From Data to Predictive Models: Robust Identification and Analysis of the Immune System

by

Brendon K. Colbert

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved September 2021 by the
Graduate Supervisory Committee:

Matthew Peet, Chair
Abhinav Acharya
Spring Berman
Luis Crespo
Sze Zheng Yong

ARIZONA STATE UNIVERSITY

December 2021

ABSTRACT

In this dissertation, we develop new data-driven techniques to solve three problems related to generating predictive models of the immune system. These problems and their solutions are summarized as follows.

The first problem is that, while cellular characteristics can be measured using flow cytometry, immune system cells are often analyzed only after they are sorted into groups by those characteristics. Instead of grouping cells we propose analyzing the cellular characteristics by generating Probability Density Functions (PDFs) to model the flow cytometry data. To generate a PDF to model the distribution of immune cell characteristics we develop a new class of random variable called Sliced-Distributions (SDs) in Chapter 3 and show that SDs outperform other state-of-the-art methods and can differentiate between immune cells from healthy mice and those with Rheumatoid Arthritis.

The second problem is that while immune system cells can be broken into different subpopulations, it is unclear which subpopulations are most significant. We, therefore, formulate a new machine learning algorithm in Chapter 4 to identify subpopulations that can best predict disease severity or the populations of other immune cells. The proposed machine learning algorithm performs well when compared to other state-of-the-art methods and is applied to an immunological dataset to identify disease-relevant subpopulations of immune cells denoted immune states.

Finally, while immunotherapies have been effectively used to treat cancer, selecting an optimal drug dose and period of treatment administration is still an open problem. In Chapter 5 we propose a method to estimate Lyapunov functions of a system with unknown dynamics. We apply this method to generate a semialgebraic set containing immunotherapy doses and period of treatment that leads to tumor elimination. The problem of selecting an optimal pulsed immunotherapy treatment

from this semialgebraic set is formulated as a Global Polynomial Optimization (GPO) problem. In Chapter 6 a new method to solve GPO problems is proposed and optimal pulsed immunotherapy treatments are identified for this system.

DEDICATION

Dedicated to my fiancée, my family, and my friends.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

The human immune system consists of over 25 billion individual white blood cells alone [117, 150] that, in coordination, act to protect the body against threats such as viruses and cancer. The adaptive immune system has been optimized over 500 million years [56] of evolutionary pressure from pathogens to eliminate and recall threats, all without any centralized control. Despite recorded interest in the immune system as early as the 5th century BC [45, 146], we still do not completely understand the complex relationships between the millions of cells that control such a vital system to human life. Without understanding how the immune system functions, one can not fully understand why it fails nor how it can be fixed. We are thus motivated by one crucial question: How do we generate better models of the immune system?

Immune system cells interact through surface receptors with other immune system cells and molecular signals to collectively respond to threats such as viruses or cancer cells. Unfortunately, determining all the characteristics (such as cell surface receptors or intracellular proteins) of the immune cells in the body is, as of yet, impossible. Fortunately, using techniques such as flow cytometry [94, 142], a sample of an individuals immune system cells can be analyzed, and a limited number of cellular characteristics (e.g. size, surface molecules, etc.) of the immune system cells can be determined. An important question then, is what immune cell characteristics need to be measured with flow cytometry to predict whether the immune systems will, for example, eliminate a virus or cancer cells.

This dissertation therefore answers the question of how to generate better predictive models of the immune system by first developing techniques to determine

what immune cell characteristics (using techniques such as flow cytometry) need to be measured so that we can accurately predict labels such as whether an individual is healthy or has a given disease. Next we focus on generating better data-driven models that use these cell characteristics to predict more complicated labels such as disease severity. Finally we generate a model to predict whether a given dosage and treatment period of an immunotherapy will eliminate tumor cells in a patient.

Potential Impact of Improved Predictive Models of the Immune System

1 **Immune Health Metrics:** The normal range of the populations of simple immune cells categories, such as the population of all white blood cells have been rigorously studied (see for instance [117]). However, being inside a normal range does not indicate that a persons immune system is operating normally, especially since even among healthy individuals there is a large diversity in the populations of immune system cells [18]. Thus better metrics to analyze diverse populations of immune system cells are necessary to determine immune health with greater precision.

2 **Disease Metrics:** Determining the immune system cells that best explain the severity of a disease is important for determining optimal treatment strategies since treatments could, for instance, be designed to target the immune system cells causing the disease. In addition, tracking a disease metric throughout treatment can be used to determine if the current treatment is modifying the immune system as expected or if a new approach may be necessary.

3 **Personalized Immunotherapy Treatments:** Successful immunotherapy treatments depends on the immune cells and, for cancer specifically, the local tumor microenviroment [79]. Therefore, for optimal performance, the immunotherapy drugs, dosage, and dosing schedule should be selected based on the populations

and characteristics of the patients own immune cells.

To motivate the methods developed in this dissertation we will first describe what is known about the immune system and highlight challenges which make it difficult to model.

1.1 The Immune System

While the immune system has, clearly, existed for the entirety of human history the first known record of the concept of immunity is attributed to Thucydides in the 5th century BC [45, 146]. Thucydides noted that survivors of the plague of Athens in 430 BC were unable to contract the illness again and were thus "immune". The first attempt by humans to intentionally modulate this theorized process of immunity occurred at least as early as 1000 AD in China where powdered smallpox lesions were inhaled to induce immunity to smallpox.

Immunity to threats is perhaps one of the most important benefits of the immune system since, as witnessed by Thucydides, it is often the best defense against pathogens. With the advent of immunotherapies and vaccines immune system mechanisms can be exploited to, for instance, confer protection to pathogens without first contracting the illness as is required for "natural" immunity. Unfortunately, the immune response is not infallible nor are its mechanisms completely understood. For instance, autoimmunity such as arthritis or type 1 diabetes are examples of when the system mistakenly targets the body itself. While the symptoms of these diseases can be treated, they can rarely be resolved [21, 49].

Thus, to understand why the immune system fails, and how these failures can be treated, much interest in the immune system is firmly rooted in the process of self versus non-self determination, wherein a potential threat is targeted for elimination if it is identified as non-self or protected if it is identified as self. In the following

sections we describe key immune system components that are related to self versus non-self determination and propose models which describe the interactions between the immune system cells as they collectively identify and eliminate threats.

1.2 Self Versus Non-Self Determination

Models of self versus nonself determination can describe, for example, how the immune system fails to recognize a threat such as cancer, or incorrectly recognizes healthy cells as a threat in autoimmunity. To describe how self versus nonself determination works, we first describe how common threats to the immune system are identified by the innate immune system. Next we illustrate, through observations, that the innate immune system alone cannot properly eliminate pathogenic threats and that the more complex adaptive immune system is necessary for self versus non-self determination. Finally we end by illustrating how the immune system cells are proposed to interact to between immune system cells using models proposed in the literature.

1.2.1 The Innate Immune System

Many pathogenic threats share common molecular patterns of infection that are not found in healthy human cells. These common patterns are exploited by the innate immune system, whose role is to eliminate these "obvious" threats. The innate system therefore has a simplistic method for self versus non-self determination that is limited to a pre-determined set of common patterns.

Characteristics of pathogenic threats are called Pathogen-Associated Molecular Patterns (PAMPs). Innate immune cells can recognize PAMPs that are associated with bacteria, fungi and viruses but not human cells [112]. Upon recognizing a PAMP innate immune system cells release signals to alert other immune system cells of a

potential threat and can engulf and process nearby molecules to present to other immune system cells. We define these signals and PAMPs as follows.

Signals and Molecules:

Cytokines: Cytokines are signaling molecules released and recognized by immune system cells. These signals are used to modulate the immune response by, for example, attracting immune system cells or promoting inflammation to increase the magnitude of the immune response.

PAMPs: PAMPs are pathogen-associated molecular patterns that are commonly found on a wide variety of pathogens. When innate immune system cells recognize PAMPs they, for instance, release cytokine signals or eliminate pathogens.

Antigens: Antigens are molecules such as proteins, peptides and polysaccharides. For example antigens can be part of a pathogen, and are thus specific to that pathogen, as opposed to PAMPs which are related to a wide variety of pathogens. While the innate immune system does not recognize antigens, they can present antigens to other immune system cells (such as adaptive immune system cells presented in the next subsection).

The role of the innate immune system then is primarily to recognize PAMPs, release cytokines to alert other immune system cells of potential threats and present antigens to other immune system cells. Some examples of innate immune system cells include:

Innate Immune System Cells

Macrophages: Macrophages are a type of immune system cell which can engulf and digest foreign substances. They are known to recognize PAMPs associated with gram-negative and gram-positive bacteria using Toll-Like Receptors (TLRs) [112].

Natural Killer (NK) Cells: Natural killer cells are a type of immune system cell which also engulf and digest foreign substances. They are known to recognize PAMPs associated with viruses, specifically single stranded and double stranded RNA [112].
Dendritic Cells: Dendritic cells can also engulf and digest foreign substances, but are primarily known as Antigen-Presenting Cells (APCs) because they will process and display antigen to other immune system cells via MHC I or MHC II molecules. They are known to recognize PAMPs associated with viruses and bacteria [112].

The response to PAMPs is nearly immediate because of the large numbers of circulating innate immune cells that can respond. Unfortunately, innate immune system cells are dependent on a limited number of receptors and do not recognize specific pathogens that have been encountered previously. Therefore the innate immune system does not have a so called "memory" that could grant immunity to reinfection that was observed by, for instance, Thucydides. In addition, when adaptive immune system cells are defective as in DiGeorge's, Wiskott-Aldrich or Job's syndrome, the individual is dependent on the innate immune system and is highly susceptible to pathogens such as viruses and bacteria [112]. The adaptive response we describe in the next subsection is thus crucial to eliminating pathogens that would otherwise threaten the entire system.

1.2.2 The Adaptive Immune System:

The adaptive immune system itself consists of hundreds of millions of cells all specific for different antigens - and thus pathogens associated to those antigens. When an adaptive immune cell recognizes an antigen it can mount a targeted response directly against the pathogen associated to the antigen and the cumulative response of all of the adaptive immune cells composes the complete immune response. In addition, adaptive immune system cells can differentiate into memory cells that live

longer to confer longer lasting immunity to the individual. These adaptive immune system cells that are antigen specific are further defined as follows.

Adaptive Immune System Cells

B Cells: B cells are a type of immune system cell which produce antibodies that are specific for a particular antigen. Antibodies are molecules that attach to antigen on pathogens marking the pathogen for elimination via phagocytosis by the innate immune system. B cells can also secrete cytokines to modulate the immune response.

Cytotoxic T Cells: Cytotoxic T cells eliminate infected cells. These cells are particularly important for eliminating viruses since viruses are not susceptible to antibodies while inside infected cells.

Helper T Cells: Helper T cells secrete cytokines to assist the immune response. The cytokines these helper T cells release include those which aid B cells and antibody production.

Regulatory T Cells: Regulatory T cells release cytokines to shut down the immune response once the threat has been eliminated. Regulatory T cells can also suppress T cells which are specific for self antigens to avoid autoimmunity.

Some innate immune system cells, such as macrophages and dendritic cells can present antigen to T and B cells and are thus involved in the adaptive immune response, though are not specifically adaptive cells. Likewise antibodies released by the adaptive response enhance the phagocytosis of pathogens by macrophages, illustrating how the two systems are interconnected.

An important question remains, however, as to how a T or B cell could be designed to specifically target an antigen that the immune system has never encountered before. The answer to this question lies in a process called somatic recombination.

Somatic Recombination in T Cells

The adaptive immune system generates adaptive cells specific for an unseen antigen by generating cells with random antigen receptors. If enough random receptors are generated then the chances that at least one adaptive cell is specific for any given antigen can be relatively high. Of course randomly generating antigen receptors could lead to dangerous autoimmunity, for instance, if a receptor specific for a self antigen is generated. The maturation process of adaptive cells is designed to filter out ineffective or self-reactive cells. For brevity we will focus on the process of somatic recombination in T cells, but note that somatic recombination also occurs in B cells.

T cell precursors travel from the bone marrow to the thymus where they proliferate to generate nearly $5 \cdot 10^7$ total cells every day. However only $2 - 4\%$ of these cells become mature T cells by passing two filtering processes and rearranging segments of their DNA to generate new antigen receptors, called T Cell Receptors (TCRs). We focus on so called "conventional" T cells whose TCRs have an α and β gene and which additionally have either the TCR co-receptor CD4 or CD8. However, immature T cells can have neither co-receptor or even both co-receptors depending on their stage of development.

For instance T cells first enter a double-negative stage and have neither CD4 or CD8 co-receptors. In the double-negative stage the β gene of the TCR is rearranged first to begin the random TCR generation process. If a β gene is rearranged and if the coding sequence is without errors then the cells are said to have passed β selection and can develop into double-positive cells.

In the double-positive stage developing T cells have both CD4 and CD8 co-receptors and the α gene is rearranged to complete the somatic recombination of the TCR. Once both the α and β genes have undergone somatic recombination the

TCR has been randomly combined and is specific for an antigen that the immune system may have never encountered before. To ensure that the TCR functions properly and is not self-reactive there are two selection stages to filter out T cells with poor TCRs.

The first filter for developing T cells is called positive selection. In positive selection thymic cells present self antigen on Major Histocompatibility Complex (MHC) molecules to developing T cells. In this stage if the TCR is functioning properly the developing T cells should have a small response to the MHC molecule. Developing T cells that either have no response, or too large of a response to the MHC and self antigen presented by the thymic cells will not pass positive selection. After positive selection the number of TCR will increase on the T cells as they finish developing.

The second filter for developing T cells is called negative selection. Since the developing T cells have more TCR on the cell surface after passing positive selection, T cells that are specific for self antigens will have larger responses in this stage. If the developing T cells have a large response to the thymic self antigen presenting cells, then the developing cell is either killed or becomes a regulatory T cell if it expresses the CD4 co-receptor [112]. After passing both positive and negative selection the T cells are fully developed and roam the body searching for antigen specific to their TCR.

1.2.3 A Circuit Model of Self Non-self Determination

While it is not completely clear exactly how self versus nonself determination works, it seems clear that helper and regulatory T cells modulate the immune response through the release of cytokines and cell-cell interactions. We consider the regulatory and helper T cell populations to be most important for the decision of self versus nonself, as they can control the cytotoxic T and B cell populations through the

release of cytokines and cellular interactions.

Furthermore, there is evidence that the dynamics of the antigen in the system also has a large effect on the adaptive immune response. Most notably, in [80, 158], it was shown that the adaptive immune response is stronger when antigen is introduced at an exponentially increasing rate. Since pathogens such as bacteria and viruses grow exponentially the ability to mount stronger responses against rapidly growing threats is clearly advantageous.

In [126] it was shown how the widely recognized delay in regulatory T cell activation (see [75, 139]), could approximate a derivative response to antigen signaling. The cells that compose the immune system are thus proposed to act as a large decentralized control system that mimic a Proportional-Integral-Derivative (PID) controller. To describe the proposed model we will next analyze the proportional, derivative, and integral response of the PID type response.

The Proportional Response: To adequately eliminate threats the immune system must respond to the antigen signal in the system. Suppose that for any time t the function $a(t)$ describes the amount of antigen in the system, $E(t)$ is the population of helper T cells and $T_c(t)$ is the population of cytotoxic T cell. In [126] the population of helper T cells grows when unactivated T helper cells recognize an antigen, producing a proportional response to the antigen and the population decays as the cells die. Likewise the helper T cells recruit unactivated cytotoxic T cells that are specific for the antigen so that the threat posed by the antigen can be eliminated. The proportional response of these populations can be described as follows.

$$\dot{E}(t) = r_{Ea}a(t)N_{eq} - d_E E(t), \qquad \dot{T}_c(t) = r_{Ec}E(t)N_c - d_{T_c}T_c(t), \tag{1.1}$$

where $d_E,\ d_{T_c} > 0$ are death rates, N_{eq} and N_c are populations of unactivated helper and cytotoxic T cells, and $r_{Ea},\ r_{Ec} > 0$ are recruitment rates. In Fig. 1.1 we illustrate

the positive proportional response of the helper T cells to antigen stimulation.

The proportional response does grow in response to an antigenic threat to protect the individual. Unfortunately the proportional response alone does not explain the observed decision making process of the immune system. For instance self antigen, which is always present in the system, should cause a consistent level of inflammation and a proportional response alone would cause the immune system to be auto-reactive.

The Derivative Response: The derivative response of the circuit model explains how the immune system can differentiate between a self antigen and a non-self antigen. The proposed mechanism of self versus non-self differentiation is based upon populations of regulatory T cells $R(t)$ which reduce immune responses through direct contact with helper T cells, or the release of cytokines. The antigen signal released by healthy cells will be roughly constant and the derivative of this signal will be relatively small. The antigen signal released by a threat, however, will begin small and increase as the threat multiplies. In the case of a threat the derivative of the antigen signal will be large.

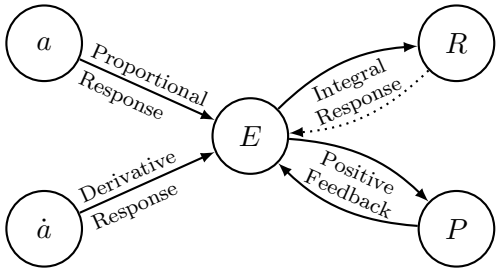


Figure 1.1: A diagram of the immune system dynamics from [126] for determining self versus non-self. In this model the derivative response with respect to the antigen is physically approximated in the immune system by the relative delay in maturation of regulatory immune system cells when compared to helper T cells.

The proposed mechanism of self versus non-self determination is based on a derivative response to the antigen signal. In [126], regulatory cells are modeled as being a delayed response due to slower maturation rates of regulatory T cells, but that also have a proportional response to antigen so that the population of regulatory T cells

is defined as follows.

$$R(t) = \frac{r_R}{d_R} a(t-\tau), \tag{1.2}$$

where r_R is the recruitment rate of the regulatory T cells and d_R is the death rate.

It is assumed that the population of regulatory T cells acts on the population of helper T cells ar a rate of r_{RE}, and the combined proportional and derivative response of the helper T cells can thus be defined as,

$$\dot{E}(t) = r_{Ea} a(t) E(t) - r_{RE} \frac{r_R}{d_R} a(t-\tau) E(t), \tag{1.3}$$

$$= (r_{Ea} - K_{RE}) a(t) E(t) + \tau K_{RE} \frac{a(t) - a(t-\tau)}{\tau} E(t) \tag{1.4}$$

where $K_{RE} = r_{RE} \frac{r_R}{d_R}$ and $\frac{a(t)-a(t-\tau)}{\tau}$ approximates the derivative of the antigen signal.

In Fig. 1.1 we illustrate the positive derivative response of the helper T cells to antigen stimulation. If $r_{Ea} \approxeq K_{RE}$ then the response is primarily dependent on the derivative of the antigen signal, thus implying that regulatory T cells can suppress immune responses to self antigens (which should be relatively constant), but can still recognize threats which produce an increasing antigen signal. In that case, the proportional response illustrated in Fig. 1.1 is nonexistent. Unfortunately a derivative response alone is not always capable of eliminating threats completely and may lead to persistent infection.

To ensure infections are eliminated, a cytokine based switching mechanism is proposed. IL-2 is a cytokine known to be important for the proliferation of immune system cells and is released by helper T cells and the number of IL-2 molecules, p, can be modeled as follows.

$$\dot{p}(t) = r_p E(t) - d_p p(t)$$

where $p(t) = \frac{r_p E(t)}{d_p}$ is the quasi-steady state approximation of the amount of IL-2 in the system.

Since IL-2 promotes proliferation of helper T cells, and is released by helper T cells, these dynamics generate a positive feedback loop. This positive feedback loop can be modeled as follows when using the quasi-steady state approximation of the amount of IL-2 in the system.

$$\begin{aligned}\dot{E}(t) &= -d_E E(t) + r_E p(t) E(t) + u(t) \\ &= -d_E E(t) + r_E \frac{r_p}{d_p} E(t)^2 + u(t)\end{aligned}$$

where d_E is the death rate, r_E is the rate of positive feedback of released IL-2 on the helper T cell population and $u(t)$ represents the dynamics of the derivative response.

In Fig. 1.1 we illustrate the positive feedback between the IL-2 population, p and the helper T cell population. If the derivative response of the system is relatively small, then the system is stable as the positive feedback can't overcome the natural decay rate of the helper T cells. However, for a large enough derivative response the system becomes unstable and the helper T cell population experiences uncontrolled exponential growth. Thus, the positive feedback loop is only "switched" on when a threat with a large enough derivative response is recognized. Large enough derivative responses triggers exponential growth so that threats can be eliminated by the immune system.

Unfortunately uncontrolled exponential growth is unrealistic. To control this growth an integral response was proposed in [126] to stabilize the system.

The Integral Response: An integral response is proposed to describe how the immune system can stop the uncontrolled exponential growth of the helper T cell population once a threat has been eliminated.

A subpopulation of regulatory T cells called iT_{reg} cells are believed to be generated by the helper T cell population. Thus the population of iT_{reg} cells, $R_i(t)$, is modeled

as,

$$\dot{R}_i(t) = -d_{R_i} R_i(t) + \nu_R \frac{r_p}{d_p} E(t)^2 \tag{1.5}$$

where d_{R_i} is the death rate and $\nu_R \frac{r_p}{d_p} E(t)^2$ models the effect of the helper T cell positive feedback loop on the population of iT_{reg} cells. The iT_{reg} cells have a relatively low death rate when compared to the other cell populations. Thus the population of iT_{reg} cells is approximately, $R_i(t) \approxeq \int_0^t \frac{\nu_R r_p}{d_p} E(s)^2 ds$.

Combining the proposed derivative and integral responses (and assuming $r_{Ea} = K_{RE}$) generates the helper T cell dynamics,

$$\dot{E}(t) = -d_E E(t) + r_E \frac{r_p}{d_p} E(t)^2 + \tau K_{RE} \frac{a(t) - a(t-\tau)}{\tau} E(t) - r_{R_i E} \int_0^t \frac{\nu_R r_p}{d_p} E(s)^2 ds, \tag{1.6}$$

where $r_{R_i E}$ is the rate at which the iT_{reg} cells inhibits the helper T cell population.

In Fig. 1.1 we illustrate the negative integral feedback from the iT_{reg} cell population on the helper T cell population. Considering all of the dynamics, the proposed model has an approximate derivative response to antigen that, if large enough, acts as a trigger for uncontrolled growth of the helper T cells via a cytokine feedback loop. This feedback loop is controlled by an integral like response from the population of iT_{reg} cells to contract the population of helper T cells once the antigen threat has been eliminated. The PID like dynamics describes how millions of immune system cells can unknowingly collaborate to determine if a threat is self or non-self and mount a controlled response to protect the individual from threats.

The problem of self versus non-self recognition is important for understanding how the immune system identifies threats and for identifying the mechanisms that cause immune driven diseases such as rheumatoid arthritis. However, we are also interested in treating diseases after the self versus nonself determination has already

been decided, but, after the immune system has been unable to eliminate the threat. To illustrate how immunotherapies can be used to help the immune system eliminate a threat, we next consider a model of the dynamics between cancer cells, the immune system, and an immunotherapy.

1.3 Antigen Recognition and Elimination

In some cases, even if a threat has been identified as nonself, poor immune responses can fail to eliminate the threat. For instance, cancers can release cytokines which inhibit the immune response allowing uncontrolled tumor growth. In these cases modeling the relationship between the threat, the immune system, and potential immunotherapies can aid in the selection of an immunotherapy, dosage, or period of treatment to ensure the immune system eliminates the threat.

Antigen in the body is captured by antigen-presenting cells such as macrophages and dendritic cells and presented on MHC I and MHC II molecules to activate adaptive immune system cells, which then proliferate extensively to eliminate the threat. Activated cytotoxic T cells can directly eliminate infected cells that present their TCR specific antigen, while B cells generate antibodies that mark the antigen for destruction by, for instance, increasing the effectiveness of macrophage phagocytosis. Helper T cells and regulatory T cells release cytokines which can increase (helper) or decrease (regulatory) the number and effectiveness of cytotoxic T cells and B cells. Thus, in many models the helper and regulatory T cells are the most important for modeling the magnitude of the immune response.

To illustrate how these dynamics can be modeled we consider the model proposed in [170] to describe the dynamics between tumor cells, a cytokine, and cytotoxic, helper and regulatory T cells.

1.3.1 An Immunological Cancer Model

The immune system usually generates immune responses to eliminate cancer cells. However, in this subsection we will define a proposed model which describes how tumor cells can release a cytokine to inhibit the immune response enabling the tumor cells to grow uncontrollably.

First denote the tumor size as T, the cytokine TGF-β as B, cytotoxic T cells as E, regulatory T cells as R and activated tumor-specific cytotoxic T-cells administered with a vaccine as V. The dynamics of each of these states is defined as follows.

$$\begin{aligned}
\dot{T} &= a_0 T(1 - c_0 T) - \delta_0 \frac{ET}{1 + c_1 B} - \delta_o TV, & \dot{B} &= a_1 \frac{T^2}{c_2 + T^2} - dB, \\
\dot{E} &= \frac{fET}{1 + c_3 TB} - rE - \delta_0 RE - \delta_1 E, & \dot{R} &= rE - \delta_1 R, \\
\dot{V} &= g(t) - \delta_1 V,
\end{aligned} \tag{1.7}$$

where the positive constants a_0, c_0, δ_0, c_1, a_1, d, c_3, f, and r define different rates at which the cancer cells and immune system cells decay or proliferate. The values of these constants were identified from other papers or were selected such that the model trajectories best captured experimental data. In Fig. 1.2 we illustrate graphically the relationship between the tumor, immune system cells and the cytokine signal, where solid lines imply a positive relationship and dashed lines imply a negative relationship between the state variables in the direction of the arrow.

We will discuss the dynamics of each of these states in more detail as follows.

Tumor Dynamics: This model assumes that the tumor follows a logistic population growth, as defined by the term $a_0 T(1 - c_0 T)$. In this case the tumor cells initially increase exponentially at a rate close to a_0 but the rate decreases as the cell population approaches the carrying capacity, $\frac{1}{c_0}$, of the tumor. Elimination of the tumor cells occurs at a rate proportional to the population of cytotoxic T and

tumor cells and inversely proportional to the levels of TGF-β, and is given by $\frac{\delta_0 ET}{1+c_1 B}$. The motivation behind this term comes from papers such as [160], which have shown that TGF-β decreases the rate at which cytotoxic T cells can eliminate tumor cells. However, since cytotoxic T cells injected via vaccination are activated outside of the patient, it is assumed that they are fully differentiated and not affected by the levels of TGF-β. Thus the injected cytotoxic T cells, V, eliminate tumor cells at a rate of $\delta_o TV$ and are independent of the levels of TGF-β.

