created a device platform to leverage the use of an EF to direct the movement of keratinocyte cells in a 2D setting using predetermined EF signals. The results were promising in controlling the cells' movement in an open-loop fashion. In [82], the authors were able to control the morphology of HeLa cells using photoresponsive 3D nanotubes and leveraging photoirradiation.

Here, we present an online machine learning (ML) approach that is adapted to account for saturation in the control output. To this end we use a projection update on the weights of the ML controller to help with saturation that can occur. To evaluate the performance of the controller, we consider an in silico experimental platform by applying the controller to the model developed in [76]. In [76], the authors created a deep learning model to be able to simulate the directedness of cells when an electric field was applied. Finally, a version of the controller is demonstrated *in vitro*, where the recruitment of macrophages is controlled in an experimental setting by regulating the electric field. This is successfully setup by coupling the feedback control algorithm with image processing and cell tracking software using time lapse microscopy.

The paper will be structured as follows. Section 7.3 will speak on the ML

controller being used, show *In silico* results of how it is an improvement and the stability of it on the system. Subsection 7.3.3 will discuss the experimental set up and the measure being controlled. Sections 7.4 and 7.5 will show the results and a discussion on them. Section 5.5 is the conclusion of the paper.

5.2 Methods

In this section, we introduce the original ML controller framework and the corresponding update law. We then propose an adapted update law that considers saturation by way of adding a projection operator when the control output exceeds operational bounds. We present some analysis in the continuous time framework to show the convergence of the algorithm.

5.2.1 ML Controller

A radial basis function neural network (RBF-NN) is a feedforward neural network. It has been shown to work well for predictions and control of biological systems [38, 57]. It consists of three layers: an input layer, a hidden layer, and an output layer. The inputs go through a transformation in the hidden layer using an activation function. The activation function here is the Gaussian function

$$\Phi(||\mathbf{z}(t) - \mathbf{c}_i||) = e^{\frac{-\|\mathbf{z}(t)-\mathbf{c}_i\|^2}{\beta_i^2}} \tag{5.1}$$

with input vector $\mathbf{z}(t)$, variance β_i and the center vector $\mathbf{c}_i$(t) for the i^{it} neuron. The output of the RBF-NN is:

$$\Omega(\mathbf{z}(t)) = \sum_{i=1}^{M} W_i \Phi(||\mathbf{z}(t) - \mathbf{c}_i||), \tag{5.2}$$

where M is the number of neurons in the hidden layer and the weights W_i are updated according to the following update law in [38]:

$$\dot{W}_i = e\Phi(||\mathbf{z} - \mathbf{c}_i||) \tag{5.3}$$

This neural network is tasked with learning a feedback control law u for the system

$$\begin{cases} \dot{x} = f(x, u) + \delta(t) \\ \dot{y} = x \end{cases} \tag{5.4}$$

such that the measured output y tracks a desired reference value r. That is, the goal is to minimize $e(t) = r(t) - y(t)$ where e is the error between the reference value r and the output y. To update the parameters of the RBF, the gradient of the cost function $C(t) = \frac{1}{2}e(t)^2$ is used. For implementation purposes, we let

$$\dot{W}_i \approx \frac{W_i(t + T_s) - W_i(t)}{\gamma} \tag{5.5}$$

where γ is the learning rate of the weights and T_s is the sampling time. Taking $\frac{\partial C}{\partial W}$ and using the following discrete-time parameter updates

$$W_i(t + T_s) = W_i(t) - \gamma \frac{\partial C}{\partial W} \tag{5.6}$$

gives the update law:

$$W_i(t + T_s) = W_i(t) + \gamma\Phi(||\mathbf{z} - \mathbf{c}_i||)e \tag{5.7}$$

The issue with this update is when saturation occurs and the reference has

not been achieved. In this case, the control output continues to grow because the parameters will continue to increase due to the error not achieving zero.

To keep the controller output from straying too far from the allowed value, the control update law is adapted when the control output exceeds permissible bounds. If the control output is above the upper limit (UL) or below the lower limit (LL), a projection method is applied to down-regulate updates on the NN weights. The new update law is

$$\dot{W}_i = \begin{cases} \gamma_a \Phi(||\mathbf{z} - \mathbf{c}_i||)e - \text{sign}(e)\alpha_a \frac{W_i W_i^T \Phi(||\mathbf{z}-\mathbf{c}_i||)}{\|W_i\|} e, & \text{if } u > UL \\ \gamma_b \Phi(||\mathbf{z} - \mathbf{c}_i||)e + \text{sign}(e)\alpha_b \frac{W_i W_i^T \Phi(||\mathbf{z}-\mathbf{c}_i||)}{\|W_i\|} e, & \text{if } u < LL \\ \gamma \Phi(||\mathbf{z} - \mathbf{c}_i||)e, & \text{otherwise.} \end{cases} \tag{5.8}$$

This helps keep the control signal from straying too far from the current bounds allowed by the experimental set up. Doing this would allow for the control value to not be delayed when a change is necessary and help keep the control values near the bounds allowed.

We look at the control problem as a tracking problem with the following representation of our system

$$\begin{cases} \dot{x} = f(x, u) + \delta(t) \\ \dot{y} = x \end{cases} \tag{5.9}$$

where x is the state of the system, u is the control input into the system, y is the output seen from the system, and δ is a continuous uniformly bounded time dependent disturbance. f is defined on a bounded domain $u_{min} < u < u_{max}$ to

be a continuously differentiable and strictly monotonic function in u such that $f(x,0) = 0$ and $f_{min} < f(x,u) < f_{max}$ where f_{min} <0 and $f_{max} > 0$. Given a reachable reference $r(t)$, there exists a $u^*(t)$ such that

$$\dot{r} = f(x,u^*) + \delta(t) \tag{5.10}$$

Assuming the existence of the control law u^*, the objective is to design u_{nn} to approximate u^* which drives (7) towards the reference trajectory. The dynamics of the error are

$$\begin{aligned} \dot{e} &= \dot{r} - \dot{x} \\ &= f(x,u^*) + \delta(t) - [f(x,u_{nn}) + \delta(t)] \\ &= f(x,u^*) - f(x,u_{nn}) \end{aligned} \tag{5.11}$$

Since f is invertible, this means $\dot{e} \to 0$ implies $u_{nn} \to u^*$.

Given the capabilities of RBF NN to universally approximate functions, we use the proposed RBF with projection to design u_{nn}

$$u_{nn} = \Omega(\mathbf{z}(t)) = \sum_{i=1}^{M} W_i \Phi(||\mathbf{z} - \mathbf{c}_i||)). \tag{5.12}$$

Using the same steps as in 5.5-5.7 to implement the update on the weights,

the discretized update law for the weights are as follows:

$$W_i(t+T_s) = \begin{cases} W_i(t) + \gamma_a \Phi(||\mathbf{z}-\mathbf{c}_i||)e - \text{sign}(e)\alpha_a \frac{W_i(t)W_i(t)^T\Phi(||\mathbf{z}-\mathbf{c}_i||)}{\|W_i(t)\|}e, \text{ if } u > UL \\ W_i(t) + \gamma_b \Phi(||\mathbf{z}-\mathbf{c}_i||)e + \text{sign}(e)\alpha_b \frac{W_i(t)W_i(t)^T\Phi(||\mathbf{z}-\mathbf{c}_i||)}{\|W_i(t)\|}e, \text{ if } u < LL \\ W_i(t) + \gamma \Phi(||\mathbf{z}-\mathbf{c}_i||)e, \qquad \text{otherwise.} \end{cases} \tag{5.13}$$

5.2.2 Experimental set up

Figure 5.1 shows the schematic of the experimental set up used in this work. Macrophages were seeded in a tissue culture dishes (Corning)-based electrotaxis chamber for the best in-vitro migration performance. The chamber was built with glass strips and sealed with high vacuum grease as previously described [88] A Direct Current (DC) electric field (EF) with a voltage of up to 3 V/cm across the chamber was applied through Ag/AgCl electrodes for inducing electrotactic cell movement. The electric current was monitored and controlled by the ML controller to achieve efficient electrotaxis at will. The power system consists of a Keithley current source. The current source transfers the current through Agar bridges and Steinberg's solution. This causes an electric field to be applied to the macrophages in the electric field chamber where a microscope takes an image of the cells at 5 minute intervals. The images are stored on a computer where MATLAB script is running that calls on the image analyzer to evaluate the cells recruitment index value. This value is feedback into the ML controller algorithm to decide the next current value to keep the recruitment index at the desired trajectory. The value is sent to the current source through a serial cable that connects it to the computer.

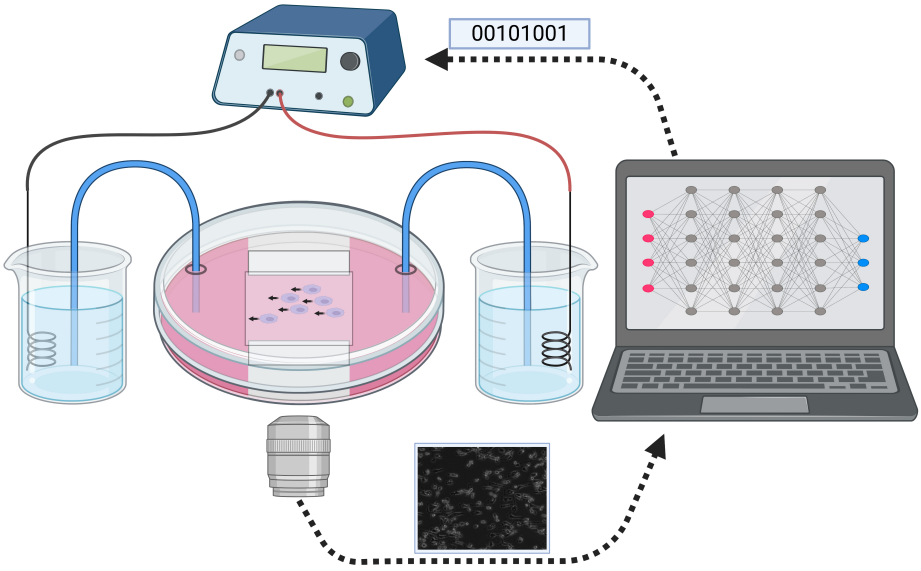

Figure 5.1: Schematic depicting the experimental setup.

5.2.3 Preparation of M0 macrophages

Bone marrow-derived macrophages (BMDMs) were generated and purified in vitro (see Fig.5.2) following standard procedures as previously described [91]. Briefly, bone marrow cells were isolated from the tibia or femur of C57BL/6 mice and cultured in (Dulbecco's Modified Eagle Medium) DMEM (Invitrogen) with 10% Fetal Bovine Serum (Invitrogen) and 1× Antibiotic-Antimycotic solution (Invitrogen), supplemented with 20% l-929 conditioned medium containing M-CSF for 6 days, followed by an extra 24-h culture without the conditioned medium. Adherent macrophages were then harvested by gentle scraping with a cell scraper and seeded in electrotaxis chambers for subsequent experiments. Cell viability was determined by trypan blue staining. C57BL/6 mice were purchased from Jackson labs and maintained under a strict 12-h light cycle and given a regular chow diet in a specific pathogen-free facility at the University of California, Davis (UCD). All animal experiments were performed in accordance with regulatory guidelines

and standards set by the Institutional Animal Care and Use Committee of UCD.

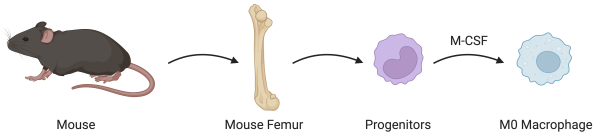

Figure 5.2: Schematic depicting how macrophages were prepared.

5.2.4 Image processing and cell tracking

The image analyzer (see Fig.5.3) is a Python script to evaluate the cells' recruitment index value called from the MATLAB script (as described in Section 2.2). The input to the image analyzer is microscope images acquired at 5-minute intervals. The cells in the images are identified and tracked over time using Trackpy [2], a particle-tracking Python package in 2D and higher dimensions. The resulting cell trajectories are analyzed to compute the recruitment index values. The computed recruitment index values are saved into a CSV (comma-separated values) file which is then used to provide feedback to the ML controller algorithm.

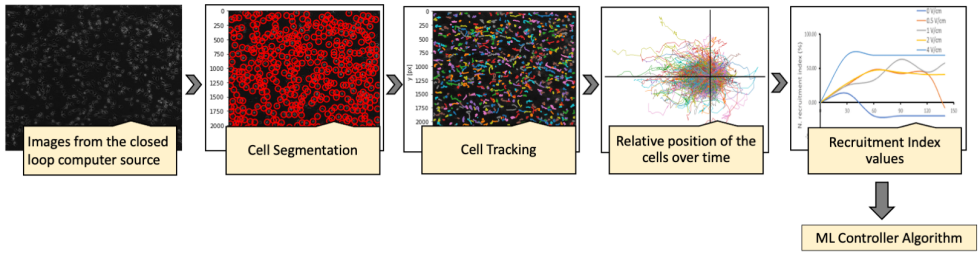

Figure 5.3: Schematic depicting the stages of Image Analyzer

5.2.5 Quantifying cellular response

The cells' relative $\hat{x}$ and $\hat{y}$ position values at every time step $\hat{t}$ are obtained from the cell trajectories and calculated based on the input image resolution. The migration speed of the cells is the ratio of accumulated distance to the time at every 5-minute, calculated as

$$\text{Migration speed} = \frac{\sum_i^n \sqrt{\hat{x}_i^2 + \hat{y}_i^2}}{i * 5} \tag{5.14}$$

Cells that exhibit migration speeds below the 25th percentile are filtered out in order to eliminate immobile cells. The directedness of cell migration, or the angle of migration, is quantified by calculating the cosine of the angle between the electric field and the line connecting the centroid of the cell from its initial to its current location. Cell directedness is calculated for every 30 minutes, i.e., 6 frames ahead as

$$cos\theta_i = \frac{\hat{x}_{i+6} - \hat{x}_i}{\sqrt{(\hat{x}_{i+6} - \hat{x}_i)^2 + (\hat{y}_{i+6} - \hat{y}_i)^2}} \tag{5.15}$$

where $\hat{x}_i$ and $\hat{y}_i$ are the relative position values of a cell at time $\hat{t}_i$.

To quantify the percentage increase of the cells by each different EF, we define a measure called Recruitment Index (RI). The RI value is feedback to determine the error closing the loop calculated using the directedness values. The RI is normalized (subtracted) to the time point 0. If all the cells move to the anode, the index is 100%, whereas the index is -100% to the cathode. The Recruitment Index is calculated using

$$\text{Recruitment Index (RI)} = \frac{\text{Cells to the anode} - \text{Cells to the cathode}}{\text{Total cell count}} * 100 \tag{5.16}$$

where

- Cells to the anode are cells with directedness > 0.01
- Cells to the cathode are cells with directedness < -0.01
- Total cells are the sum of the cells in anode and cathode, including the cells with $-0.01 <$ directedness < 0.01

5.3 Results

Here we apply the feedback control algorithm to our in silico platform and finally demonstrate an in vitro implementation. For the in vitro analysis we use a qualitative stochastic model of directedness, to demonstrate the advantages of using the feedback control algorithm with projection. The new algorithm is tested in an experimental setup and compared to a PID controller. The PID controller was chosen for comparison given that it is an industry standard.

5.3.1 *In silico*

Model

For the in silico demonstration we choose a simplified model to demonstrate and verify the stability of the control algorithm. In experiments, the feedback control is done on a metric based on the measured directedness of the cells. Here we test the controller on a mathematical model describing the temporal dynamics of directedness as a function of EF. In [76], the authors present data showing the response of cells as measured by directedness to various EF strengths. We use

this data to guide model parameters and general qualitative behavior.

$$directedness = \begin{cases} 1 & \text{if} \quad dir(2 * rand - 1)\frac{\frac{s}{EF+1}EF}{3} > 1 \\ -1 & \text{if} \quad dir(2 * rand - 1)\frac{\frac{s}{EF+1}EF}{3} < -1 \\ dir(2 * rand - 1)\frac{\frac{s}{EF+1}EF}{3} & otherwise \end{cases} \tag{5.17}$$

Let *dir* be the direction in which the cell migrates while under an electric field (i.e., towards or against the anode). EF is the strength of the electric field being applied to the cell. s is the strength towards a specific direction in the electric field. *rand* is a small random number between (0, 1) that adds randomness to the cell movement. Since directedness values should be between -1 and 1, the model makes each value the minimum and maximum if the derivation exceeds those values.

Figure 4.4 shows a comparison between the data we received and the model output. Both have the same EF applied and it can be seen that the trends of how the outputs are ordered by EF are the same. The main differences are that the model seems to be more stochastic in nature and there isn't as sharp of an increase of directedness from EF=1. We treat the model as a qualitative model of the real-life system we planned on using the RBF with projection algorithm on. One important thing to note is that the cells from the data are keratinocytes. The real-life cell we performed experiments on were marcophage M0 cells. Although the cells are different, we hope that the information transfers over and that the cells behave similarly enough to that the model will be informative of the controller's performance.

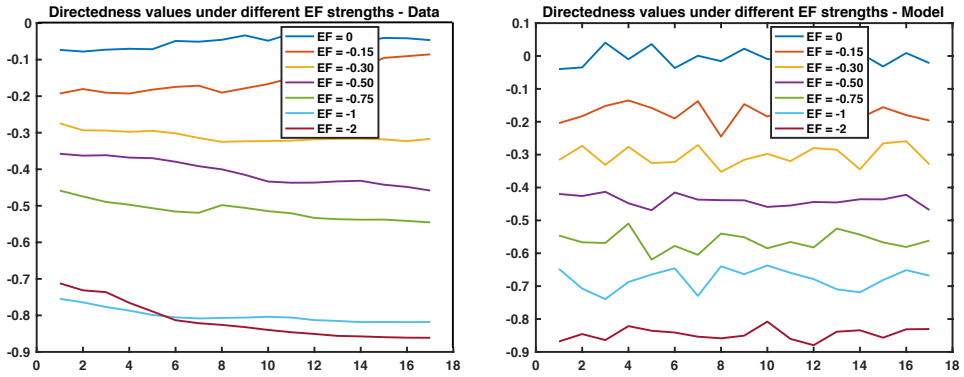

Figure 5.4: Plots showing the data given and showing the model based on the data using the same electric field values.

Results

For the simulation, s is set to 4 making it so the directedness of the cells travel similarly from positive or negative EF. 100 simulations are done at each step and averaged out for the output of the model. The EF is bounded between [-0.9,0.9] to demonstrate the difference in algorithms.

Figure 5.5 shows simulation results between the proposed ML algorithm with projection on the weight updates and the standard weight update law for the ML algorithm. The top plots have the output of the model in blue and the desired reference value in red. The middle plots have a comparison of the saturated control output and the actual output the controller wanted to apply. The saturated value is in blue and the actual control output is in red. The bottom plot is a zoomed-in view of the proposed algorithm with projection. It demonstrates the algorithm's ability to push the control output back toward the minimum allowed voltage rather than continually decreasing as the standard algorithm does. We see that this continually decreasing output by the standard algorithm causes a delay in

changing the voltage to the correct values when the reference trajectory changes.