Cytokine Dynamics: Experimental evidence in [121], suggests that the rate at which TGF-β is produced is low for small populations of tumor cells, but rapidly increases as the tumor population increases. The dynamics of the level of TGF-β is described in [5] where it increases at a rate of $a_1 \frac{T^2}{c_2+T^2}$ and naturally decays at a rate of d.

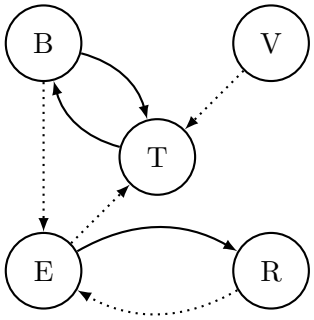


Figure 1.2: A diagram of the immune system dynamics described in Eq. (1.7).

Cytotoxic T Cell Dynamics: For the cytotoxic T cell dynamics it is assumed that the amount of antigen in the system is proportional to the volume of tumor cells. In this model the rate of cytotoxic T cell activation is directly proportional to the number of interactions between the T cells and tumor cells and is proportional to ET. To account for the negative effect of TGF-β on the activation and recruitment of cytotoxic T cells, the rate of increase of E is given by $\frac{fET}{1+c_3 TB}$.

Regulatory T cells act to shut down the immune response and, in this model, the effect on the population of cytotoxic T cells is given by the term $-\delta_0 RE$. Furthermore,

this model assumes that a proportion, r of cytotoxic T cells will be converted into regulatory T cells as also seen in models in [91]. Finally the cytotoxic T cells will naturally decay at a rate of $-\delta_1$.

Regulatory T Cell Dynamics: In this model the regulatory T cells are assumed to be converted from cytotoxic T cells. Although regulatory T cells can originate from helper T cells as well, this simplification is used to generate a model of minimal complexity. Therefore the regulatory T cells increase at a rate proportional to the cytotoxic T cell population, rE, and decay at a rate of $\delta_1 R$.

Administered Cytotoxic T Cells: The administered cytotoxic T cells are assumed to be fully differentiated and thus no longer dividing, so the population increases only when an injection of cells is given according to the function $g(t)$. An injection of activated T cells is a type of immunotherapy called adoptive T cell transfer. The injected T cells are usually taken from a patients own blood, and then grown in large numbers outside of the patient before reinjection. This treatment has been used to treat metastatic melanoma as described in [136].

These dynamics illustrate a simplified model of the complex interaction between immune system cells and a threat (in this case tumor cells).

While this dissertation proposes data-driven techniques that do not require researchers to generate dynamical models ()such as those described in this section) the models help illustrate how the immune system cells in the previous section are proposed to interact. Furthermore, in Chapter 5 and 6, we use the immunotherapy model to simulate trajectory data that could be measured in a real experiment, and use this simulated data with a proposed data-driven technique.

1.4 Summary of Contributions and Organization

Predictive models can be broken into two categories - physics based or data-driven. We have just illustrated examples of the former category, where fundamental physical laws or observed relationships are used to generate models between proposed explanatory variables and outputs of interest. However, when fundamental physical laws are unknown or too complex to derive, for instance as often occurs when modeling biological systems, data-driven models represent a tractable alternative.

Data driven decision making has already had a positive impact on quality of life, for instance its application within the healthcare industry has decreased re-admission and mortality rates and can provide patient specific treatments that are more effective [76, 103]. However the data-driven approach, when applied to the identification and analysis of the immune system, comes with its own set of problems. To generate predictive models of the immune system we have identified three problems whose solutions will enable better modeling of complex systems.

The first two problems are identified in Subsection 1.4.1 and Subsection 1.4.2 and are solved using methods developed in Chapters 3 and 4 respectively. The final problem, defined in Subsection 1.4.3 is solved using methods developed in Chapters 5 and 6, and thus it is recommended that these two chapters are read in order. In addition, we provide relevant background information in Chapter 2 which may be helpful if readers are unfamiliar with the topic at hand. A graphical illustration of the recommended reading order is included in Fig. 1.3.

1.4.1 Problem 1: Generating Models of the Distribution of Measured Data

By performing flow cytometry on a sample of immune system cells it is possible to measure characteristics (e.g. size, cell surface receptors) that determine how a cell

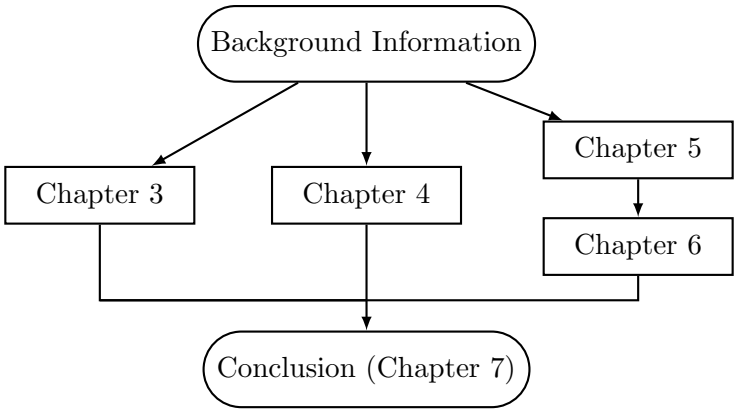


Figure 1.3: Recommended reading order of the material in this dissertation - All chapters but Chapters 5 and 6 are independent of each other.

will interact with antigen, other immune system cells or cytokine signals. To analyze the populations of immune cells they are usually sorted into different populations in a process called gating. Unfortunately, a large variation in gating has been recorded between researchers [102] implying that analyzing flow cytometry data is a subjective process.

We propose treating immune cell populations as random variables and modeling the distribution of the cellular characteristics as opposed to gating the flow cytometry data. The distribution of the cellular characteristics can then be compared between flow cytometry data sets without the need for gating.

Unfortunately, the distribution of the cellular characteristics of immune system cells do not fall into any known distribution and are therefore difficult to model. The first identified problem then is, given multivariate data from some unknown Data Generating Mechanism (DGM), to identify methods to model the complex distribution of that data.

To model complex distributions of data, in Chapter 3 we propose a new class of Probability Density Functions (PDFs) and formulate convex optimization problems to

fit the parameters of the PDF to model the distribution of the cellular characteristics. We show that the class of PDFs is dense in the set of all continuous PDFs. This means that a PDF from this class exists that can perfectly model a random variable if its corresponding PDF is continuous. This implies that we do not need to make strict, and often invalid, assumptions that the data is normally distributed [6, 12].

To demonstrate the effectiveness of this approach we compare the proposed method to other state of the art approaches to generate a model of the PDF of four publicly available data sets. By withholding a testing partition of data, we use a standard set of metrics to show that the proposed method outperforms the other state of the art methods.

We also illustrate, using a mass cytometry dataset, how to generate models of the PDF of the cellular characteristics of immune cells taken from populations of healthy patients, and those taken from populations of patients with Rheumatoid Arthritis. Despite differences in the PDFs among patients within the same groups, these models are able to identify "immune features" that are consistently shared among the patients in each group. These immune features therefore capture components of the immune system that are assumed to be relevant to the disease.

Finally we note that this portion of the research is based on the work in [32, 33].

1.4.2 Problem 2: Generating Optimal Machine Learning Algorithms

Next, there exists hundreds of different immune cells and signals which could potentially be classified as either helper [165, 105] or regulatory [138, 90]. Generating a model which uses every potential subset of immune system cells would require a large amount of difficult to acquire data to determine how these different immune cells interact. Given populations of immune system cells (as opposed to cytometry data as in the previous subsection), the second problem is to identify a subset of those

populations which capture the essential features of the immune system.

Given inputs $\{x_i\}_{i=1}^m$ (e.g. populations of immune cells) and outputs $\{y_i\}_{i=1}^m$ (e.g. disease severity), predictive models such as neural networks and kernel learning algorithms generate a data-driven model, f, to predict the relationship between the inputs and outputs. While neural networks and kernel learning algorithms have advanced significantly in recent years, these machine learning algorithms have fundamental flaws. For instance, while the activation functions used in neural networks have shown an ability to represent complex patterns in data, there is no convex formulation of the problem of training such a network - implying that trained neural networks may be locally optimal but not globally optimal. In fact it was shown in [137] that even the simplest two-layer ReLU neural networks have many local minimum and thus neural network algorithms may return sub-optimal solutions. Whereas when a minimum is found for a convex optimization problem it is guaranteed that that solution is globally optimal.

Kernel methods also have state of the art performance on a number of benchmarks. However, kernel methods require the selection of a kernel function, a problem that is highly dependent on the data itself. To minimize the "human element" of selecting a kernel function, kernel learning algorithms use a data-driven approach to select an optimal kernel function. While these kernel learning algorithms generally have convex formulations, they unfortunately learn only simple combinations of kernel functions that must be selected a priori, therefore not completely removing the "human element" of the algorithms performance. To judge the quality of a set of kernel functions three nonempirical criteria were defined that simple combinations of kernel functions are, unfortunately, unable to meet.

In Chapter 4 we develop the first machine learning algorithm which, to our knowledge, meets all three of the proposed criteria for sets of kernel functions. Our proposed

method (see [29, 30]) improves the accuracy of kernel based machine learning algorithms such as the support vector machine by automatically selecting an optimal kernel function (with respect to the data) from the class of Tessellated Kernel (TK) functions we have developed. We show that support vector machines with optimal TK kernels on average have improved accuracy when compared to other state of the art methods of machine learning on a set of 12 datasets. Finally we use the machine learning algorithm proposed in this chapter to identify populations of immune system cells that capture key components of the immune system (populations we denote "immune states") - including populations that are related to the disease severity of Rheumatoid Arthritis.

Finally we note that this portion of the research is based on the work in [28, 29, 30].

1.4.3 Problem 3: Generating Constrained Predictive Models and Identifying Optimal Treatments

When the immune system is malfunctioning by, for instance, targeting the body itself as in autoimmunity or failing to eliminate cancer cells, a number of immunotherapies have been designed to modify the populations of immune system cells by blocking cell surface receptors or releasing cytokines. Examples include Ipilumumab and Nivolumab which can deactivate effector cells [74, 163]. While these and other immunotherapies have shown promising results as cancer treatments, an important question is how to determine an optimal immunotherapy drug, dosage, the timing of drug administration. This is made difficult because an optimal immunotherapy treatment must depend on the number of immune system cells and their cellular characteristics, especially in the case of cancer immunotherapy [79]. The final problem posed in this dissertation therefore is how to identify optimal immunotherapy treatments based upon the population and characteristics of immune system cells, such as the number

of effector and regulatory type cells.

We assume that the dynamics that govern the immune system response vary slightly between different patients, and that the initial populations of immune system cells are most important for predicting if an immunotherapy treatment will lead to complete elimination of a tumor. We then generate a model of the Lyapunov function of the system (that is dependent upon the selected treatment strategy) using trajectory measurements of tumor growth starting from different initial populations of immune system cells, treatment strategies, and different patients with slight variations in the immune system dynamics.

For a selected treatment strategy, if the initial conditions of the patient are within the Region Of Attraction (ROA) of the system (based on the Lyapunov function model) then the treatment strategy is predicted to lead to complete tumor elimination for that patient. The Lyapunov function model makes no assumptions on the form of the underlying system dynamics. While Lyapunov functions are often used to prove stability or to find regions of attraction of a system with known dynamics there exist few methods wherein Lyapunov function estimates can be generated by trajectory measurements alone.

We develop a new method to model Lyapunov functions from trajectory data in Chapter 5. However, since Lyapunov functions are globally positive, we first develop a convex optimization problem to select a positive function that best models given input and output data according to either the least absolute deviations or least squares metrics. We then develop a method of generating values of a converse Lyapunov function using measurements of trajectories from different initial conditions, thus generating a model of the Lyapunov function using only measured trajectory data.

We demonstrate the proposed approach by modeling the Lyapunov function of simulated data from the cancer dynamics model described in Eq. (1.7). Fortunately,

we may represent the treatment strategies which (for the given initial conditions) are predicted to stabilize the system as a semi-algebraic set. We then show that selecting an optimal pulsed immunotherapy treatment can be formulated as a global polynomial optimization problem using the Lyapunov function model. This approach assumes that the dynamical models of each patient will be identical. To address this assumption we show that if the patient models are not identical (by varying the dynamical models to simulate different patients) we are able to find robust immunotherapy treatments that can effectively treat patients with varying immune dynamics, but are no longer optimal for any given patient.

In Chapter 6 we formulate a new algorithm to solve global polynomial optimization problems. To demonstrate that the algorithm, in conjunction with the model of the Lyapunov function, can find optimal treatment strategies, we simulate 1000 random patients and use the GPO algorithm to select optimal treatments. The simulations of the immunotherapy model show that over 30% of the selected treatments are within 10% of the optimal treatment strategy, and that all of the treatments lead to complete cancer elimination within 120 days - all based solely on measured trajectories of the system dynamics.

Finally we note that this portion of the research is based on the work in [27, 31].

Chapter 2

BACKGROUND MATERIAL

In this chapter we lay the foundation for the methods proposed in this dissertation. In Section 2.1 and 2.2 we provide background material on convex optimization - specifically delving into the topics of semidefinite programming and polynomial optimization that are important for developments within all chapters of this dissertation. Furthermore, in Section 2.3 we review Lyapunov theory which is most relevant to Chapter 5 and in Section 2.4 we review methods of classification and regression using support vector machines that are most relevant to Chapter 4. Finally in Section 2.5 we briefly review some uncertainty quantification methods which lays the foundation for Chapter 3.

2.1 Convex Optimization Problems

All of the methods proposed in this dissertation generate predictive models by solving optimization problems. An *optimization problem* has the form,

$$\begin{aligned} \min_{z \in \mathbb{R}^n} \quad & f_0(z) \\ \text{subject to:} \quad & f_i(z) \leq 0 \ \ \forall \ \ i = 1, ..., m, \end{aligned} \tag{2.1}$$

where each $f_i : \mathbb{R}^n \to \mathbb{R}$ is a function of the vector $z \in \mathbb{R}^n$ called the decision variables. The function $f_0(z)$ is called the objective function, and the functions $f_i(z)$ for $i = 1, ..., m$ are constraint functions.

If we define the set of all feasible points as $Z := \{z \in \mathbb{R}^n \mid f_i(z) \leq 0 \, \forall \, i = 1, ..., m\}$, then $z^* \in Z$ is optimal if $f_0(z^*) \leq f_0(y)$ for all $y \in Z$.

Optimization problems are divided into different classes depending on the types of functions that compose the objective function and constraints. The first types of optimization problem we will define are the class of convex and nonconvex optimization problems. If the functions $f_i(z)$ for all $i = 0, ..., m$ that compose the objective and constraint functions are convex then we say the optimization problem is convex, otherwise the optimization problem is nonconvex. We define convex functions as follows.

Definition 1. *A function $g : X \to \mathbb{R}$ is convex if*

$$g(tz + (1-1)y) \leq tg(z) + (1-t)g(y)$$

for all $z, y \in X$ and all $t \in [0, 1]$.

An important property of convex optimization problems are that any locally optimal decision variables are globally optimal. This implies that if a locally optimal decision variable is found, it must be the globally optimal solution. Nonconvex optimization problems do not have this property and there are seldom any ways to verify if a locally optimal decision variable is globally optimal.

The classes of convex optimization can be further divided into the class of Linear Programs (LPs), Quadratic Programs (QPs), and semidefinite programs (SDPs). Of most significance to this dissertation is the class of semidefinite programs (which contains the classes of LPs and QPs), which we cover in greater detail in the following subsection.

2.1.1 Semidefinite Programming

A semidefinite programming problem in its standard form can be represented as follows.

$$\begin{aligned} \min_{Z \in \mathbb{R}^{n \times n}} \quad & \text{trace}(CZ) \\ \text{subject to:} \quad & \text{trace}(A_i Z) = b_i \ \ \forall \ \ i = 1, ..., m, \\ & Z \succeq 0, \end{aligned} \tag{2.2}$$

where $C \in \mathbb{R}^{n \times n}$ and $A_i \in \mathbb{R}^{n \times n}$ for all $i = 1, ..., m$.

Optimization problems in the form of Eqn. (2.2) are convex optimization problems that can be solved in polynomial time (see [3]) using solvers such as Sedumi and Mosek [4, 153]. The constraint, $Z \succeq 0$, restricts the decision variable Z to be within the cone of symmetric positive semidefinite $n \times n$ matrices.

Semidefinite programming has found numerous implementations in optimal control with linear matrix inequalities. Linear Matrix Inequalities (LMIs) have been applied to determine stability of linear systems and design optimal H_2 and H_∞ controllers and observers for nominal and uncertain plant models. More examples of the use of LMIs to solve problems related to control systems can be found in [47]. In this dissertation however we focus on the application of semidefinite programming to the optimization of polynomial functions.

2.1.2 Polynomial Optimization

We define *polynomial optimization* as an optimization problem where the vector of decision variables z parameterize at least one degree bounded, possibly multivariate, polynomial function. We denote this ring of polynomial functions in $x \in \mathbb{R}^n$ as $\mathbb{R}[x]$ and the degree d bounded ring of polynomial functions as $\mathbb{R}_d[x]$.

To show that any polynomial in $\mathbb{R}_d[x]$ can be linearly parameterized by a vector of decision variables we must first denote the monomials of the variables $x \in \mathbb{R}^n$. The monomials in variables $x \in \mathbb{R}^n$ are denoted as $x^\alpha := \prod_{i=1}^n x_i^{\alpha_i}$ where $\alpha \in \mathbb{N}^n$. Monomials can be ordered using various orderings on $\mathbb{N}^n$. In this dissertation the graded lexicographical ordering is used. This ordering is defined inductively as follows. For $a, b \in \mathbb{N}^n$, $a \leq b$ if $\sum_{i=1}^n a_i < \sum_{i=1}^n b_i$, or $a_1 = b_1$ and $[a_2, \cdots, a_n] \leq [b_2, \cdots, b_n]$. Denote by $Z(x)$ the infinite ordered vector of all monomials, where $x^\alpha < x^\beta$ if $\alpha < \beta$. Because we have used the graded lexicographical ordering, if we restrict ourselves to the first $\binom{d+n}{d}$ elements of Z, then this is the vector of all monomials of degree d or less. We denote this truncated vector as $Z_d(x)$ and the length of Z_d as $q := \binom{d+n}{d}$. Using this definition, it is clear that any polynomial has a vector representation as $p_d(x; z) = z^T Z_d(x)$ for some vector $z \in \mathbb{R}^q$, where d is the degree of polynomial p.

In addition for any given $y \in \mathbb{R}^n$ we have that $p(y; z)$ is a linear function with respect to the decision variables z and can be represented as,

$$p(y; z) = \sum_{j=1}^{q} z_j \prod_{i=1}^{n} y_i^{\alpha_i}.$$

Therefore evaluating p at any point y is a linear function of the decision variables and is a convex function of the decision variable z.

Furthermore, consider constraints of a polynomial function evaluated not over a single given point y, but over an infinite amount of points $y \in Y$. For instance consider the constraint,

$$p(y; z) \geq 0, \text{ for all } y \in Y,$$

where there exist an infinite number of constraints on the decision variables z. We will discuss methods of enforcing this constraint in the following section.

2.2 Optimization of Positive Polynomials

In this section a parametrization of a set of nonnegative polynomials is defined using the Sum-Of-Squares (SOS) polynomial approach. Next a parametrization of a set of polynomials nonnegative over a semialgebraic set is given based upon the Positivstellensatz result. Finally we define the greatest lower bound problem and the closely related Global Polynomial Optimization problem.

2.2.1 Sum-of-Squares Polynomials

In this subsection we define the set of SOS polynomials, and show that the set of SOS polynomials is a convex set defined by positive semidefinite matrices. Formally, we denote the set of SOS polynomials as

$$\mathbb{S}[x] := \left\{ s \in \mathbb{R}[x] \,:\, s(x) = \sum_{i=1}^{l} p_i^2(x),\, p_i \in \mathbb{R}[x],\, l \in \mathbb{N} \right\}, \tag{2.3}$$

and the set of degree d bounded sum-of-squares polynomials is $\mathbb{S}_d[x]$.

Clearly any element of $\mathbb{S}[x]$ is a nonnegative polynomial. Furthermore, using the vector representation of polynomials we can show that any SOS polynomial has a matrix representation as follows.

$$\sum_{i=1}^{l} p_i^2(x) = \sum_{i=1}^{l} Z_d(x)^T z_i z_i^T Z_d(x) = Z_d(x)^T \left(\sum_{i=1}^{l} z_i z_i^T \right) Z_d(x) = Z_d(x)^T M Z_d(x),$$

where $M \in \mathbb{R}^{q \times q}$ is positive semidefinite since the matrix $z_i z_i^T \succeq 0$. Moreover, any polynomial,

$$h(x) = Z_d(x)^T M Z_d(x).$$

is an SOS polynomial if $M \succeq 0$, because

$$Z_d(x)^T M Z_d(x) = Z_d(x)^T \sum_{i=1}^{q} \left(c_i c_i^T \right) Z_d(x) = \sum_{i=1}^{q} Z_d(x)^T z_i z_i^T Z_d(x) = \sum_{i=1}^{l} p_i^2(x),$$

where the vectors c_i are the eigenvectors of the matrix M multiplied by the square root of the corresponding eigenvalue. Therefore we have shown that any SOS polynomial has a representation of this form,

$$h_{2d}(x; M) = Z_d^T(x) M Z_d(x),$$

for some $M \succeq 0$.

A positive semidefinite M is a certificate of positivity on $\mathbb{R}^n$ for the polynomial h_{2d}. However, it is important to note that even if a polynomial cannot be parameterized with a positive semidefinite M it still may be nonnegative on $\mathbb{R}^n$. For instance the motzkin polynomial is an example of a degree $2d$ bounded polynomial that is not an SOS polynomial but is nonnegative on the domain $\mathbb{R}^n$ [111].

Next we use the Positivstellensatz result to find certificates for nonnegative polynomials on subsets of $\mathbb{R}^n$.

2.2.2 Putinar's Positivstellensatz, Quadratic Modules and the Archimedean Property

In this section, we review Putinar's positivstellensatz which gives necessary conditions for a polynomial to be positive on the semiaglebraic set $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0,\ i = 1, \ldots, l\}$, where $g_i \in \mathbb{R}[x]$, $S \neq \emptyset$ and S is compact. The degree bounded

Putinar's Positivstellensatz uses the polynomials g_i that define S to deduce a cone of polynomials which are non-negative on S. This cone is the quadratic module which we define as follows.

Definition 2. *Given a finite collection of polynomials $g_i \in \mathbb{R}[x]$, we define the quadratic module as*

$$A := \{p \mid p = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i \quad \sigma_i \in \mathbb{S}[x]\},$$

and the degree-k bounded quadratic module as

$$A^{(k)} := \{p \mid p = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i \quad \sigma_i \in \mathbb{S}_k[x]\}.$$

Clearly, any polynomial in A is nonnegative on S. Furthermore, since for any $f, g \in \mathbb{R}[x]$ we have that $fg \in \mathbb{R}[x]$, the constraint $p \in A^{(k)}$ can be represented as a polynomial in matrix form and the constraint can be represented as an LMI. Furthermore, if the module satisfies the Archimedean property, then Putinar's Positivstellensatz states that any polynomial which is positive on S is an element of A. That is, A parameterizes the cone of polynomials positive on S.

A quadratic module A is said to be Archimedean if there exists some $p \in A$ and $R \neq 0$ such that $p(x) = R^2 - \sum_{i=1}^{n} x_i^2$. We say that $\{g_i\}$ is an Archimedean representation of S if the associated quadratic module is Archimedean. Note that the Archimedean property is a property of the functions g_i which then define the quadratic module and not a property of S. Specifically, if S is compact, and $\{g_i\}_{i=1}^{l}$ is not Archimedean, then we may construct an Archimedean representation $\{g_i\}_{i=1}^{l+1}$ where $g_{l+1}(x) = R^2 - \sum_{i=1}^{n} x_i^2$ for any sufficiently large $R > 0$.

A consequence of Putinars Postivstellensatz is that the problem of computing the greatest lower bound of a polynomial function over a semi-algebraic set can be formulated as a convex optimization problem.

2.2.3 The Greatest Lower Bound Problem

The Greatest Lower Bound (GLB) problem is defined as,

$$\begin{aligned} \lambda^* := \max_{\lambda \in \mathbb{R}} \quad & \lambda \\ \text{such that:} \quad & f(x) - \lambda > 0, \quad \forall x \in S, \end{aligned} \tag{2.4}$$

where we define the feasible set S as

$$S := \{x \in \mathbb{R}^n \,:\, g_i(x) \geq 0,\ h_j(x) = 0\}. \tag{2.5}$$

Unfortunately, there is no algorithm which solves the GLB exactly in polynomial time. However asymptotically exact approaches to solving the GLB problem such as SOS programming in [123], and its dual Moment-relaxation problem in [98] can be used to approximate the GLB problem when the functions f, g_i and h_j are polynomials. Both these approaches are well-studied and have associated Matlab toolboxes, including SOSTOOLS [130] and Gloptipoly [73]. Both approaches yield a hierarchy of primal/dual semidefinite programs (SDPs) with increasing sequences of optimal values $\{p_k^*\}_{k\in\mathbb{N}}$ (SOS) and $\{d_k^*\}_{k\in\mathbb{N}}$ (Moment) such that $p_k^* \leq d_k^* \leq f^*$. Furthermore, under mild conditions, the algorithms are asymptotically accurate in the sense that $\lim_{k\to\infty} p_k^* = \lim_{k\to\infty} d_k^* = f^*$ [141].

Moreover, if the feasible set, S, as defined in (2.5), is nonempty and compact, then there exist constants c_1 and c_2, depending on polynomials f, h_j, and g_i such that $|p_k^* - f^*| \cong \frac{c_2}{\sqrt[c_1]{\log(k)}}$ [116].

A problem closely related to the GLB is the Global Polynomial Optimization Problem.

2.2.4 Global Polynomial Optimization

Global Polynomial Optimization (GPO) is defined as optimization of the form

$$\begin{aligned} f^* := \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{such that:} \quad & g_i(x) \geq 0 \quad \text{for } i = 1,\ldots,s \\ & h_j(x) = 0 \quad \text{for } j = 1,\ldots,t, \end{aligned} \tag{2.6}$$

where f, g_i, and h_i are real-valued polynomials in decision variables x. Clearly the GLB and GPO problems are closely related in that $\lambda^* = f^* = f(x^*)$, but are not

equivalent in that the GLB problem does not find x^*.