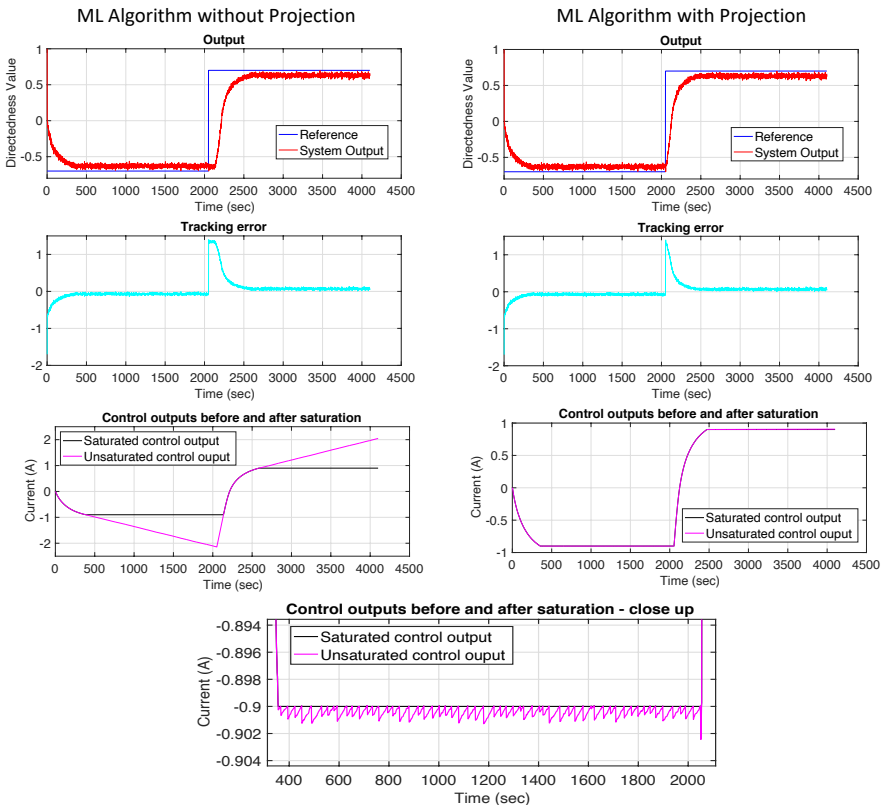

Figure 5.5: Performance of the ML algorithm with projection is compared to the original ML algorithm. The controller is applied to the qualitative stochastic model for cell migration under an electric field. a) The reference signal is in blue and the red line shows the response of the systems under feedback control. b) The tracking error between the reference signal and system response is plotted. c) The controller output (pink) is compared to the saturated control signal being sent to the device. d) A re-scaled plot of the left image in panel c) showing the effects of the projection component of the algorithm.

5.3.2 *In vitro*

Figure 5.6 shows experimental results of feedback control on the recruitment index of macrophage M0 cells using the purposed ML algorithm (left three figures) and a PID controller (right three figures). The initial goal was to have the

recruitment index exceed the reference value of 60% RI. Once this was achieved, the reference changed to -60% RI, with the goal being to track the reference value. In the plot titled output, the blue line is the reference value, and the red is the measured recruitment index value of the M0 cells during the experimental run (a). The plot titled tracking error shows the error in light blue (b). The plot titled comparison of saturated control output and control output shows the saturated control output applied to the cells and derived control output before the saturation bounds were applied (c). Table 5.1 shows the parameters used for the ML controller, including the input z used.

We see that the value used for γ_a was not in the correct range to keep the control effort from growing far from the saturation limits for the beginning part of the experiment. Once the initial goal was met, however, the controller quickly changes signals to achieve the changed reference value. When saturation values are reached, the control output starts to get pushed back toward the bounded limits of the experimental setup. We see that the final 3 values applied by the saturated control output are the same as the values decided on by the controller.

Table 5.1: Detailed design parameters of the RBF network used in EF cell migration experiment.

	Example 1
Sampling Time (T_s)	5 $[min]$
η_w	$9e-5$
β	1
γ_a	1
γ_b	1
γ	1
α_a	0.4
α_b	0.4
Number of Neurons (M)	101
Number of Inputs (N)	6
$w_i(0)$	$rand(1, 101) * 1e-5$
$\mathbf{z}(n)$ for Controller 1	$[r(n+1), y(n-1), y(n-2), y(n-3), y(n-4), y(n-5)]^T$

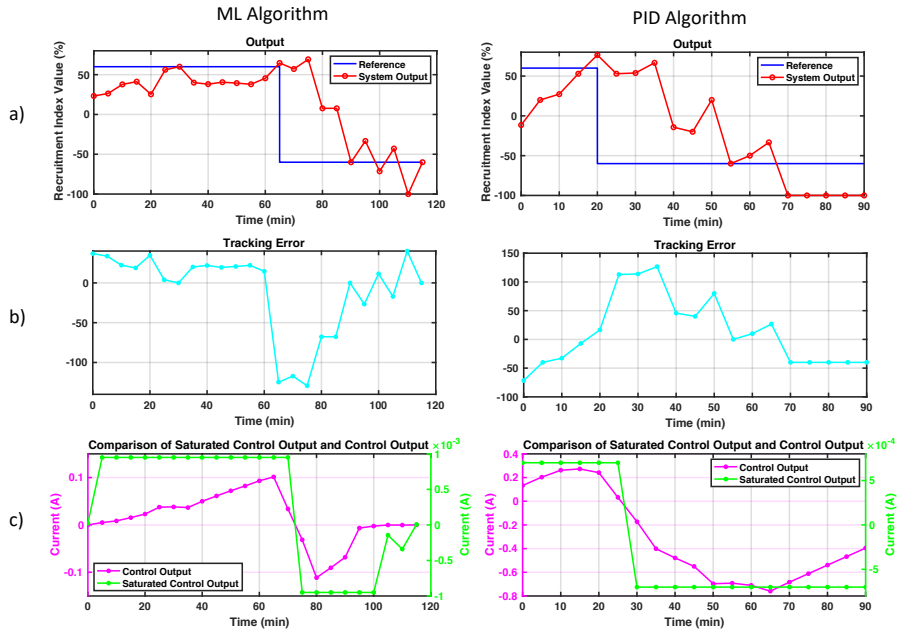

Figure 5.6: Plots in row (a) show the experimental results for feedback control on the recruitment index of macrophage M0 cells using the purposed ML algorithm and a PID controller. Once the initial positive reference value of 60% RI, in blue, is surpassed, the reference changes to -60% RI, and the goal is to track the reference value from there on out. The red is the measured recruitment index value of the M0 cells during the experimental run. Plots in row (b) show the tracking error in light blue. The plots in row (c) show the saturated control output applied to the cells, in green, and the control output without the saturation limits, in magenta.

5.4 Discussion

Part of the difficulty in applying the algorithm above comes from not knowing how large the norm of the weights would be prior to the experiments. This led to the controller not always staying near the stability bounds. Due to the limited number of experimental runs available, insufficient data was collected to find a good set of parameters for the controller to be fully utilized. Further improvements can be made to the update law to compensate.

The projection method works in this setup because of the activation function chosen and the experiment's goal. A Gaussian function does not have negative values as an output. This means the only way to have negative values for the control output is through the weights. If the goal was to keep the RI in only positive values, the controller could help with the output growing too large, but it would not be able to return it to the saturation bounds without possibly causing issues with stability.

Even with promising results using the proposed ML algorithm for control, other challenges need to be addressed where this controller falls short to solve. Inducing a strong electric field for long periods of time on the macrophage cells can cause polarization [43], which could have an adverse effect on wound healing time[45]. This means picking a control strategy that minimizes the control effort while also directing the cells towards the desired reference. This leads to the idea of model predictive control (MPC) as a possible solution. The cost function that is minimized while finding the optimal control value at each time step allows us to penalize control effort and error. The solution obtained wouldn't necessarily use the maximum/minimum values allowed during experiments meaning polarization is less likely to occur.

The work presented has relevance in wound healing. M1 cells tend to abundant

during the beginning of inflammation where they help in the cleaning of the wound. M2 cells are plentiful at the end of inflammation and beginning of proliferation where they help with growing new tissue[55]. With this understanding, an external EF can be modulated to increase recruitment of these cells at the correct stages in the hopes of expediting the wound healing process. This would require a way of monitoring cell migration *in vivo* and in real-time. This can potentially be addressed through proxy for M1 and M2 abundance and composition through the measurement of biomarkers or chemical compounds in the wound bed.

The methods in this paper can be extended to broader applications. Self-regulation is an important attribute for the health of many systems. It can be seen in ecological systems where the dynamics of the system can be modeled by a predator-prey model[6]. In biology it is done more in a feedback control manner where once the body senses a change the system adjusts to keep everything in a stable state. One example being during exercise when vasodilation occurs allowing for increased blood flow, which allows for more oxygen delivery to the muscles[46]. Another example is in wound healing where particular cells are sent out during different stages to help treat the wound[75]. Utilization of control of biological systems has already been shown to be beneficial in precision medicine[61]. Some of the barriers to extending the work includes challenges in sensing system response *in vivo*.

5.5 Conclusion

The ML controller with a projection on the weights showed improvement over the original ML controller in simulations. The controller was then used *in vitro* to recruit macrophage M0 cells towards a desired trajectory. A PID controller was used for the same *in vitro* experiment to compare a standard controller used

in industry vs the designed controller in this paper. The ML controller showed better results in tracking the cells' recruitment index than the PID controller. The control algorithm seems suitable for situations where the norm of the weights can be predetermined or enough experimental runs can be done to find an approximation of the weights' norm and where the control signal has to change between positive and negative values.

Chapter 6

Modeling of a Proton Pump

6.1 Introduction

Advancements in technology has allowed for further integration between man made devices and biological systems. Bioelectronics devices have been created that allow for monitoring blood sugar levels, controlling stem cell fate, applying electrical stimulation and deliverying therapeutic drugs [67][26][84]. Ion pumps are bioeletronic devices that allow for the movement of ions and charged drugs to move towards a targeted area using electrical signals. They are capable of helping in the case of precision medicine where applying a concentration of a drug at a specific region and time is more effective than current drug delivery methods such as digested pills [90].

Ion movement is important in a biological system because a system in general, has a healthy range of pH values it stays between. Being outside of the healthy range of pH could be harmful and cause sickness or even death [87]. Delivery

of ion H+ is capable of helping to maintain a healthy pH balance. Maintaining these healthy pH values can help with gene expression and maintaining a healthy neuronal function [80].

One area where bioelectronics comes to play is feedback control. Biological systems tend to have a form of feedback where it self-regulates to stay in a healthy state. Feedback control is when after sensing a change, the system adjusts itself to push itself back to its steady state. An example of this is during blood clotting where a signal goes out once the body senses a cut that activates blood platelets which in turn stops bleeding and forms blood clots [54]. Using bioelectronics it is possible to reproduce this type of feedback. Such an application was demonstrated in the control of the resting potential of cells [38].

As technology becomes more advanced more time is needed to thoroughly test devices. Though well-monitored experiments are necessary to make conclusive results *in silco* experiments are a means of helping bridge the gap when proper experimentation is slow/expensive to iterate. This could slow down the process of tuning parameters in a feedback control algorithm. This requires a predictive model.

Trying to model these devices that interact with biology can be a difficult task. Being unable to fully understand the system can lead to poor performance of the model when compared to the actual data. The devices have unforeseen hidden dynamics that aren't understood. Manufacturing differences between devices lead to differing responses requiring a more specific model per device.

Through all these difficulties, I present a mathematical model that is capable of capturing the dynamics of a bioelectronic device used in experiments. With the use of literature, I was able to apply assumptions that allowed the model to have appropriate bounds. Finally, using parameter fitting techniques, proper values for

the parameters were found that allowed the model to have proper dynamics when compared to the data used.

The rest of the paper is outlined as follows: Chapter II is an introduction, Chapter III is the results, Chapter IV is a discussion of the results, Chapter V is a mention of future works and Chapter VI is the conclusion of the paper.

6.2 Background

6.2.1 Proton Pump

The proton pump is an ion pump that delivers positively charged molecules. The proton pump used and being characterized here is one that adds/removes H+ ions. Figure 1 shows a schematic of the device. The ions travel through an ion bridge ((C) in Figure 1) which is connected to a reservoir ((A) in Figure 1) and a membrane that leads to the area where pH change is being monitored ((B) in Figure 1), which is the target solution. It is capable of doing this by sending electrical signals to an electrode/electrodes, which allows for ions to go through a polymer membrane, which separates a reservoir containing a large number of ions and a target solution. A negative charge will remove ions from the place where pH is being monitored and back to the ion bridge/reservoir. A positive charge will have the reverse effect, pushing ions into the place where pH is being monitored. The amount of ions in the reservoir is considered to be infinite, meaning that moving ions to the target solution to reach a target pH value should not be limited by the supply of H+. It is important to note that the ion bridge has no fluids, strictly only allowing ion movement. The individual pumps are aligned in a 4x5 grid. Each pump can be activated individually, or a group of them can be activated to add/remove ions.

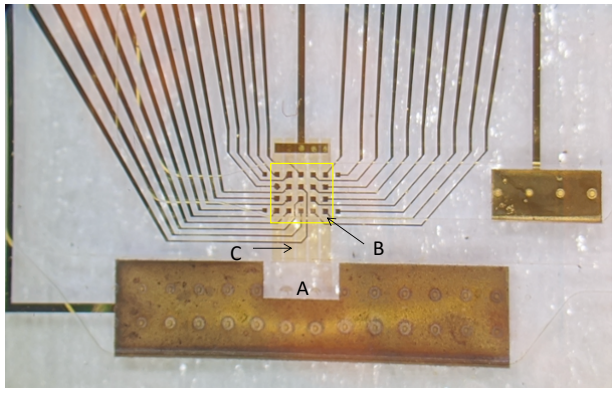

Figure 6.1: Image of the Proton Pump. (A) shows the reservoir of the pump. (B) is higlighting the yellow square which is containing the media which is where pH is attempted to be controlled. (C) shows the ion bridge.

6.2.2 How Data was Gathered

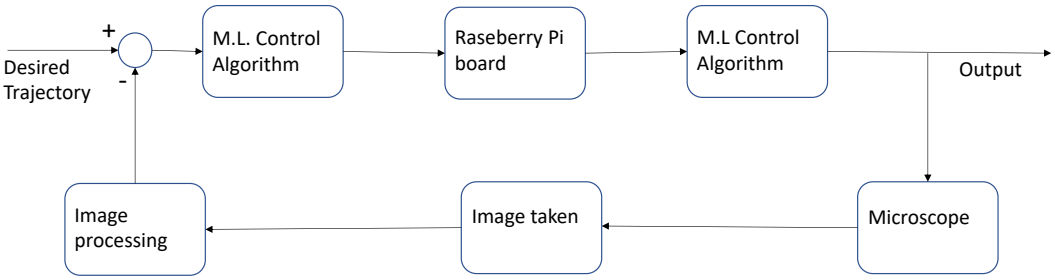

Figure 6.2: Schematic of how experiments were carried. The figure shows how the overall system uses feedback to make corrections for the voltage in real time.

The data used in this worked was obtained from [38], where the authors demonstrated control of pH using a proton pump. An experiment was carried out where a change in the pH of the media in the device is controlled using a machine learning algorithm in real-time. In order to see a change in pH, a fluorescent dye (SNARF-1) is added to the media. The strength of the fluorescence changes depending on the change in the pH of the system. SNARF-1 is best used for seeing changes in pH from ranges 6-8 in experiments without cells and 7-8 when cells are used [72]. In this experiments no cells were used, just the change in the pH in the media. A microscope using a Texas red filter is used to take images of the media to capture real-time changes in the pH media. The sampling time during these experiments was 2 seconds.

To make sure the pH is being driven towards a desired trajectory, a machine learning algorithm is used for the controller. The algorithm is a 3 layer radial basis function neural network with an input layer, hidden layer, and output layer. Using the images taken from the microscope, a small portion next to the proton pump(s) being actuated is considered. The fluorescence mean pixel value (FMPV) of the portion of the image tracked is evaluated using Matlab and fed back into the algorithm, which adjusts the voltage in order to push the FMPV toward the desired trajectory, thus providing closed-loop feedback control.

The data gathered is shown in Figures 6.3-6.8, wherein each figure on the top panel shows a reference trajectory (blue) and the actual output of the system (red). The lower panel shows the voltage/input being applied to the system.

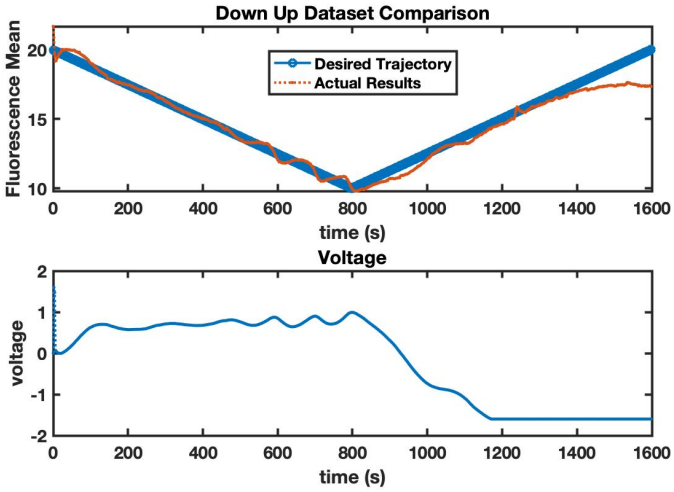

Figure 6.3: Closed loop experiment of controller tracking a reference signal linearly decreasing followed by a linearly increasing trend. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

Looking through the data gathered, it was decided to throw out three of the datasets: Square Wave B (Figure 6.6), Square Wave C (Figure 6.7), and Square

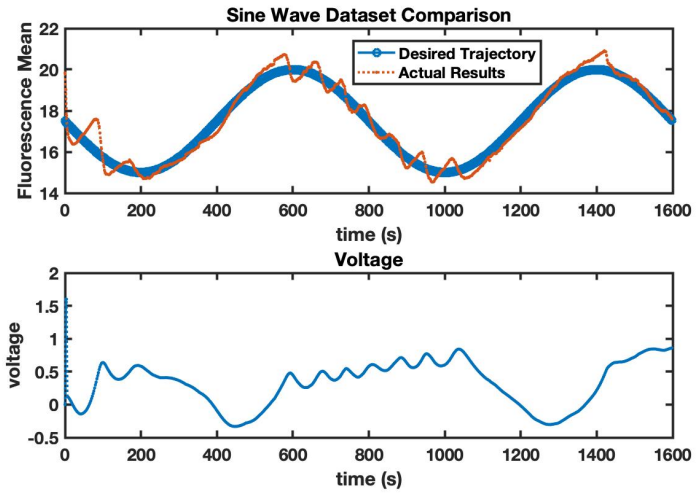

Figure 6.4: Closed loop experiment of controller trying tracking a sine wave. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

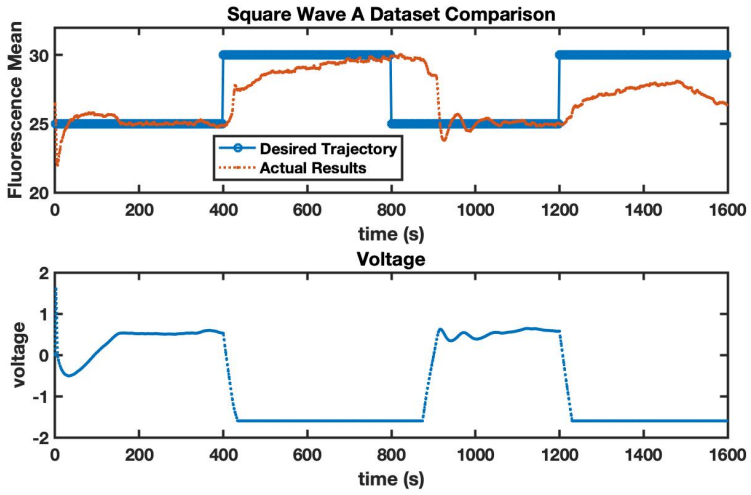

Figure 6.5: Closed loop experiment of controller tracking a square wave. This run is called Square Wave A. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

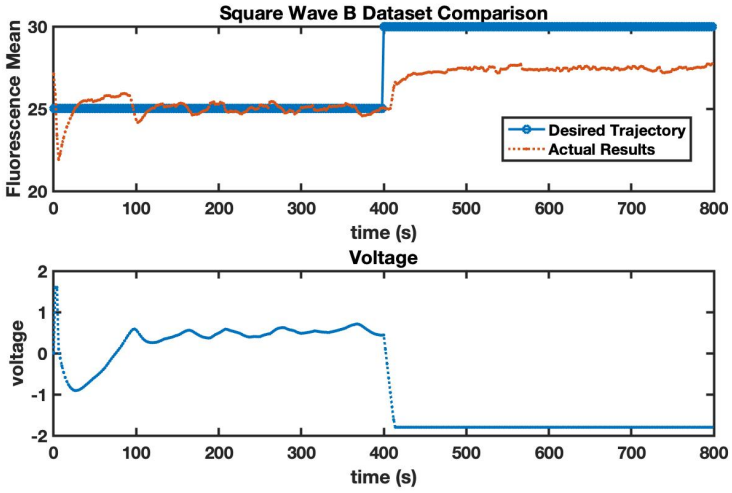

Figure 6.6: Closed loop experiment of controller tracking a square wave. This run is called Square Wave B. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

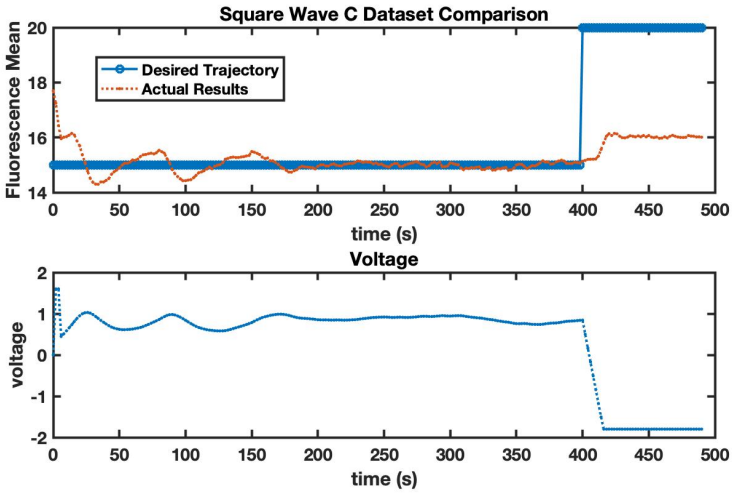

Figure 6.7: Closed loop experiment of controller tracking a square wave. This run is called Square Wave C. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

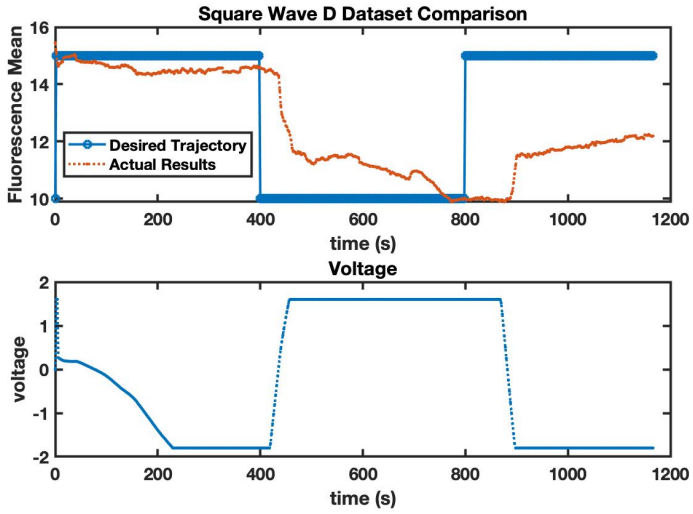

Figure 6.8: Closed loop experiment of controller tracking a square wave. This run is called Square Wave D. The top graph depicts the trajectory trying to be tracked (blue) and the actual value of the system (red). The bottom graph shows the voltage/input being applied to the system during this time.

Wave D (Figure 6.8). This was due to how the output values in these experiments didn't respond as well to the input values given compared to the other experiments. It would be challenging to try to capture the responses in Square Wave A and Square Wave B using a single model considering how both start at the same fluorescence value and are given similar voltages to reach the same reference trajectory but only one achieves it while the other one is not able to reach the target. This type of dynamics can't be captured in a single time-invariant model. Ideally, the system should maintain consistent input-to-output responses.

Looking at the remaining datasets which are used, we see that they cover a large range of different applied voltages. These datasets also respond well compared to the datasets not chosen, making them the best representatives to characterize the proton pump. It is important to note that there is a bound on the voltage applied to the system. Square Wave A covers primarily at the minimum

value of voltage allowed to be applied. It nears a voltage of one at its highest positive value. Sine Wave primarily stays between a voltage of (-0.5,1). The Down Up dataset is the most unique in terms of the voltages applied and is seen using voltages similar to Sine Wave, but also ends at the minimum voltage value.

A saturation of the system seems to occur when the minimum voltage is applied for too long, as shown in Square Wave A and Down Up datasets. The Sine Wave and Down Up datasets tend to suggest it is easier lower the fluorescence mean value compared to increasing it. This implication can be seen from the oscillations that occur when the fluorescence value is being decreased. The controller has a difficult time keeping the decrease in value smooth while the increase is capable of being tracked smoothly.

6.3 Results

6.3.1 Mathematical Model

In this section, we derive a mathematical model of the bioelectronic device, an ion pump. In order to try to capture the dynamics of the ion pump, a hybrid ODE model is used. The reason behind a hybrid model is due to how the range in voltage being applied changes the dynamics of ion transport across the membrane. From the data shown earlier [Figure 3-5] when the voltages are not near the upper and lower bounds of the voltage the system responds well and can generally be kept at a trajectory. When the input is near the minimum/maximum voltages the system tends to respond differently. The state x_1 in the schematic in Figure 10 and in the model represents the rations in the portion of the image considered and is proportional to the fluorescence mean value in the imaged area in Figure 9. In [32] I present a simplified version of the proposed model used

here. The fact that the model an ODE facilitates in showing stability of the controller in [32]. In general, PDEs are used to model systems where diffusion occurs. This is due to diffusion having a spatial component that ODEs don't take into account. Although diffusion is naturally occurring in the system, an ODE model is a suitable model since we are taking only a small portion of the image to obtain the fluorescence value. This can alleviate the need to model the diffusion of the system.

There were many challenges that came with attempting to model the ion pump due to all the variability within the device and how experiments were conducted. Due to manufacturing being done by hand, each device could have slight variances. This could cause certain proton pumps to have different efficiency compared to others across devices and even between proton pumps on the same device. Fresh media in the ion pump would normally allow for a better response from the pump. The media wasn't always refreshed between experiments which resulted in variations in the system response. The resistance of the ion bridge might have gotten worse as experiments ran. There have been reported issues where the response of the device worsens as it is used more. Lack of calibrations before experiments made it impossible to know the exact values of the pH in the system as experiments ran. This makes it so the initial condition of the pH at the start of the experiments is an educated guess using literature. The light emission of the microscope and media affect the fluorescence values. The fluorescence values are then arbitrary and without calibration can't give an exact relationship to how the pH is changing in the system.

The voltage applied seems to be an indicator as to how the system will most likely respond. The voltage is bounded in the experiments. Some have a min/max value of -1.6/1.6V, while others had a min/max value of -1.8/1.8V. From the data

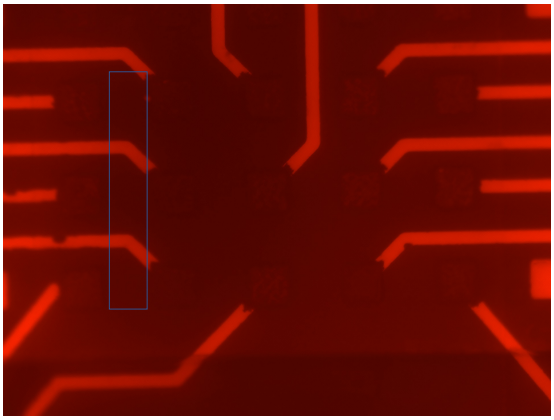

Figure 6.9: Image taken by microscope. The blue rectangle indicates the portion of the image used for the fluorescence value in the data.

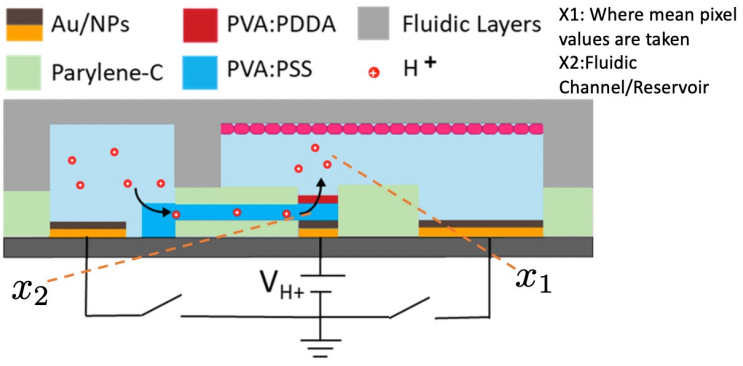

Figure 6.10: Ion Pump Schematic

gathered it can be seen that pushing the voltage to its minimum value seems to cause saturation in the system. Depending on the experiment, this saturation value can be larger or smaller. All the datasets at some point hit the minimum voltage value except for the Sine Wave data (Figure 6.4). Here the system seems to work well without any issues trying to hit the reference value.

One thing noticed is that there appear to be more oscillations when pushing the system down than up. This tends to suggest that the rate of change for adding and removing ions should be different. Due to saturation occurring near the min/max voltage values and the system responding well when not near those values it is reasonable to split the model into four regions. To separate these regions ϵ and δ terms are introduced. The parameter ϵ separates the system into two regions where the voltage is positive. The upper region ($u \geq \epsilon$) is where a saturation of the system may occur although there is only one dataset that pushes the max voltage value. The upper middle region ($\epsilon \leq u > 0$) tends to act linearly with a differing slope depending on the voltage being applied. The parameter δ does the separation in the negative region. The low middle region ($0 \geq u \geq \delta$) tends to act linearly with a differing slope depending on the voltage being applied. The lower region ($\delta > u$) is where the system tends to saturate especially if the minimum voltage is being applied for a long period of time.

As stated before x_1 in the model is the ion concentration in the area where images are taken and used for data[image of photo]. The state x_2 is the ion concentration that are in the reservoir and ion bridge. This value is considered to be infinite, suggesting that the pH can be changed without hitting a saturation, although the data does not support this.

It is assumed that the rate of transport of ions is bounded. This leads to the use of a nonlinear saturating function that is used in the model. This hill type

function will be a function of the voltage, considering that is how the majority of the exchange of ions between x_1 and x_2 occurs. The variable u is the applied voltage and is the input of the system. The parameters c_i and cc_i are the maximum rates of change of ions in the hill function due to u. Parameters d_i and dd_i are the half maximals of the hill function. Although x_1 is the ion concentration in only a small portion of where the image is taken, diffusion should still be accounted for since ions can still flow freely through the area. To take this into account, a leakage term g is added. Natural diffusion is considered through the membrane when no voltage is applied as the term D. The completed model is shown below.

$$
\begin{aligned}
\mathrm{u} > \epsilon &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{c_1 x_2}{d_1 + x_2} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{c_1 x_2}{d_1 + x_2} \end{cases} \\
\epsilon \geq \mathrm{u} > 0 &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{cc_1 x_2}{dd_1 + x_2} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{cc_1 x_2}{dd_1 + x_2} \end{cases} \\
0 \geq \mathrm{u} \geq \delta &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{cc_2 x_1}{dd_2 + x_1} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{cc_2 x_1}{dd_2 + x_1} \end{cases} \\
\delta > \mathrm{u} &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{c_2 x_1}{d_1 + x_1} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{c_2 x_1}{d_2 + x_1} \end{cases}
\end{aligned}
\tag{6.1}
$$

6.3.2 Fluorescence Values to pH Mapping

The movement of ions and change in ion concentration from x_1 to x_2 and vice versa are the dynamics the model is attempting to capture. In order to do this a mapping function needs to be used that will take the fluorescence mean pixel values to ion concentrations. The formulation of the mapping function is further discussed below.

The data was given in terms of FMPV, which is an arbitrary value depending on the lamp used in the microscope and the intensity the image is taken at. Although this does allow one to see a change in the pH value of the system it is not possible to quantify the exact pH values. To know the exact change in the pH, careful calibrations are required where two images are taken at around the same time at different emission values and two known pH points are known, one acidic and one basic [72].

Unfortunately, due to no calibrations done on the device, there is no exact relationship between the FMPV and the pH of the media/water. There is also no exact value for the initial pH of the experiments ran. Although this is a difficult obstacle there are some things that can be said to find a proper transformation. It is known that the range of the initial condition is 7-7.3. There is also a relationship between pH and ion concentration which is

$$\mathrm{H} = 10^{-\mathrm{pH}} \tag{6.2}$$

Also, from [Snarf paper] there should be a linear relationship between the FMPV and the pH values when calibrated correctly. It also states SNARF-1 works best for reading measurements between 6-8 pH for experiments without cells. The linear relationship is given by

$$c + x\mathrm{FMPV} = \mathrm{pH} \tag{6.3}$$

Using this information, a mapping function is created that will go from FMPV to pH as shown above. First, we assume that the initial pH for all experiments ran is 7.1 and that the minimum pH value attained from the set of experiments is 6.1

as shown below

$$\begin{cases} c+\text{FMPV}_{IC} * x = 7.1 \\ c+\text{FMPV}_{min} * x = 6.1 \end{cases} \tag{6.4}$$

Another assumption being made is the pH value never went above or below the thresholds of reliable readings for SNARF-1 during these experiments where the data was gathered. In order to make this assumption hold, the largest difference in fluorescence value between the initial FMPV of each individual data set and the largest difference from that value was taken as the slope. In this case, the Down up data set has the largest difference and is used for finding the slope that will be used universally for the mapping function. To be sure the initial condition is the same for the mappings, the initial value of the fluorescence for each set is used to get the value of the bias c. This sets up two equations with two unknowns resulting in slope and bias of

$$x = \frac{1}{\text{FMPV}_{IV_{DU}} - \text{FMPV}_{min_{DU}}} \tag{6.5}$$

$$c = 7.1 - x\text{FMPV}_{IV} \tag{6.6}$$

This is important considering the values that come from the FMPV are arbitrary and can change depending on the lamp used and the intensity of the light of the microscope which is why calibrating is typically very important to know how pH is changing. Since the actual dynamics of the system are the change of ions this mapping function, along with [ph to ion con eq], allows the model to capture the appropriate dynamics.

6.3.3 Parameter Fitting

In order to fit the parameters for the model, time series data from the experiments in [38] were used. The grey box model estimation from the system identification toolbox is used [52]. Different algorithms can be used to fit the parameters of the data. Initially, a gradient descent method was used but the results didn't capture what was physically happening and weren't used. The algorithm chosen to solve the nonlinear least square problem was the trust region reflective Newton algorithm [11]. At first, the parameter fitting for each individual data set was fairly accurate but the parameters were overfit to the datasets. Certain parameters varied more than others. This led to using a different algorithm since there were difficulties due to the fact that the fluorescence values-to-pH mapping had to be adjusted to each dataset since the minimum value of the FMPV is different for each dataset.

The new algorithm used was a genetic algorithm [85]. This would allow for the parameters to be tuned to all three datasets simultaneously. This method did prove to be computationally expensive and seemed to suggest that there wasn't a set of universal parameters. This suggested that due to variations in how the devices responded there might be a global set of parameters along with parameters specific to each dataset. This could be due to how long the device was in use or how images were taken at the time.

In order to get these global and specific parameters the grey box model estimation was used again with the trust region reflective Newton algorithm. This allowed for set of global parameters and specific parameters for the model to be found such that the model was capable of following the dynamics of the datasets. The results of this parameter estimation are shown in Tables 1-4. Table 1 shows the values for the global parameters and Tables 2-4 show specific parameters for

each data set with corresponding simulations in Figures 11-13.

Parameters	Values
D	0.0043
g	-9.34E-04
k	0.1460
ϵ	0.8767
δ	-0.6855

Table 6.1: Shared Estimated Parameters of the Pump Model

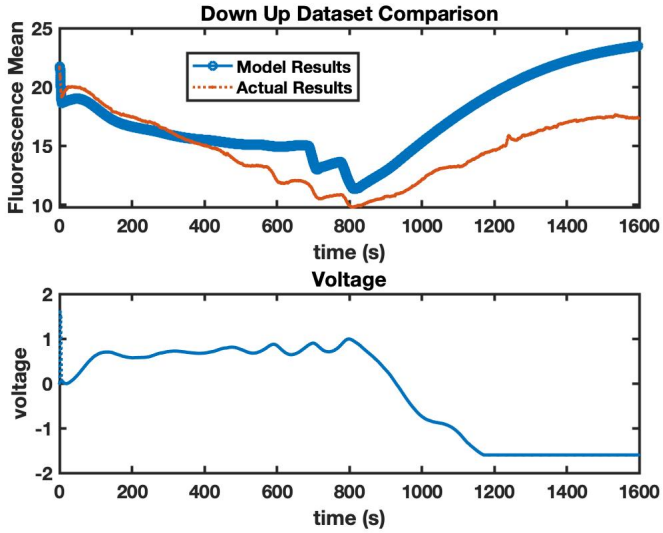

Figure 6.11: Down Up experimental data vs model simulation.

6.4 Discussion

The model has shown it is capable of capturing the trends of each individual dataset when a subset of the total parameters are optimized for all datasets. However, it is unable to capture all three datasets with one parameter fitting. This can be due to many variables that occur during experimentation.

Parameters	Values
$c1$	0.9724
$c2$	4.62E-06
$d1$	0.1694
$d2$	34.995
$cc1$	0.1923
$cc2$	1.47E-04
$dd1$	2.9066
$dd2$	34.8764

Table 6.2: Estimated Parameters for Down Up Experiment

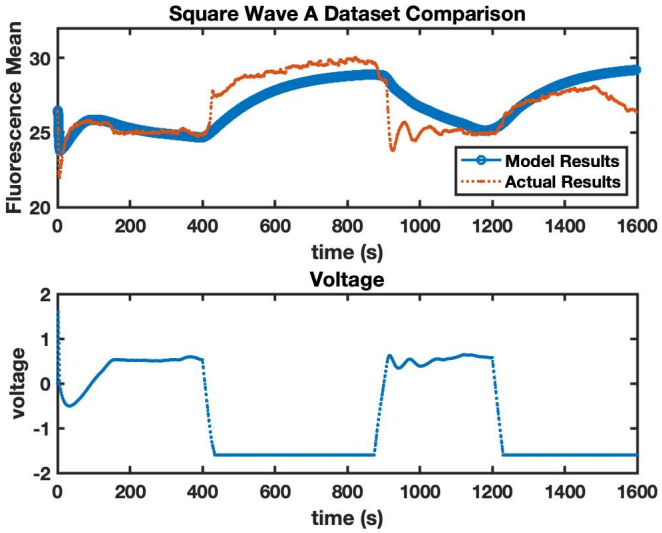

Figure 6.12: Square Wave A experimental data vs model simulation.