As defined in Eq. 2.6, the GPO problem encompasses many well-studied subclasses including Linear Programming (LP) [88, 85], Quadratic Programming (QP) [106], Integer Programming (IP), Semidefinite Programming (SDP) and Mixed-Integer Nonlinear Programming (MINLP) [99]. Because of its generalized form, almost any optimization problem can be cast or approximately cast as a GPO, including certain NP-hard problems from economic dispatch [122], optimal power flow [64] and optimal decentralized control [101]. As applied to control theory, GPO can be used for stability analysis of polynomial dynamical systems by, e.g., verifying polytopic invariants as in [110].

When f and g_i are convex and h_i are affine, the GPO problem is convex and may be solved using barrier functions and gradient descent. When the GPO problem is not convex, there also exist special cases in which the global optimum is guaranteed to be found. For example, in [154] the unconstrained problem was solved using Groebner bases. In the special case of $x \in \mathbb{R}^1$, the problem was solved in [144, 145]. In addition, there exist several widely used heuristics which often yield reasonably suboptimal and approximately or exactly feasible solutions to the GPO problem (e.g. [35, 151]), but which we will not discuss here in depth.

The moment approach can be used to obtain an algorithm with finite termination time, unfortunately, however, it has been shown that the class of problems for which these methods terminate is a strict subset of the general class of GPO problems [115] and furthermore, there are no tractable conditions verifying if the algorithm will terminate or bounds on computational complexity in the case of finite termination. Therefore, in Chapter 6 we propose a new method for solving GPO problems.

2.3 Lyapunov Theory

Lyapunov Theory refers to the work of A. M. Lyapunov in which the definitions of stability were standardized and methods for determining stability of systems were proposed. Lyapunov considered systems of ordinary differential equations of the form

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0, \tag{2.7}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ and $x_0 \in \mathbb{R}^n$ are respectively denoted the vector field and initial conditions. If we define $g(x_0, t)$ to be the solution map of Eq. (2.7), where $g(x, 0) = x(0)$ and $\frac{d}{dt}g(x, t) = f(g(x, t))$ for all $x \in \mathbb{R}^n$ and $t \geq 0$ then we may define asymptotic stability of a system on a set X as follows.

Definition 3. *Given a nonlinear ODE, $\dot{x} = f(x)$, the point $x = 0$ is asymptotically stable on the set X if,*

$$\lim_{t\to\infty} g(x, t) = 0, \quad \forall x \in X \tag{2.8}$$

where $\partial_t g(x, t) = f(g(x, t))$ and $g(x, 0) = x$.

Lyapunov based methods estimate the region of attraction on a compact set $X \in \mathbb{R}^n$ by searching for a Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$, and a scalar c such that the time derivative of the Lyapunov function is negative for all values of x on the set $D := \{\ x \mid V(x) \leq c,\ x \neq 0\}$. The Lyapunov function, V, must be positive on the set D but can be any type of function - for instance we see logarithmic Lyapunov functions in [1].

If we consider polynomial Lyapunov functions then the search for Lyapunov proofs of stability can be cast as a semi-definite programming problem using the Sum-of-Squares (SOS) optimization discussed in the previous section. The toolbox SOSTools [129], provides a general purpose sum-of-squares programming solver that can

be used to solve multiple control systems problems using Semi-Definite Programming (SDP) solvers such as SeDuMi [153]. Note, however, that even when the ODE of a system is known, $\dot{x} = f(x)$, and the function $f : \mathbb{R}^n \to \mathbb{R}^n$ is polynomial, methods for finding the region of attraction involve a bilinear SOS optimization problem that can be, at least approximately, solved using bisection [82] or genetic algorithms [70]. SDP techniques have also been used to find Lyapunov functions as a certificate of stability for switched [15] and hybrid systems [81].

Most relevant to this dissertation are data based methods (of which there are very few). However when an ODE and Lyapunov function are given, but the ODE does not capture all the system dynamics, a data-based approach to estimate these unmodeled dynamics and return an estimate of the region of attraction for the system can be found in [9].

If an ODE does have an equilibrium point, then the set of converse Lyapunov theorems guarantee the existence of Lyapunov functions that prove stability on the region of attraction [69]. For example, such a converse Lyapunov function,

$$V(x) = \int_0^\infty \|g(x,t)\|^2 \, dt, \tag{2.9}$$

is guaranteed to exist.

Non-Lyapunov based methods for calculating stability and the region of attraction generally involve numerically integrating the vector field of the ODE. One such numerical method (see [26]) involves identifying the equilibrium points whose unstable manifolds contains initial conditions that approach the equilibrium point of interest, which is usually performed by integrating the vector field. The union of these manifolds are then within the region of attraction. Other methods involve numerically integrating the vector field in the forward and reverse direction and observing the trajectories of a set of initial conditions as in [62].

In Chapter 5 we develop a data based method for modeling a converse Lyapunov function for an unknown system and use this model to estimate the region of attraction.

2.4 Support Vector Machines

One of the fundamental problems in machine learning is that of using labeled data to predict the values of unlabeled data. For instance, assume a data generating mechanism has generated data points $\{x_i\}_{i=1}^m \subset \mathbb{R}^n$ with corresponding labels $\{y_i\}_{i=1}^m \subset \mathbb{R}$, then a common problem is to predict the labels of future points drawn from the DGM.

Support Vector Machines (SVMs) are a machine learning method for binary classification ($y_i \in \{-1, 1\}$) and regression problems ($y_i \in \mathbb{R}$) developed primarily by Vladmir Vapnik in the 1990s [14, 36]. The support vector machine was designed to minimize the error of predicting unlabeled data based on minimizing a bound on the prediction error provided by Vapnik-Chervonenkis (VC) theory. In this section we will pose both the 1-norm soft margin support vector machine as well as the epsilon Support Vector Regression (ϵ-SVR) problem. We will propose improvements to these machine learning algorithms in Chapter 4.

2.4.1 The 1-norm Soft Margin SVM:

Suppose we are given a set of m *training data* points $\{x_i\}_{i=1}^m \subset \mathbb{R}^n$, each with associated *label* $y_i \in \{-1, 1\}$ for $i = 1, \cdots, m$. We want to find a classifier, f, that correctly classifies the training points (i.e. $f(x_i) = y_i$) as well as points that are not included in the training data. A point is misclassified if $f(x_i) = -y_i$ and we impose a penalty $C \in \mathbb{R}^+$ on points in the training data that have been misclassified. We define the primal version of the *linear* 1-norm soft margin problem as

$$\min_{w\in\mathbb{R}^n,\ \zeta\in\mathbb{R}^m,\ b\in\mathbb{R}} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{m}\zeta_i \qquad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, \quad (2.10)$$

where the learned map (*classifier*) from inputs to outputs is then $f : \mathbb{R}^n \to \{-1, 1\}$ where $f(z) = \text{sign}(w^T z + b)$.

This map maximizes the margin (distance) between training data with a negative label, and those with a positive label. If the relationship between the inputs and outputs is not linear, then we may introduce a positive kernel function, k to generate nonlinear classifiers.

Definition 4. *We say a function $k : Y \times Y \to \mathbb{R}$ is a* **positive kernel function** *if*

$$\int_Y \int_Y f(x)k(x, y)f(y)dxdy \geq 0$$

for any function $f \in L_2[Y]$.

For any given positive kernel k we may associate a function Φ such that $k(x, y) = \langle\Phi(x), \Phi(y)\rangle$ where $\langle\cdot, \cdot\rangle$ is the dot product. In this case Optimization Problem (2.10) might be posed as

$$\begin{aligned} \min_{w\in\mathbb{R}^n,\ \zeta\in\mathbb{R}^m,\ b\in\mathbb{R}} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^{m}\zeta_i \\ \text{s.t.} \quad & y_i(\langle w, \Phi(x_i)\rangle + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, \end{aligned} \quad (2.11)$$

and given the optimal values w^* and b^*, the classifier would be $f(x) = \text{sign}\left(\langle w^*, \Phi(x)\rangle + b^*\right)$ which maximizes the margin between points of negative and positive label in the kernel space $\Phi(x)$. Although the primal form of SVM has certain advantages - see [133], it is ill-suited to kernel learning. For this reason, we consider the dual formulation,

$$\begin{aligned} \max_{\alpha\in\mathbb{R}^m} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j\langle\Phi(x_i), \Phi(x_j)\rangle \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \ \ \forall \ \ i = 1, ..., m. \end{aligned} \quad (2.12)$$

In this case we may eliminate Φ from the optimization problem using $\langle \Phi(x_i), \Phi(x_j) \rangle = k(x_i, x_j)$ where the elements $k(x_i, x_j)$ define the kernel matrix. In this case, the resulting classifier is only a function of k and becomes

$$f(z) = \text{sign}\left(\sum_{i=1}^{m} \alpha_i^* y_i k(x_i, z) + b \right).$$

Note that b can be found a posteriori as the average of $y_j - \sum_{i=1}^{m} \alpha_i y_i k(x_j, x_i)$ for all j such that $0 < \alpha_j < C$ - See [140]. This implies that the primal variable w is not explicitly required for the calculation of b, and that the resulting learned classifier, f, may be expressed solely in terms of α and the kernel function.

Commonly used positive kernel functions include the gaussian kernel $k_1(x, y) = \mathrm{e}^{(-\beta ||x-y||^2)}$, where β is the bandwidth (and must be chosen a priori) and the polynomial kernel $k_2(x, y) = (1 + x^T y)^d$ where d is the degree of the polynomial. We next formulate the Kernel Learning problem for the ϵ-SVR problem.

2.4.2 The ϵ-SVR Problem:

Suppose we are given a set of m *training data* points $\{x_i\}_{i=1}^m \subset \mathbb{R}^n$, each with associated *output* $y_i \in \mathbb{R}$ for $i = 1, \cdots, m$. For a given "penalty" parameter $C \in \mathbb{R}^+$ and allowable loss $\epsilon \in \mathbb{R}^+$, we define the primal version of the *linear* epsilon Support Vector Regression (ϵ-SVR) problem as

$$\begin{aligned} \min_{w \in \mathbb{R}^n,\ \xi \in \mathbb{R}^m,\ \xi^* \in \mathbb{R}^m,\ b \in \mathbb{R}} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{m} (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - w^T x_i - b \leq \epsilon + \xi_i, \quad \xi_i \geq 0, \\ & w^T x_i + b - y_i \leq \epsilon + \xi_i^*, \quad \xi_i^* \geq 0, \end{aligned} \tag{2.13}$$

where the learned predictor from inputs to outputs is then $f : \mathbb{R}^n \to \mathbb{R}$ where $f(x) = w^T x + b$.

As in the case of the binary support vector machine, the use of a positive kernel

function, k, can be used to modify the learned predictor. Using the transformation Φ, such that $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$, Optimization Problem (2.13) can be posed as

$$\begin{aligned} \min_{w \in \mathbb{R}^n,\ \xi \in \mathbb{R}^m,\ \xi^* \in \mathbb{R}^m,\ b \in \mathbb{R}} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{m} (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - w^T \Phi(x_i) - b \leq \epsilon + \xi_i, \quad \xi_i \geq 0, \\ & w^T \Phi(x_i) + b - y_i \leq \epsilon + \xi_i^*, \quad \xi_i^* \geq 0, \end{aligned} \tag{2.14}$$

Given a solution, the predictor would be $f(z) = \langle w, \Phi(z) \rangle + b$.

As in the 1-norm soft margin problem we will consider the dual formulation,

$$\begin{aligned} \max_{\alpha \in \mathbb{R}^m, \alpha^* \in \mathbb{R}^m} \quad & -\frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \langle \Phi(x_i), \Phi(x_j) \rangle \\ & - \epsilon \sum_{i=1}^{m} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i (\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \ \ \forall \ \ i = 1, ..., m. \end{aligned} \tag{2.15}$$

In this case we may eliminate Φ from the optimization problem using $\langle \Phi(x_i), \Phi(x_j) \rangle = k(x_i, x_j)$ where the elements $k(x_i, x_j)$ define the kernel matrix. The resulting predictor is only a function of k and becomes

$$f(z) = \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) k(x_i, z) + b.$$

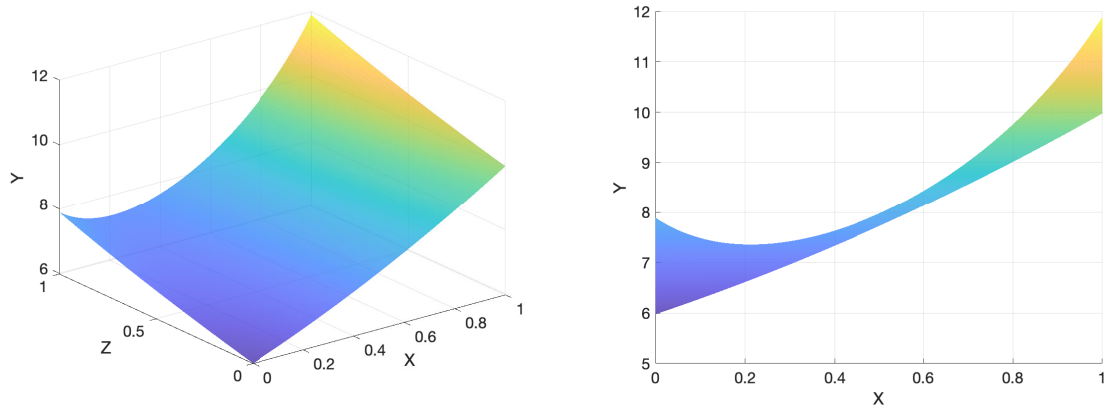
Note that b can be found a posteriori through exploiting the Karush-Kuhn-Tucker (KKT) conditions - See [147]. This implies that the primal variable w is not explicitly required for the calculation of b, and that the resulting predictor, f, may be expressed solely in terms of α, α^* and the kernel function. In Chapter 4 we will propose a new class of kernel function and a new algorithm to optimize this kernel function with respect to the given data.

2.5 Uncertainty Quantification

Consider the problem of using numerical algorithms to model the probability density function of a continuous random variable based on data sampled from that variable. Specifically, suppose we are given a parameterized set of probability distributions and would like to select the distribution which best models a set of data, using some metric for fit (e.g. Maximum Likelihood Estimation (MLE)). This problem is often referred to as uncertainty quantification and is a critical part of analysis in such fields such as climate change [124, 168], control [50, 25], and economic policy [167, 84]. Uncertainty quantification is a well-studied field of research [43, 156], and as a result, there are many candidates for the parameterized class of models and associated fitting algorithms.

As an illustration of how uncertainty can affect measured data consider Figures 2.1(a) and (b). In Fig. 2.1(a) we show a function with two inputs (X and Z) and one output (Y). However, assume that the input Z represents varying operating conditions, noise, or some unmeasured characteristic. Then in Fig. 2.1(b) we show the value of the output, Y, for different values of the input, X, over all possible values of the unmeasured value Z. Clearly, the function no longer returns a single value for a given input, but will vary depending upon the value of the unknown Z. When measuring data from a DGM, the unmeasured affects imply that the exact relationship between the inputs and outputs will be inexact, as illustrated in Fig. 2.1(b).

By quantifying the variability in the measured data, robust predictive models can be generated to account for the unknown factors causing the variation in the measured data. In this dissertation we use Probability Density Functions to model the variability of data drawn from a DGM.



(a) A function with two inputs (X,Z) and one output (Y). All inputs are measured and the output has no uncertainty.

(b) A function with one input (X), one unmeasured input (Z) and one output (Y). Since Z is unmeasured the output is uncertain.

Figure 2.1: A plot of a function with two inputs (X,Z) and one output (Y). When both inputs are known the output is always known (a), when only one input is known the output is uncertain (b).

2.5.1 Important Definitions

We define $L_1(\Delta)$ as the set of all Lebesgue measurable functions f on $\Delta \subset \mathbb{R}^n$ that satisfy $\int_\Delta f(\delta)d\delta < \infty$ and $L_{1+}(\Delta)$ as all L_1 functions that are positive for all values in Δ.

A Probability Density Function (PDF) on a bounded compact domain Δ is a convex subset of L_1 functions [48] and is defined as follows.

Definition 5. *We say a function $f \in L_{1+}(\Delta)$ is a* **Probability Density Function** *on a bounded compact domain Δ if,*

$$\int_\Delta f(\delta)d\delta = 1 \quad \textit{and,} \quad f(\delta) \geq 0 \textit{ for all } \delta \in \Delta.$$

From this definition we may define F_Δ as the set of all bounded PDFs on a compact

domain Δ.

$$F_\Delta := \left\{ g \in L_{1+}(\Delta) \;\middle|\; g(\delta) = \frac{f(\delta)}{\int_\Delta f(\delta)d\delta}, \quad f \in L_{1+}(\Delta), \quad \sup_{\delta\in\Delta} f(\delta) < \infty \right\}.$$

Density Two metrics for comparing the distance between two PDFs, f and g are The squared Hellinger distance, $d_H^2(f,g) := \int_\Delta \left(\sqrt{f(x)} - \sqrt{g(x)}\right)^2 dx$, and the L_1 distance, $d_T(f,g) := \frac{1}{2}\int_\Delta |f(x) - g(x)|$.

In this paper we define density between two sets of PDFs using the Hellinger metric.

Definition 6. *Given two sets of PDFs F and G we say that F is dense in G if for any $g \in G$ and epsilon > 0 there exists an $f \in F$ such that $d_H^2(f,g) \leq \epsilon$.*

Note, however, that the relationship between the Hellinger distance and the L_1 distance is $d_H^2(f,g) \leq d_T(f,g)$ [93], implying that this definition of density holds if we replace the squared Hellinger distance with the L_1 distance.

2.5.2 Metrics for Selecting PDFs to Model Random Variables

To select a predictive model which best models the variability of measured data we will use two different metrics - the likelihood and worst case likelihood metrics.

Likelihood Metrics

Suppose we are given a set of data, $\mathcal{D} := \{\delta_i\}_{i=1}^m$, drawn from some unknown DGM where $\delta \in \mathbb{R}^n$ is a continuous random variable. Here we will propose two metrics for selecting a PDF from a given class to model the DGM.

The first metric is the likelihood of the data. For a PDF f, with given parameters θ, the likelihood of the data is given by,

$$L_f(\mathcal{D}) = \prod_{\delta\in\mathcal{D}} f(\delta;\theta). \tag{2.16}$$

Finding the parameters, θ, which result in the maximum likelihood of the data is called the Maximum Likelihood Estimation (MLE) optimization problem. This approach is covered extensively in work such as [113].

A second metric used to evaluate a PDF with respect to given data is the worst case likelihood of the data,

$$W_f(\mathcal{D}) = \min_{\delta \in \mathcal{D}} f(\delta; \theta). \tag{2.17}$$

Finding the parameters, θ, which result in the maximum worst case likelihood is called the maximum Worst Case likelihood Estimation (WCE) optimization problem. The level sets of the WCE optimized PDF functions are often used to characterize sets of minimal volume that are likely to contain future data points.

Uncertainty Quantification with Gaussian Distributions

For example, consider the class of Gaussian PDFs. For any $\mu \in \mathbb{R}^n$ and positive matrix $P \succeq 0 \in \mathbb{R}^{n \times n}$, we may obtain a Gaussian Probability Density Function of the form

$$f_G(\delta; \mu, P) := \frac{e^{-\frac{(\delta-\mu)^T P(\delta-\mu)}{2}}}{(2\pi)^{n/2}\sqrt{\det(P^{-1})}}. \tag{2.18}$$

For a given set of data the MLE optimization problem (for a Gaussian PDF) is the following optimization problem.

$$\max_{\mu \in \mathbb{R}^n, P \in \mathbb{R}^{n \times n}} \quad L_{f_G}(\mathcal{D}) = \prod_{\delta \in \mathcal{D}} f_G(\delta; \mu, P)$$

The solution to the maximum likelihood problem for a Gaussian PDF has an analytical solution that can be computed efficiently where, μ^* is the mean of the data, and P^* is the inverse of the empirical covariance matrix, and is called the precision matrix.

For the same set of data the WCE problem (again for a Gaussian PDF) is the following optimization problem,

$$\max_{\mu\in\mathbb{R}^n, P\in\mathbb{R}^{n\times n}} \quad W_{f_G}(\mathcal{D}) = \max_{\mu\in\mathbb{R}^n, P\in\mathbb{R}^{n\times n}} \min_{\delta\in\mathcal{D}} f_G(\delta;\mu,P).$$

Generally the worst case likelihood approach does a poor job of modeling the PDF of the resulting DGM. However, it can be used to characterize volumes where future data is likely to fall based on the level sets of the model PDF. The level sets of the Gaussian distribution are semi-algebraic and are defined as follows.

$$H_\beta = \{\delta : (Z_d(\delta)-\mu)^T P(Z_d(\delta)-\mu) \leq \beta\}, \tag{2.19}$$

The worst case likelihood approach tends to generate sets that contain all of the data with significantly smaller volume than the maximum likelihood approach. In fact if the worst case likelihood approach is used to find μ and P, and $\beta = \max_{\delta\in\mathcal{D}}(Z_d(\delta)-\mu)^T P(Z_d(\delta)-\mu)$, then we show in Chapter 3 that the set S is exactly the ellipsoid of minimal volume containing all of the points in $\mathcal{D}$.

Chapter 3

UNCERTAINTY QUANTIFICATION USING POLYNOMIAL AND SUM-OF-SQUARES OPTIMIZATION

To generate models that capture the diversity of immune system cells found in, for instance flow cytometry or mass cytometry datasets, we will model the Probability Density Function (PDF) of characteristics of the immune system cells. Analyzing the PDF of the cellular characteristics of a patients immune system cells returns a holistic model of the immune system when compared to sorting the cells into a finite set of groups. PDF models can then be compared using metrics such as the Hellinger distance [72] to determine immune system similarity between two individuals. However, the PDF of cytometry datasets often do not fall under any known distribution like, for instance a Gaussian. Modeling the cellular characteristics with a PDF thus requires state of the art techniques.

One approach to modeling the PDF of multivariate data is to use Gaussian Mixture Models (GMMs) - a summation of multivariate normal PDFs. GMM distributions are a significantly richer class of distributions than multivariate normals and examples of the use of GMMs to model random variables include adjacent vehicle motion in [169], and modeling wind power generation in [87]. Unfortunately the use of numerical algorithms for fitting GMMs to data suffers from the problem of non-convexity of the optimization problem. Specifically, the MLE optimization problem introduced in Chapter 2 for GMMs is a non-convex optimization problem. Thus any distribution obtained from an expectation-maximization algorithm as applied to the MLE problem for GMMs is likely to be sub-optimal, with no provable bounds on performance [11].

In this chapter we propose new sets of PDFs to model the distribution of multivariate data which are dense in the set of all bounded PDFs (F_Δ) on a compact domain (Recall Definition 6 from Subsection 2.5.1 for the definition of density for sets of PDFs). However, unlike the GMMs, the parameters of these sets are convex with respect to the MLE optimization problem and the globally optimal parameters can be found.

In Section 3.1 we propose a new set of distributions called Sliced Distributions (SDs). We then define in Sections 3.2 and 3.3 two subsets of Sliced Distributions, the Sliced-Exponential (SE) and Sliced-Normal (SN) distributions and prove that, for any $\epsilon > 0$ and PDF in $f \in F_\Delta$, their exists a SE PDF g such that the squared Hellinger distance between f and g is less than ϵ. Furthermore we introduce convex optimization problems that can be used to fit a SD to a given set of data drawn from a DGM. Then in Section 3.5 we compare SDs to classes of PDFs such as the Gaussian and GMMs on standard metrics, and show that SDs generate superior models of the distribution of publicly available datasets. We then apply the SDs to model the difference in the PDF of immune cell characteristics between healthy patients and those with rheumatoid arthritis using a mass cytometry dataset. Furthermore we show that the PDF models can be used to predict whether a patient has RA or does not have RA with an estimated accuracy of 92.86%.

3.1 Sliced-Distributions

In this section we define the set of Sliced Distributions which we will use to represent the joint probability distribution of random variables. The general form of the Probability Density Function (PDF) of a SD is as follows.

Definition 7. *Let $\delta \in \mathbb{R}^n$ be a random variable with support $\Delta \subset \mathbb{R}^n$, $Z : \mathbb{R}^n \to \mathbb{R}^q$ with $q > n$, and $p_f : \mathbb{R}^q \times \mathbb{R}^k \to \mathbb{R}$ with support Δ_Z be a given probability density*

function. Then the sliced PDF is given by,

$$p_{SD}(\delta) = \begin{cases} \frac{1}{c} p_f(Z(\delta); \theta) & \text{for all } \delta \in \Delta \\ 0 & \text{otherwise.} \end{cases}$$

where $c = \int_\Delta p_f(Z(x))dx$ *and* θ *represents parameters of the probability density function.*

If Z is an injective non-surjective function then the set,

$$S_Z := \{z \in \mathbb{R}^q | z \neq Z(\delta) \text{ for any } \delta \in \Delta\}$$

is non-empty and the Sliced-Distribution, $p_{SD}(\delta)$, is a slice of the PDF $p_f(z)$ on Δ_Z / S_Z where S_Z is "sliced" out of the distribution. We classify Sliced Distributions by the function Z which lifts the data to a higher dimensional space, and by the distribution in the higher dimensional space, p_f.

To further illustrate the connection between the functions p_f and Z, we define the physical and feature spaces as follows.

Definition 8. *For a given random vector* $\delta \in \mathbb{R}^q$*, a function* $Z : \mathbb{R}^n \to \mathbb{R}^q$ *and corresponding PDF* p_f *with support* $\Delta_Z \subset \mathbb{R}^q$*, we say the* **physical space** *is* $\mathbb{R}^n$ *and contains samples* $\delta^{(i)}$ *of the random vector* δ *and the* **feature space** *is* $\mathbb{R}^q$ *and contains transformed samples of the random vector* $z^{(i)} := Z(\delta^{(i)})$*.*

Therefore p_f is a joint density function in the feature space, while p_{SD} is a joint density function in the physical space.

3.2 The Set of Sliced-Exponential Random Variables

In this section we propose the set of Sliced-Exponential (SE) random variables and define two convex optimization problems that can be used to select the parameters

of the SE that best model the distribution of given data, $\mathcal{D} := \{\delta_i \in \mathbb{R}^n\}_{i=1}^m$. We make no assumptions on the data $\mathcal{D}$ except that it is drawn from some unknown Data Generating Mechanism (DGM).

We define the set of SE random variables as follows.

Definition 9. *Given bounded $\Delta \in \mathbb{R}^n$, λ, and d the* **Set of Sliced-Exponential PDFs** *on Δ is defined as $E_\Delta := \{f \in F_\Delta \mid f = f_\delta(\delta; \lambda, d, \Delta), \quad d \in \mathbb{N}\}$, where the PDF, $f_\delta(\delta; \lambda, d, \Delta)$ is given by*

$$f_\delta(\delta; \lambda, d, \Delta) = \begin{cases} e^{-\lambda^T Z_d(\delta) - log(c)} & \text{if } \delta \in \Delta \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

and $Z_d(\delta)$ are the monomials of degree less than d of δ and $c = \int_\Delta e^{-\lambda^T Z_d(\delta)} d\delta$.

In Theorem 11 we will show that E_Δ is dense in F_Δ, the set of all bounded PDFs. In addition we prove that the likelihood of the data is convex with respect to the λ parameter of the SE PDF. We use this fact in Subsections 3.2.2 and 3.2.3 to show that the MLE and WCE optimization problems are convex.

3.2.1 Properties of SE Random Variables

We will show in this subsection that the set of SE random variables is dense in the set of all random variables defined by bounded PDFs on a compact domain.

Lemma 10. *On compact Δ, for any $f \in F_\Delta$ and $\epsilon > 0$, there exists an $h \in L_1(\Delta)$ such that*

$$\int_\Delta \left| \frac{e^{h(\delta)}}{\int_\Delta e^{h(x)} dx} - f(\delta) \right| d\delta < \epsilon.$$

Proof. Since $f \in F_\Delta$ we have that there exists an $M < \infty$ such that $\sup_{\delta \in \Delta} f(\delta) \leq M$ and since Δ is compact $V_\Delta = \int_\Delta \mathbb{1}_\Delta(\delta) d\delta$ exists. Next for any $\alpha > 0$, define the

function

$$log^{(\alpha)}(x) = \begin{cases} log(x) & x > \alpha \\ log(\alpha) & x \leq \alpha, \end{cases}$$

and let $f_n(\delta) = log^{(\frac{1}{n})}(f(\delta))$ which is an L_1 function since $f(\delta)$ is L_1. Furthermore, we have that $\int_\Delta e^{f_n(\delta)} d\delta \leq \int_\Delta e^{f(\delta)} + \frac{1}{n} d\delta \leq 1 + \frac{1}{n} V_\Delta$ and therefore,

$$\sup_{\delta \in \Delta} \left| f(\delta) - \frac{e^{f_n(\delta)}}{\int_\Delta e^{f_n(\delta)} d\delta} \right| \leq \begin{cases} \max_{\delta \in \Delta} f(\delta) - \frac{f(\delta)}{1 + \frac{1}{n} V_\Delta}, & \text{for } f(\delta) > \frac{1}{n} \\ \frac{\frac{1}{n}}{1 + \frac{1}{n} V_\Delta} & \text{for } f(\delta) \leq \frac{1}{n} \end{cases}$$

In the first case, $\max_{\delta \in \Delta} f(\delta) - \frac{f(\delta)}{1 + \frac{1}{n} V_\Delta} = M - \frac{M}{1 + \frac{1}{n} V_\Delta}$, since M is the maximum value of $f(\delta)$ on Δ. Therefore we have that,

$$\int_\Delta \left| \frac{e^{f_n(\delta)}}{\int_\delta e^{f_n(x)} dx} - f(\delta) \right| d\delta \leq V_\Delta \max \left\{ M - \frac{M}{1 + \frac{1}{n} V_\Delta}, \frac{\frac{1}{n}}{1 + \frac{1}{n} V_\Delta} \right\} \tag{3.2}$$

$$\leq V_\Delta \max \left\{ \frac{M V_\Delta}{n + V_\Delta}, \frac{V_\Delta}{n + V_\Delta} \right\} \tag{3.3}$$

Let $N = \max \left\{ \frac{V_\Delta^2}{\epsilon} - V_\Delta, \frac{M V_\Delta^2}{\epsilon} - V_\Delta \right\}$. Then in the first case when $M \geq 1$,

$$\int_\Delta \left| \frac{e^{f_N(\delta)}}{\int_\Delta e^{f_N(x)} dx} - f(\delta) \right| d\delta \leq \frac{M V_\Delta^2}{N + V_\Delta} \leq \frac{M V_\Delta^2}{\frac{M V_\Delta^2}{\epsilon}} \leq \epsilon,$$

and in the second case when $M \leq 1$,

$$\sup_{\delta \in \Delta} \left| \frac{e^{f_N(\delta)}}{\int_\Delta e^{f_N(\delta)} d\delta} - f(\delta) \right| \leq \frac{V_\Delta}{N + V_\Delta} \leq \frac{V_\Delta}{\frac{V_\Delta}{\epsilon}} \leq \epsilon.$$

Therefore $h = f_N \in L_1(\Delta)$ completes the proof. □

We next apply Lemma 10 to show that the set of Sliced-Exponential PDFs is dense in the set of all bounded PDFs F_Δ.

Theorem 11. *For any $f \in F_\Delta$ and any $\epsilon > 0$ there exists an $h \in E_\Delta$ such that, $d_H^2(f, h) < \epsilon$.*

Proof. By Lemma 10 we have that for any $\epsilon > 0$ there exists a function $h \in L_1(\Delta)$ such that $\int_\Delta \left| \frac{e^{h(\delta)}}{\int_\Delta e^{h(x)}dx} - f(\delta) \right| d\delta < \epsilon$. Polynomial functions are dense in the set of continuous functions on compact sets, which are dense in $L_1(\Delta)$ [57], therefore there exists a polynomial $q_\epsilon(\delta)$ such that, $\int_\Delta \left| \frac{e^{q_\epsilon(\delta)}}{\int_\Delta e^{q_\epsilon(x)}dx} - f(\delta) \right| d\delta < \epsilon$.

Let $h(\delta) = q_\epsilon(\delta) - \log(\int_\Delta e^{q_\epsilon(\delta)} d\delta)$ then $h(\delta) = \frac{e^{q_\epsilon(\delta)}}{\int_\Delta e^{q_\epsilon(\delta)} d\delta}$ and $h(\delta) \in E_\Delta$. Therefore $\int_\Delta |h(\delta) - f(\delta)| \, d\delta < \epsilon$, and

$$d_H^2(h, f) \leq \int_\Delta |h(\delta) - f(\delta)| \, d\delta < \epsilon.$$

□

We will show in the numerical results that even SEs of finite degree can model the distribution of unknown DGMs better than other state of the art methods. Next we will develop an optimization problem for optimizing the λ parameter of the SE.

3.2.2 Solving the MLE Optimization Problem for Sliced-Exponentials

This subsection describes how to select an optimal parameter λ to model the distribution of a given set of data $\mathcal{D} := \{\delta^{(i)}\}_{i=1}^m$ by solving the MLE optimization problem for fixed values of Δ and d.

Maximizing the Likelihood of the Data

We first define the MLE problem with respect to the set of Sliced-Exponentials. We then define a convex optimization problem and show that it solves the MLE problem.

The likelihood of a given set of data $\mathcal{D}$, for a model f is,

$$L_f(\mathcal{D}) = \prod_{i=1}^m f_\delta(\delta^{(i)}), \tag{3.4}$$

and the solution of the MLE problem is the model f within a given set of PDFs that maximizes $L_f(\mathcal{D})$.

Recall that for a support set Δ, our Sliced-Exponential PDF can be defined as,

$$f_d(\delta;\lambda,\Delta) = \begin{cases} e^{-\lambda^T Z_d(\delta)-\log(c)} & \text{if } \delta \in \Delta \\ 0 & \text{otherwise.} \end{cases} \tag{3.5}$$

The MLE problem with respect to the SE set of random variables and data from a DGM $\mathcal{D} := \{\delta^{(i)}\}_{i=1}^m$, is to find the parameter λ_{ML} such that,

$$\lambda_{\text{ML}} = \arg\max_{\lambda\in\mathbb{R}^q} O_{d,\delta,s}(\lambda) = \prod_{i=1}^m e^{-\lambda^T Z_d(\delta^{(i)})-\log(c)}, \tag{3.6}$$

however since there is no known analytical solution for c we will calculate its value using Monte Carlo sampling.

First let $\mathcal{S} := \{s^{(i)}\}_{i=1}^b$ contain b points uniformly sampled from the set Δ. Then we may estimate the normalization constant c using Monte Carlo sampling as,

$$c = \int_\Delta e^{-\lambda^T Z_d(\delta)} d\delta \approx \frac{v_\Delta}{b}\sum_{i=1}^b e^{-\lambda^T Z_d(s^{(i)})}, \tag{3.7}$$

where c becomes exact as b approaches infinity. We therefore estimate $\log(c)$ as,

$$\log(\tilde{c_b}(\lambda,d,\mathcal{S})) = \log(v_\Delta) - \log(b) + \log\left(\sum_{i=1}^b e^{-\lambda^T Z_d(s^{(i)})}\right). \tag{3.8}$$

We will discuss how many samples, b to select for accurate estimation of the normalization constant in Appendix A.1 and further analysis can be found in [40].

Now we may simplify the objective function $O_{d,\delta,s}(\lambda)$ as,

$$\begin{aligned} O_{d,\delta,s}(\lambda) &:= \prod_{i=1}^m e^{-\lambda^T Z_d(\delta^{(i)})-\log(\tilde{c_b}(\lambda,d,s))} \\ &= -\sum_{i=1}^m \log\left(\tilde{c_b}(\lambda,d,s)\right) + \lambda^T Z_d(\delta) \\ &= \sum_{i=1}^m -\log\left(\frac{v_\Delta}{b}\sum_{i=1}^b e^{-\lambda^T Z_d(s^{(i)})}\right) - \lambda^T Z_d(\delta) \\ &= \sum_{i=1}^m -\log\left(\frac{v_\Delta}{b}\right) - \log\left(\sum_{i=1}^b e^{-\lambda^T Z_d(s^{(i)})}\right) - \lambda^T Z_d(\delta). \end{aligned} \tag{3.9}$$

Since the term $m \log(\frac{v_\Delta}{b})$ is constant, minimizing the following expression,

$$\min_{\lambda \in \mathbb{R}^q} \left\{ \sum_{i=1}^{m} \log \left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})} \right) + \lambda^T Z_d(\delta^{(i)}) \right\}, \tag{3.10}$$

is equivalent to maximizing equation (3.9) and thus returns the solution to the MLE problem.

Next we will prove that this optimization problem is convex.

The MLE Optimization Problem is Convex

Here we will find the gradient and Hessian of the objective function of Optimization Problem (3.6). For a given degree d, data set δ, and sampled set, S, the objective function of Optimization Problem (3.9) is

$$O_{d,\delta,s}(\lambda) := \sum_{j=1}^{m} \log \left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})} \right) + \lambda^T Z_d(\delta^{(j)}). \tag{3.11}$$

The gradient and Hessian of this objective function with respect to λ is therefore a sum of the gradient and Hessian of functions of the following form,

$$f(\lambda, s, \delta) = \log \left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})} \right) + \lambda^T Z_d(\delta). \tag{3.12}$$

The partial derivative of $f(\lambda, s, \delta)$ with respect to each element in λ is

$$\frac{\partial f}{\partial \lambda_j} = \frac{\sum_{i=1}^{b} -Z_d^j(s^{(i)}) e^{-\lambda^T Z_d(s^{(i)})}}{\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}} + Z_d^j(\delta) \tag{3.13}$$

where $Z_d^j(s^{(i)})$ is the jth element of the monomial basis of the ith datum in s. The element corresponding to the k'th row and j'th column of of the Hessian can be calculated as,

$$H_{k,j} = \frac{\left(\sum_{i=1}^{b} V_i^{(k)} V_i^{(j)} E_i\right)\left(\sum_{i=1}^{b} E_i\right) - \left(\sum_{i=1}^{b} V_i^{(k)} E_i\right)\left(\sum_{i=1}^{b} V_i^{(j)} E_i\right)}{\left(\sum_{i=1}^{b} E_i\right)^2} \tag{3.14}$$

where $E_i = e^{-\lambda^T Z_d(s^{(i)})}$ and $V_i^{(j)} = Z_d^j(s^{(i)})$.

Next we will prove that the Hessian matrix is positive semi-definite and, thus, the function $f(\lambda, s, \delta)$ in Eq. (3.12) is convex with respect to λ.

Lemma 12. *For $V \in \mathbb{R}^{b\times q}$ where $V^{(j)} \in \mathbb{R}^b$ is the jth column vector in the matrix V, and $E \in \mathbb{R}^{q+}$ where E_i is the ith element in E and $s = \sum_{i=1}^b E_i$, then the Hessian matrix $H \in \mathbb{R}^{q\times q}$ is defined elementwise as,*

$$H_{k,j} = \frac{\left(\sum_{i=1}^b V_i^{(k)} V_i^{(j)} E_i\right)\left(\sum_{i=1}^b E_i\right) - \left(\sum_{i=1}^b V_i^{(k)} E_i\right)\left(\sum_{i=1}^b V_i^{(j)} E_i\right)}{\left(\sum_{i=1}^b E_i\right)^2}$$

is positive semi-definite.

Proof. To prove that H is positive semi-definite and since $\left(\sum_{i=1}^b E_i\right)^2$ is a positive constant we will find a decomposition $H\left(\sum_{i=1}^b E_i\right)^2 = RR^T$ which proves H is positive semi-definite.

Let $\beta \in \mathbb{N}^{\binom{b}{2}\times 2}$ be every unique combination of $i, l \in \{1, 2, \cdots, b\}$ without replacement. Then let $r_i^k = \sqrt{E_{\beta_{i,1}}}(V_{\beta_{i,1}}^{(k)} - V_{\beta_{i,2}}^{(k)})\sqrt{E_{\beta_{i,2}}}$ which exists because $E_i \geq 0$ for all $i = 1, ..., b$. Then we have the following relationship,

$$\begin{aligned}
\left(\sum_{i=1}^b E_i\right)^2 H_{k,j} &= \left(\sum_{i=1}^b V_i^{(k)} V_i^{(j)} E_i\right)\left(\sum_{i=1}^b E_i\right) - \left(\sum_{i=1}^b V_i^{(k)} E_i\right)\left(\sum_{i=1}^b V_i^{(j)} E_i\right) \\
&= \sum_{i=1}^b \left(\sum_{l=i+1}^b E_i (V_i^{(k)} - V_l^{(k)})(V_i^{(j)} - V_l^{(j)}) E_l\right) \\
&= \sum_{i=1}^b \left(\sum_{l=i+1}^b \sqrt{E_i}(V_i^{(k)} - V_l^{(k)})\sqrt{E_l}\sqrt{E_i}(V_i^{(j)} - V_l^{(j)})\sqrt{E_l}\right) \\
&= \left(r^k\right)^T r^j
\end{aligned}$$

Now define, $R^T = \begin{bmatrix} r^{(1)} & r^{(2)} & \cdots & r^{(q)} \end{bmatrix}$ and we have that $H = \frac{RR^T}{\left(\sum_{i=1}^b E_i\right)^2}$, implying that H is itself positive semi-definite since RR^T is positive semi-definite and $\left(\sum_{i=1}^b E_i\right)^2$ is a positive scalar. □

We have now proven that the Hessian matrix is positive semi-definite.

Theorem 13. *Optimization Problem* (3.6) *is convex.*

Proof. The objective function of Optimization Problem (3.6) is a positive summation of functions $f(\lambda, s, \delta) = \log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right) + \lambda^T Z_d(\delta)$ which have a positive semi-definite Hessian matrix with respect to λ as proven in Lemma 12. Since positive sums of convex functions are convex Optimization Problem (3.6) is convex. □

To demonstrate the SEs with parameters optimized by solving the MLE problem, we will next model the distribution of a publicly available data set using Optimization Problem (3.9).

Example 1 For this example we use the publicly available Iris [55] data set, $\mathcal{D} := \{\delta^{(i)} \in \mathbb{R}^4\}_{i=1}^{150}$. For reference, Fig. 3.1 shows a scatterplot of the Iris dataset after it has been scaled to fit within a hyper-cube centered at 0 with side lengths of 1. We have selected the Iris dataset to use as an example since it is clearly multi-modal, and does not fall into any known distribution.

In Fig. 3.2 we plot SE PDFs fit to lower-dimensional subsets of the Iris dataset. Along the diagonal we plot the marginal PDFs of each variate δ_1 through δ_4 using degree 8 SEs. In each of the non-diagonal subplots we plot the SE PDFs describing two of the variates using degree 6 SEs. These plots show that low degree Sliced-Exponentials may model a wide variety of random variables.

To show that SEs can characterize the multi-modal dependencies in higher dimensional data sets we train a fourth degree SE on the Iris data set. Using slice sampling we generate and plot 150 simulated data points from the trained SE in Fig. 3.3. For comparison the log likelihood value of a normal distribution fit to this data is 596.4, compared to a log likelihood value of 725.8 for the SE. This is a significant increase

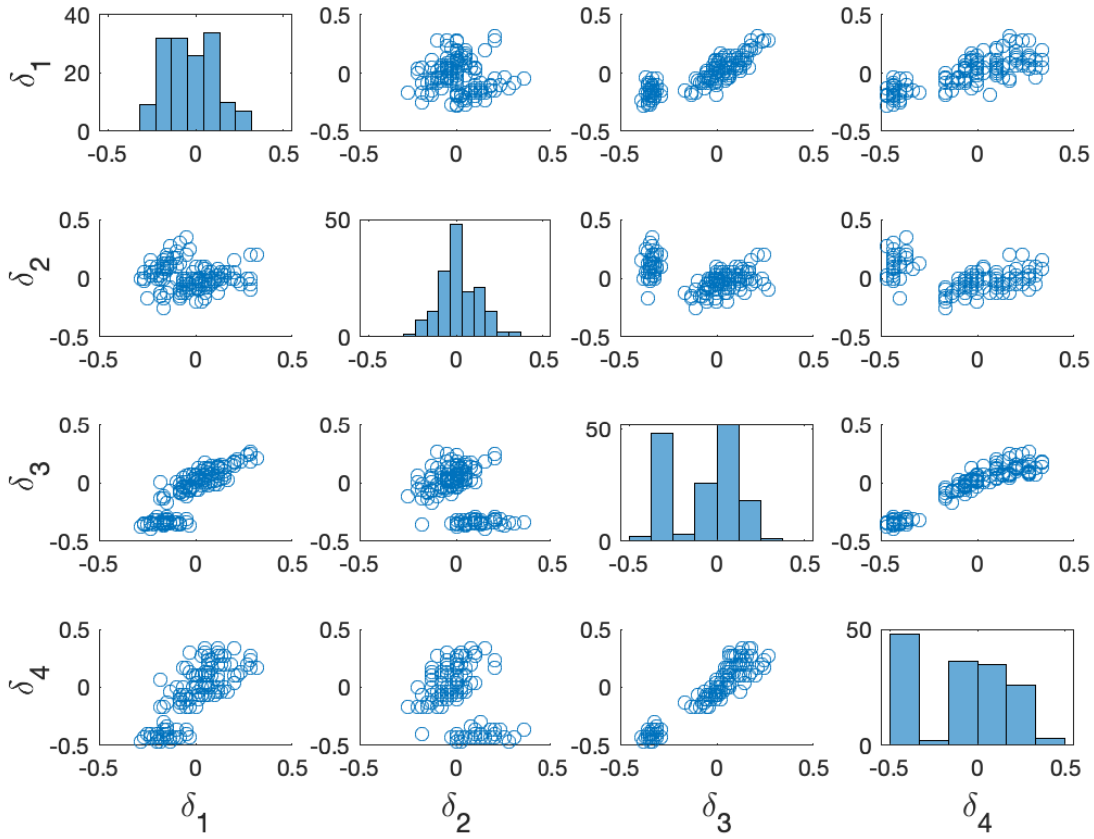


Figure 3.1: Subplots along the jth diagonal element show a histogram of measured δ_j data, while subplots corresponding to the jth and ith position show a scatterplot of measured data of δ_j versus δ_i for a rescaled version of the Iris dataset [55].

in the log likelihood of 19.57%.

Next we consider the WCE problem for generating optimal parameters λ.

3.2.3 A Convex Optimization Problem to Solve the Worst Case Estimation (WCE) Problem

Optimizing the SE with respect to the worst case likelihood estimation yields SE PDFs whose sets enclose the data more tightly than SEs optimized by solving the maximum likelihood estimation problem. We first formally define the WCE problem

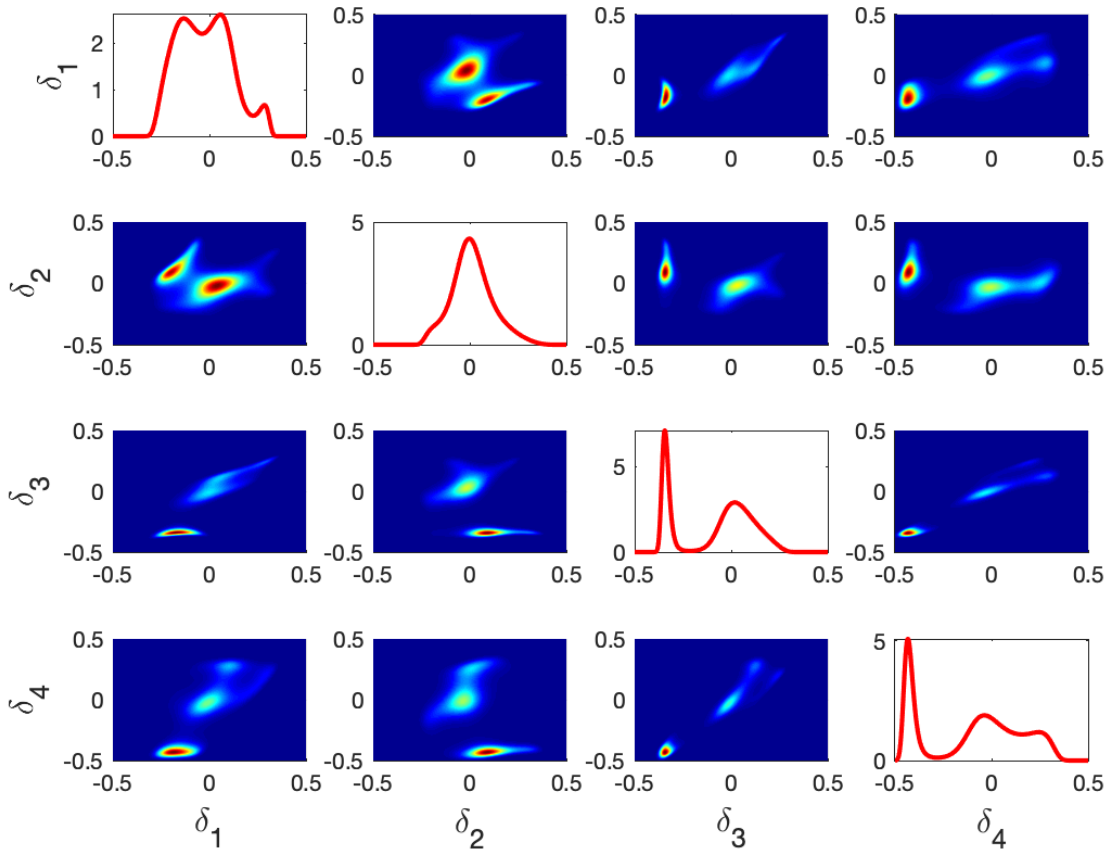


Figure 3.2: Sliced-Exponential PDFs fit using Optimization Problem (3.10), subplots along the jth diagonal element show a Sliced-Exponential PDF of degree 8 fit to measured δ_j data, while subplots corresponding to the jth and ith position show a Sliced-Exponential PDF of measured data of δ_j versus δ_i of the Iris dataset [55].

and show that we may find the value of λ, for a given degree d, which maximizes the worst case likelihood on a set of data points, $\mathcal{D}$, by solving a convex optimization problem. The selection of an optimal degree d can be identified using cross-validation techniques such as those proposed later in Section 3.5.1.

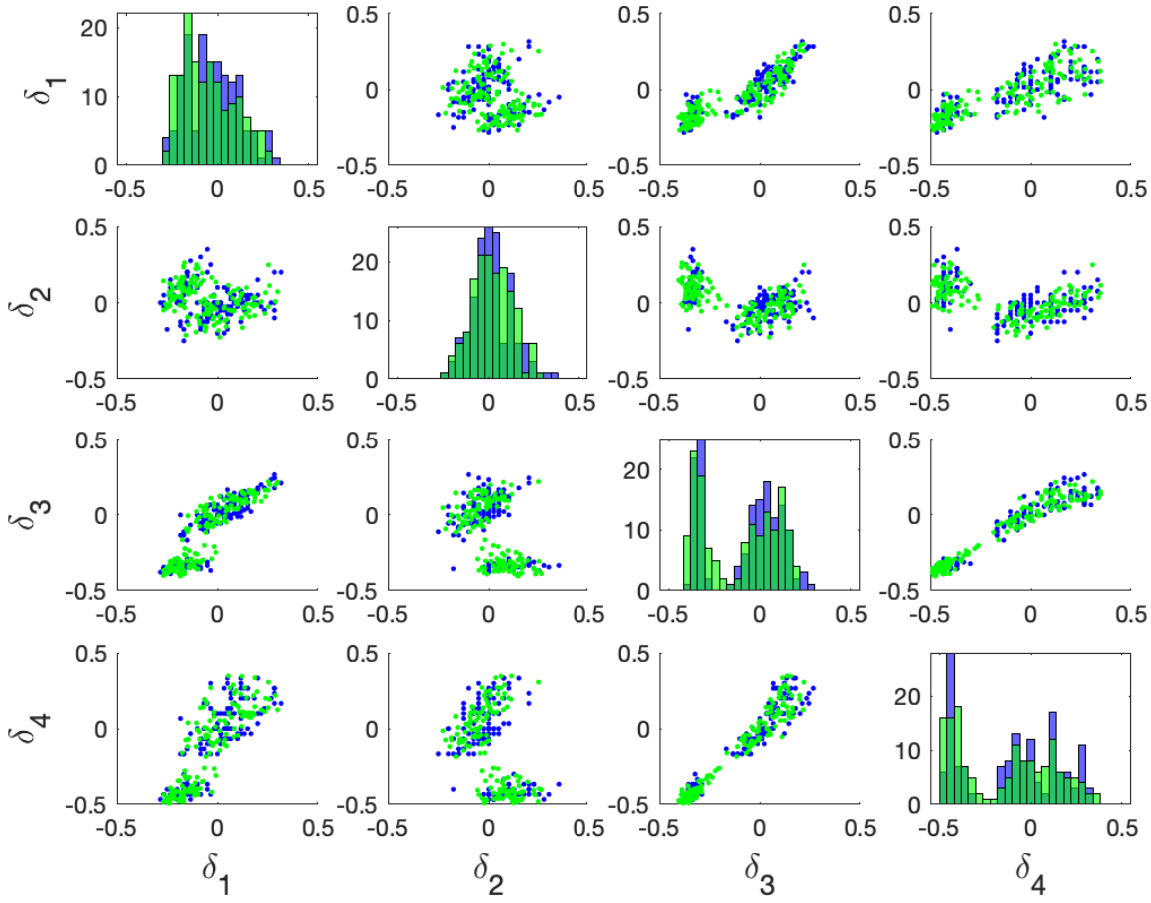


Figure 3.3: Plot of a rescaled version of the Iris dataset [55] (blue) and data sampled from the SE PDF (green) optimized using the solution of the MLE optimization problem. Subplots along the jth diagonal element show a histogram of the sampled δ_j data, while subplots corresponding to the jth and ith position show a scatterplot of sampled data of δ_j versus δ_i.