Parameters	Values
$c1$	0.9982
$c2$	0.0902
$d1$	0.6467
$d2$	34.8044
$cc1$	0.0398
$cc2$	0.4830
$dd1$	0.1037
$dd2$	34.9872

Table 6.3: Estimated Parameters for Square Wave A Experiment

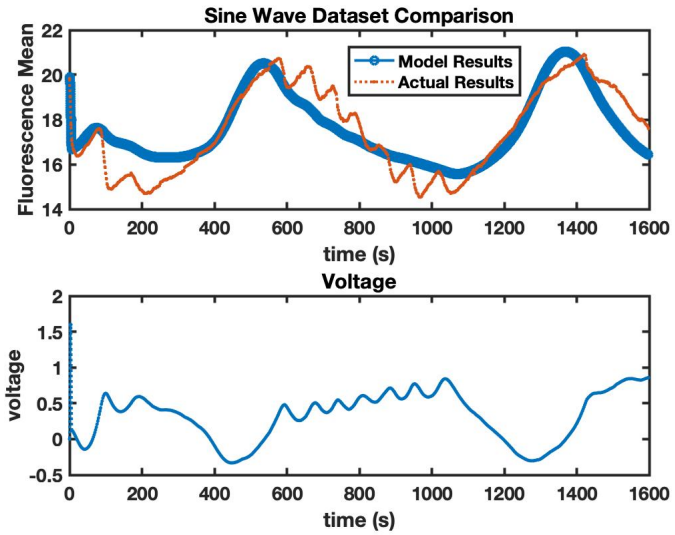

Figure 6.13: Sine Wave experimental data vs model simulation.

Parameters	Values
$c1$	1.000
$c2$	0.0267
$d1$	2.60E-05
$d2$	11.9037
$cc1$	0.1779
$cc2$	0.1684
$dd1$	34.9994
$dd2$	2.22E-14

Table 6.4: Estimated Parameters for Sine Wave Experiment

Due to no calibration of the device and not knowing the exact relationship of FMPV and pH the exact initial condition of each experiment isn't known. The range that the initial condition could be is 7-7.3 pH and for the simulations, an initial condition of 7.1 pH was assumed. The reason for this range in the initial value of the pH is what media was used in the device. Water has a pH value of 7, while other media could be as high as 7.4 pH. This, coupled with not knowing how exactly the pH value is changing in time, can lead to results that vary from what is actually happening in the system.

The mapping from fluorescence to pH might have been another issue in terms of trying to fit the parameters of the system. When just reading values at one emission, as is the case here, a nonlinear function might have worked better. Picking values for a nonlinear mapping function still runs into the same problem as the current linear mapping. The values in the nonlinear function are still arbitrary considering there is no exact initial condition or known pH value at any other reading of the fluorescence. Although a nonlinear mapping could improve the model performance the drawback of not exactly knowing how the dynamics remains a problem.

The amount of data that was considered good enough to use for modeling

might have been too small. More data could have helped in better understanding the dynamics of the system in the different regions of the model. The upper region ($u > \epsilon$), upper middle ($\epsilon \geq u > 0$), lower middle ($0 \geq u \geq \delta$), and lower region ($\delta > u$) are all represented by the data but in limited capacity depending on the dataset. An example of this is the Sine Wave data. The data there shows how most of the voltage applied was within the middle regions and had none occurring in the lower region. When this is optimized it would take into account the middle regions and the parameters necessary to push it into the middle regions quickly from the upper region as the data requires.

Variations on how each experiment was carried out can cause differences in outcomes. The best results for the experiments typically occurred when the media in the device was refreshed. The refreshing of media is what is assumed for the three datasets chosen for evaluating the model due to responses being better than the other datasets. Although the media is assumed to be refreshed as the experiments go on, the fluorescence response could lessen. Over time the devices could have different dynamics due to the value of the current going through the device getting worse.

Manufacturing of devices is done by hand, which makes creating devices to be exactly identical a difficult task. This allows for each device to have slight variations that could lead to different dynamics. Due to these differences, a global model that works for all devices might not be attainable.

With there being parameters specific to each individual dataset it can be helpful to see how the differences are between each dataset at each region.

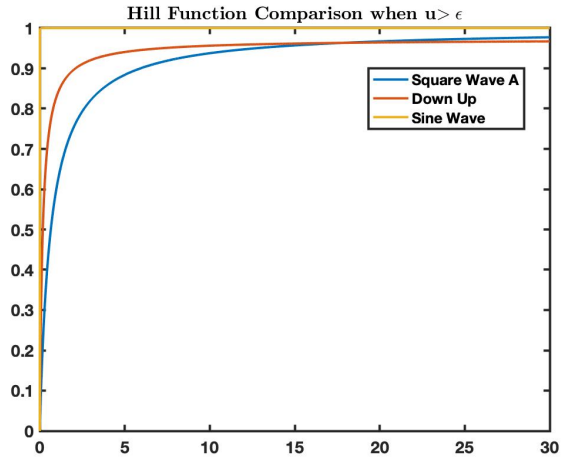

Figure 6.14: Hill functions of the model capturing three datasets in the upper region. Square Wave A is blue, Down Up is red and Sine Wave is yellow

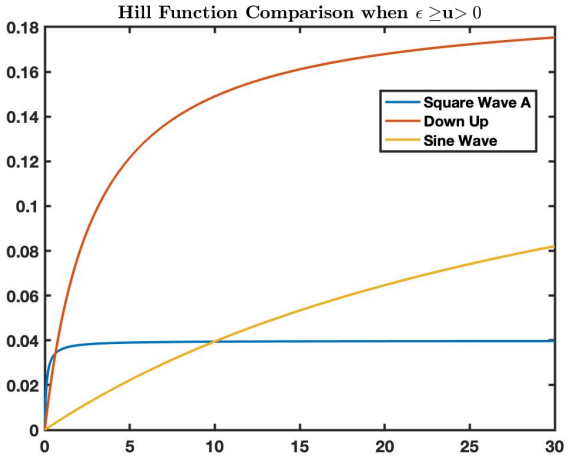

Figure 6.15: Hill functions of the model capturing three datasets in the upper middle region. Square Wave A is blue, Down Up is red and Sine Wave is yellow

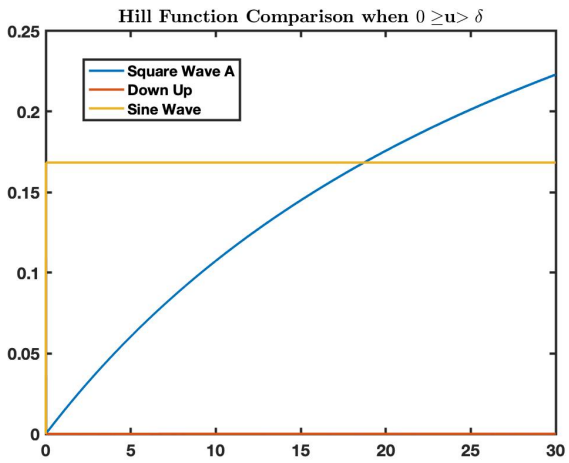

Figure 6.16: Hill functions of the model capturing three datasets in the lower middle region. Square Wave A is blue, Down Up is red and Sine Wave is yellow

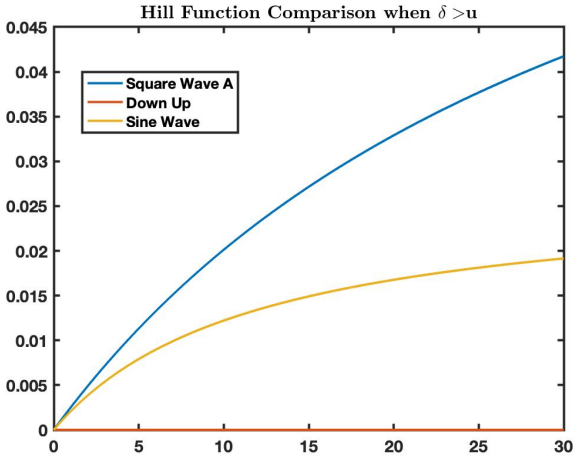

Figure 6.17: Hill functions of the model capturing three datasets in the lower region. Square Wave A is blue, Down Up is red and Sine Wave is yellow

In the upper region ($u > \epsilon$) (Fig. 14) the hill functions are all relatively close in terms of their maximum value. Sine Wave hits the maximum value immediately, while the other data sets took a little longer. This might suggest that more parameters could be shared, at least at this region of the model, although it would make the individual simulations less accurate.

In the upper middle region ($\epsilon \geq u > 0$) (Fig. 15), the hill functions vary more. The overall maximum values of the hill functions are smaller than before by a large margin. Square Wave A hits its maximum value faster than the other sets, which are still increasing at the end of the simulations. This could be due to the fact that Square Wave A is trying to hit a saturation as quickly as possible in this region while the other datasets are just decreasing somewhat linearly.

In the lower middle region ($0 \geq u \geq \delta$) (Fig. 16) the plots show very different responses. Sine Wave again reaches its maximum value immediately. Square Wave A increases almost linearly. Down Up is a very small value and also increases but slowly. This most likely has do with how the three datasets behave in this region. Square Wave A attempts to quickly increase/decrease in order to hit the min/max values of the square wave given. Due to this Square Wave A dataset doesn't have as much time in these regions as the others. Down Up remains in this region for a while before the voltage value moves to the lower region of the system. The Sine Wave dataset is primarily in the middle regions. It differs from the others in that it immediately hits its max value.

The lower region ($\delta > u$) (Fig. 17) is more in line with the lower middle region, except that the values are smaller. One thing that should be noted is that the Sine Wave dataset never has voltages that causes it to be in this region so the hill function for this can be disregarded.

6.5 Future Works

Although a model that is able to generalize the datasets fairly well was derived, it does have things that could be considered faults, such as no full set of global parameters, i.e. one overall model for all the datasets. The inability to derive a universal model could be due to a more complicated ODE model being required. Different functions could have been used over the hill function to cause saturation of the system. More time could have been used to find the proper form of another saturating, such as a sigmoid function.

Another approach could be using a PDE. The fact that diffusion does play a role in the actual system might be better characterized using a PDE model. A PDE also allows for other complexities in the system that might have been overlooked when first modeling the device.

The initial attempt at modeling the device was using machine learning methods. Although these didn't yield good results using, the fluorescence value-to-pH mapping wasn't used. Using the mapping could allow for the machine learning methods to do a better job of capturing the dynamics of the system. This might be an avenue to pursue in order to model the proton pump.

A model that captures the dynamics of the proton pump can be used in simulations to help test new control algorithms. A goal is to have a biological system that this device would be used on and to create a controller that is capable of keeping it at a target value. The meaning of the device can change as in rather than pumping ions, it could be administrating medicine. The controller would attempt to keep a target value through the constraints on the voltage, as stated before, and the dynamics of the new system. Simulations would be run *in silico*.

6.6 Conclusion

Modeling the proton pump proved to be a difficult task. Due to variability in the nature of the experiments and devices, it was challenging to be able to create an ODE-based model that could capture all the datasets with one set of parameters. A model was created where some parameters are shared between all the datasets that make sense in terms of how the device functions. The variability for the non-shared parameters can be attributed to things such as time usage of the device, media not always being re-freshened between experiments, and other contributing factors.

In order to achieve more universal results, devices need to be created such that they have the same input-to-output response. Also, if they happen to vary in time this variation needs to be across all devices and not be a constraint on only one.

Calibrations are required to be able to model the devices in a way that represents what is actually happening with the system as different inputs are being applied. Not having knowledge of the pH change makes it challenging to construct a model since the values of the FMPV are arbitrary. It isn't known if the saturation is due to system dynamics or the dye losing effectiveness or other factors.

Despite all the setbacks, the model was capable of capturing the overall dynamics of the system presented in the datasets. This allows for *in silico* experiments which are helpful since experiments can't always be run to test how a controller will work with the actual device.

Chapter 7

Current control using a sliding mode controller

Portions of this chapter are currently under review. The dissertation author was the co-first author of this paper.

7.1 Abstract

Precision medicine tailors treatment toward a patient's individual needs. This is beneficial, considering people respond differently to the same treatments. One way of implementing precision medicine is through bioelectronics equipped with real-time sensing and intelligent actuation. Bioelectronic devices such as ion pumps can be utilized to deliver therapeutic drugs. To be able to perform precision medicine, medical devices need to be able to deliver drugs with high accuracy. For this, closed-loop control is required to be able to change the treatment strategy as new information about the response and progression of the biological system is

received. To this end, a sliding mode controller is utilized given its ability to perform satisfactory control actions when there is model uncertainty. The controller is used in an experiment with the goal of delivering a pre-determined dosage of fluoxetine throughout a period of time.

7.2 Introduction

Recent developments in bioelectronics hold promise for advancing precision medicine [105]. Bioelectronics devices have been created that allow for monitoring blood sugar levels, controlling stem cell fate, applying electrical stimulation and delivering therapeutic drugs [26, 67, 84]. Ion pumps are bioelectronic devices that allow for the movement of ions and charged drug molecules to move towards a targeted area electrophoretically [98]. They are capable of helping in the case of precision medicine where applying a concentration of a drug at a specific region and time is more effective than current drug delivery methods such as digested pills [89].

Such applications require the ability to carefully control the delivery of therapeutics with precision. In [79], the authors discuss the ability of feedback control to enhance the capabilities of bioelectronic devices. However, implementing feedback control in bioelectronic devices presents unique challenges including variability in device performance. It is difficult to fabricate bioelectronics in a way that the end process is exactly the same. Due to variations occurring in the fabrication process, the bioelectronic devices can differ in their properties. Components in the device can also vary in their performance when the device is used for an extended period of time [14]. This can lead to different responses from the devices when used in an experimental setting.

To address the above mentioned challenges, in previous work, a neural network

(NN)-based machine learning algorithm was used with ion pumps to successfully control the pH level of a target solution [38]. A feedback control algorithm was able to regulate pH levels according to a target time-varying trajectory. In [81], the authors were able to use a similar control algorithm to control the membrane potential of stem cells by way of regulating pH of the extracellular environment. Thus, an adaptive feedback control algorithm helps to mitigate variations in system response even when they evolve in time.

The second primary challenge in the control of bioelectronic devices is their limited operating range. That is, the voltages that can be applied to the device are limited. This causes a saturation issue that needs to be taken into account. In [33], the authors presented an alternative approach to arriving at an adaptive algorithm that has the benefits of the NN-based controller but explicitly handles saturations with guaranteed convergence. This algorithm was tested *in silico*.

In this paper, a similar algorithm as presented in [33] is leveraged to control the amount a biochemical being delivered by an ion pump. We applied a heuristic switching algorithm for modification of the controller gains. The biochemical chosen is fluoxetine which has been shown to be a relevant drug in wound healing in diabetic mice [63]. Three different reference signals were used where the average value for all three was the same. This was done to show the controller's ability to keep the current at the reference regardless of the signal used as well as to allow for different methods of delivery. Due to interface limitations, the controller output cannot be exactly executed at all times. Regardless of this, we demonstrate that the controller is able to perform at a reasonable level, keeping the current near the reference throughout the experimental time.

The paper will be structured as follows. Subsection "Bioelectronic Ion Pump" 7.3.1 will speak on the ion pump and its capability to deliver flouxetine. Subsection

"Controller" 7.3.2 will discuss the sliding mode controller being used. Subsection "Experimental Setup" 7.3.3 will discuss the experimental setup and how things are integrated together. Sections "Results" 7.4 and "Discussion" 7.5 will show the results and discussion, respectively .

7.3 Materials and methods

In this section, we introduce the ion pump and discuss its ability to deliver fluoxetine. We then discuss the sliding mode controller and give a brief outline of how the controller works. Finally, we present an overview of the experimental setup used.

7.3.1 Bioelectronic Ion Pump

Ion pumps are bioelectronic devices that can perform precise electrophoretic delivery of ions and molecules from a region of high concentration in the device, known as the reservoir, to the region of interest, known as the target [78, 98]. Over the years, ion pumps have been used to modulate pH [90] , treat neurological disorders [73, 94], recruit macrophage [15] and deliver hormones in plants [7]. In this work, we use the ion pump to deliver fluoxetine, a biochemical which is a selective serotonin reuptake inhibitor known to have immunomodulatory effects [63].

The ion pump transfers the biochemical from the reservoir to the target, with a voltage (V_{pump}, typically between $0.5V$ and $2V$) applied between the working electrode (Ag) and the reference electrode(Ag/AgCl) (see Fig 7.1(a)). A negatively charged hydrogel selectively transports the biochemical between the target and the reservoir. In the reservoir, we have $0.01M$ of fluoxetine hydrochloride.

Fluoxetine exists as a positively charged species at a pH of 5. For a positive V_{pump}, the fluoxetine moves through the hydrogel-filled capillary and is transported to the target. Fig 7.1(b) shows an image of the device in a six-well plate with a buffer solution. This acts as our target for the experiments.

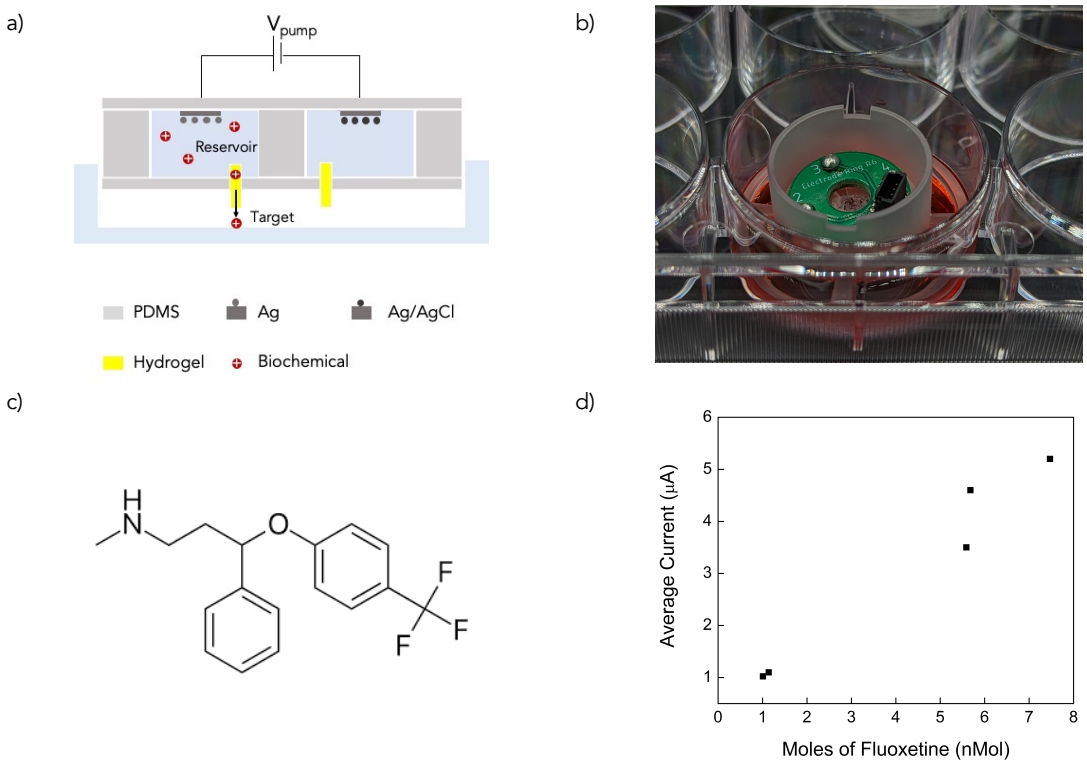

Figure 7.1: (a) Schematic of the bioelectronic ion pump. (b) Image of the ion pump in a 6-well plate with buffer solution. (c) Chemical structure of fluoxetine (d) Graph showing the relationship between the current produced by the ion pump and amount of fluoxetine delivered.

We collect the target solution after completion of delivery and use High Performance Liquid Chromatography (HPLC) to estimate the amount of fluoxetine in the target. We establish a relationship between the current produced during the biochemical delivery and the corresponding amount of fluoxetine delivered to calibrate the ion pump's performance (see Fig 7.1(d)) Based on these results, we found the devices to be delivering the biochemical with an efficiency of approxi-

mately 20%, where efficiency is defined as follows

$$\text{Efficiency} = \frac{\text{Number of moles of fluoxetine delivered}}{\text{Number of moles of electrons transferred}} \times 100\% \tag{7.1}$$

7.3.2 Controller

Sliding mode control is known for its ability to deal with uncertainties and unmodelled dynamics of a system [83]. This is due to their design, where once the controller drives the system to the designed sliding manifold, the system stays there, sliding along the manifold until it reaches equilibrium. The major issue with the control design is the chattering of the control signal that tends to appear [101]. There have been many methods to deal with this so the control method can be applied in real-life systems. It is applied in robotics, process control, induction motors, and power converters [23, 68, 100].

A similar sliding mode controller developed in [33] is being utilized for this output feedback control experiment (see Fig 7.2). The controller developed in [33] was shown to perform well with a mechanistic model of an ion pump where saturation can occur. These challenges are ones that we are expected to encounter in these experiments and inspired confidence that the proposed controller would be a viable.

To begin, we consider our system to be an affine nonlinear system with input saturation represented by the following state space model

$$\begin{cases} \dot{x} = f(x) + g(x)\phi(u) + \delta(t) \\ \dot{u} = \nu \\ y = h(x), \end{cases} \tag{7.2}$$

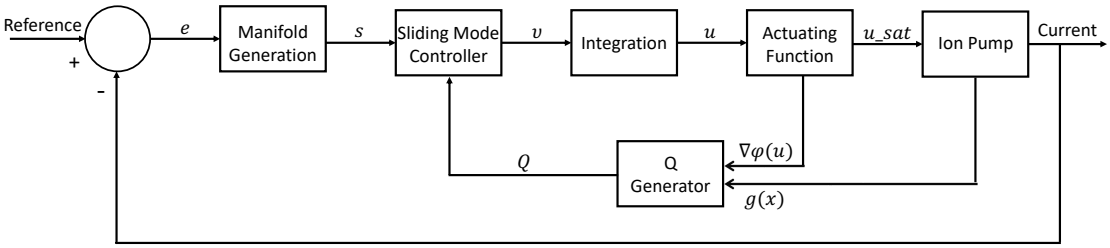

Figure 7.2: Closed-loop control architecture of actuating the ion pump to achieve the desired current value in order to control the concentration of fluoxetine.

where $x \in \mathbb{R}^n$ is the state vector. In our *in vitro* experimental setup, y is current being read from the system. An artificial control signal $\nu \in \mathbb{R}^m$ is introduced to help reduce the known chattering issue with sliding mode controllers. The output of the actuating function $\phi(u) \in \mathbb{R}^m$ is the control input vector. In this application this is the voltage applied to the system. $y \in \mathbb{R}^n$ is the output vector of the system. In the experiment here $y = x$ which is the read current from the system. The function $f(x) \in \mathbb{R}^n$ is an unknown locally Lipschitz nonlinear function and $g(x) \in \mathbb{R}^{n \times m}$ is an unknown input coefficient value. The time-varying variable $\delta(t) \in \mathbb{R}$ is a sufficiently smooth disturbance occurring throughout the experiment. We design the manifold as

$$s = Ke + \dot{e}, \tag{7.3}$$

where K is a positive constant, and e is the error defined as

$$e(t) = x(t) - r(t), \tag{7.4}$$

$x(t)$ is the current value, and $r(t)$ is the desired reference value for the current.

The sliding mode controller then calculates the artificial control input to be

used for calculating the actual control signal to be sent to the ion pump

$$\nu = \mu\rho sign(Qs), \tag{7.5}$$

where ρ is a positive coefficient and Q is obtained as

$$Q = g(x)\nabla_u \phi(u). \tag{7.6}$$

We require the sign of $g(x)$ which is found through experiments to be $\mu = -1$, which is the case when the response of the system is monotonically increasing with the input. In some cases, the rate of change in the response could be faster or slower in one direction than the other. Therefore, we introduce a heuristic switching algorithm 4 for better adaptation to these conditions.

Algorithm 4 Gain update Algorithm

```
Set parameters ρ1, ρ2
if e(i) > 0 then
    Set ρ = ρ1
else
    Set ρ = ρ2
end if
```

The actuating function used here is defined as

$$\phi(u) = A_{max} sin(u), \tag{7.7}$$

where A_{max} is the max/min voltage allowed by the ion pump. For the experiments, we set $A_{max} = 3$. To help reduce the chattering signal that tends to occur with sliding mode controllers, ν is integrated to get u. Integration is done using the trapezoidal rule where we have a sampling time of $T = 1$ sec. It should be noted that using the trapezoidal rule causes us to start the algorithm at $t = 2$ sec.

Finally, to make sure we stay in the saturated region, we applied $\phi(u)$ to get the final voltage that will be applied to the ion pump

$$u_{sat} = \phi(u(t)). \tag{7.8}$$

Algorithm 5 shows the detailed description of the implementation of the developed algorithm.

Algorithm 5 Sliding Mode Control Algorithm

Set parameters $K, n, r(i), T, \mu, u(1)$
for $i = 2 : n,$ **do**
  Read the current output $y(i)$
  $e(i) = y(i) - r(i)$
  $s(i) = Ke(i) + \dot{e}(i)$
  Call **Algorithm** 4
  $\nu(i) = \mu\rho\text{sign}\Big(s(i)A_{max}\cos\left(u_{i-1}\right)\Big)$
  $u(i) = u(i-1) + \left(\frac{\nu(i)+\nu(i-1)}{2T}\right)$
  $u_{sat}(i) = A_{max}\sin\Big(u(i)\Big)$
end for

For details regarding the stability proof of the developed controller, readers are referred to our previous work [33]. One thing to make note of is, that stability cannot be proven in application due to not having accurate model information. Through experimental trial and error, the parameters can be adjusted to have the controller work properly.

7.3.3 Experimental Setup

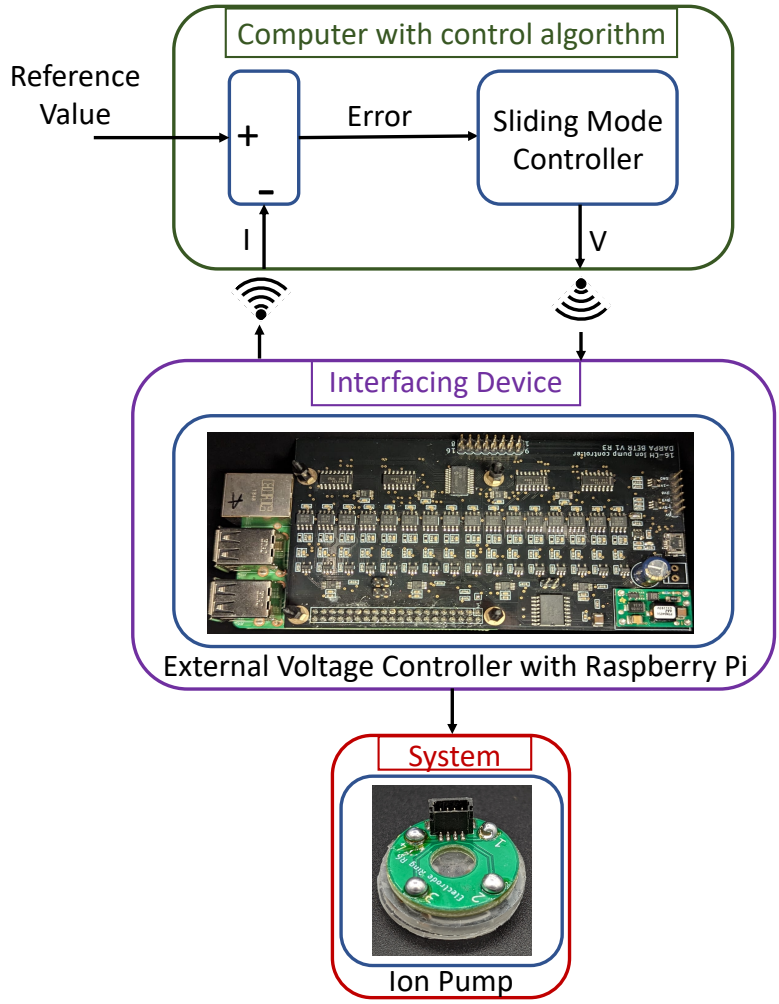

Figure 7.3: Schematic of the experimental closed-loop set up. The error is computed using the values of the current output read from the ion pump and the desired reference value. The sliding mode controller evaluates the next voltage value. This is sent to the external voltage controller with Raspberry Pi through a WiFi connection with the laptop running the control algorithm. The external voltage controller applies the value through a connected cable from the ion pump. After the voltage has been applied, the external voltage controller reads the current from the ion pump and sends it back to the laptop with the control algorithm through WiFi closing the loop.

The ion pump is integrated with a printed circuit board (PCB) interface which connects the electrodes in the pump to an external power supply. We use silver pins and a conductive silver paste to connect the Ag/AgCl electrodes in the reser-

voir to the PCB which is then soldered to the ion pump. Jumper cables are used to connect this PCB to a Raspberry Pi based external voltage controller which can connect to the WiFi [69]. The device is seated in a six-well plate with a buffer solution acting as the target for delivery. The external voltage controller applies the control commands it receives from the sliding mode controller to the ion pump. The setup is able to successfully apply up to 16 different voltages with a resolution of $1.95mV$ per channel. In addition, it is able to successfully read up to 16 current values from the ion pump but at a potential sacrifice in the accuracy in the reading, where the resolution of the current reading is $0.125nA$. This could lead to extra uncertainty in the system.

Fig 7.3 shows the closed-loop experimental setup. The error is calculated using the value of the current output read from the ion pump and the desired reference value. The sliding mode controller evaluates the next voltage value to drive the current towards the desired reference. This is sent to the Raspberry Pi through a WiFi connection between the Raspberry Pi and a laptop where the control algorithm is running on. The Raspberry Pi applies the voltage value to the ion pump through a connected cable. This closes the loop.

7.4 Results

7.4.1 *In silico*

Model

To demonstrate the ability of the sliding mode controller to control the current in an ion pump and to verify its stability, the model of the proton pump developed chapter 6 is used. To briefly remind the reader, the data used for the model was collected during controlling the change in the mean fluorescence intensity of media

in a proton pump in chapter 4. The model is as follows

$$
\begin{aligned}
\mathrm{u} > \epsilon &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{c_1 x_2}{d_1 + x_2} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{c_1 x_2}{d_1 + x_2} \end{cases} \\
\epsilon \geq \mathrm{u} > 0 &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{cc_1 x_2}{dd_1 + x_2} - g x_1 \\ \dot{x_2} D(x_1 - x_2) + u\frac{cc_1 x_2}{dd_1 + x_2} \end{cases} \\
0 \geq \mathrm{u} \geq \delta &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{cc_2 x_1}{dd_2 + x_1} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{cc_2 x_1}{dd_2 + x_1} \end{cases} \\
\delta > \mathrm{u} &\begin{cases} \dot{x_1} = D(x_2 - x_1) + u\frac{c_2 x_1}{d_1 + x_1} - g x_1 \\ \dot{x_2} = D(x_1 - x_2) + u\frac{c_2 x_1}{d_2 + x_1}. \end{cases}
\end{aligned}
\tag{7.9}
$$

x_1 is the ion concentration in the area we are trying to control the mean fluorescence intensity and x_2 is the ion concentration in the ion reservoir and ion bridge. Variable D is the natural diffusion rate of the dye into the media. The variable u is the applied voltage into the system. The parameters c_i and cc_i are the maximum rates of change of the ions in the hill function due to u. Parameters d_i and dd_i are the half maximals values of the hill function. Although x_1 is the ion concentration in only a small portion of where the image is taken, diffusion should still be accounted for since ions can still flow freely through the area. The variable g is added as a leakage term to account for this.

The ability of the sliding mode controller is shown using the model and performing output feedback control using the algorithm. Although an actual value can't be found to show stability, from the proof, it is reasonable to assume that a large enough value for ρ will allow for stability. The drawback of using a large value for ρ will be an increased amount of chattering in the control signal, but it

is an acceptable tradeoff for stability. We demonstrate the ability of the controller to perform well using two different trajectories: a slow decline and a constant trajectory.

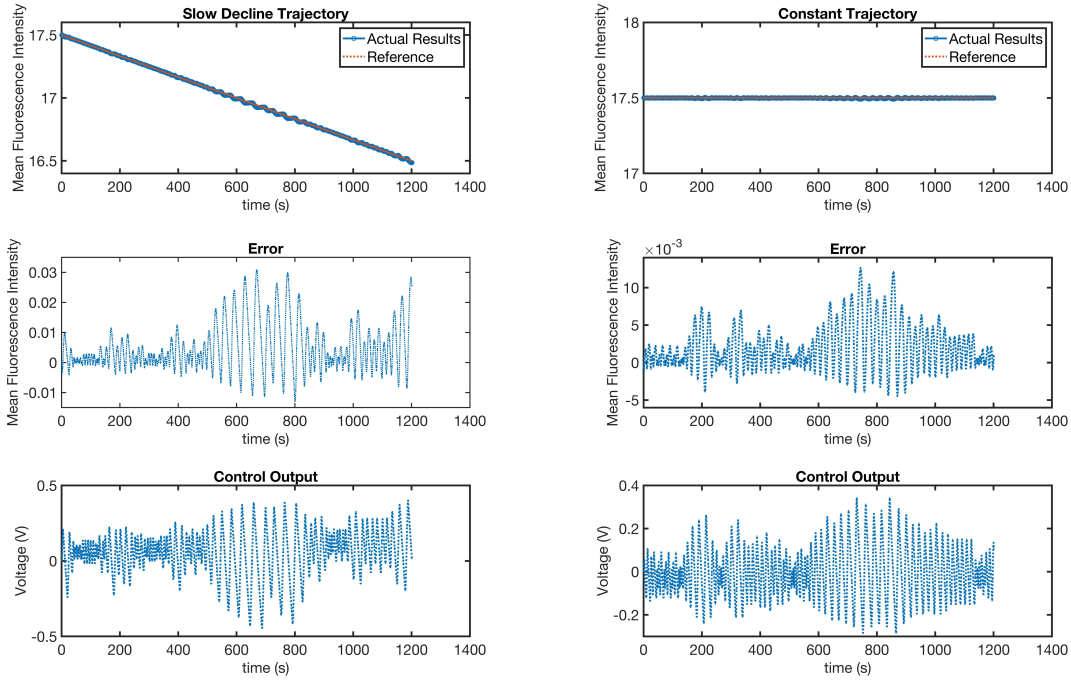

Figure 7.4: *In silico* results of the sliding mode controller. Left, are the results when the trajectory is a slow decline. Right, are the results when the trajectory is constant. The top plots is the output of the model, the middle shows the error between the system output and reference, and the bottom is the control value being applied to the system.

The results of the *in silico* study are shown in Fig 7.4. The top plots are the system responses in blue and the reference being tracked in red. Middle plots show the error. Bottom plots show the control effort being the sliding mode control. The left plots are for a decreasing reference value while the right plots are for a constant reference of 17.5 fluorescence mean pixel value. We show sliding mode controller does chatter a bit, but is able to keep the system the fluorescence intensity, x_1, near the desired trajectory. We see that the error is kept low throughout both

simulations.

7.4.2 *In vitro*

Fig 7.5 shows experimental results for feedback control on current in the ion pump device using sliding mode control. In the first column the blue lines indicate the desired current set as a reference for the feedback control algorithm. The red line is current being measured from the device in real-time. The second column shows the controller output (i.e., voltage) being applied to the ion pump in green. The third column shows corresponding tracking error in cyan.

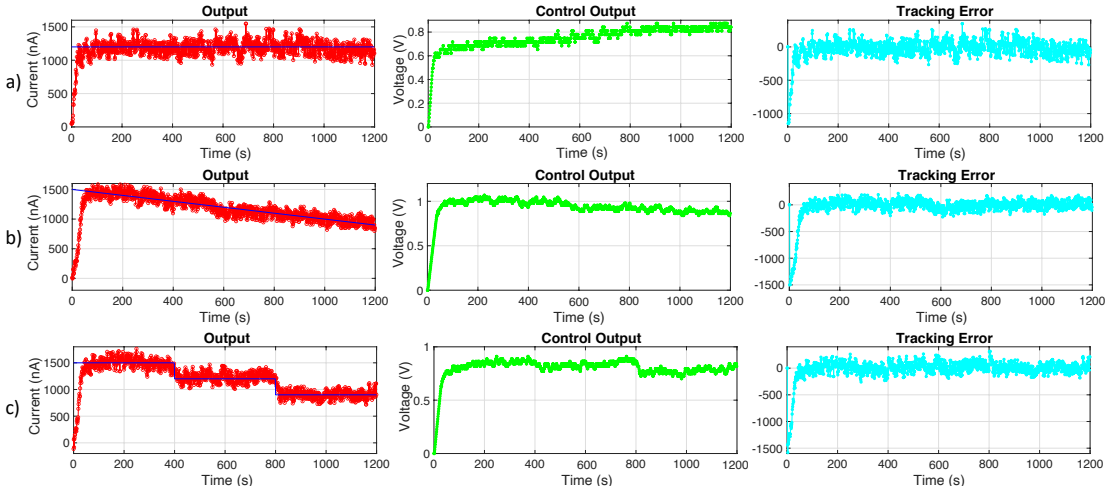

Figure 7.5: Experimental results for feedback control on current in the ion pump device using sliding mode control. Row (a) shows ion pump response to a constant reference signal at $1200nA$. Row (c) shows ion pump response to step changes in the reference signal starting at $1500nA$ dropping by $300nA$ each 400 seconds. Row (b) shows ion pump response to a gradual decline reference signal beginning at $1500nA$ and ending at $900nA$.The blue line in the first column indicates the desired current set as a reference for the feedback control algorithm. The red curve represents the measured current from the device in real-time. The second column shows the control output that is delivered to the device in green. The third column shows the error between the desired and measured current referred to as tracking error in cyan.

We establish a value for the reference signal by estimating the amount of

current required to deliver a dosage of $0.45mg$ fluoxetine. Based on device calibration, we estimate that a current of $1.2\mu A$ ($1200nA$) is required to meet the desired dosage of biochemical.