Solving the WCE Optimization Problem for Sliced-Exponentials

We first derive the WCE problem with respect to the set of Sliced-Exponentials. We then define a convex optimization problem and show that it solves the MLE problem. The worst case likelihood of a model f on a given set of data $\mathcal{D} := \{\delta^{(i)}\}_{i=1}^m$ is defined

as,

$$W_f(\mathcal{D}) = \min_{i \in \{1,\cdots,m\}} f_\delta\left(\delta^{(i)}\right),$$

and finding the model f from a set of models that maximizes $W_f(\mathcal{D})$ is called the Worst Case Estimation (WCE) problem.

For a support set Δ, recall that the Sliced-Exponential PDF is defined as,

$$f_\delta(\delta;\lambda) = \begin{cases} e^{-\lambda^T Z_d(\delta) - \log(c)} & \text{if } \delta \in \Delta \\ 0 & \text{if } \delta \notin \Delta. \end{cases} \tag{3.15}$$

Since there are no analytical solutions for the normalization constant of the PDF, we will use the Monte Carlo based numerical integration constant defined in Eq. (3.8).

The solution of the WCE problem for the SE set is the optimal parameter λ_{WC}, given by,

$$\lambda_{\text{WC}} := \operatorname*{argmax}_{\lambda} \min_{i \in \{1,\cdots,m\}} e^{-\lambda^T Z_d(\delta^{(i)}) - \log(\tilde{c_b}(\lambda,d,s))}. \tag{3.16}$$

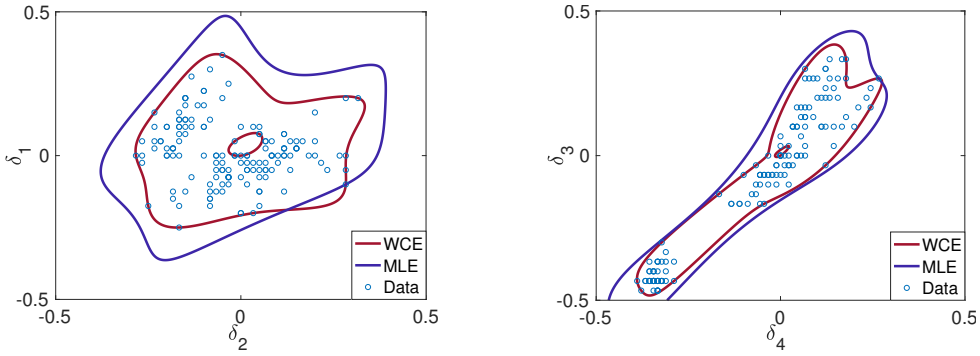
We may simplify the objective function as follows.

$$\begin{aligned} e^{-\lambda^T Z_d(\delta^{(i)}) - \log(\tilde{c_b}(\lambda,d,s))} &= -\log\left(\tilde{c_b}(\lambda,d,s)\right) - \lambda^T Z_d(\delta^{(i)}) \\ &= -\log\left(\frac{v_\Delta}{b}\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right) - \lambda^T Z_d(\delta^{(i)}) \\ &= -\log\left(\frac{v_\Delta}{b}\right) - \log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right) - \lambda^T Z_d(\delta^{(i)}). \end{aligned} \tag{3.17}$$

A convex lower bound on Eq. (3.17) may be implemented by constraining a decision variable, t as follows,

$$t \leq -\log\left(\frac{v_\Delta}{b}\right) - \log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right) - \lambda^T Z_d\left(\delta^{(i)}\right) \quad \forall i \in \{1,\cdots,m\}, \tag{3.18}$$

and maximizing t subject to this constraint maximizes the worst case likelihood on the data in $\mathcal{D}$ as defined in Eq. (3.16) for the degree d SE.



(a) Level sets of SE PDFs that contain all of the first and second variates of the Iris data set where λ was optimized by solving the WCE (red) or the MLE (blue) optimization problem.

(b) Level sets of SE PDFs that contain all of the third and fourth variates of the Iris data set where λ was optimized by solving the WCE (red) or the MLE (blue) optimization problem.

Figure 3.4: Level sets of SE PDFs fit using Optimization Problem (3.19) that contain bi-variate subsets of the Iris data set (red) compared to level sets of the SE PDFs fit using Optimization Problem (3.10) to the same data (blue).

Therefore, solving the optimization problem,

$$\min_{t\in\mathbb{R},\lambda\in\mathbb{R}^q}\left\{-t \mid t+\log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right)+\lambda^T Z_d(\delta^{(i)})\leq 0 \ \ \forall \ \ i\in\{1,\cdots,m\}\right\}, \tag{3.19}$$

is equivalent to maximizing equation (3.17) minus the constant term $\log(\frac{v_\Delta}{b})$.

Theorem 14. *Optimization Problem* (3.19) *is convex.*

Proof. The objective function of Optimization Problem (3.6) is affine and thus convex. The function $f(\lambda,s,\delta)=\log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right)+\lambda^T Z_d(\delta)$ has a positive semi-definite Hessian matrix with respect to λ as proven in Lemma 12 and is thus convex. Since positive sums of convex functions are convex each constraint, $t+\log\left(\sum_{i=1}^{b} e^{-\lambda^T Z_d(s^{(i)})}\right)+\lambda^T Z_d(\delta^{(i)})\leq 0$ is convex and Optimization Problem (3.6) is therefore a convex optimization problem. □

For any SE with PDF f_δ, we define the level set of smallest volume that contains all of the points in $\mathcal{D}$ as,

$$H_\beta := \{\delta \quad | \quad f_\delta(\delta;\lambda) \geq \beta\}, \tag{3.20}$$

for $\beta = W_{f_\delta}(\mathcal{D})$. For any value of $\beta \in \mathbb{R}^+$, $H_\beta := \{\delta \quad | \quad f_\delta(\delta;\lambda) \geq \beta\}$ is defined by a polynomial inequality and is thus a semi-algebraic set. This can be seen with the following algebraic manipulation,

$$0 \leq \log(f_\delta(\delta;\lambda)) - \log(\beta) \leq -\lambda^T Z_d(\delta) - \log(c) - \log(\beta) \leq -\lambda^T Z_d(\delta) - \kappa,$$

where $\kappa = \log\left(\frac{c}{\beta}\right)$ and H_β is equivalent to the semi-algebraic set $\{\delta | -\lambda^T Z_d(\delta) - \kappa \geq 0\}$.

To demonstrate the advantages of the SEs optimized by solving the WCE using Optimization Problem (3.19) we will analyze the data-enclosing level sets of the SE fit to a publicly available data set.

Example 2 For this example we also use the publicly available Iris [55] data set, $\mathcal{D} := \{\delta^{(i)} \in \mathbb{R}^4\}_{i=1}^{150}$. For reference, Fig. 3.1 shows a scatterplot of the Iris dataset after it has been scaled to fit within a hyper-cube centered at 0 with side lengths of 1.

In Fig. 3.4 we plot the level set $H_{W_{f_\delta}(\mathcal{D})}$, which is the set of minimal volume that contains all of the Iris dataset, for SEs optimized by solving the MLE and WCE optimization problem. Fig. 3.4 shows the bi-variate PDFs of the ML SEs corresponding to the ith and jth parameters. Recall that as in Fig. 3.2 these sets correspond to lower-dimensional subsets of the iris data set. Notice that the data-enclosing set for a four dimensional SE can't be shown in the same format. However, uniformly distributed samples over such a set are shown in figure 3.5.

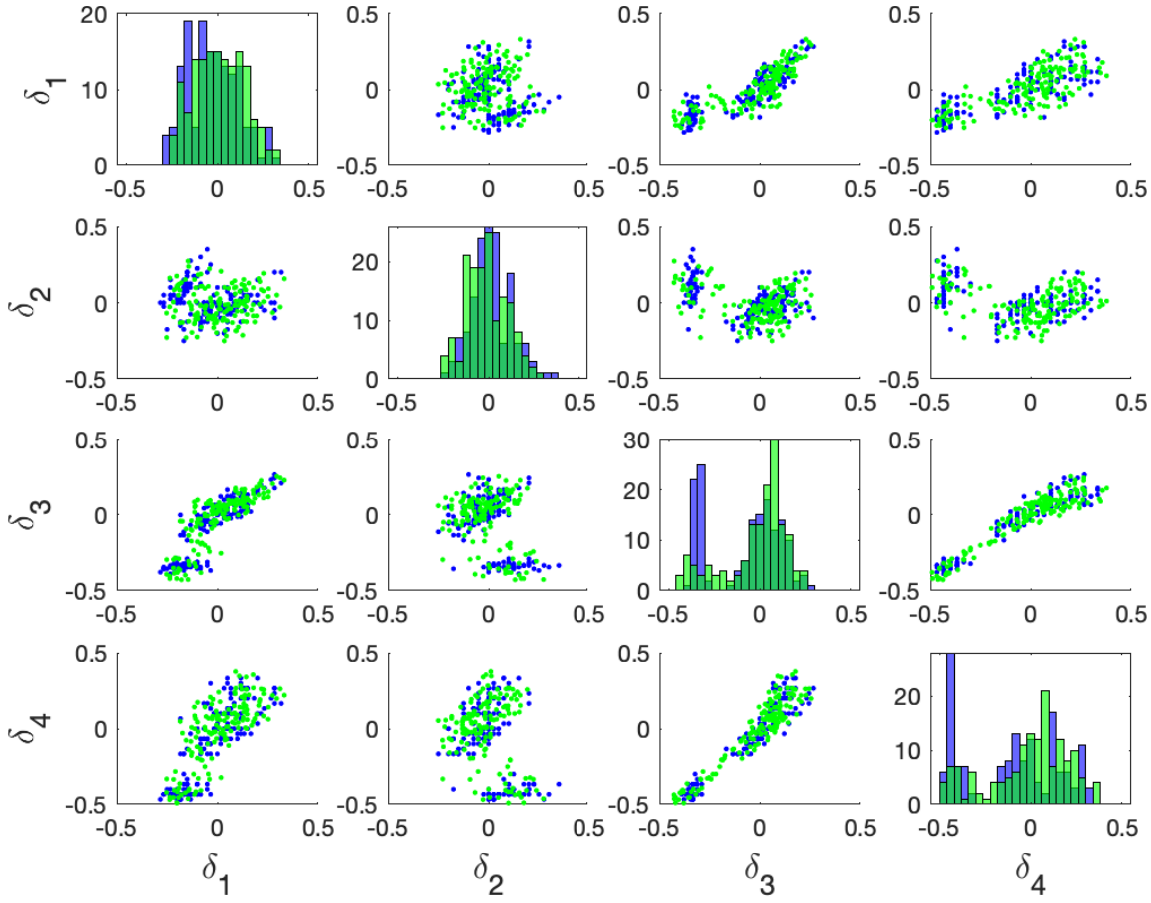


Figure 3.5: Plot of a rescaled version of the Iris dataset [55] (blue) and data uniformly sampled from the level set $H_{W_{f_\delta}(\mathcal{D})}$ of an SE PDF (green) optimized by solving the WCE optimization problem. Subplots along the jth diagonal element show a histogram of the sampled δ_j data, while subplots corresponding to the jth and ith position show a scatterplot of sampled data of δ_j versus δ_i.

We calculated the volume of the minimum volume ellipsoid that contains the Iris data to be 0.0164, whereas the volume of the sliced-exponential level set is 0.0036. This implies that the volume of the sliced-exponential level set was 128% less than volume of the minimum volume ellipsoid.

The level sets of SE PDFs can be used with polynomial optimization methods such as sum-of-squares optimization [131, 123] to carry out formal robustness analysis and

optimization, such as the polynomial level sets used in [50, 25]. In addition, the reliability of such sets, which bound the probability of future samples drawn from the same DGM falling within the set, can be assessed using scenario theory as in [41].

We perform a numerical analysis of the effect of the number of samples used to calculate the numerical integration constant, and the effect of increasing the number of samples or dimension of $\mathcal{D}$ on the solution to the MLE and WCE problems in Appendix A.1.

Next we consider a closely related set of sliced distributions called sliced-normals.

3.3 Sliced-Normal Distributions

Here we define a subset of Sliced Distributions called the Sliced-Normal (SN) distribution. As in the previous section we first prove that the set of Sliced-Normal distributions is dense in the set of all bounded PDFs F_Δ.

Unfortunately for an equivalent degree, SNs have more parameters than SEs and are thus more expensive to fit to data. Therefore we propose solving optimization problems closely related to the MLE and WCE optimization problems that have a significantly reduced computational complexity when compared to the Sliced-Exponential MLE and WCE optimization problems.

In the feature space Sliced-Normal Distributions are multivariate normal distributions where the transformation $Z(\delta)$ returns the monomial basis of δ. Specifically we can write a degree bounded SN as,

$$p_{SN}^{(d)}(\delta) = \begin{cases} \frac{1}{c} e^{-\frac{1}{2}(Z_d(\delta)-\mu)^T P (Z_d(\delta)-\mu)} & \text{for all } \delta \in \Delta \\ 0 & \text{otherwise.} \end{cases} \tag{3.21}$$

where $c = \int_\Delta e^{-\frac{1}{2}(Z_d(\delta)-\mu)^T P (Z_d(\delta)-\mu)} dx$, $P > 0$ is the positive definite precision matrix,

and the set of SN distributions is,

$$F_N := \{p_{SN}^{(d)}(\delta) \mid d \in \mathbb{N}\}. \tag{3.22}$$

The SN distributions differ from the SE distributions in that the polynomial argument is a sum-of-squares polynomial and is thus globally positive. Sum-of-squares polynomials have more parameters than the polynomial parameterization used with SEs.

3.3.1 Properties of Sliced-Normals

In this subsection we show that the SN set is dense in the set of all bounded PDFs F_Δ. We begin by showing that for any SE distribution we may find an equivalent SN distribution on Δ. Therefore the set of SN distributions must, like the set of SE distributions, be dense in F_Δ.

Lemma 15. *For any polynomial function, $p(\delta)$ and $\epsilon > 0$ there exists a sum-of-squares polynomial function $q(\delta)$ such that,*

$$\int_\Delta \left| \frac{1}{c_p} e^{(p(\delta))} - \frac{1}{c_q} e^{q(\delta)} \right| d\delta < \epsilon,$$

where $c_p = \int_\Delta e^{p(\delta)} d\delta$ and $c_q = \int_\Delta e^{q(\delta)} d\delta$.

Proof. Let $m = \min_{\delta \in \Delta} p(\delta)$, which must exist since polynomials are bounded on compact sets. Then we have that,

$$\begin{aligned} \frac{1}{c_p} e^{p(\delta)} &= \frac{1}{e^{-m} c_p} e^{-m} e^{p(\delta)} \\ &= \frac{1}{e^{-m} c_p} e^{p(\delta)-m} \\ &= \frac{1}{c_f} e^{-f(\delta)}, \end{aligned}$$

where $f = p(\delta) - m$ and $c_f = \int_\Delta e^{f(\delta)} d\delta$. In addition $f(\delta)$ is nonnegative on Δ, and the set of SOS polynomials is dense in the set of nonnegative polynomials on any compact

set [97]. Therefore there exists an SOS polynomial $q(\delta)$ such that $q(\delta) = f(\delta) = p(\delta)$ for all $\delta \in \Delta$ and we have that

$$\int_{\Delta} \left| \frac{1}{c_p} e^{p(\delta)} - \frac{1}{c_q} e^{q(\delta)} \right| d\delta < \epsilon$$

□

Therefore on any compact Δ the SN distributions is dense in the set of SE distributions.

Lemma 16. *The set of SN PDFs N_Δ is dense in the set of bounded PDFs F_Δ.*

Proof. From Lemma 15 we have that the set of sliced-normal PDFs N_Δ are dense in the set of sliced-exponential PDFs E_Δ and from Theorem 11 we have that the set of sliced-exponentials is dense in F_Δ. □

While both the SE and SN sets are dense in F_Δ, the set of degree bounded SEs has fewer decision variables for a given degree d polynomial. Therefore when solving the MLE and WLE optimization problems exactly it is more computationally efficient to solve these problems using the SE set. However, there exist a number of computationally efficient solutions for finding SNs using metrics related to the MLE and WLE problems that cannot be applied to the set of SEs. These solutions often result in comparable models but at significantly less computational expense.

3.4 Efficient Modeling Approaches using Approximate Solutions

The MLE and WLE optimization problems in the physical space for the SN and SE distributions can be computationally expensive to solve. This is primarily due to the Monte Carlo sampling method which might require millions of samples to accurately compute the normalization constant for each computation of the gradient,

hessian or objective value. We next develop computationally inexpensive methods by solving the MLE optimization problem in feature space.

3.4.1 Solving the MLE Optimization Problem in Feature Space

In feature space an SN PDF is equivalent to a normal PDF where there are analytical solutions for the mean, μ and precision matrix P and thus Monte Carlo sampling is not required to optimize these parameters.

Specifically, the MLE optimization problem in feature space is formulated as

$$\max_{P\in\mathbb{R}^{q\times q},\, \mu\in\mathbb{R}^q} \left\{ \log \prod_{\delta\in\mathcal{D}} \frac{e^{-\frac{(Z_d(\delta)-\mu)^T P(Z_d(\delta)-\mu)}{2}}}{(2\pi)^{q/2}\sqrt{\det(P^{-1})}} : P \succeq 0 \right\}, \tag{3.23}$$

because a SN distribution is a multivariate normal distribution in the feature space and the normalization constant for a multivariate Gaussian is $c = (2\pi)^{q/2}\sqrt{\det(P^{-1})}$.

This optimization problem is a special case of optimization problems of the form

$$\max_{P\in\mathbb{R}^{q\times q},\, \mu\in\mathbb{R}^q} \left\{ \log \prod_{i=1}^{m} \frac{e^{-\frac{(h_i-\mu)^T P(h_i-\mu)}{2}}}{c\sqrt{\det(P^{-1})}} : P \succeq 0 \right\}. \tag{3.24}$$

Such optimization problems admit an analytic solution, as can be found in, e.g. [67]. Specifically, for given $\{h_i\}$, we have that the solution is $\mu^* = \frac{1}{m}\sum_{i=1}^m h_i$ and $P^* = \Sigma^{-1}$, where $\Sigma = \frac{1}{m}\sum_{i=1}^m h_i h_i^T$. Thus μ is the empirical mean and P is the empirical precision matrix of the data in feature space.

Unfortunately the optimal P^* and μ^* selected by optimizing a multivariate Gaussian distribution in the feature space may have a poor likelihood value in the physical space. Thus we next consider a second method to improve the SN parameters to increase the likelihood of the data.

3.4.2 Rescaling the Feature Space Precision Matrix

In this section, we propose a convex optimization problem that maximizes the likelihood of the SN for a given data sequence by scaling the suboptimal precision matrix P obtained in the prior subsection. In this case, we assume that a degree d has been selected and we are given previously selected values of the hyperparameters $P^* \in \mathbb{R}^{q\times q}$ and $\mu^* \in \mathbb{R}^q$, presumably found using Optimization Problem (3.23). We now consider SN 'candidates' of the following form,

$$p_{SN}^{(d)}(\delta;\gamma) = \begin{cases} \frac{1}{c}e^{-\gamma\frac{1}{2}(Z_d(\delta)-\mu^*)^T P^*(Z_d(\delta)-\mu^*)} & \text{for all } \delta \in \Delta \\ 0 & \text{otherwise.} \end{cases} \tag{3.25}$$

where P^* and μ^* are fixed, $\Delta \subset \mathbb{R}^n$ is the support set of the SN, and γ is the rescaling factor. This distribution can be cast as a SE distribution where the monomial basis has been replaced with a polynomial function $B(\delta) = \frac{1}{2}(Z_d(\delta) - \mu^*)^T P^*(Z_d(\delta) - \mu^*)$ thus implying $\lambda \in \mathbb{R}$ and this problem can be solved using the SE optimization methods defined in Section 3.2. Since there is only one scalar decision variable, this problem is computationally inexpensive compared to using a larger monomial basis such as in the full set of SN or SE distributions. If $\gamma^* > 0$, we have that P^* is still positive definite, thus making the polynomial a sum of squares polynomial and the distribution a SN distribution.

To demonstrate the SNs with parameters optimized by rescaling the MLE optimal feature space solution, we will next model the distribution of a publicly available data set using Optimization Problem (3.25).

Example 3 For this example we use the publicly available Iris [55] data set, $\mathcal{D} := \{\delta^{(i)} \in \mathbb{R}^4\}_{i=1}^{150}$. For reference, Fig. 3.1 shows a scatterplot of the Iris dataset after it

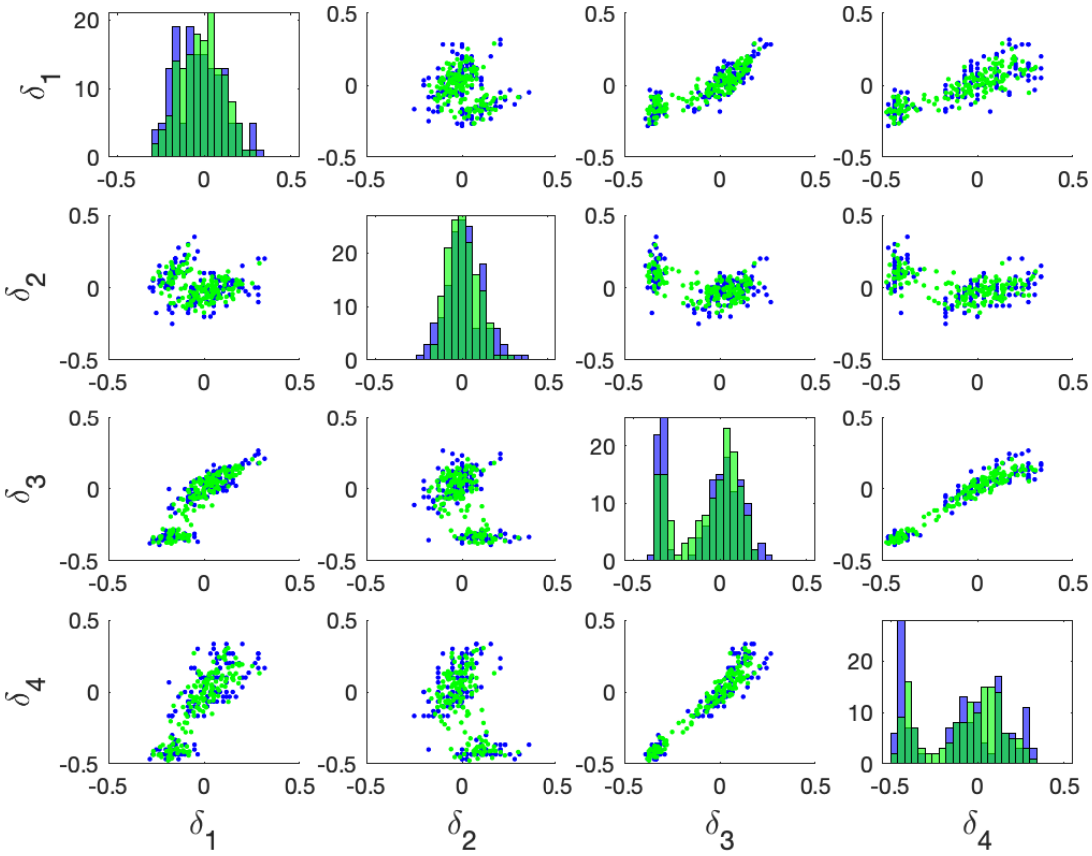


Figure 3.6: Plot of a rescaled version of the Iris dataset [55] (blue) and data sampled from the SN PDF (green) optimized using the rescaled precision matrix. Subplots along the jth diagonal element show a histogram of the sampled δ_j data, while subplots corresponding to the jth and ith position show a scatterplot of sampled data of δ_j versus δ_i.

has been scaled to fit within a hyper-cube centered at 0 with side lengths of 1. We have selected the Iris dataset to use as an example since it is clearly multi-modal, and does not fall into any known distribution.

To show that SNs can characterize the multi-modal dependencies in higher dimensional data sets we train a fourth degree SN on the Iris data set. Using slice sampling we generate and plot 150 simulated data points from the trained SN in Fig. 3.6. For

comparison the log likelihood value of a normal distribution fit to this data is 596.4, compared to a log likelihood value of 875.7 for the SN. This is a significant increase in the log likelihood of 37.94%.

We next consider approximate solutions to the WLE optimization problem.

3.4.3 Solving the WCE Optimization Problem in Feature Space

We may also compute approximate solutions to the WLE optimization problem for the set of SN distributions. First we will show that the WLE optimization problem for the set of degree 1 SN distributions is identical to the minimum volume ellipsoid problem as defined in [162, 161]. Then we use the solution of the minimum volume ellipsoid problem in the feature space to efficiently render SNs that are optimal in the feature space as an alternative to solving the computationally expensive WCE problem for SEs.

The minimum volume ellipsoid problem can be formulated as the following optimization problem [162],

$$\begin{aligned} \min_{M \in \mathbb{R}^{q \times q}} \quad & -\log(\det M) \\ \text{such that:} \quad & \left[\begin{smallmatrix} 1 \\ Z_d(\delta) \end{smallmatrix} \right]^T M \left[\begin{smallmatrix} 1 \\ Z_d(\delta) \end{smallmatrix} \right] \leq n \ \forall \ \delta \in \mathcal{D}, \quad \text{and} \quad M > 0. \end{aligned} \tag{3.26}$$

Optimization Problem (3.26) can be solved efficiently even for cases where the number of points $m > 100,000$ and $n > 50$ using methods from [161]. Note that, since the volume of the ellipse is not related to the first column and row of M an equivalent optimization problem is given by,

$$\begin{aligned} \min_{M \in \mathbb{R}^{q \times q}} \quad & -\log(\det M_{2:q,2:q}) \\ \text{such that:} \quad & \left[\begin{smallmatrix} 1 \\ Z_d(\delta) \end{smallmatrix} \right]^T M \left[\begin{smallmatrix} 1 \\ Z_d(\delta) \end{smallmatrix} \right] \leq q \ \forall \ \delta \in \mathcal{D}, \quad \text{and} \quad M > 0, \end{aligned} \tag{3.27}$$

where $M_{2:q,2:q}$ is the matrix M without the first row and column.