In the first experimental setup (see Fig 7.5(a)), we test the controller's performance on a constant reference signal of $1200nA$. In the second experiment (see Fig 7.5(b)), we test the controller's performance on a gradual descending reference signal starting at $1500nA$ dropping to $900nA$ by the end of the experiment. In the third experiment (see Fig 7.5(c)), we test the controller's performance on a step wise descending reference signal starting at $1500nA$, dropping by $300nA$ every 400 seconds. Note the fluctuations around the reference signals at the steady-state has relative error of less than 7% in all the experiments. We set the steady-state to be after the first 100 seconds of the experiment to allow for all three runs to reach the reference from the starting current of $0nA$. Since the amount of biochemical delivered is determined by integrating over the measured current, we don't expect that the high frequency fluctuations in current measurements has a high impact on the overall uncertainty in the biochemical delivery. Although the fluctuations make it difficult to keep the current exactly at the reference, the average current throughout each experiment can be calculated to show the desired amount of fluoxetine is being delivered. The averages for each reference signal as as follows: constant - $1171.49nA$, gradual decline - $1166.45nA$, decreasing steps - $1184.75nA$. Table 7.1 shows the relative error information along with the averages.

Table 7.1: Quantitative measures of all the experiments.

Experiment	average output signal	average steady-state relative error
constant	$1171.49nA$	7%
gradual decline	$1166.45nA$	6%
decreasing steps	$1184.75nA$	7%

7.5 Discussion

We see that the controller had to gradually increase the voltage applied to the ion pump in order for it to maintain the constant reference. This is due to the device's decaying performance as the experiment wears on. This highlights the need for an adaptive control strategy and why open-loop control could not deliver the amount of concentration wanted. One thing to notice is that the average current throughout the experiment was always lower than the actual reference. This leads to the idea of increasing the reference slightly passed the wanted reference to offset the consistent error being seen.

A challenge that needs to be addressed to achieve better control of the ion pump is the selection of the tuning parameters. The reason for this is that the efficiency of the ion pump tends to decline over time. This would mean that the response will change later in the experiments which could affect the performance of the sliding mode controller. Adaptive parameters could lead to better performance since they would adjust as needed to the varying response of the ion pump.

Some of the oscillatory behaviour of the current could be due to the interfacing board used. When the controller sends a voltage value that is smaller than the resolution of $1.95mV$ then the board is unable to send the exact voltage value. This can cause slight oscillatory behavior with the current as small changes in voltage are able to cause large changes in the current as seen in Fig 7.5. Although, exact control of the current at the reference is not achievable, we see in Table 7.1 that the average of the current is near the desired value and is within acceptable relative error bounds.

This application can have an impact on the wound-healing field. People can respond differently to the same applied medical treatment for reasons such as age, ethnicity, and genetics [58]. This makes it difficult for physicians to treat

people optimally. Through precision medicine, custom treatment options can be created depending on an individual's needs. Closed-loop control allows for the customization of the treatment strategy by changing the strategy in real-time as new information is being given [24]. This can be done by interfacing the feedback control regulated ion pump presented here with a higher-level control algorithm that provides the reference signal. In this way, wound healing might be treated by sensing how well a wound is responding to a specific treatment and changing the dosage of biochemicals as needed in real-time.

7.6 Conclusion

We demonstrated a method for delivering a desired amount of biochemical using closed-loop control. By finding the efficiency of the ion pump in delivering fluoxetine, we know the amount of fluoxetine being delivered using the current of the device. A sliding mode controller was used *in vitro* and it successfully kept the current at the desired reference, which allowed the desired amount of fluoxetine to be delivered.

Chapter 8

Control of cell migration using model predictive control

This chapter is currently finishing revisions for publication. The dissertation author was the primary investigator and author of this paper.

8.1 Introduction

Cell migration is required for proper development and health. It is involved in stem cell homing, tissue regeneration, and immune response [16, 59]. Cell migration also plays a role in disease, where cancer cells use movement toward the vasculature to cause cancer metastasis. It also can cause arthritis and asthma, where an abundance of inflammatory cells collect during cell migration [93]. In an effort to elicit wanted responses from biological systems using electric fields, bioelectronics, have shown their ability to help when leveraging feedback control. An example being [81], where the authors were able to control the membrane

voltage of mammalian stem cells.

One way cell migration occurs naturally in the body is through electrical signals. Physiological electric fields (EF) have been sensed during wound healing and embryonic development [20]. Leveraging an electric field to direct cell migration can be done utilizing bioelectronics [102]. One concern is eliciting an undesired system response by applying an electric field. This is why an effective closed-loop control setup is necessary to reduce offset effects while achieving the desired goal.

A control method that is capable of minimizing the control effort is model predictive control [30]. However, this can be difficult to achieve since predictive models of biological systems are hard to derive due to their complexity and how important models are for developing control algorithms. Due to models not being able to capture the system exactly, another control method that can be viable is sliding mode control. This is because of its robustness to uncertainties, although it comes with a tradeoff of oscillation in the system response [19].

Although a mathematical model can be difficult to derive for a biological system, deep learning has proven to be an alternative method to derive a model of a system. Long short-term memory (LSTM) deep learning models have shown an ability to capture temporal patterns for time series predictions [36]. In [76], the authors developed a predictive deep-learning model for the migration of Cranial Neural Crest Cell (CNCC) under an electric field. They were able to show the models' ability to predict CNCC's cell migration for one-time step ahead. In addition, they have shown the models' ability to be adapted for one-time step ahead prediction of other cell types (i.e., keratocyte and keratinocyte) migration using transfer learning. With a predictive model, it is possible to develop a control algorithm for cell migration while minimizing the control effort. Minimizing the EF applied will help limit any offset effects, such as the polarization of macrophages

[12].

In this paper, we propose a switching controller that utilizes a model predictive controller and a sliding mode controller. The sliding mode control will be used when the error exceeds a certain bound. The design of the sliding mode controller can be found in [33]. Once the controller is within the bound, the model predictive controller is used to compute the control signal sent to the system. This combination should allow for the system to be driven to the reference with minimal oscillations.

8.2 Methods

In this section, we introduce the deep-learning model used and briefly describe how it works. We then discuss the model predictive controller, propose a switch control and briefly describe how the *in silico* setup works.

8.2.1 Deep Learning Model

In [76], LSTM models were developed for cell migration under an electric field. It was trained on data using cell movement under different electric field strengths. Figure 8.1 shows how the data was quantified and the model created. In Figure 1-a, we see that the original positions of each cell tracked is given. This is then quantified by a metric termed directedness at each time step using the cosine of θ (i.e., $cos(\theta)$). The θ angle is calculated based on the original position of the cell at time 0 acting as the center. One line extends the cell's position at the current timestep, and the other line that gives us angle θ is the direction of the electric field. This leaves us with directedness values for all the tracked cells at each timestep. This data was then used to train the LSTM model. In Figure 1-b,

we see that the input to the LSTM model is the previous 20 directedness values of the cells, the previous 19 EF inputs, and the current EF value applied to the system (20 total). The LSTM model uses this information to predict the cells' directedness values for the next time step.

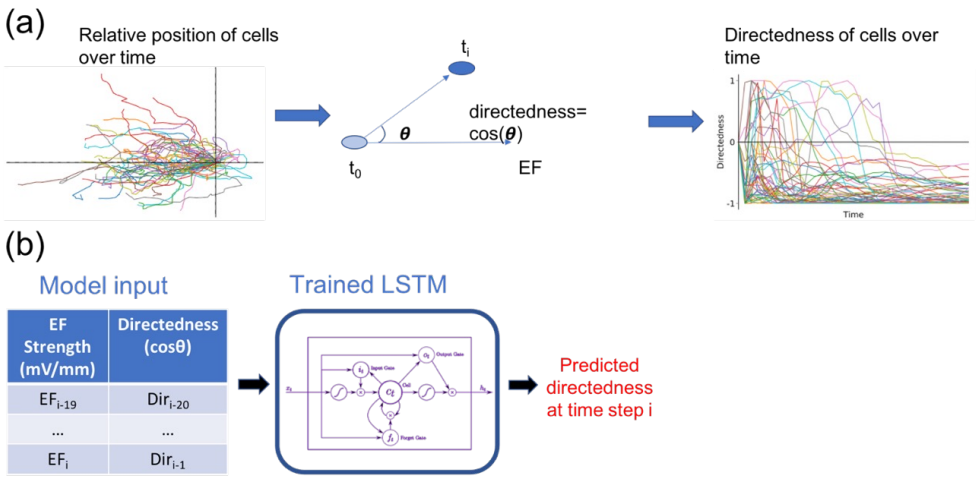

Figure 8.1: a) Quantifying cell movement from the data. b) How the LSTM model works.

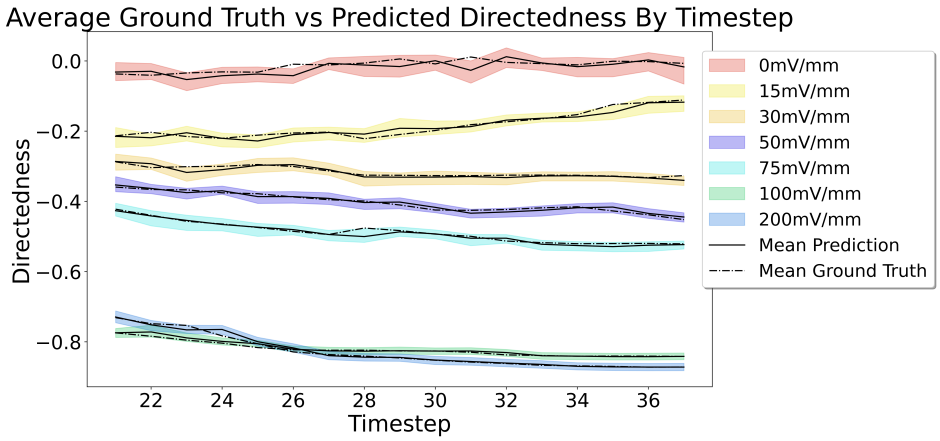

Figure 8.2: The LSTM models ability to prediction of CNCC's cell migration under various EFs.

We see in Figure 8.2 that the models successfully predict the CNCC's cell migration data. This gives confidence that the models can be used for a model

predictive control framework.

8.2.2 Controller

Model predictive control leverages a model to achieve an optimal goal designed in the cost function. The cost function normally tries to minimize the error between the desired trajectory and the output of the system and control effort. To effectively perform this, the MPC controller uses future predictions within a receding horizon. This allows the controller to search the possible solutions throughout the horizon to pick the best control value to push the system toward the reference within those time steps. This is an advantage since looking at multiple future time steps allows the controller to account for future dynamics that other control methods, such as PID, are not able to do. The difficulty for MPC to succeed lies in the model. If the model isn't able to properly predict the system response, it can lead to poor control performance. To help mitigate this, the system error is also added to the cost function shown in equation 8.1.

We set up the control problem as follows

$$\begin{aligned} \underset{u}{\text{minimize}} \quad & J(x(k), u, e_s) \\ \text{subject to} \quad & x(k+1) = f(x(k), u) \\ & x(k) = x_0 \\ & 0 \leq u \leq 0.2 \end{aligned}$$

where the cost function is defined as

$$J(x(k), u) = \frac{1}{2M} \sum_{i=1}^{M} \Big(x_{k+i} - r_{k+i} \Big)^2 + \frac{1}{2} e_s^2 + \frac{1}{2} u^2. \tag{8.1}$$

The receding horizon length is M, the reference is r_{k+i}, the model predictions are x_{k+1}, and $e_s = y_k - r_k$ is to account for the actual system error at the current time step. This allows us to add information about the system response since the model does not represent the system perfectly. The input of the system is represented by u. The constraint on u is because the data used to create the LSTM models is bounded by [0,0.2]. One important thing to mention is that the control effort calculated is used throughout the entire horizon rather than at individual time steps in the horizon, as shown in Figure 8.3. This was done for two reasons: 1) computation time and 2) that the correct control value would work throughout the whole horizon as seen in the data [76].

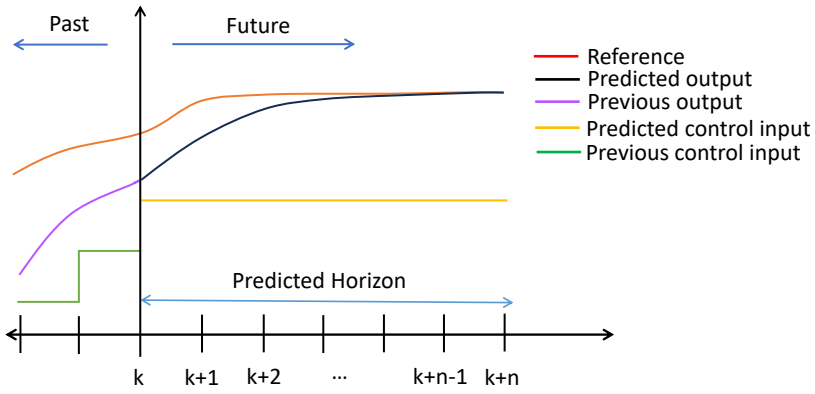

Figure 8.3: Figure illustrating the receding horizon strategy

To help with keep ensure that the MPC controller is working within an interval around the reference trajectory, a sliding mode controller is used to push the system within an ϵ bound. When the system is within bounds, the MPC controller takes over and is used to try and keep the system around the reference. Figure 8.4 depicts how the switching controller works. The sliding mode controller is used due its robustness to uncertainties [19] and the design can be seen in Chapter 7.

For the overall control strategy, an error is calculated between the reference and the system output. The error is sent to both controllers, and the MPC

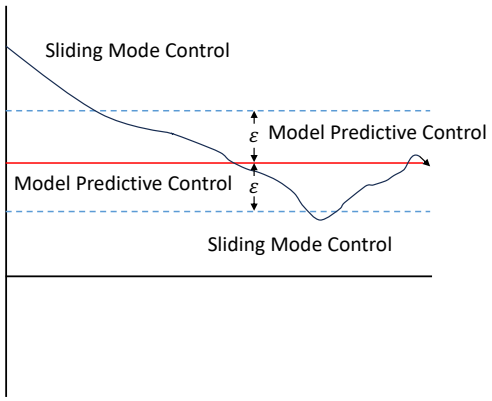

Figure 8.4: Schematic of the switching setup

controller is also given the output information of the system. The MPC controller uses the SQP optimizer to find the control value that minimizes the cost function throughout the horizon. At the same time, the sliding mode controller picks a value that will drive the system to the reference trajectory. The sliding mode controller value is used if the error is outside the bound. The MPC controller value is used if the error is within the bound. This is sent to the system that gives the next output. The output is returned to the MPC controller and is then returned with the reference to get an error. Figure 8.5 shows a block diagram of the overall control strategy.

8.2.3 Simulation setup

For the *in silico* study, two different LSTM models from [76] are used to represent the model for the MPC and the actual system. The models have similar behavior but exhibit unique transients. This was done since models, even with deep learning, are hard to make such that they perfectly capture the response of the system to an actuation. The goal is to have the average directedness of the cells track the reference while minimizing the control effort.

The receding horizon chosen is 8. This is due to earlier simulations suggest-

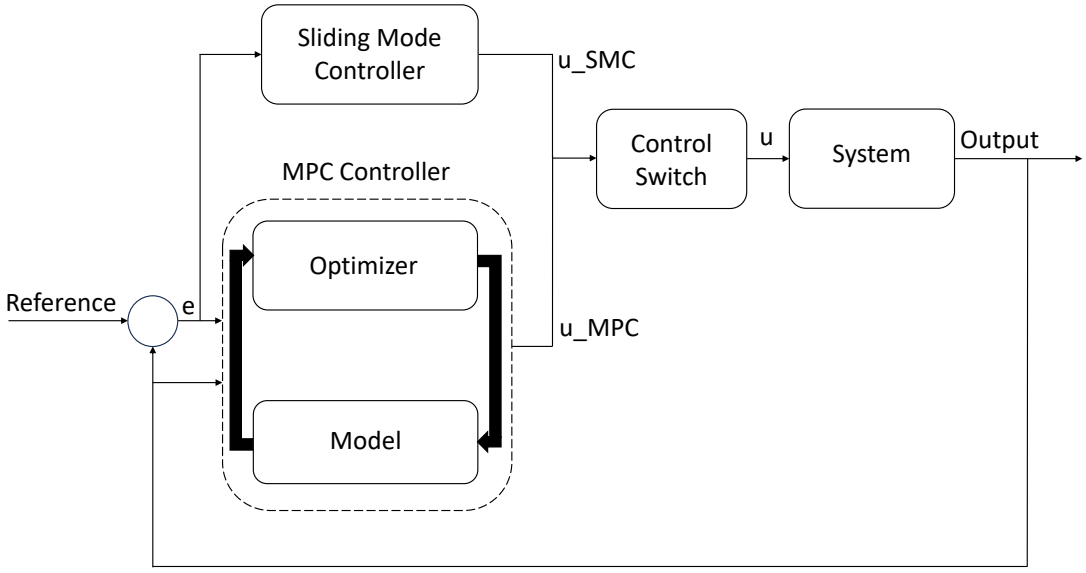

Figure 8.5: Schematic of the switching controller setup

ing this was a good compromise between computation time and accuracy. For optimization, fmincon leveraging SQP is used due to its faster computation time. The error threshold and step size change for the control input found through optimization is made larger to allow faster convergence at the expense of accuracy. The model is run for 100 timesteps. Each timestep represents 5 minutes.

We run a simulation where the parameters on the optimization scheme are set to default and there is no time constraint put on the optimization process. This was to verify that the MPC controller would work well when allowed to run as long as it needed. It also shows the differences between the LSTM models used for the model and system in the MPC set up. Then a total of 9 simulations are run comparing three control strategies: MPC, sliding mode control, and the proposed combination control utilizing both MPC and SMC for various parameter settings. The paramaters for the fmincon SQP optimizer are as follows: max iterations allowed was 25, Function tolerance was 0.005, and Step tolerance was 0.01. There was also a stopping function to stop the optimizer after 190 seconds of run time.

All follow a constant reference where the first reference is set to -0.55, and then the reference decreases each time by -0.025 until it reaches -0.75 for the final simulation. The voltage allowed is bounded between [0,0.2] since these are the bounds of the voltage the LSTM model was trained on. 50 cells are initialized for the LSTM model, and the average directedness value of the cells is used as the output.

8.3 Results

Figure 8.6 shows the ability of the MPC controller to drive the system to the reference when we set the reference to -0.7. The issue is the computation time which we see, takes longer than allowed for the experiments in Chapter 5. The average computation time for the simulation was 863 seconds, about 3 times more than wanted since the sampling time in *in vitro* experiments are 300 seconds.

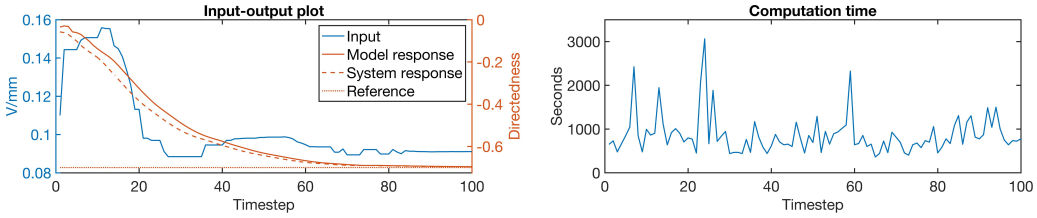

Figure 8.6: *In silico* result with no thresholds. Reference is set to -0.7.

All the simulations below meet the requirement of under 5 minutes of computation time after changing the parameters for the optimization and adding a stopping function to limit the time allowed for the optimizer. This is wanted to show that this method could be applied to the EF experiments shown in chapter 5.

Figure 8.7 shows the results for the three highest reference values: -0.55, -0.575, -0.6. We see that the MPC controller was unable to drive the average directedness

of the system toward the reference. The proposed SMC+MPC switching controller tends to take on similar control inputs as the sliding mode controller for the two lower reference value plots. When the reference value is -0.6, we see that the SMC+MPC switching controller starts to have a smoother control input and oscillates less.

Figure 8.8 shows the results for the three middle reference values: -0.625, -0.65, -0.675. Here, we see that the MPC controller performs better than in Figure 8.7. The MPC controller is able to drive the system response toward the reference value but only settles on it when the reference is -0.65. The sliding mode controller continues to oscillate but begins to oscillate less in the final plots when the reference is -0.675. The SMC+MPC continues to use the MPC portion of the controller for the majority for the simulations except for in -0.625 where the system response seems to go out of the ϵ bound twice, and the SMC portion is used to drive back into the bound.

Figure 8.9 shows the results for the three lowest reference values: -0.7, -0.725, -0.75. These average directedness values would mean a strong moment of cells towards to the cathode. The MPC controller overshoots the reference for the 3 different simulations shown. The sliding mode controller has less oscillations in these simulations when compared to 8.7 and 8.8. The SMC+MPC did overshoot and go beyond the ϵ bound but was pushed back in after the SMC controller was switched on for a short time.

Tables 8.1 and 8.2 show that the MPC controller always performed worse regarding the error. MPC generally had the lowest control input when it was the most inaccurate. Once it performed better, the mean control input was the second highest. The MPC controller was always the worst in terms of mean absolute error (MAE), as seen in 8.1. This was due to it taking longer to drive

the system toward the reference. The sliding mode controller would oscillate in its control input which would cause an oscillation in the system response. Despite the oscillation, the SMC controller was second in error and came in second or third for the mean control input value. The proposed SMC+MPC controller performed the best in terms of mean absolute error for all the simulations. It also ended up having the highest mean control effort as shown in Table 8.2. Before those, it was typically second in control effort.

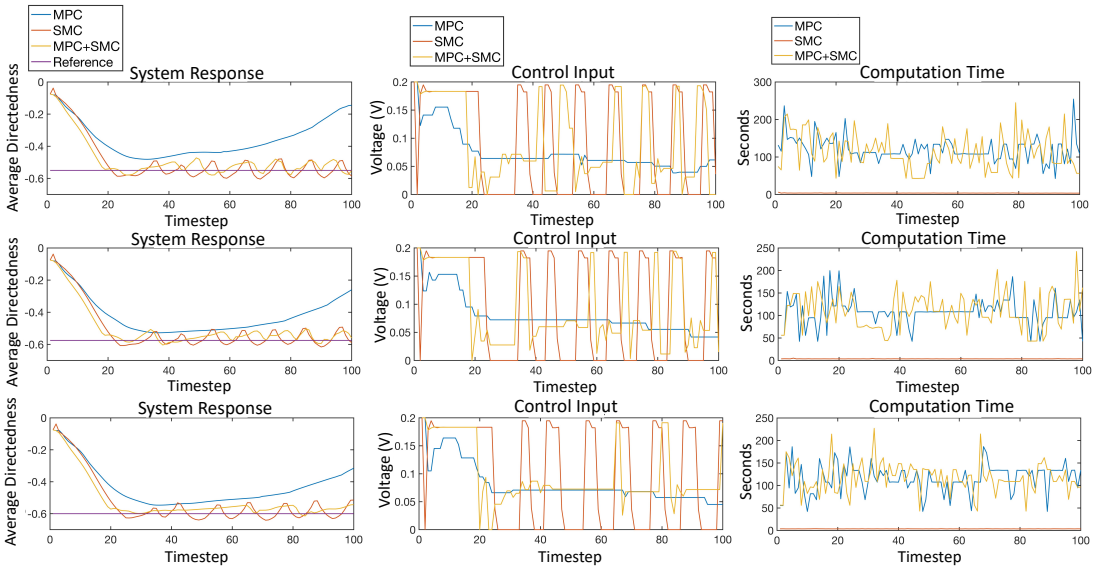

Figure 8.7: *In silico* results for 3 experimental runs. The top row has a reference value of -0.55, the second row has a reference value of -0.575, and the bottom row has a reference value of -0.6. The first column is the system response of each simulation, the second column is the control input applied to the system, and the final column is the computation time.

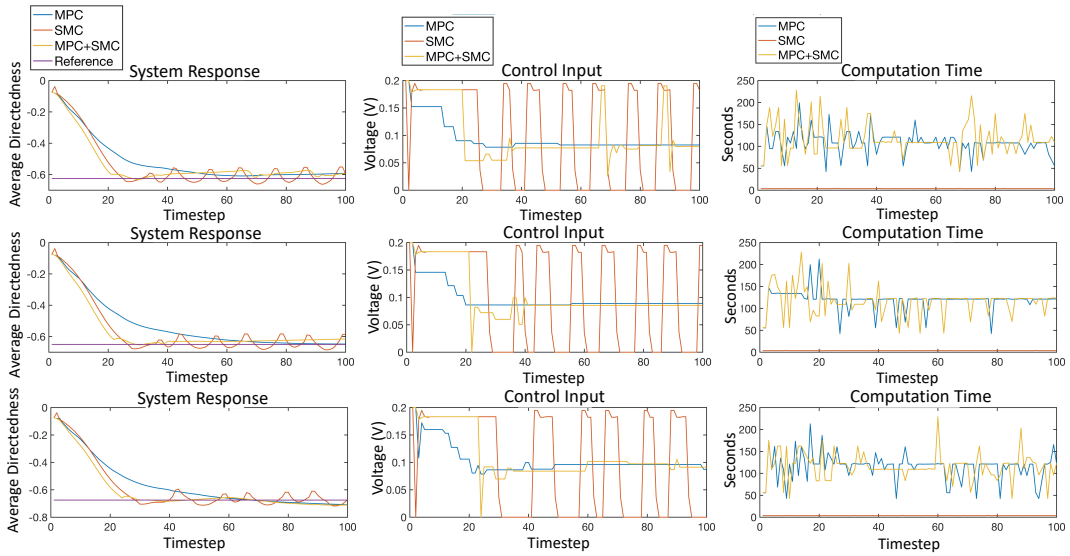

Figure 8.8: *In silico* results for 3 experimental runs. The top row has a reference value of -0.625, the second row has a reference value of -0.65, and the bottom row has a reference value of -0.675. The first column is the system response of each simulation, the second column is the control input applied to the system, and the final column is the computation time.

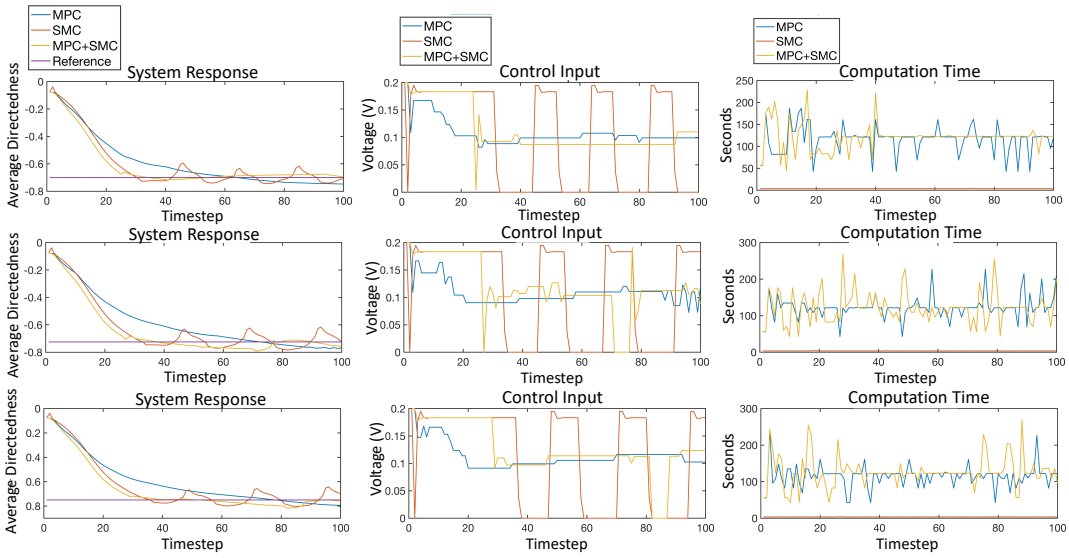

Figure 8.9: *In silico* results for 3 experimental runs. The top row has a reference value of -0.7, the second row has a reference value of -0.725, and the bottom row has a reference value of -0.75. The first column is the system response of each simulation, the second column is the control input applied to the system, and the final column is the computation time.

Table 8.1: Mean absolute error for the three different control strategies across all references used.

	MPC	SMC	SMC+MPC
MAE (-0.55)	0.1974	0.0829	0.0722
MAE (-0.575)	0.1581	0.0851	0.0768
MAE (-0.6)	0.1687	0.0917	0.0808
MAE (-0.625)	0.1118	0.0953	0.0873
MAE (-0.65)	0.1222	0.0999	0.0862
MAE (-0.675)	0.1249	0.1064	0.0876
MAE (-0.7)	0.1345	0.1168	0.0924
MAE (-0.725)	0.1570	0.1297	0.1076
MAE (-0.75)	0.1574	0.1375	0.1108

Table 8.2: Mean control effort for the three different control strategies across all references used.

	MPC	SMC	SMC+MPC
MCI (-0.55)	0.0746	0.0926	0.0902
MCI (-0.575)	0.0797	0.0963	0.0936
MCI (-0.6)	0.0813	0.0978	0.0965
MCI (-0.625)	0.0945	0.1014	0.0997
MCI (-0.65)	0.0978	0.1033	0.1029
MCI (-0.675)	0.1033	0.1024	0.1115
MCI (-0.7)	0.1083	0.1013	0.1129
MCI (-0.725)	0.1096	0.1119	0.1207
MCI (-0.75)	0.1135	0.1137	0.1246

8.4 Discussion

When comparing the simulation where default optimization settings and no stopping function was used vs. where they were put in place, we see that the MPC controller loses some performance in tracking, but overall is acceptable considering the decrease in computation time. The controller can still achieve the wanted response of the system with a lower control effort than seen in Chapter 5.

We see that all the control algorithms did lower the magnitude of the mean control effort compared to what was seen in chapter 5. Although the MPC controller failed to perform well when the reference value was higher than -0.65, it could drive the system toward the reference when lowered. Considering the experimental goals in Chapter 5 to control cell migration, we should expect references between $\pm[0.6, 0.8]$ to most likely be used.

The reference values between $[-0.5, -0.625]$ showed that the MPC controller, with all the constraints and proposed receding horizon, failed to drive the system to the reference throughout the simulation. This is where the sliding mode controller showed that it could perform well, although it did oscillate about the reference for all simulations. The proposed switching controller was able to take the best parts of both of these and drive the system toward the reference.

Since all control methods were under the time constraint, it shows that all methods would be viable in a similar experimental setting seen in Chapter 5. Although the proposed switching controller did have a slightly higher mean control value in most situations, it did have the lowest mean absolute error. This gives confidence that it would be an appropriate control strategy for controlling cell migration while minimizing the control effort. This is important since electric fields are known to help cause the polarization of macrophage cells [12, 43]. An untimely increase of M2 cells during the wound-healing process can hinder wound

healing time rather than expedite it [45].

The proposed switching control algorithm could still be improved by choosing different thresholds, using a radial basis function neural-network estimator to estimate the difference between the model and the system output, and optimizing the code to run faster. Chapter 3 showed that the radial basis function neural network can be used as a next-step predictor in an online fashion. This could increase the accuracy in the early timesteps where the model and system differ in their responses.

8.5 Conclusion

Minimizing the control effort (voltage) to drive cell migration is beneficial to help reduce unwanted side effects. To this end, a switching controller is proposed, which uses a sliding mode controller and a model predictive controller. An *in silico* study is performed using the three controllers across several reference values. All three did lower the magnitude of the mean control effort. The MPC controller never had to use the max control value (0.2v) allowed. The difference between the three control methods came in their ability to drive the system to the reference and computation time. In keeping the system near the reference, the switching controller performed better than the other two controllers for all references chosen and had the lowest mean absolute error.

Chapter 9

Conclusion

This work was motivated by the need to control bioelelectronics to aid in precision medicine. Bioelectronics are now better able to integrate with biology allowing them to help in areas never thought of before. Currently, most methods using bioelectronics are in an open-loop fashion where predetermined delivery of therapeutics is done. The issue with this is that people tend to react differently to different treatments and this could cause overexposure to some therapeutics. In order to provide personal medicine to a patient, closed-loop control of bioelectronics is needed. We initially leverage data-driven control for this task since modeling complex systems can be impossible. Through the use of an RBF-NN to approximate the ideal control law, we were able to control the change pH and cell migration in a desired way. A sliding mode controller was then used for

9.1 Summary of work

Chapter 3 introduces the radial basis function neural network as a predictor for an optogenetic dataset. A predictor is important because you are able to better control a system when you know what to expect when there isn't a model or a complete model of the system. The RBF-NN showed it was a capable predictor of the dataset in an online fashion (i.e., no training of the neural network was needed).

In Chapter 4, we addressed how to cope with controlling biological systems with the absence of prior knowledge of the system. This was due to the complexities in attempting to model the biological system. The RBF-NN was able to successful in trying to keep the pH value at the wanted reference. Although, it did work for many runs, issues appeared when keeping the minimum value of the EF allowed for too long. Saturation would occur causing delays for the controller when a change in the reference occurred.

In Chapter 5, we addressed this issue be developing an updated version of the RBF-NN used. The updated version lost the bias term but added projection to the weights in an effort to keep the control output near the bounds of the allowed EF. The method showed improvement in dealing with saturation when using a model of cell directedness. The controller was than utilized in an experiment and compared with the industry standard controller, a PID. The controller performed better than the PID but did have issues staying in the bounds due to parameter values not being tuned well enough.

Chapter 6 had us building a qualitative model of the proton pump. The reason behind this was to verify that future controllers would be able to stabilize the system since most of the experimental work is done using these proton pumps. In Chapter 7 he model is used to show that a sliding mode controller is capable

of keeping the pH at a wanted reference *in silico*. It was than used *in vitro* to control the delivery of Fluoxetine, a therapeutic drug.

In the final chapter of work, Chapter 8, we revisit the goal of controlling the migration of cells using an electric field. We noticed in 5 that the control strategy would typically be at the min/max current values allowed. Although the strategy worked, we wanted to mitigate the polarization of cells due to an EF. To do this we propose a switching controller which uses a sliding mode controller and a model predictive controller. Deep learning models were developed in [76]. We use two of these models to build the switching control framework. We show that the controller performs better than the sliding mode controller and the model predictive controller separately.

9.2 Areas for future work

We saw that the switching controller that utilized both a model predictive control and sliding mode control could drive the system toward the reference with less control effort required from experiments. It did have some issues keeping the system at the reference, though, and would allow for some overshooting or undershooting and just staying within a certain bound. To help mitigate this, more work can be done on the cost function, optimization thresholds, and adding an estimator to account for the differences in the model compared to the actual system. We saw in 3 that the RBF-NN can act as an online predictor. This could be utilized in the model to help estimate the difference between the model and the system response.

Adaptive learning rates could be an area of improvement for the sliding mode controller and the RBF-NN controller. We saw in 5 that due to poor parameters, the control output was able to grow beyond the control constraints by a large

margin. Also, due to the differing performance of the bioelectronic ion pumps, we saw that the sliding mode controller had to increase the control effort to maintain a constant reference. This suggests that adaptive parameters that change as the system does might help in overall performance of the controllers.

We have been accumulating more and more macrophage cell migration data but with varying amounts of timesteps. This makes training a deep learning model difficult. A goal would be to learn different methods that allow you to use varying sets of data with different time steps to train a model still to give good predictive results.

Finally, we see that tracking a state such as recruitment index or directedness is difficult due to the large time steps required in experiments. This can lead to some overshoot from long exposure to a high control input. To help mitigate this, another control strategy, such as funnel control, could be utilized. Funnel control aims to keep the error within a certain bound of a wanted reference. Since the cell migration goal aims at having a strong directedness, keeping the directedness within a certain bound should accomplish the task.

Bibliography

[1] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831, 2015.

[2] Daniel B. Allan, Thomas Caswell, Nathan C. Keim, Casper M. van der Wel, and Ruben W. Verweij. soft-matter/trackpy: Trackpy v0.5.0, April 2021.

[3] M Allen. Pacemakers and implantable cardioverter defibrillators. *Anaesthesia*, 61(9):883–890, 2006.

[4] David Angeli and Eduardo D Sontag. Multi-stability in monotone input/output systems. *Systems & Control Letters*, 51(3-4):185–202, 2004.

[5] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. Deep learning for computational biology. *Molecular systems biology*, 12(7):878, 2016.

[6] György Barabás, Matthew J Michalska-Smith, and Stefano Allesina. Self-regulation and the stability of large ecological networks. *Nature ecology & evolution*, 1(12):1870–1875, 2017.

[7] Iwona Bernacka-Wojcik, Miriam Huerta, Klas Tybrandt, Michal Karady, Mohammad Yusuf Mulla, David J Poxson, Erik O Gabrielsson, Karin Ljung, Daniel T Simon, Magnus Berggren, et al. Implantable organic electronic ion pump enables aba hormone delivery for control of stomata in an intact tobacco plant. *Small*, 15(43):1902189, 2019.

[8] Paul T Boggs and Jon W Tolle. Sequential quadratic programming. *Acta numerica*, 4:1–51, 1995.

[9] Fred Brauer, Carlos Castillo-Chavez, and Carlos Castillo-Chavez. *Mathematical models in population biology and epidemiology*, volume 2. Springer, 2012.

[10] Walter B Cannon. Organization for physiological homeostasis. *Physiological reviews*, 9(3):399–431, 1929.

[11] Andrew R Conn, Nicholas IM Gould, and Philippe L Toint. *Trust region methods*. SIAM, 2000.

[12] Xiaohan Dai, Boon Chin Heng, Yunyang Bai, Fuping You, Xiaowen Sun, Yiping Li, Zhangui Tang, Mingming Xu, Xuehui Zhang, and Xuliang Deng. Restoration of electrical microenvironment enhances bone regeneration under diabetic conditions by modulating macrophage polarization. *Bioactive materials*, 6(7):2029–2038, 2021.

[13] R DeCarlo and S Zak. A quick introduction to sliding mode control and its applications. *Universita'Degli Studi di Cagliari*, 2008.

[14] Harika Dechiraju, Manping Jia, Le Luo, and Marco Rolandi. Ion-conducting hydrogels and their applications in bioelectronics. *Advanced Sustainable Systems*, 6(2):2100173, 2022.

[15] Harika Dechiraju, John Selberg, Manping Jia, Pattawong Pansodtee, Houpu Li, Hao-Chieh Hsieh, Cristian Hernandez, Narges Asefifeyzabadi, Tiffany Nguyen, Prabhat Baniya, et al. On-chip on-demand delivery of k+ for in vitro bioelectronics. *AIP Advances*, 12(12):125205, 2022.