The solution to the WLE optimization problem in feature space on the other hand is given by,

$$P^*, Q^* = \underset{P \in \mathbb{R}^{q \times q}, Q \in \mathbb{R}^{q+1 \times q+1}}{\arg\max} \left\{ \log|P| - \alpha : \; z_i^T Q z_i \leq \alpha, \; P \succeq 0, Q \succeq 0, Q_{2:q,2:q} = P \right\}. \tag{3.28}$$

By a change of variable $R = \frac{q}{\alpha} Q$, we have that the objective function of the WLE optimization problem in feature space can be formulated as,

$$\begin{aligned} -\log|Q_{2:q,2:q}| + \alpha &= -\log\frac{\alpha^q}{q}|R| + \alpha \\ &= -\log\frac{\alpha^q}{q}|R_{2:q,2:q}| + \alpha \\ &= -(q)\log(\alpha) + (q)\log(q) - \log(|R_{2:q,2:q}|) + \alpha, \end{aligned}$$

and the constraint can be formulated as,

$$z_i^T R z_i \leq q.$$

Since α is unconstrained we find that $\alpha^* = \frac{1}{q}$ and that R^* can be found from,

$$R^* = \underset{R \in \mathbb{R}^{q \times q}}{\arg\min} \left\{ -\log|R_{2:q,2:q}| : \; z_i^T R z_i \leq q, \; R \succeq 0 \right\}, \tag{3.29}$$

which is identical to the minimum volume ellipsoid problem. Thus, if we solve the minimum volume ellipsoid problem with the data, z_i, we may recover the solution to the WLE optimization problem for a SN. Let R^* be the solution to the minimum volume ellipsoid problem and we may recover the solution to Optimization Problem (3.28) as $Q^* := \frac{R^*}{q^2}$.

Efficient algorithms developed specifically for the minimum volume ellipsoid problem such as those defined in [161], can therefore be used to find suboptimal solutions to the WLE optimization problem for SNs.

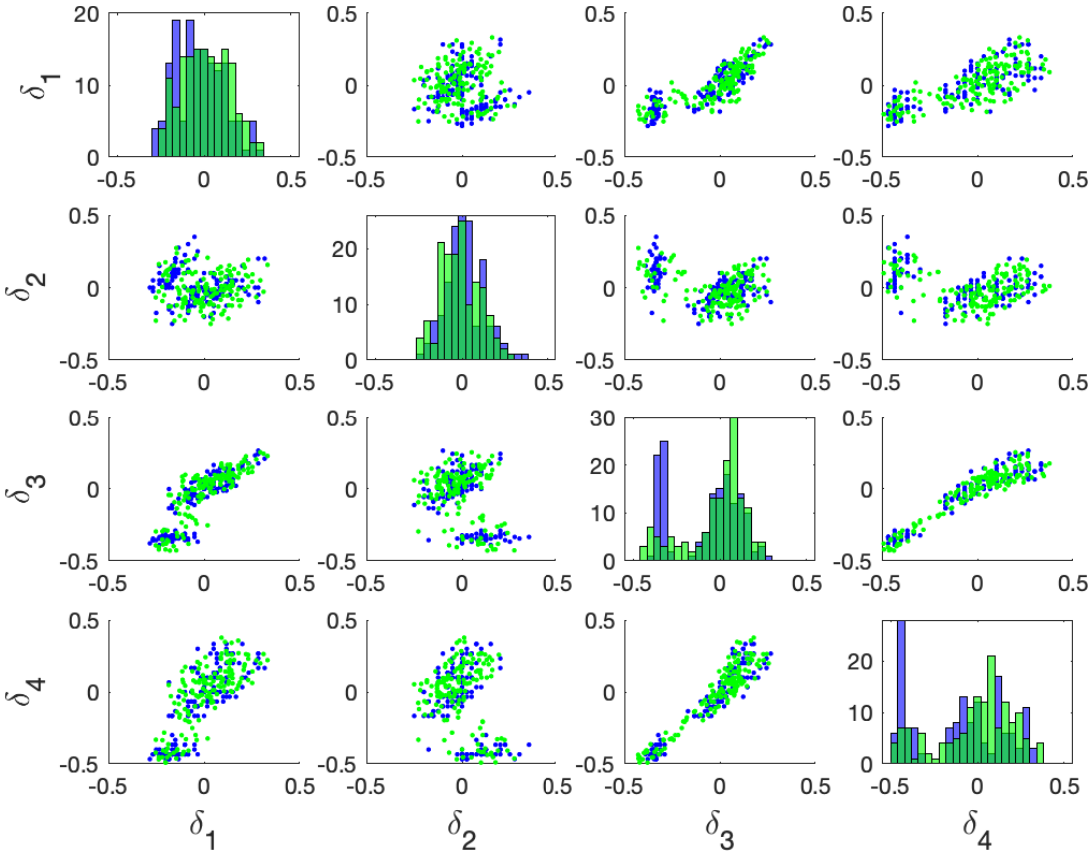


Figure 3.7: Plot of a rescaled version of the Iris dataset [55] (blue) and data uniformly sampled from the level set $H_{W_{f_\delta}(\mathcal{D})}$ of an SN PDF (green) optimized by solving the WCE optimization problem in feature space. Subplots along the jth diagonal element show a histogram of the sampled δ_j data, while subplots corresponding to the jth and ith position show a scatterplot of sampled data of δ_j versus δ_i.

Example 4 For this example we also use the publicly available Iris [55] data set, $\mathcal{D} := \{\delta^{(i)} \in \mathbb{R}^4\}_{i=1}^{150}$. For reference, Fig. 3.1 shows a scatterplot of the Iris dataset after it has been scaled to fit within a hyper-cube centered at 0 with side lengths of 1. To demonstrate the WLE method for SNs, we generate an optimal SN model by solving the WLE optimization problem in feature space. We then plot uniformly distributed samples over the level set of minimal volume, $H_{W_{f_\delta}(\mathcal{D})}$, using sliced sampling

in Fig. 3.7.

The volume of the minimum volume ellipsoid that contains the Iris data is 0.0164, whereas the volume of the level set of the sliced-normal is 0.0013. This implies that the volume of the sliced-normal level set was 170% less than the volume of the minimum volume ellipsoid.

Next we compare SDs to other distributions by modeling the distribution of a set of publicly available data sets and examining their performance on a test partition of the data.

3.5 Numerical Results of SD Optimization

First we compare the SDs to the sets of Gaussians and Gaussian mixture models. Then we apply the SDs to the problem of identifying cell populations that most differ between patients with, and without Rheumatoid Arthritis. By identifying the populations that differ the most between the two types of patients we will identify cell populations that are potentially correlated to the disease.

3.5.1 Numerical Accuracy of SD Optimization

In this subsection we will use the MLE and WCE optimization problems to find optimal SE and SN random variables that model the uncertainty in publicly available data sets. We use a separate set of testing points to compare the accuracy of the SD random variables to the set of Gaussian random variables and the set of Gaussian mixture models.

Methodology: Specifically we partition the data into a training, validation, and testing data partition. Some methods have regularization parameters that can be used to increase the generalization of the algorithm and avoid overfitting. For these

Table 3.1: The log likelihood (LL) of the uncertainty models on the test partition data points and the computation time (T) for the MLE MN, EM GMM, MLE SE, and WCE SE implementations. The data sets have dimension (n) and number of training data points (m).

Dataset	Model	LL	Time (s)
VOS	MLE MN	474.524 ± 31.363	0.001 ± 0.001
$n = 1$	EM GMM	452.452 ± 20.898	0.053 ± 0.024
$m = 5844$	MLE SE	498.924 ± 28.124	1.015 ± 0.042
	MLE SN	474.645 ± 30.892	0.231 ± 0.041
TV	MLE MN	-130.635 ± 47.948	0.002 ± 0.001
$n = 2$	EM GMM	4131.136 ± 13.554	3.459 ± 0.463
$m = 40973$	MLE SE	9461.196 ± 153.642	5.792 ± 0.766
	MLE SN	6372.847 ± 274.507	4.567 ± 0.106
BP	MLE MN	17838.586 ± 71.228	0.003 ± 0.001
$n = 3$	EM GMM	19853.676 ± 25.695	1.724 ± 0.299
$m = 51850$	MLE SE	33007.733 ± 517.990	34.396 ± 17.589
	MLE SN	26306.683 ± 327.225	1.923 ± 1.512
Iris	MLE MN	90.431 ± 8.091	0.000 ± 0.000
$n = 4$	EM GMM	76.037 ± 3.285	0.040 ± 0.019
$m = 127$	MLE SE	99.778 ± 10.862	6.878 ± 6.859
	MLE SN	87.894 ± 13.550	0.882 ± 0.424

methods the validation partition is used to select the regularization parameter, and then the method is retrained (with the selected regularization parameter) using the training and validation partition. Model performance is then analyzed on the testing partition of the data. We use the following data sets.

Data: In this section we use the data sets, VOS, TV[1], BP, and IR as can be found in [128, 119, 83, 55] in the UCI machine learning or OpenML databases.

To compare the proposed SEs and SNs to other methods we use the following implementations. We repeat all tests 5 times with different partitions of the training, validation, and test partitions and report the average and standard deviations of all values.

[**MLE SE**] The MLE optimization problem (Optimization Problem (3.6)) is used to select the optimal λ parameter of the SE. We use cross-validation to select the degree of the SE random variable which generated the largest likelihood on the validation partition of the data;

[**FS MLE SN**] The Feature Space (FS) based MLE optimization problem (Optimization Problem (3.25)) is used to select the suboptimal rescaled P and μ parameters of the SN. Optimal SNs can also be found using the techniques in [41], but will have a larger computation time than an MLE SE of the same degree and is thus omitted. We use cross-validation to select the degree of the SN random variable which generated the largest likelihood on the validation partition of the data;

[**WCE SE**] The WCE optimization problem (Optimization Problem (3.6)) is used to select the optimal λ parameter of the SE. We use cross-validation to select the degree of the SE random variable which generated the largest worst case likelihood on the validation partition of the data;

[**FS WCE SN**] The Feature Space (FS) WCE optimization problem (Optimization Problem (3.28)) is used to select the optimal μ and P parameters of the SN. We use cross-validation to select the degree of the SN random variable which generated the

[1]The data set TV originally contains data points with more than two dimensions, however, we extract the time of day and traffic volume from the original data (the real valued variables) to create our two dimensional data set.

largest worst case likelihood on the validation partition of the data;

[**MLE MN**] The MLE optimization problem is solved to optimize the mean and covariance matrix of a multivariate normal random variable as in [51]. In this case no regularization parameters were used, however to ensure a fair comparison between methods the normalization constant is computed using the same Monte Carlo samples as are used in the SE and SN cases;

[**EM GMM**] The expectation-maximization problem is solved to optimize the mean and covariance matrix of a mixture of k multivariate normal random variables. We use, k, the number of multivariate normal distributions as a regularization parameter and select $k \in \{1, \cdots, 30\}$ which had the largest likelihood on the validation partition of the data. As in the MLE MN case we ensure a fair comparison between methods by computing the normalization constant using the same Monte Carlo samples as are used in the SE and SN cases.

Maximum Likelihood Comparison First we analyze the difference in likelihood of the models with respect to the test data partition. A larger likelihood value on the test partition implies that the model is a better fit for the data. In Table 3.1 we compare the performance of the MLE SE, MLE SN, MLE MN, and EM GMM implementations on the test partition set. We report the likelihood on the test partition and the computation time in Table 3.1.

In all cases the MLE SE model has a higher likelihood on the test partition than all other implementations. This implies that in all cases the MLE SE model was more likely to have generated the data in the test partition than the other models. The FS-MLE SN implementation was second best in all cases but the Iris data set and was faster than the MLE SE implementation. In cases with greater numbers of data points and higher dimensional problems the FS-MLE SN is more computationally

Table 3.2: The volume (V) of the level sets of the uncertainty models containing all of the test partition data points and the computation time (T) for the MLE MN, EM GMM, MLE SE, MLE SN, WCE SN and WCE SE implementations. The data sets have dimension (n) and number of training data points (m).

Dataset	Model	V	Time (s)
TV	MLE MN	0.977 ± 0.000	0.003 ± 0.004
n: 2	EM GMM	0.978 ± 0.000	0.010 ± 0.000
m: 40973	SE MLE	0.992 ± 0.004	0.809 ± 0.129
	SE WCE	0.742 ± 0.006	483.454 ± 319.870
	SN FS WCE	0.899 ± 0.003	0.717 ± 0.122
BP	MLE MN	0.924 ± 0.000	0.005 ± 0.003
n: 3	EM GMM	0.756 ± 0.008	0.671 ± 0.581
m: 51850	SE MLE	0.744 ± 0.071	3.466 ± 0.777
	SE WCE	0.255 ± 0.044	1546.791 ± 477.062
	SN FS WCE	0.780 ± 0.002	0.384 ± 0.006
IR	MLE MN	0.048 ± 0.004	0.004 ± 0.002
n: 4	EM GMM	0.087 ± 0.021	0.003 ± 0.001
m: 127	SE MLE	0.034 ± 0.013	6.149 ± 6.410
	SE WCE	0.013 ± 0.006	11.948 ± 11.261
	SN FS WCE	0.012 ± 0.007	1.412 ± 0.486

efficient than the MLE SE implementation.

Worst Case Set Comparison We next train each algorithm on the training partition of the TV, BP and IR data sets and analyze the volume of the level sets that contain all of the test data partition. If we let $\mathcal{D}_t$ be the testing partition, then in Table 3.2 we report the volume of the sets $H_{W_{f_\delta}(\mathcal{D}_t)}$ and the computation time for each algorithm.

The level set of minimal volume which contained the testing partition of data had the smallest volume when modeled using the WCE SE algorithm in all cases but

the Iris data set where the SN FS-WCE implementation was slightly better. This implies that the WCE SE algorithm most tightly enclosed the data in almost every case, and generalized well to new data points. The time taken to generate the set of minimal volume was longest for the WCE optimization problem. The FS-WCE SN implementation performed well on all but the BP data set, where it was outperformed by all but the MLE MN implementation. However, when compared to the SE WCE implementation, the SN FS-WCE was significantly faster while offering comparable performance in some cases. This makes the SN FS-WCE a good first choice for generating tightly enclosing sets, while the SE WCE implementation can be used for generating tighter enclosing sets but at a greater computational cost.

3.5.2 Applications of SDs to a Mass Cytometry Dataset

In this subsection we use a Sequential Forward Selection (SFS) algorithm to identify immune cell characteristics that differ the most between the average patient with rheumatoid arthritis and an average healthy patients using a publicly available mass cytometry dataset.

The Immune Dataset

The mass cytometry dataset consists of 5 samples of immune system cells taken from healthy patients and 9 samples of immune system cells taken from RA patients. This data was taken from the flowrepository [] and the methods of data collection are described in [].

We analyze a group of 5 healthy patients $H^k = \{h_i^k \in [0,1]^{41}\}_{i=1}^{m_h^k}$ which contains $\sum_{k=1}^{5} m_h^k = 1,251,491$ total cell measurements and a second group of 9 RA patients $R^k = \{r_i^k \in [0,1]^{41}\}_{k=1}^{m_r^k}$ which contains $\sum_{k=1}^{9} m_r^k = 3,231,161$ total cell measurements.

A Metric of Similarity

We treat all of the measured samples in H and R as random variables generated by healthy patients and those with RA. To analyze the mass cytometry data from both populations we will model the random variables as sliced-exponentials to generate SE models for the healthy and RA patients respectively.

As in the definition of density between two sets of PDFs we use the squared Hellinger distance as a metric of distance between PDFs. If $d_H^2(f, g) = 0$, then the PDFs f and g are identical on Δ, whereas larger values indicate a larger distance between the PDFs.

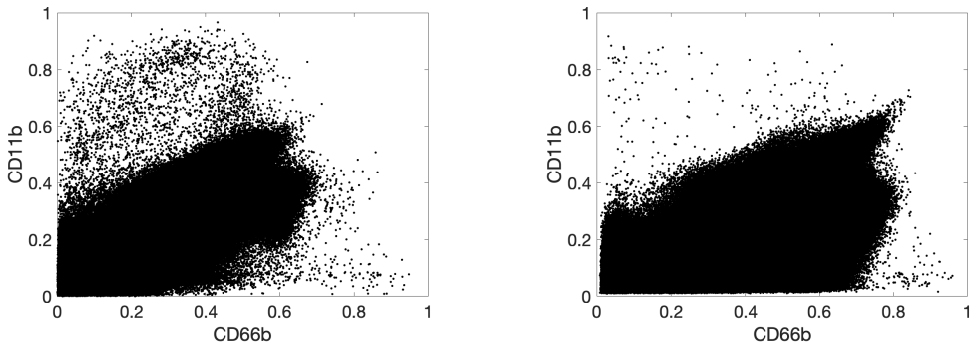
Given a possible set of feature indices $F := \{1, \cdots, n\}$, we define the set of partitions of F as $\mathcal{P}(F)$, and the set of all possible partitions of F of length $w \leq n$ as follows.

$$B_w := \{v \in \mathbb{N}^w \mid v \in \mathcal{P}(F)\}$$

For a given selection of features, $b \in B_w$, we denote the associated projection $P_b : \mathbb{R}^n \to \mathbb{R}^w$ so that $(P_b(\delta))_i = \delta_{b_i}$ for $\delta \in \mathbb{R}^n$ and $i = 1, \cdots, w$. Suppose we let $f^{(H)}_{P_b(\delta)}$ be a model of the distribution of the b features in δ for the healthy population of patients and $f^{(R)}_{P_b(\delta)}$ be a model of the distribution of the b features in δ for the population of patients with RA. Then, if we restrict the search to w features, the solution to the following combinatoric optimization problem returns the features, b, which vary the most between the healthy and RA patient populations.

$$\max_{b \in B_w} \quad d_H^2(f^{(H)}_{P_b(\delta)}, f^{(R)}_{P_b(\delta)}) \tag{3.30}$$

As in Chapter 4 we will use a Sequential Feature Selection (SFS) algorithm as described in [20] to perform feature selection to solve Optimization Problem (3.30). SFS algorithms begin with an empty (or full) set of features and sequentially add (or



(a) Dot plot of CD66b and CD11b receptors measured using mass cytometry for healthy patients.

(b) Dot plot of CD66b and CD11b receptors measured using mass cytometry for patients with RA.

Figure 3.8: Dot plot of CD66b and CD11b receptors measured using mass cytometry for healthy patients (a) and patients with RA (b).

remove) the highest value (or cost) feature until the set of features is a certain size or meets a performance metric. The SFS algorithm used in this paper is as described in [42]. This SFS algorithm begins with $b := \emptyset$, and iteratively selects a locally optimal feature (with respect to the objective function of Optimization Problem (3.30)) at each step.

Results:

We solve Optimization Problem (3.30) using the SFS algorithm as described in [20] using sliced-exponential models of degree one through five to model the PDFs of the healthy and diseased cells. We select $w = 3$ to limit the search to the top three most important cellular characteristics.

To determine which degree of sliced-exponential best captures the difference between cells from healthy patients and those with RA, we will analyze how well the two models can be used to differentiate between RA and healthy patients.

Let $f^{(H)}$ and $f^{(R)}$ be sliced-exponential models of the healthy patient data in H

Table 3.3: The leave-one-out classification accuracy $A_L(H, R)$ of predicting whether a patient has RA or does not have RA using varying degree SE models of immune system cells from healthy patients and RA patients.

SE Degree	1	2	3	4	5
$A_L(H, R)$	85.71%	78.57%	78.57%	92.86%	92.86%

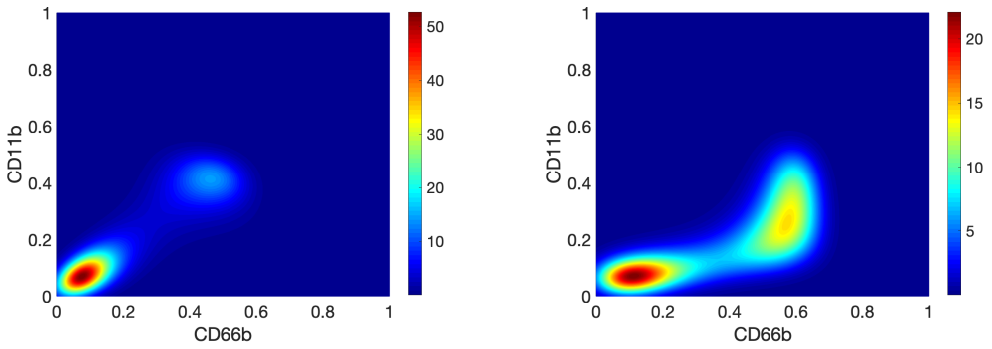
and RA patient data in R respectively. Given a set of data, $\mathcal{D}$ from a new patient we will use the likelihood of both models on the data to determine if the patient is healthy or has RA. Specifically, if $L_{f^{(H)}}(\mathcal{D}) > L_{f^{(R)}}(\mathcal{D})$ then the patient is predicted to be healthy, otherwise they are predicted to have RA.

To estimate the predictive accuracy of the sliced-exponential models on new data we will use the leave-one-out classification accuracy. Let, $f^{(H,k)}$ and $f^{(R,k)}$ be the models of the healthy patient data and RA patient data - excluding for the k'th patient in that set. Let $I(x)$ be an indicator function that is equal to one if $x > 0$ and zero otherwise and the leave-one-out classification error can then be defined as,

$$A_L(H, R) = \frac{1}{14}\left(\sum_{k=1}^{5} I\left(L_{f^{(H,k)}}(H^k) - L_{f^{(R)}}(H^k)\right) + \sum_{k=1}^{9} I\left(L_{f^{(R,k)}}(R^k) - L_{f^{(H)}}(R^k)\right)\right).$$

Thus $A_L(H, R)$ is the accuracy of predicting whether a patient in the training data set is healthy or has RA using the healthy and RA models when they are trained without that patients data. This is a better estimate of the predictive accuracy for future patients whose data was not used to train the models.

We show the leave-one-out classification accuracy for varying degrees of sliced-exponential models in Table 3.3. The best leave-one-out classification accuracy was generated by the degree four and five sliced-exponentials which both achieved an accuracy of 92.86% after misidentifying one healthy patient as having RA. This analysis shows that more complex models than exponential or normal distributions are required to effectively model the differences in the densities of the average healthy



(a) Sliced-Exponential PDF of healthy immune system cells with CD66b and CD11b measured using mass cytometry.

(b) Sliced-Exponential PDF of RA immune system cells with CD66b and CD11b measured using mass cytometry.

Figure 3.9: Sliced-Exponential PDF of healthy immune system cells (a) and RA immune system cells (b) with CD66b and CD11b measured using mass cytometry.

patient and the average RA patient.

Given equal performance between two models, it is good practice to select the least complex model. Therefore we will further analyze the degree four sliced-exponential model.

The three cell characteristics that most differ between the healthy and diseased populations (based on the degree 4 sliced-exponential models) consist of the CD66b, CD11b and CD23 receptors. We show a dot plot of the cellular measurements taken from healthy and RA patients for the first two receptors in Fig. 3.8(a) and 3.8(b).

To demonstrate that the populations of cells have complex densities (necessitating the use of SEs or other complex distributions) we show the joint PDFs of the sliced-exponential model for the first two receptors of the healthy and RA patients in Fig. 3.9. The complex bimodal density of the cellular characteristics captured by the degree 4 sliced-exponential is what makes the leave-one-out classification accuracy higher than the lower degree sliced-exponential models.

Further testing is necessary to determine a more accurate estimate of the predictive

accuracy of this model. Furthermore, collecting mass cytometry data from patients with other autoimmune diseases is necessary to determine if the models can differentiate between patients with RA and other autoimmune diseases, or if the models are capturing densities of cell populations that may be similar in other autoimmune diseases.

3.6 Conclusion

This chapter proposes convex optimization problems to select optimal parameters of sliced distributions that solve the MLE and WCE optimization problems to model the distribution of given data. We showed that two sets of SD PDFs are dense in the set of all bounded PDFs with respect to the Hellinger distance. The developments herein allow for the efficient characterization of the distribution of data.

We use two metrics to demonstrate that the proposed methods, based on Sliced Distributions, have superior performance with respect to modeling the distribution of data when compared to Gaussian mixture models and multivariate normal models. The first metric is based on the likelihood of the models to generate a test partition of the data, while the second is based on the volume of the level sets of the model PDFs that most tightly contain the test partition of the data. The models optimized with respect to the maximum likelihood metric are superior with respect to the first metric, while those optimized with respect to the worst case likelihood metric are superior in the second metric.

Furthermore we show that SDs may be used to model mass cytometry data of the average healthy patient and the average RA patient. By comparing the models generated for each group, we identified a set of immune system characteristics whose PDF models differed most between healthy patients and those with RA. These models can differentiate between mass cytometry datasets taken from healthy and RA pa-

tients with high accuracy, illustrating that the models captured important differences in the immune system cells between the two groups. However, further testing with larger datasets is necessary to more accurately determine the predictive accuracy of the proposed models.

Chapter 4

MACHINE LEARNING WITH POSITIVE KERNELS PARAMETERIZED BY POSITIVE MATRICES

There exists hundreds of different subpopulations of immune cells and cytokine signals which could potentially be classified as either helper [165, 105] or regulatory [138, 90]. Analyzing data to determine which of these potential cells and signals is most relevant to the immunogenic or tolerogenic responses to antigen requires sophisticated machine learning methods. One such state of the art class of machine learning algorithms is the class of kernel methods.

Kernel methods are a class of machine learning algorithm (the most relevant to this chapter being the support vector machine) that return a function, $f \in \mathcal{F}$, designed to map a set of inputs to corresponding outputs. Specifically kernel methods can be used on problems such as classification, regression and the clustering of data. The set of functions, $\mathcal{F}$, from which the kernel method may select a function is dependent on the selection of some a priori selected kernel function. Therefore, the selection of a kernel function determines the class of functions that can be searched over, directly affecting the accuracy of the learned map from the inputs to outputs.

Kernel Learning (KL) algorithms such as those found in [171, 149, 172] automate this task by finding the kernel, $k \in \mathcal{K}$ which optimizes an achievable metric such as the soft margin (for classification). The set of kernels, $k \in \mathcal{K}$, over which the algorithm can optimize, however, strongly influences the performance of the resulting classifier or predictor.

To understand how the choice of $\mathcal{K}$ influences performance and robustness, we propose three properties to characterize the set $\mathcal{K}$ - tractability, density, and universality.

Specifically, $\mathcal{K}$ is tractable if $\mathcal{K}$ is convex (or, preferably, a linear variety) - implying the KL problem is solvable using, e.g. the algorithms in [134, 78, 96, 132, 66]. The set $\mathcal{K}$ has the density property if, for any $\epsilon > 0$ and any positive kernel, k^* there exists a $k \in \mathcal{K}$ where $\|k - k^*\| \leq \epsilon$. The density property implies the kernel generalizes well implying good performance on untrained data. The set $\mathcal{K}$ has the universal property if any $k \in \mathcal{K}$ is universal (see Definition 17) - ensuring a classifier/predictor exists that maps each unique training input to its corresponding output using any kernel function in $\mathcal{K}$.

Previously, there was no class of kernel that meets all three criteria - e.g. Gaussians are not tractable or accurate; polynomials are not scalable. Therefore we have proposed a new class that meet all three criteria - the Tessellated Kernel (TK) class. Specifically, the TK class: admits a linear parameterization using positive matrices; is dense in all kernels; and every element in the class is universal. This implies that the use of TK kernels for learning the kernel can obviate the need for selecting candidate kernels in other Kernel Learning (KL) algorithms or parameters such as the bandwidth of the Gaussian kernel. Numerical testing on soft margin Support Vector Machine (SVM) problems show that algorithms using TK kernels outperform other kernel learning algorithms, neural networks, and random forest algorithms. Finally we apply KL with TKLs to the problem of identifying immune system cells in an animal model that are correlated to the immune state of mice with Rheumatoid Arthritis. Given a set of immune system populations, this analysis identifies the key populations that are correlated to the disease severity and progression of the disease.

4.1 Introduction to Kernel Learning

This chapter addresses the problem of the automated selection of an optimal kernel function for a given kernel-based machine learning problem (e.g. soft margin SVM).

Kernel functions implicitly define a linear parametrization of nonlinear candidate maps $y = f(x)$ from vectors x to scalars y. Specifically, for a given kernel, the 'kernel trick' allows optimization over a set of candidate functions in the kernel-associated hypothesis space without explicit representation of the space itself. The kernel selection process, then, is critical for determining the class of hypothesis functions and, as a result, is a well-studied topic with common kernels including polynomials, Gaussians, and many variations of the Radial Basis Function. In addition, specialized kernels include string kernels [104, 52], graph kernels [59], and convolution kernels [71, 34]. The kernel selection process heavily influences the accuracy of the resulting fit and hence significant research has gone into the optimization of these kernel functions in order to select the hypothesis space which most accurately represents the underlying physical process.

Recently, there have been a number of proposed kernel learning algorithms. For support vector machines, the methods proposed in this chapter are heavily influenced by the SDP approach proposed in [96] which directly imposed kernel matrix positivity on a subspace defined by the linear combination of candidate kernel functions. There have been several extensions of the SDP approach, including the hyperkernel method in [120]. However, because of the complexity of semidefinite programming, more recent work has focused on alignment methods for MKL as in, e.g. [39] or gradient methods for convex and non-convex parameterizations of positive linear combinations of candidate kernels, such as SimpleMKL [134] or the several variations in SHOGUN [149]. These MKL methods rely on kernel operations (addition, multiplication, convolution) to generate large numbers of parameterized kernel functions as in [37]. Examples of non-convex parameterizations include GMKL [78], and LMKL [65]. Work focused on regularization includes the group sparsity metric [155] and the enclosing ball approach [58]. See, e.g. [66] for a comprehensive review of

MKL algorithms.

In this chapter, we focus on the class of "Universal Kernels" formalized in [109]. For a given compact metric space (input space), $\mathcal{X}$, it is said that a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a Positive Kernel (PK) if for any $N \in \mathbb{N}$ and any $\{x_i\}_{i=1}^N \subset \mathcal{X}$, the matrix defined elementwise by $K_{ij} = k(x_i, x_j)$ is symmetric and Positive SemiDefinite (PSD).

Definition 17. *A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be universal on the compact metric space $\mathcal{X}$ if it is continuous and there exists an inner-product space $\mathcal{W}$ and feature map, $\Phi : \mathcal{X} \to \mathcal{W}$ such that $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{W}}$ and where the unique Reproducing Kernel Hilbert Space (RKHS),*

$$\mathcal{H} := \{f \ : \ f(x) = \langle v, \Phi(x) \rangle, \ v \in \mathcal{W}\}$$

with associated norm $\|f\|_{\mathcal{H}} := \inf_v \{\|v\|_{\mathcal{W}} \ : \ f(x) = \langle v, \Phi(x) \rangle\}$ is dense in $\mathcal{C}(\mathcal{X}) := \{f : \mathcal{X} \to \mathbb{R} \ : f$ is continuous$\}$ where $\|f\|_{\mathcal{C}} := \sup_{x \in X} |f(x)|$.

Note that for any given PD kernel, the RKHS $\mathcal{H}$ exists, is unique, and can be characterized (as described in [157]) using the Riesz representation theorem as the closure of $\text{span}\{k(y, \cdot) \ : \ y \in \mathcal{X}\}$ with inner product defined for any $f(x) = \sum_{i=1}^n c_i k(y_i, x)$ and $g(x) = \sum_{i=1}^m d_i k(z_i, x)$ as

$$\langle f, g \rangle_{\mathcal{H}} := \sum_{i=1}^{n} \sum_{j=1}^{m} c_i d_j k(y_i, z_j).$$

Universal kernels are preferred when large amounts of data are available, due to the fact that the dimension of the hypothesis space increases for every additional data point - resulting in the ability to construct highly specialized and accurate classifiers.

The most well-known example of a universal kernel is the Gaussian (generalized in [174]). However, many other common kernels are not universal, including, significantly, the polynomial class of kernels. This is significant because the class of

generalized polynomial kernels (Eq. (4.9)) have the density and tractability property, while the set of Gaussian kernels have the universality property but not the tractability property.

The Class of Tessellated Kernels (TK) We proposed the class of kernel functions (called Tessellated Kernels) which are not polynomials, yet which are defined by polynomials and admit a linear parametrization in [29]. These kernels define classifiers on a tessellated domain, each sub-domain (or tile) of which is a hyper-rectangle with vertices defined by the *input data* - $\{x_i\}_{i=1}^m$. In this way, each data point further divides any tiles within which any of its features lie, resulting in increasing numbers of disjoint tiles. The classifier itself, then, is piecewise polynomial - being polynomial when restricted to any particular tile.

TK kernels have three important properties which make them uniquely well-suited for kernel learning problems. First, these kernels admit a linear parameterization using positive semidefinite matrices - meaning we can use convex optimization to search over the entire class of such kernels (tractability), which is proven in Corollary 28 and implemented in Optimization Problem (4.23). This is like the class of generalized polynomial kernels (See Eq. (4.9)) yet unlike other universal kernel classes such as the Gaussian/RBF, wherein the bandwidth parameter appears in the exponential. Second, the TK class is dense in all kernels (accuracy), meaning there exists a TK kernel that can approximate any given kernel arbitrarily well. This is like the generalized polynomial class yet unlike the Gaussian/RBF class, wherein the resulting kernel matrix is restricted to having all positive elements. Third, any kernel of the TK class has the universal property (scalability). This is like the Gaussian/RBF class and unlike the generalized polynomial kernels, none of which are universal. The TK class is thus unique in that no other currently known class of kernel functions has all

three properties of tractability, accuracy, and scalability.

4.1.1 Kernel Learning for Classification and Regression

We first present two standard algorithms for solving the kernel learning problem for both classification and regression. These algorithms are general in the sense that they apply to any given linear parameterization of kernel functions.

The Kernel Function: To find nonlinear models using SVMs we must introduce a positive kernel function, k.

Definition 18. *We say a function $k : Y \times Y \to \mathbb{R}$ is a* **positive kernel function** *if*

$$\int_Y \int_Y f(x)k(x,y)f(y)dxdy \geq 0$$

for any function $f \in L_2[Y]$.

Recall from Chapter 2 that applying a kernel function to the 1-norm soft margin SVM problem yields the optimization problem (2.12),

$$\begin{aligned} \max_{\alpha \in \mathbb{R}^m} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \ \ \forall \ \ i = 1, ..., m. \end{aligned}$$

Furthermore applying a kernel function to the ϵ-SVR Problem yields the dual formulation in optimization problem (2.15),

$$\begin{aligned} \max_{\alpha \in \mathbb{R}^m, \alpha^* \in \mathbb{R}^m} \quad & -\frac{1}{2}\sum_{i,j=1}^m (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) \\ & -\epsilon \sum_{i=1}^m (\alpha_i + \alpha_i^*) + \sum_{i=1}^m y_i(\alpha_i - \alpha_i^*) \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \ \ \forall \ \ i = 1, ..., m. \end{aligned}$$

Unfortunately Optimization Problems 2.12 and 2.15 require that the kernel function, $k(x, y)$, be chosen a priori, a choice which significantly influences the accuracy of the resulting classifier f. Next we alter the optimization problem by considering the kernel itself to be an optimization variable, constrained to lie in a given convex set of candidate positive kernel functions, $\mathcal{K}$.

The Kernel-learning Problem (1-norm SVM):

Since Optimization Problems 2.12 and 2.15 are the dual of the SVM/SVR optimization problems, we want to select the kernel function $k \in \mathcal{K}$ that minimizes the objective function of the dual.

For the 1-norm soft margin SVM, when the kernel itself is parameterized as an optimization variable we therefore have the following convex optimization problem.

$$\begin{aligned} \min_{k\in\mathcal{K}} \max_{\alpha\in\mathbb{R}^m} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad 0 \le \alpha_i \le C \ \ \forall \ \ i = 1, ..., m \end{aligned} \tag{4.1}$$

Likewise, in the case of the ϵ-SVR optimization problem the KL problem is given as the following convex optimization problem.

$$\min_{k\in\mathcal{K}} \max_{\substack{\alpha\in\mathbb{R}^m, \\ \alpha^*\in\mathbb{R}^m}} \quad -\frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) - \epsilon\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i(\alpha_i - \alpha_i^*) \tag{4.2}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \ \ \forall \ \ i = 1, ..., m.$$

Having formulated the kernel learning problems, we now present two standard approaches to parameterizing the set of candidate kernels, $\mathcal{K}$, and solving the resulting convex optimization problem.

4.1.2 SDP-based kernel learning using positive kernel matrices

We first consider the method in [96], for the 1-norm soft margin SVM wherein positive matrices were used to parameterize $\mathcal{K}$ for a given set of candidate kernels $\{k_i\}_{i=1}^{l}$ as

$$\mathcal{K} := \left\{ k(x,y) = \sum_{i=1}^{l} \mu_i k_i(x,y) : \ \mu \in \mathbb{R}^l, \ K_{ij} = k(x_i, x_j), \ K \succeq 0 \right\}, \tag{4.3}$$

where the $\{x_i\}_{i=1}^{m} \subset \mathbb{R}^n$ are the training points of the SVM problem and the k_i were chosen a priori to be, for instance, Gaussian and polynomial kernels. It is significant to note that the PSD constraint on the kernel matrix K, enforces that the kernel matrix is PSD for the set of training data, but does not necessarily enforce that the kernel function itself is PD - meaning that kernels in $\mathcal{K}$ are not necessarily positive kernels.

Using this parameterized $\mathcal{K}$, the kernel optimization problem for the 1-norm soft margin support vector machine was formulated in [96] as the following semi-definite program, where e is the vector of all ones.

$$\begin{aligned} \min_{\mu \in \mathbb{R}^l, \ t \in \mathbb{R}, \ \gamma \in \mathbb{R}, \ \nu \in \mathbb{R}^m, \ \delta \in \mathbb{R}^m} \quad & t \\ \text{subject to:} \quad & \begin{pmatrix} G & e + \nu - \delta + \gamma y \\ (e + \nu - \delta + \gamma y)^T & t - 2C\delta^T e \end{pmatrix} \succeq 0 \\ & \nu \geq 0, \qquad \delta \geq 0, \qquad G_{ij} = k(x_i, x_j) y_i y_j \\ & k(x,y) = \sum_{i=1}^{l} \mu_i k_i(x,y) \end{aligned} \tag{4.4}$$

Note that here the original constraint $K \geq 0$ in $\mathcal{K}$ has been replaced by an equivalent constraint on G. This problem can now be solved using well-developed interior-point methods as in [3] with implementations such as MOSEK in [4].

We next consider the method of [114], for the ϵ-SVR problem wherein, like the

method in [96], a linear combination of positive matrices were used to parameterize $\mathcal{K}$. Using this parameterization of $\mathcal{K}$, the kernel optimization problem for the ϵ-SVR problem, as formulated in [114], is as follows, where again e is the vector of all ones.

$$\begin{aligned} \min_{\mu\in\mathbb{R}^l,\ t\in\mathbb{R},\ \gamma\in\mathbb{R}^m,\ \nu_u^+\in\mathbb{R}^m,\ \nu_u^-\in\mathbb{R}^m,\ \nu_l^-\in\mathbb{R}^m} \quad & t \\ \text{subject to:} \quad & \begin{pmatrix} 2K & \gamma \\ \gamma^T & t - 2Ce^T(\nu_u^+ + \nu_u^-) \end{pmatrix} \geq 0 \\ & \nu_u^+, \nu_u^-, \nu_l^- \geq 0, \quad \epsilon e + \nu_u^+ + \nu_u^- - \nu_l^- \geq 0, \\ & k(x,y) = \sum_{i=1}^{l} \mu_i k_i(x,y) \end{aligned} \tag{4.5}$$

In Optimization Problem (4.4) and (4.5), the size of the SDP constraint is $(m+1)\times(m+1)$ which is problematic in that the complexity of the resulting SDP grows as a polynomial in the number of training data. Methods that do not require this large semi-definite matrix constraint are explored next.

4.1.3 Minimax Kernel Learning

In this subsection, we again take a set of basis kernels $\{k_i\}_{i=1}^l$ and consider the set of positive linear combinations,

$$\mathcal{K} := \left\{ k \ : \ k(x,y) = \sum_{i=1}^{l} \mu_i k_i(x,y),\ \mu_i \geq 0 \right\}. \tag{4.6}$$

Any element of this set is a positive kernel, replacing the matrix positivity constraint by a LP constraint. For classification the Kernel Learning problem in minimax form is as follows.

$$\begin{aligned} \min_{\mu\geq 0}\ \max_{\alpha\in\mathbb{R}^m} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \sum_{k=1}^{l} \mu_k \alpha_i \alpha_j y_i y_j k_k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad \alpha_i \in [0, C] \ \ \forall \ \ i = 1, ..., m \end{aligned}$$

Likewise for regression the Kernel Learning problem in minimax form is as follows.

$$\min_{\mu \geq 0} \max_{\substack{\alpha \in \mathbb{R}^m, \\ \alpha^* \in \mathbb{R}^m}} -\frac{1}{2} \sum_{k=1}^{l} \sum_{i,j=1}^{m} \mu_k (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k_k(x_i, x_j) + \sum_{i=1}^{m} y_i(\alpha_i - \alpha_i^*) - \epsilon(\alpha_i + \alpha_i^*)$$

$$\text{s.t.} \sum_{i=1}^{m} \alpha_i - \alpha_i^* = 0, \quad \alpha_i, \alpha_i^* \in [0, C] \ \ \forall \ \ i = 1, ..., m.$$

Use of these formulations is generally referred to as Multiple Kernel Learning (MKL) and is solved directly in the minimax formulation rather than as an SDP in [96, 114]. These formulations are an LP minimization problem in μ for fixed α and a QP maximization problem in α for fixed μ. Recently, a number of highly efficient two-step methods have been proposed which exploit this formulation, including SimpleMK [134]. These methods alternate between fixing μ and optimizing α, then fixing α and optimizing μ, adding the constraint that $\sum_i \mu_i = 1$ using a projected gradient descent. Other two-step solvers include [66]. Two-step MKL solvers typically have a significantly reduced computational complexity compared with SDP-based approaches and can typically handle thousands of data points and thousands of basis kernels. In Section 4.4 and 4.5 we apply the SDP and the minimax approach respectively to the problem of Kernel Learning with Tessellated Kernels.

4.2 Positive matrices parameterize positive kernels

The methods proposed in the prior section generally require the kernel function to be a positive summation, or sometimes a positive product, of a priori selected kernel functions. In this section we review a new framework for using positive matrices to parameterize positive kernels that we first proposed in [29]. This is a generalization of a result initially proposed in [135]. In Subsection 4.2.1 we apply this framework to obtain generalized polynomial kernels. In Subsection 4.2.2, we use the framework to obtain the TK class.

Proposition 19. *Let N be any bounded measurable function $N : \mathcal{X} \times Y \to \mathbb{R}^q$ on compact $\mathcal{X}$ and Y and $P \in \mathbb{R}^{q \times q}$ be a positive semidefinite matrix $P \geq 0$. Then*

$$k(x, y) = \int_{\mathcal{X}} N(z, x)^T P N(z, y) dz \tag{4.7}$$

is a positive kernel function.

Proof. Since N is bounded and measurable, $k(x, y)$ is bounded and measurable. Since $P \geq 0$, there exists $P^{\frac{1}{2}}$ such that $P = (P^{\frac{1}{2}})^T P^{\frac{1}{2}}$. Now for any $f \in L_2[Y]$ define

$$g(z) = \int_Y P^{\frac{1}{2}} N(z, x) f(x) dx.$$

Then

$$\begin{aligned}
\int_Y \int_Y f(x) k(x, y) f(y) dx dy &= \int_Y \int_Y \int_{\mathcal{X}} f(x) N(z, x)^T P N(z, y) f(y) dz dx dy \\
&= \int_{\mathcal{X}} \left(\int_Y P^{\frac{1}{2}} N(z, x) f(x) dx \right)^T \left(\int_Y N(z, y) P^{\frac{1}{2}} f(y) dy \right) dz \\
&= \int_{\mathcal{X}} g(z)^T g(z) dz \geq 0.
\end{aligned}$$

□

For a given N, the map $P \mapsto k$ in Proposition 19 is linear. Specifically,

$$k(x, y) = \sum_{i,j} P_{i,j} G_{i,j}(x, y), \quad \textit{where}, \quad G_{i,j}(x, y) = \int_{\mathcal{X}} N_i(z, x) N_j(z, y) dz.$$

4.2.1 Generalized Polynomial Kernels (GPK)

Let $Y = \mathbb{R}^n$ and define $Q_d : \mathbb{R}^n \to \mathbb{R}^q$ to be the vector of monomials of degree d. If we now define $N_P(z, y) = Q_d(y)$, then k as defined in Proposition 19 is a polynomial of degree $2d$. The following result is from [127].

Lemma 20. *A polynomial k of degree $2d$ is a positive polynomial kernel if and only if there exists some $P \geq 0$ such that*

$$k(x, y) = Q_d(x)^T P Q_d(y). \tag{4.8}$$

This lemma implies that a representation of the form of Equation (4.7) is necessary and sufficient for a generalized polynomial kernel to be positive. For convenience, we denote the set of generalized polynomial kernels of degree d as follows.

$$\mathcal{K}_P^d := \{k \ : \ k(x,y) = Q_d(x)^T P Q_d(y) \ : \ P \geq 0\} \tag{4.9}$$

Unfortunately, however, polynomial kernels are never universal and hence we propose the following universal class of TK kernels, each of which is defined by polynomials, but which are not polynomial.

4.2.2 Tessellated Kernels

To begin, we define the indicator function for the positive orthant as

$$I_+(z) = \begin{cases} 1 & z \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

where $z \geq 0$ means $z_i \geq 0$ for all i. Now define $Z_d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^q$ to be the vector of monomials of degree d in $\mathbb{R}^{2n}$. We now propose the following choice of $N : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{2q}$.

$$N_T^d(z,x) = \begin{bmatrix} Z_d(z,x) I_+(z-x) \\ Z_d(z,x) I_+(x-z) \end{bmatrix} = \begin{cases} \begin{bmatrix} Z_d(z,x) \\ 0 \end{bmatrix} & z \geq x \\ \begin{bmatrix} 0 \\ Z_d(z,x) \end{bmatrix} & x \geq z \\ 0 & \text{otherwise} \end{cases} \tag{4.10}$$

Equipped with this definition, we define the class of Tessellated Kernels as follows.

$$\mathcal{K}_T^d := \left\{ k \ : \ k(x,y) = \int_{\mathcal{X}} N_T^d(z,x)^T P N_T^d(z,y) dz, \ P \geq 0 \right\}, \qquad \mathcal{K}_T := \{k \ : \ k \in \mathcal{K}_T^d, \ d \in \mathbb{N}\}$$

4.2.3 Representation of TK kernels using polynomials

The following result shows that any $k \in \mathcal{K}_T$ is piecewise polynomial. Specifically, if we define the partition of $\mathbb{R}^n$ into 2^n orthants - parameterized by $\beta \in \{0,1\}^n$ as $\{X_\beta\}_{\beta\in\{0,1\}^n}$ where

$$X_\beta := \left\{ x \in \mathbb{R}^n \ : \ \begin{smallmatrix} x_j \geq 0 \text{ for all } j:\beta_j=0, \\ x_i \leq 0 \text{ for all } i:\beta_i=1 \end{smallmatrix} \right\}, \tag{4.11}$$

then for any $k \in \mathcal{K}_T$, $k(x,y) = k_\beta(x,y)$ for any $x - y \in X_\beta$.

Lemma 21. *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a,b]$, N is as defined in Eqn.* (4.10)*,*

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \succ 0,$$

k is as defined in Eqn. (4.7) *and $\{X_\beta\}_{\beta\in\{0,1\}^n}$ is defined in Eqn.* (4.11)*. Then $k(x,y) = k_\beta(x,y)$ for any $x - y \in X_\beta$ where the k_β are polynomials defined as*

$$k_\beta(x,y) = \int_{\beta_1 y_1 + (1-\beta_1)x_1}^{b_1} \cdots \int_{\beta_n y_n + (1-\beta_n)x_n}^{b_n} Z_d(z,x)^T Q_1 Z_d(z,y) dz + k_0(x,y),$$

where

$$k_0(x,y) = \int_x^b Z_d(z,x)^T Q_2 Z_d(z,y) dz + \int_y^b Z_d(z,x)^T Q_3 Z_d(z,y) dz + \int_a^b Z_d(z,x)^T P_{22} Z_d(z,y) dz,$$

and

$$Q_1 = P_{11} - P_{12} - P_{21} + P_{22}, \quad Q_2 = P_{12} - P_{22}, \quad Q_3 = P_{21} - P_{22}.$$

Proof. Given N as defined above, if we partition $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$ into equal-sized blocks, we have

$$k(x,y) = \int_{\mathcal{X}} N(z,x)^T P N(z,y) dz = \sum_{i,j=1}^{2} \int_{(x,y,z)\in\mathcal{X}_{ij}} Z_d(z,x)^T P_{i,j} Z_d(z,y) dz$$

where

$$\mathcal{X}_{ij} := \{(x,y,z) \in \mathbb{R}^{3n} \ : \ I_+((-1)^j(z-x))I_+((-1)^j(z-y)) = 1\}.$$

From the definition of $\mathcal{X}_{ij}$ we have that,

$$\begin{aligned}
\mathcal{X}_{11} =& \{z \in \mathcal{X} \ : \ z_i \geq p_i^*(x,y), \ i = 1, \cdots, n\} \\
\mathcal{X}_{12} =& \{z \in \mathcal{X} \ : \ z_i \geq x_i, \ i = 1, \cdots, n\} / \mathcal{X}_{11} \\
\mathcal{X}_{21} =& \{z \in \mathcal{X} \ : \ z_i \geq y_i, \ i = 1, \cdots, n\} / \mathcal{X}_{11} \\
\mathcal{X}_{22} =& \mathcal{X} / \left(\mathcal{X}_{11} \cup \mathcal{X}_{12} \cup \mathcal{X}_{21}\right).
\end{aligned}$$

where $p_i^*(x,y) = \max\{x_i, y_i\}$ and $p_i^*(x,y) = \beta_i y_i + (1-\beta_i)x_i$. By the definitions of $\mathcal{X}_{11}, \mathcal{X}_{12}, \mathcal{X}_{21}$, and $\mathcal{X}_{22}$ we have that,

$$\begin{aligned}
&k(x,y) = \\
&\int_{p^*(x,y)}^{b} Z_d(z,x)^T (P_{11} - P_{12} - P_{21} + P_{22}) \, Z_d(z,y) dz + \int_x^b Z_d(z,x)^T (P_{12} - P_{22}) \, Z_d(z,y) dz \\
&+ \int_y^b Z_d(z,x)^T \left(P_{21} - P_{22}\right) Z_d(z,y) dz + \int_a^b Z_d(z,x)^T P_{22} Z_d(z,y) dz. \qquad (4.12)
\end{aligned}$$

□

Note that the number of domains X_β used to define the piecewise polynomial k is 2^n, which does not depend on q (the dimension of P_{ij}). Thus, even if $Z_d = 1$, the resulting kernel is partitioned into 2^n domains. The size of $Z_d(x,y) \in \mathbb{R}^q$ only influences the degree of the polynomial defined on each domain.

The significance of the partition does not lie in the number of domains of the kernel, however. Rather, the significance of the partition lies in the resulting classifier, which, for a given set of training data $\{x_i\}_{i=1}^m$, has a domain tessellated into $(m+1)^n$ tiles, X_γ, where $\gamma \in \{0, \cdots, m\}^n$. Although the training data is unordered, we create an ordering using $\Gamma(i,j) : \{0, \cdots, m\} \times \{1, n\} \to \{1, \cdots, m\}$ where $\Gamma(i,j)$ indicates that among the jth elements of the training data, $x_{\Gamma(i,j)}$ has the ith largest value.

That is,

$$[x_{\Gamma(i-1,j)}]_j \leq [x_{\Gamma(i,j)}]_j \leq [x_{\Gamma(i+1,j)}]_j \qquad \forall i = 1, \cdots, m-1, \quad j = 1, \cdots, n.$$

Now, for any $\gamma \in \{0, \cdots, m\}^n$, we may define an associated tile

$$X_\gamma := \left\{ z \ : \ [x_{\Gamma(\gamma_j,j)}]_j \leq z_j \leq [x_{\Gamma(\gamma_j+1,j)}]_j, \ j = 1, \cdots, n \right\}.$$

The classifier may now be represented as

$$\begin{aligned} f(z) &= \sum_{i=1}^{m} \alpha_i y_i k(x_i, z) + b \\ &= f_\gamma(z) \ \ \forall z \in X_\gamma. \end{aligned}$$

To define the f_γ, we associate with every tile γ and datum i an orthant $\beta(i, \gamma)$ which denotes the position of tile T_γ relative to datum x_i - i.e. T_γ is in the orthant $\beta(i, \gamma)$ centered at the point x_i. Specifically,

$$\beta(i,\gamma)_j = \begin{cases} 0 & [x_{\Gamma(\gamma_j,j)}]_j \geq [x_i]_j \\ 1 & \text{otherwise.} \end{cases}$$

Now we may define

$$f_\gamma(z) = \sum_{i=1}^{m} \alpha_i y_i k_{\beta(i,\gamma)}(x_i, z)$$

which is a polynomial for every γ. In this way, each data point further divides the domains which it intersects, resulting in $(m+1)^n$ disjoint sub-domains, each with associated polynomial classifier.