[16] Peter Devreotes and Alan Rick Horwitz. Signaling networks that regulate cell migration. *Cold Spring Harbor perspectives in biology*, 7(8):a005959, 2015.

[17] Michael Egmont-Petersen, Dick de Ridder, and Heinz Handels. Image processing with neural networks—a review. *Pattern recognition*, 35(10):2279–2301, 2002.

[18] Verónica Eisner, Martin Picard, and György Hajnóczky. Mitochondrial dynamics in adaptive and maladaptive cellular stress responses. *Nature cell biology*, 20(7):755–765, 2018.

[19] Kemalettin Erbatur, M Okyay Kaynak, and Asif Sabanovic. A study on robustness property of sliding-mode controllers: A novel design and experimental investigations. *IEEE Transactions on Industrial Electronics*, 46(5):1012–1018, 1999.

[20] Paulo Luiz Farber, Felipe Contoli Isoldi, and Lydia Masako Ferreira. Electric factors in wound healing. *Advances in wound care*, 10(8):461–476, 2021.

[21] Gene F Franklin, J David Powell, and Abbas Emami-Naeini. *Feedback control of dynamic systems*, volume 33. Pearson London, 2015.

[22] Gene F Franklin, J David Powell, Abbas Emami-Naeini, and J David Powell. *Feedback control of dynamic systems*, volume 4. Prentice hall Upper Saddle River, 2002.

[23] SJ Gambhire, D Ravi Kishore, PS Londhe, and SN Pawar. Review of sliding mode based control techniques for control system applications. *International Journal of dynamics and control*, 9:363–378, 2021.

[24] Patrick D Ganzer and Gaurav Sharma. Opportunities and challenges for developing closed-loop bioelectronic medicines. *Neural regeneration research*, 14(1):46, 2019.

[25] Tingting Gao, Yan-Jun Liu, Lei Liu, and Dapeng Li. Adaptive neural network-based control for a class of nonlinear pure-feedback systems with time-varying full state constraints. *IEEE/CAA Journal of Automatica Sinica*, 5(5):923–933, 2018.

[26] Michelle F Griffin, Peter E Butler, Alexander M Seifalian, and Deepak M Kalaskar. Control of stem cell fate by engineering their micro and nanoenvironment. *World journal of stem cells*, 7(1):37, 2015.

[27] Martin T Hagan and Howard B Demuth. Neural networks for control. In *American Control Conference (ACC), 1999*, pages 1642–1656. IEEE, 1999.

[28] Baris Hancioglu, David Swigon, and Gilles Clermont. A dynamical model of human immune response to influenza a virus infection. *Journal of theoretical biology*, 246(1):70–86, 2007.

[29] Simon Haykin. *Neural Networks and Learning Machines*. Pearson, 3 edition, 2009.

[30] KS Holkar and Laxman M Waghmare. An overview of model predictive control. *International Journal of control and automation*, 3(4):47–63, 2010.

[31] Maximilian Hörner, Nadine Reischmann, and Wilfried Weber. Synthetic

biology: programming cells for biomedical applications. *Perspectives in biology and medicine*, 55(4):490–502, 2012.

[32] Bashir Hosseini Jafari, Ksenia Zlobina, Giovanny Marquez, Mohammad Jafari, John Selberg, Manping Jia, Marco Rolandi, and Marcella Gomez. An architecture for feedback control of biological systems via bioelectronic device with applications to wound healing. *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, 2021.

[33] Bashir Hosseini Jafari, Ksenia Zlobina, Giovanny Marquez, Mohammad Jafari, John Selberg, Manping Jia, Marco Rolandi, and Marcella Gomez. A feedback control architecture for bioelectronic devices with applications to wound healing. *Journal of the Royal Society Interface*, 18(185):20210497, 2021.

[34] Zhong-Sheng Hou and Zhuo Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013.

[35] Robert J Howlett and Lakhmi C Jain. *Radial basis function networks 2: new advances in design*, volume 2. Springer Science & Business Media, 2001.

[36] Yuxiu Hua, Zhifeng Zhao, Rongpeng Li, Xianfu Chen, Zhiming Liu, and Honggang Zhang. Deep learning with long short-term memory for time series prediction. *IEEE Communications Magazine*, 57(6):114–119, 2019.

[37] Mohammad Jafari and Marcella Gomez. Online machine learning based controller for coupled tanks systems. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2019.

[38] Mohammad Jafari, Giovanny Marquez, John Selberg, Manping Jia, Harika Dechiraju, Pattawong Pansodtee, Mircea Teodorescu, Marco Rolandi, and Marcella Gomez. Feedback control of bioelectronic devices using machine learning. *IEEE Control Systems Letters*, 5(4):1133–1138, 2020.

[39] Mohammad Jafari, Vahid Sarfi, Amir Ghasemkhani, Hanif Livani, Lei Yang, Hao Xu, and Rasool Koosha. Adaptive neural network based intelligent secondary control for microgrids. In *2018 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–6. IEEE, 2018.

[40] Mohammad Jafari, Alireza Mohammad Shahri, and Saeed Bagheri Shouraki. Attitude control of a quadrotor using brain emotional learning based intelligent controller. In *2013 13th Iranian Conference on Fuzzy Systems (IFSC)*, pages 1–5. IEEE, 2013.

[41] Mohammad Jafari, Hao Xu, and Luis Rodolfo Garcia Carrillo. Brain emotional learning-based intelligent controller for flocking of multi-agent systems. In *American Control Conference (ACC), 2017*, pages 1996–2001. IEEE, 2017.

[42] Michael A Johnson and Mohammad H Moradi. *PID control*. Springer, 2005.

[43] Heemin Kang, Sungkyu Kim, Dexter Siu Hong Wong, Hee Joon Jung, Sien Lin, Kaijie Zou, Rui Li, Gang Li, Vinayak P Dravid, and Liming Bian. Remote manipulation of ligand nano-oscillations regulates adhesion and polarization of macrophages in vivo. *Nano letters*, 17(10):6415–6427, 2017.

[44] Mina Khoshdeli and Bahram Parvin. Feature-based representation improves color decomposition and nuclear detection using a convolutional neural network. *IEEE Transactions on Biomedical Engineering*, 65(3):625–634, 2018.

[45] Sang Yong Kim and Meera G Nair. Macrophages in wound healing: activation and plasticity. *Immunology and cell biology*, 97(3):258–267, 2019.

[46] Ronald J Korthuis. Skeletal muscle circulation. In *Colloquium Series on Integrated Systems Physiology: From Molecule to Function*, volume 3, pages 1–144. Morgan & Claypool Life Sciences, 2011.

[47] Annu Lambora, Kunal Gupta, and Kriti Chopra. Genetic algorithm-a literature review. In *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 380–384. IEEE, 2019.

[48] Eugene Lavretsky and Kevin A Wise. Robust adaptive control. In *Robust and adaptive control*, pages 317–353. Springer, 2013.

[49] Maciej Ławryńczuk. Modelling and nonlinear predictive control of a yeast fermentation biochemical reactor using neural networks. *Chemical Engineering Journal*, 145(2):290–307, 2008.

[50] Gabriele Lillacci and Mustafa Khammash. Parameter estimation and model selection in computational biology. *PLoS computational biology*, 6(3):e1000696, 2010.

[51] Jinkun Liu. *Radial Basis Function (RBF) neural network control for mechanical systems: design, analysis and Matlab simulation*. Springer Science & Business Media, 2013.

[52] Lennart Ljung. System identification toolbox. *The Matlab user's guide*, 1988.

[53] Vinícius Gonçalves Maltarollo, Káthia Maria Honório, and AB Ferreira da Silva. Applications of artificial neural networks in chemical problems. *Artificial neural networks-architectures and applications*, pages 203–223, 2013.

[54] Kenneth G Mann, Richard J Jenny, and Sriram Krishnaswamy. Cofactor proteins in the assembly and expression of blood clotting enzyme complexes. *Annual review of biochemistry*, 57(1):915–956, 1988.

[55] Jiayi Mao, Lu Chen, Zhengwei Cai, Shutong Qian, Zhimo Liu, Binfan Zhao, Yuguang Zhang, Xiaoming Sun, and Wenguo Cui. Advanced biomaterials for regulating polarization of macrophages in wound healing. *Advanced Functional Materials*, 32(12):2111003, 2022.

[56] Djikolngar Maouyo, Shaoyou Chu, and Marshall H Montrose. ph heterogeneity at intracellular and extracellular plasma membrane sites in ht29-c1 cell monolayers. *American Journal of Physiology-Cell Physiology*, 278(5):C973–C981, 2000.

[57] Giovanny Marquez, Bethany Johnson, Mohammad Jafari, and Marcella Gomez. Online machine learning based predictor for biological systems. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 120–125. IEEE, 2019.

[58] Sarah E McDowell, Jamie J Coleman, and RE Ferner. Systematic review and meta-analysis of ethnic differences in risks of adverse reactions to drugs used in cardiovascular medicine. *Bmj*, 332(7551):1177–1181, 2006.

[59] Francisco Merino-Casallo, Maria Jose Gomez-Benito, Silvia Hervas-Raluy, and Jose Manuel Garcia-Aznar. Unravelling cell migration: defining movement from the cell surface. *Cell Adhesion & Migration*, 16(1):25–64, 2022.

[60] Jorge J Moré and Danny C Sorensen. Newton's method. Technical report, Argonne National Lab., IL (USA), 1982.

[61] Pooria Mostafalu, Ali Tamayol, Rahim Rahimi, Manuel Ochoa, Akbar Khalilpour, Gita Kiaee, Iman K Yazdi, Sara Bagherifard, Mehmet R Dokmeci, Babak Ziaie, et al. Smart bandage for monitoring and treatment of chronic wounds. *Small*, 14(33):1703509, 2018.

[62] Kumpati S Narendra and Kannan Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on neural networks*, 1(1):4–27, 1990.

[63] Chuong Minh Nguyen, Danielle Marie Tartar, Michelle Dawn Bagood, Michelle So, Alan Vu Nguyen, Anthony Gallegos, Daniel Fregoso, Jorge Serrano, Duc Nguyen, Doniz Degovics, et al. Topical fluoxetine as a novel therapeutic that improves wound healing in diabetic mice. *Diabetes*, 68(7):1499–1507, 2019.

[64] Katsuhiko Ogata. *Modern control engineering fifth edition*. 2010.

[65] Evan J Olson, Lucas A Hartsough, Brian P Landry, Raghav Shroff, and Jeffrey J Tabor. Characterizing bacterial gene circuit dynamics with optically programmed gene expression signals. *Nature methods*, 11(4):449, 2014.

[66] Evan J Olson, Constantine N Tzouanas, and Jeffrey J Tabor. A photoconversion model for full spectral programming and multiplexing of optogenetic systems. *Molecular systems biology*, 13(4):926, 2017.

[67] Róisín Owens, Peter Kjall, Agneta Richter-Dahlfors, and Fabio Cicoira. Organic bioelectronics-novel applications in biomedicine. preface. *Biochimica et biophysica acta*, 1830(9):4283–4285, 2013.

[68] VM Panchade, RH Chile, and BM Patre. A survey on sliding mode control

strategies for induction motors. *Annual Reviews in Control*, 37(2):289–307, 2013.

[69] Pattawong Pansodtee, John Selberg, Manping Jia, Mohammad Jafari, Harika Dechiraju, Thomas Thomsen, Marcella Gomez, Marco Rolandi, and Mircea Teodorescu. The multi-channel potentiostat: Development and evaluation of a scalable mini-potentiostat array for investigating electrochemical reaction mechanisms. *Plos one*, 16(9):e0257167, 2021.

[70] Jooyoung Park and Irwin W Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

[71] Yongjin Park and Manolis Kellis. Deep learning for regulatory genomics. *Nature biotechnology*, 33(8):825, 2015.

[72] Molecular Probes. Snarf ph indicators: Product information. 2003.

[73] Christopher M Proctor, Andrea Slézia, Attila Kaszas, Antoine Ghestem, Isabel Del Agua, Anna-Maria Pappa, Christophe Bernard, Adam Williamson, and George G Malliaras. Electrophoretic drug delivery for seizure control. *Science advances*, 4(8):eaau1291, 2018.

[74] James B Rawlings. Tutorial overview of model predictive control. *IEEE control systems magazine*, 20(3):38–52, 2000.

[75] Melanie Rodrigues, Nina Kosaric, Clark A Bonham, and Geoffrey C Gurtner. Wound healing: a cellular perspective. *Physiological reviews*, 99(1):665–706, 2019.

[76] Brett Sargent, Mohammad Jafari, Giovanny Marquez, Abijeet Singh Mehta, Yao-Hui Sun, Hsin-ya Yang, Kan Zhu, Roslyn Rivkah Isseroff, Min Zhao,

and Marcella Gomez. A machine learning based model accurately predicts cellular response to electric fields in multiple cell types. *Scientific reports*, 12(1):1–13, 2022.

[77] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel. Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6):1327–1349, 2021.

[78] Maria Seitanidou, Robert Blomgran, Giggil Pushpamithran, Magnus Berggren, and Daniel T Simon. Modulating inflammation in monocytes using capillary fiber organic electronic ion pumps. *Advanced healthcare materials*, 8(19):1900813, 2019.

[79] J Selberg, M Jafari, C Bradley, M Gomez, and M Rolandi. Expanding biological control to bioelectronics with machine learning. *APL Materials*, 8(12):120904, 2020.

[80] John Selberg, Marcella Gomez, and Marco Rolandi. The potential for convergence between synthetic biology and bioelectronics. *Cell systems*, 7(3):231–244, 2018.

[81] John Selberg, Mohammad Jafari, Juanita Mathews, Manping Jia, Pattawong Pansodtee, Harika Dechiraju, Chunxiao Wu, Sergio Cordero, Alexander Flora, Nebyu Yonas, et al. Machine learning-driven bioelectronics for closed-loop control of cells. *Advanced Intelligent Systems*, 2(12):2000140, 2020.

[82] Soumya Sethi, Tomoko Emura, Kumi Hidaka, Hiroshi Sugiyama, and Masayuki Endo. Photo-controlled dna nanotubes as stiffness tunable matrices for controlling cellular behavior. *Nanoscale*, 2023.

[83] Yuri Shtessel, Christopher Edwards, Leonid Fridman, Arie Levant, et al. *Sliding mode control and observation*, volume 10. Springer, 2014.

[84] Daniel T Simon, Erik O Gabrielsson, Klas Tybrandt, and Magnus Berggren. Organic bioelectronics: bridging the signaling gap between biology and technology. *Chemical Reviews*, 116(21):13009–13041, 2016.

[85] SN Sivanandam and SN Deepa. Genetic algorithm optimization problems. In *Introduction to genetic algorithms*, pages 165–209. Springer, 2008.

[86] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*, volume 199. Prentice hall Englewood Cliffs, NJ, 1991.

[87] M Hanief Sofi, Radhika Gudi, Subha Karumuthil-Melethil, Nicolas Perez, Benjamin M Johnson, and Chenthamarakshan Vasu. ph of drinking water influences the composition of gut microbiome and type 1 diabetes incidence. *Diabetes*, 63(2):632–644, 2014.

[88] Bing Song, Yu Gu, Jin Pu, Brian Reid, Zhiqiang Zhao, and Min Zhao. Application of direct current electric fields to cells and tissues in vitro and modulation of wound electric field in vivo. *Nature protocols*, 2(6):1479–1489, 2007.

[89] Xenofon Strakosas, M Seitanidou, Klas Tybrandt, Magnus Berggren, and Daniel T Simon. An electronic proton-trapping ion pump for selective drug delivery. *Science Advances*, 7(5):eabd8738, 2021.

[90] Xenofon Strakosas, John Selberg, Xiaolin Zhang, Noah Christie, Peng-Hao Hsu, Adah Almutairi, and Marco Rolandi. A bioelectronic platform modulates ph in biologically relevant conditions. *Advanced Science*, 6(7):1800935, 2019.

[91] YH Sun, G Luxardi, G Xu, K Zhu, B Reid, BP Guo, CB Lebrilla, E Maverakis, and Min Zhao. Surface glycans regulate salmonella infection-dependent directional switch in macrophage galvanotaxis independent of nanh. *Infection and immunity*, 90(1):e00516–21, 2022.

[92] Chris Tofallis. A better measure of relative prediction accuracy for model selection and model estimation. *Journal of the Operational Research Society*, 66(8):1352–1362, 2015.

[93] Xavier Trepat, Zaozao Chen, and Ken Jacobson. Cell migration. *Comprehensive Physiology*, 2(4):2369, 2012.

[94] Ilke Uguz, Christopher M Proctor, Vincenzo F Curto, Anna-Maria Pappa, Mary J Donahue, Magali Ferro, Róisín M Owens, Dion Khodagholy, Sahika Inal, and George G Malliaras. A microfluidic ion pump for in vivo drug delivery. *Advanced Materials*, 29(27):1701217, 2017.

[95] Miguel Vicente-Manzanares and Alan Rick Horwitz. Cell migration: an overview. *Cell migration*, pages 1–24, 2011.

[96] Wilfried Weber and Martin Fussenegger. Emerging biomedical applications of synthetic biology. *Nature Reviews Genetics*, 13(1):21–35, 2012.

[97] Robert L Williams, Douglas A Lawrence, et al. *Linear state-space control systems*. John Wiley & Sons, 2007.

[98] Adam Williamson, Jonathan Rivnay, Loïg Kergoat, Amanda Jonsson, Sahika Inal, Ilke Uguz, Marc Ferro, Anton Ivanov, Theresia Arbring Sjöström, Daniel T Simon, et al. Controlling epileptiform activity with organic electronic ion pumps. *Advanced Materials*, 27(20):3138–3144, 2015.

[99] Jarek Wosik, Wei Chen, Kuang Qin, Rafik M Ghobrial, Jacek Z Kubiak, and Malgorzata Kloc. Magnetic field changes macrophage phenotype. *Biophysical Journal*, 114(8):2001–2013, 2018.

[100] Ligang Wu, Jianxing Liu, Sergio Vazquez, and Sudip K Mazumder. Sliding mode control in power converters and drives: A review. *IEEE/CAA Journal of Automatica Sinica*, 9(3):392–406, 2021.

[101] K David Young, Vadim I Utkin, and Umit Ozguner. A control engineer's guide to sliding mode control. *IEEE transactions on control systems technology*, 7(3):328–342, 1999.

[102] Tom J Zajdel, Gawoon Shim, Linus Wang, Alejandro Rossello-Martinez, and Daniel J Cohen. Scheepdog: programming electric cues to dynamically herd large-scale cell migration. *Cell systems*, 10(6):506–514, 2020.

[103] Boyang Zhang, Xiuxia Sun, Shuguang Liu, and Xiongfeng Deng. Recurrent neural network-based model predictive control for multiple unmanned quadrotor formation flight. *International Journal of Aerospace Engineering*, 2019, 2019.

[104] Min Zhao. Electrical fields in wound healing—an overriding signal that directs cell migration. In *Seminars in cell & developmental biology*, volume 20, pages 674–682. Elsevier, 2009.

[105] Ksenia Zlobina, Mohammad Jafari, Marco Rolandi, and Marcella Gomez. The role of machine learning in advancing precision medicine with feedback control. *Cell Reports Physical Science*, page 101149, 2022.