Thus we see that the number of domains of definition of the classifier grows quickly in m, the number of training data points. For instance, with n = 2 there are 100 tiles for just 9 data points. This growth is what makes TK kernels universal - as will be seen in Section IV.

In Figure 4.1(a) we see the function, $f(z) = \sum_{i=1}^{m} \alpha_i y_i k(x_i, z) + b$, for a degree 1 TK kernel trained for a 1-dimensional labeling problem as compared with a Gaussian kernel. We see that the TK classifier is continuous, and captures the shape of the generator better than the Gaussian. Note that the TK classifier is not continuously differentiable and the derivative can change precipitously at the edges of the tiles. However, if we decrease the inverse regularity weight C in the objective function of Optimization Problem (2.11), then this has the effect of smoothing the resulting classifier. In Figure 4.1(a), as C decreases we see that the changes in slope at edges of the tiles decrease.

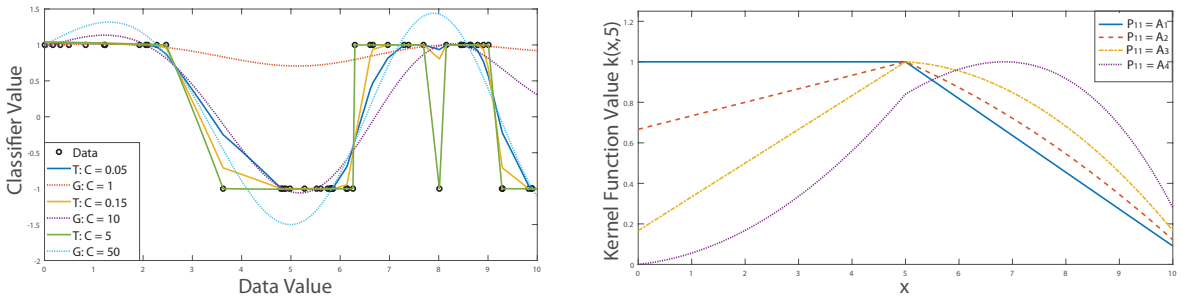
To illustrate that the function $k(x_i, z)$ is a piecewise polynomial tessellated by the training datum, we plot the value of an assortment of TK kernels in one dimension in Figure 4.1(b). We use training datum $x_i = 5$, and a selection of different positive matrices where $P_{1,2} = P_{2,1} = P_{2,2} = 0$ and $P_{1,1} = A_i$ for $i = 1, \ldots, 4$ where

$$A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 0 \\ 0 & .1 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.13}$$

In the first three cases the monomial basis is of degree 1, while in the fourth case the monomial basis is of degree 2 - for simplicity we exclude monomials with z. These different matrices all illustrate changes in slope which occur at the training datum.

4.3 Properties of the tessellated class of kernel functions

In this section, we prove that all TK kernels are continuous and universal and that the TK class is pointwise dense in all kernels.



(a) Optimal classifier, $f(z)$ using a degree one TK (solid lines), and a positive combination of Gaussian kernels (dotted lines) with three different penalty weights C.

(b) Normalized kernel function $k(5, z)$ using $P_{1,1} = A_i$ from (4.13) and $P_{1,2} = P_{2,1} = P_{2,2} = 0$.

Figure 4.1: This figure depicts the optimal classifier for labeling a 1-dimensional data set compared to Gaussian classifiers as well as the normalized kernel function, $k(5, z)$, using different $P_{1,1}$ matrices and $\mathcal{X} = [0, 10]$.

4.3.1 TK kernels are continuous

Let us begin by recalling that for any $P \geq 0$ and $N(z, x)$,

$$k(x, y) = \int_{\mathcal{X}} N(z, x)^T P N(z, y) dz$$

is a positive kernel and recall that for the TK kernels, we have

$$N(z, x) = \begin{bmatrix} Z_d(z, x) I_+(z - x) \\ Z_d(z, x) I_+(x - z) \end{bmatrix}.$$

By the representer theorem this implies that the classifiers consist of functions of the form

$$f(y) = \sum_{i=1}^{m} \alpha_i \int_{\mathcal{X}} N(x_i, z)^T P N(y, z) dz.$$

The following theorem establishes that such functions are necessarily continuous.

Theorem 22. *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a, b]$, $P \geq 0$, N is as defined in Eqn.* (4.10) *for some $d \geq 0$ and k is as defined in Eqn.* (4.7)*. Then k is continuous and for any $\{x_i\}_{i=1}^m$ and $\alpha \in \mathbb{R}^m$, the function*

$$f(z) = \sum_{i=1}^{m} \alpha_i k(x_i, z),$$

is continuous.

Proof. Partition P as follows

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} > 0.$$

To prove that $f(z)$ is continuous we need only prove that $k(x, y)$ is continuous. Applying Lemma 3 we may define $k(x, y)$ as

$$k(x, y) = \begin{cases} k_\beta(x, y) & \text{if } x - y \in X_\beta. \end{cases} \tag{4.14}$$

where the k_β are polynomials defined as

$$k_\beta(x, y) = \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} Z_d(z, x)^T Q_1 Z_d(z, y) dz + k_0(x, y),$$

where $x - y \in X_\beta$, $\theta_{\beta,i}(x, y) = \beta_i y_i + (1 - \beta_i) x_i$, $Q_1 = P_{11} - P_{12} - P_{21} + P_{22}$, and $k_0(x, y)$ is a polynomial. To expand $k_\beta(x, y)$, we use multinomial notation for the monomials in Z_d. Specifically, we index the elements of Z_d as $Z_d(z, x)_i = z^{\gamma_i} x^{\delta_i}$ where $\gamma_i, \delta_i \in \mathbb{N}^n$ for $i = 1, \cdots, q$ and where therefore $z^{\gamma_i} x^{\delta_i} = \prod_{j=1}^n z_j^{\gamma_{i,j}} x_j^{\delta_{i,j}}$. Then

$$\begin{aligned} \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} Z_d(z, x)^T Q_1 Z_d(z, y) dz &= \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} \sum_{k,l} (Q_1)_{k,l} \, x^{\delta_k} z^{\gamma_k} z^{\gamma_l} y^{\delta_l} dz \\ &= \sum_{k,l} (Q_1)_{k,l} \, x^{\delta_k} y^{\delta_l} \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} z^{\gamma_k + \gamma_l} dz. \end{aligned} \tag{4.15}$$

Expanding the integrals in (4.15), each has the form

$$\begin{aligned}\int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} z^{\gamma_k+\gamma_l}\, dz &= \prod_{j=1}^{n} \int_{\theta_{\beta,j}(x,y)}^{b_j} z_j^{\gamma_{k,j}+\gamma_{l,j}}\, dz_j \\ &= \prod_{j=1}^{n} \frac{z_j^{\gamma_{k,j}+\gamma_{l,j}+1}}{\gamma_{k,j}+\gamma_{l,j}+1}\Big|_{\theta_{\beta,j}(x,y)}^{b_j} \\ &= \prod_{j=1}^{n} \frac{b_j^{\gamma_{k,j}+\gamma_{l,j}+1}}{\gamma_{k,j}+\gamma_{l,j}+1} - \frac{\theta_{\beta,j}(x,y)^{\gamma_{k,j}+\gamma_{l,j}+1}}{\gamma_{k,j}+\gamma_{l,j}+1}.\end{aligned}$$

Since $\theta_{\beta,j}(x,y)$ is equivalent to $\max(x_j, y_j)$, and can be written as the continuous function,

$$\theta_{\beta,j}(x,y) = \frac{1}{2}(x_j + y_j + |x_j - y_j|),$$

we conclude that $k(x,y)$ is the product and summation of continuous functions and therefore k and the resulting classifiers are both continuous. □

4.3.2 TK kernels are Universal

In addition to continuity, we show that any TK kernel with $P \succ 0$ has the universal property. The following theorem shows that any TK kernel with $P \succ 0$ is necessarily universal.

Theorem 23. *Suppose k is as defined in Eqn.* (4.7) *for some $P \succ 0$, $d \in \mathbb{N}$ and N as defined in Eqn.* (4.10). *Then k is universal for $Y = \mathcal{X} = [a,b]$, $a < b \in \mathbb{R}^n$.*

Proof. Without loss of generality, we assume $Y = \mathcal{X} = [0,1]^n$. If $P > 0$, then there exist ϵ_i such that $P = P_0 + \sum_i \epsilon_i P_i$ where $P_0 > 0$ and

$$P_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} e_1, 0, \ldots, 0 \end{bmatrix}$$

where $\{e_1\}$ is the first canonical basis of $\mathbb{R}^n$. In this case

$$k(x,y) = \hat{k}(x,y) + \underbrace{\prod_{i=1}^{n} \epsilon_i \min\{x_i, y_i\}}_{k_1(x,y)},$$

where $\hat{k}$ is a positive kernel. Since the hypothesis space satisfies the additive property (See [166, 13]), if k_1 is a universal kernel, then k is a universal kernel.

Recall that for a given kernel, the hypothesis space, $\mathcal{H}$, can be characterized as the closure of $\text{span}\{k(y,\cdot) : y \in \mathcal{X}\}$. Now, consider

$$\text{span}\{k_1(y,\cdot) : y \in \mathcal{X}\},$$

which consists of all functions of the form

$$f(x) = \sum_j c_j \prod_{i=1}^{n} \min\{[y_j]_i, x_i\}.$$

Now

$$\min\{[y_j]_i, x_i\} = \begin{cases} x_i, & \text{if } x_i \leq [y_j]_i \\ [y_j]_i, & \text{otherwise.} \end{cases}$$

For $n = 1$, we may construct a triangle function of height 1 centered at y_2 as

$$f(x) = \sum_{i=1}^{3} \frac{\alpha_i}{\epsilon} k_1(y_i, x) = \begin{cases} 0, & \text{if } x < y_1 \\ \delta(x - y_1), & \text{if } y_1 \leq x < y_2 \\ 1 - \delta(x - y_2), & \text{if } y_2 \leq x < y_3 \\ 0, & \text{if } y_3 < x, \end{cases}$$

where $\delta = y_1 - y_2 = y_2 - y_3$, and

$$\alpha_1 = -\delta, \quad \alpha_2 = 2\delta, \quad \alpha_3 = -\delta.$$

By taking the product of triangle functions in each dimension, we obtain the pyramid functions which are known to be dense in the space of continuous functions on a

compact domain (See [143]). We conclude that k_1 is a universal kernel and hence k is universal. □

This theorem implies that even if the degree of the polynomials is small, the kernel is still universal. Specifically, in the case when $n = 1$ and $d = 0$, the set $\mathcal{K}_T^0$ is universal yet contains only three parameters (the elements of the symmetric $P \in \mathbb{R}^{2\times 2}$).

4.3.3 TK kernels are pointwise dense in all kernels

In the previous two subsections, we have shown that TK kernels are continuous and universal. Furthermore, as shown in Section 4.2, the TK class admits a linear parameterization. The remaining question, then, is whether TK kernels are superior in some *performance* metric to other classes of universal kernels such as Gaussian kernels. First, note that the universal property is of the kernel itself and which is extended to a class of kernels by requiring all kernels in that class to satisfy the property. However, although a kernel may be universal, it may not be well-suited to SVM. Expanding on this point, although it is known that any universal kernel may be used to separate a given set of data, it can be shown that for any given set of normalized data, $\{x_i, y_i\}$, there exists a universal kernel, k, for which the objective function of the solution to 1-norm soft margin problem is arbitrarily suboptimal - e.g. by increasing the bandwidth of the Gaussian kernel.

To address the question of performance, we propose the *pointwise density* property. This property is defined on a set of kernels and guarantees that there is some kernel in the set of kernels for which the solution to the 1-norm soft margin problem is optimal. Specifically, we have the following.

Definition 24. *The set of kernels $\mathcal{K}$ is said to be* ***pointwise dense*** *if for any positive kernel, k^*, any set of data $\{x_i\}_{i=1}^m$, and any $\epsilon > 0$, there exists $k \in \mathcal{K}$ such*

that $\|k(x_i, x_j) - k^*(x_i, x_j)\| \leq \epsilon$.

This definition implies that a set of kernels can approximate any given positive kernel arbitrarily well. To illustrate the importance of the pointwise density property we show that for a large class of kernel learning problems, the value of the optimal kernel is not pointwise positive - i.e. $k(x, y) \not\geq 0$ for all $x, y \in \mathcal{X}$. This is significant because almost all commonly used kernels are pointwise non-negative. Indeed we find that the elements of the optimal kernel matrix are negative as frequently as they are positive.

Optimal kernels are not pointwise positive

To demonstrate the necessity of negative values in optimal kernel matrices, we will analytically solve an SDP to find the optimal kernel matrix for the 1-norm SVM KL problem.

The Optimal Kernel Matrix for 1-norm SVM

To demonstrate the necessity of negative values in optimal kernel matrices, we analytically solve the following SDP derived from Optimization Problem (4.4) which determines the optimal kernel matrix (K^*) given the labels y of a problem and a "penalty" parameter C, but with no constraint on the form of the kernel function (other than it be PD).

$$\begin{aligned} \min_{t\in\mathbb{R}, K\in\mathbb{R}^{m\times m}, \gamma\in\mathbb{R}, \nu\in\mathbb{R}^m, \delta\in\mathbb{R}^m} \quad & t, \\ \text{subject to:} \quad & \begin{pmatrix} G & e+\nu-\delta+\gamma y \\ (e+\nu-\delta+\gamma y)^T & t - C\delta^T e \end{pmatrix} \geq 0 \\ & \nu \geq 0, \quad \delta \geq 0, \quad K \geq 0, \quad \text{trace}(K) = m, \quad G_{i,j} = y_i K_{i,j} y_j \end{aligned} \tag{4.16}$$

The following theorem finds an analytic solution of this optimization problem.

Theorem 25. *Let $y_i \in \{1, -1\}$ for $i = 1, \cdots, m$ and $C \geq \frac{2}{m}$, then the solution to Optimization Problem 4.16 is,*

$$\nu^* = 0, \quad \gamma^* = -\sum_{i=1}^{m} \frac{y_i}{m}, \quad \delta^* = 0, \quad t^* = \frac{\|e - \gamma^* y\|_2}{m},$$

and $K^* = \frac{m}{\|e+\gamma^* y\|_2^2} \mathcal{Y}(e + \gamma^* y)(e + \gamma^* y)^T \mathcal{Y}$ where $\mathcal{Y} = diag(y)$.

Proof. We first show that $K^* = U \Sigma U^T$, where

$$U = \mathcal{Y} \begin{bmatrix} \frac{(e+\nu^*-\delta^*+\gamma^* y)}{\|(e+\nu^*-\delta^*+\gamma^* y)\|_2} & \cdots \end{bmatrix}, \Sigma = \begin{bmatrix} m & 0 \\ 0 & 0 \end{bmatrix}.$$

Optimization Problem (4.16) is equivalent to

$$\min_{K \in \mathbb{R}^{m \times m}, \gamma \in \mathbb{R}, \nu \in \mathbb{R}^m, \delta \in \mathbb{R}^m} \quad (e + \nu - \delta + \gamma y)^T (\mathcal{Y} K \mathcal{Y})^{-1} (e + \nu - \delta + \gamma y) + 2C\delta^T e \quad (4.17)$$

$$\text{subject to:} \quad \nu \geq 0, \qquad \delta \geq 0, \qquad K \geq 0, \qquad \text{trace}(K) = m.$$

This problem can be separated into subproblems as

$$\min_{\substack{\gamma \in \mathbb{R}, \nu \in \mathbb{R}^m, \delta \in \mathbb{R}^m \\ \nu \geq 0,\ \delta \geq 0}} \qquad \min_{\substack{K \in \mathbb{R}^{m \times m}, \\ K \geq 0,\ \text{trace}(K) = m}} \quad (e + \nu - \delta + \gamma y)^T (\mathcal{Y} K \mathcal{Y})^{-1} (e + \nu - \delta + \gamma y) + 2C\delta^T e.$$

Now, for any feasible K, we have that $K \geq 0$ and $\bar{\sigma}(K) \leq m$ and hence

$$\begin{aligned} (e + \nu - \delta + \gamma y)^T (\mathcal{Y} K \mathcal{Y})^{-1} (e + \nu - \delta + \gamma y) &\geq \frac{1}{\bar{\sigma}(K)} \|e + \nu - \delta + \gamma y\|_2^2 \\ &\geq \frac{1}{m} \|e + \nu - \delta + \gamma y\|_2^2 . \end{aligned}$$

Now, we propose $K = U \Sigma U^T$ and show that it is optimal, where

$$U = \mathcal{Y} \begin{bmatrix} \frac{(e+\nu^*-\delta^*+\gamma^* y)}{\|(e+\nu^*-\delta^*+\gamma^* y)\|_2} & V \end{bmatrix}, \quad \Sigma = \begin{bmatrix} m & 0 \\ 0 & 0 \end{bmatrix}.$$

and V is any unitary completion of the matrix U. Then $K \succeq 0$, $\text{trace}(K) = m$, and

$$
\begin{aligned}
&(e+\nu-\delta+\gamma y)^T (\mathcal{Y}K\mathcal{Y})^{-1}(e+\nu-\delta+\gamma y) \\
&= (e+\nu-\delta+\gamma y)^T \left(\left[\tfrac{(e+\nu-\delta+\gamma y)}{\|(e+\nu-\delta+\gamma y)\|_2} V \right] \begin{bmatrix} m & 0 \\ 0 & 0 \end{bmatrix} \left[\tfrac{(e+\nu-\delta+\gamma y)}{\|(e+\nu-\delta+\gamma y)\|_2} V \right]^T \right)^{-1} (e+\nu-\delta+\gamma y) \\
&= \frac{\|(e+\nu-\delta+\gamma y)\|_2^2}{m}.
\end{aligned}
$$

We conclude that this K solves the first sub-problem and hence Optimization Problem (4.16) reduces to

$$\min_{\substack{\gamma\in\mathbb{R},\nu\in\mathbb{R}^m,\delta\in\mathbb{R}^m \\ \nu\geq 0,\ \delta\geq 0,}} \quad \frac{\|(e+\nu-\delta+\gamma y)\|_2^2}{m} + 2C\delta^T e. \tag{4.18}$$

Now let $\nu^*, \delta^*, \gamma^*$ be as defined in the theorem statement. For the convex objective

$$f(\delta,\nu,\gamma) = \frac{\|e+\nu-\delta+\gamma y\|_2^2}{m} + 2C\delta^T e$$

let $\bar{y} = \frac{1}{m}\sum_{i=1}^m y_i$ and we have that

$$\frac{\partial f}{\partial \nu_i}(\nu^*,\delta^*,\gamma^*) = \frac{2+2\bar{y}y_i}{m} \geq \frac{2+-2(1)(1)}{m} \geq 0,$$

and for $C \geq \frac{2}{m}$

$$\frac{\partial f}{\partial \delta_i}(\nu^*,\delta^*,\gamma^*) = \frac{2mC-2-2\bar{y}y_i}{m} \geq \frac{4-2-2(1)(1)}{m} \geq 0.$$

Finally

$$\frac{\partial f}{\partial \gamma}(\nu^*,\delta^*,\gamma^*) = \frac{1}{m}\sum_{i=1}^m 2y_i - 2\bar{y} = -2\bar{y} + 2\frac{1}{m}\sum_{i=1}^m y_i = 0.$$

Hence the KKT conditions are satisfied and since the optimization problem is convex, $(\nu^*,\delta^*,\gamma^*)$ is optimal. □

This result shows that for binary labels, the optimal kernel matrix has an analytic solution. Furthermore, if we consider the case where $\sum_{i=1}^m y_i = 0$, then $\lambda^* = 0$ and

hence $K^* = yy^T$ and $K^*_{i,j} = y_i y_j$. This implies that the optimal kernel matrix consists of an equal number of positive and negative entries - meaning that kernels functions with globally positive values will not be able to approximate the optimal kernel matrix well. Furthermore, for values of C less than $\frac{2}{m}$, we numerically find that the same kernel matrix is still optimal - only the values of δ^* and γ^* are different.

The GPK and TK classes are pointwise dense in all kernels

Having demonstrated the significance of pointwise density, we now establish that both the GPK and TK kernel sets satisfy this property. For this analysis, we relax the strict positivity constraint $P > 0$ in the definition of the TK class. In this case, the GPK class becomes a subset of the TK class. We then prove pointwise density of the GPK class - a property which is then inherited by the TK class. The following lemma shows that the GPK class is a subset of the TK class.

Lemma 26. $\mathcal{K}^d_P \subset \mathcal{K}^d_T$

Proof. If $k_p \in \mathcal{K}^d_P$, there exists a $P_1 \geq 0$ such that $k_p(x, y) = Z_d(x)^T P_1 Z_d(y)$. Now let J be the matrix such that $JZ_d(z, x) = Z_d(x)$ and define

$$P = \frac{1}{\prod_{j=1}^n (b_j - a_j)} \begin{bmatrix} J^T P_1 J & J^T P_1 J \\ J^T P_1 J & J^T P_1 J \end{bmatrix} \geq 0.$$

Now let k be as defined in Equation (4.7). Then $k \in \mathcal{K}_T^d$ and

$$\begin{aligned}
k(x,y) =& \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_{p^*(x,y)}^b Z_d(z,x)^T J^T \left(P_1 - P_1 - P_1 + P_1\right) J Z_d(z,y) dz \\
&+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_x^b Z_d(z,x)^T J^T \left(P_1 - P_1\right) J Z_d(z,y) dz \\
&+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_y^b Z_d(z,x)^T J^T \left(P_1 - P_1\right) J Z_d(z,y) dz \\
&+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_a^b Z_d(z,x)^T J^T P_1 J Z_d(z,y) dz \\
=& \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_a^b Z_d(x)^T P_1 Z_d(y) dz \\
=& k_p(x,y).
\end{aligned}$$

We conclude that $k_p = k \in \mathcal{K}_T^d$. □

We now use polynomial interpolation to prove that GPK kernels are pointwise dense.

Theorem 27. *For any kernel matrix K^* and any finite set $\{x_i\}_{i=1}^m$, there exists a $d \in \mathbb{N}$ and $k \in \mathcal{K}_P^d$ such that if $K_{i,j} = k(x_i, x_j)$, then $K = K^*$.*

Proof. Since $K^* \geq 0$, $K^* = M^T M$ for some M. Using multivariate polynomial interpolation (as in [60]), for sufficiently large d, we may choose Q such that

$$Q \begin{bmatrix} Z_d(x_1) & \cdots & Z_d(x_m) \end{bmatrix} = M.$$

Now let

$$k(x,y) = Z_d(x)^T P Z_d(y)$$

where $P = Q^T Q \geq 0$. Now partition M as

$$M = \begin{bmatrix} m_1 & \cdots & m_m \end{bmatrix}.$$

Then $QZ_d(x_i) = m_i$ and hence

$$\begin{aligned} K_{ij} &= Z_d(x_i)^T Q^T Q Z_d(x_j) \\ &= m_i^T m_j \\ &= K^*_{ij}. \end{aligned}$$

□

GPK and TK kernels converge quickly to the optimal kernel

We next show that in practice, a degree of only 4 or 5 can be sufficient to approximate the optimal kernel matrix with minimal error using either GPK and TK kernels.

Specifically, we consider the problem of approximating the optimal kernel matrix for a given set of data $\{x_i\}$ and given set of kernels, $\mathcal{K}$, using both the element-wise matrix $\|\cdot\|_1$ and $\|\cdot\|_\infty$ norms.

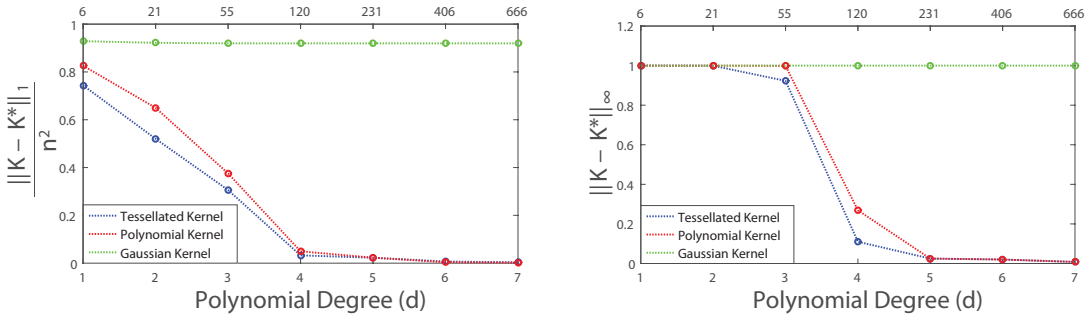
$$\min_{k\in\mathcal{K}} \frac{\|K - K^*\|_1}{n^2} \quad s.t. \;\; K_{i,j} = k(x_i, x_j) \tag{4.19}$$

$$\min_{k\in\mathcal{K}} \|K - K^*\|_\infty \quad s.t. \;\; K_{i,j} = k(x_i, x_j) \tag{4.20}$$

The sets of kernel functions we consider are: $\mathcal{K}_G^\gamma$ - the sum of K Gaussians with bandwiths γ_i; $\mathcal{K}_P^d$ - the GPKs of degree d; and $\mathcal{K}_T^d$ - the TK kernels of degree d. That is, we choose $\mathcal{K} \in \{\mathcal{K}_G^\gamma,\ \mathcal{K}^d,\ \mathcal{K}_T^d\}$ where for convenience, we define the class of sums of Gaussian kernels of bandwidths $\gamma \in \mathbb{R}^K$ as follows.

$$\mathcal{K}_G^\gamma := \left\{ k \ : \ k(x,y) = \sum_{i=1}^K \mu_i \mathrm{e}^{\frac{||x-y||_2^2}{\gamma_i}} \ : \ \mu_i > 0 \right\} \tag{4.21}$$

We now solve Optimization Problems (4.19) and (4.20) for $\mathcal{K}_G^\gamma$, $\mathcal{K}_P^d$, and $\mathcal{K}_T^d$ as a function of the degree of the polynomials, d and the number of bandwidths selected (K). For this test, we use the spiral data set with 20 samples and corresponding labels such that $\sum_{i=1}^m y_i = 0$. Since half of the entries in K^* are -1, and since the



(a) $\frac{\|K-K^*\|_1}{n^2}$ for the TK and GPK classes of degree d and for a positive combination of m Gaussian kernels.

(b) $\|K-K^*\|_\infty$ for the TK and GPK classes of degree d and for a positive combination of m Gaussian kernels.

Figure 4.2: The objective of Optimization Problem 4.19 and 4.20 for the TK and GPK classes of degree d and for a positive combination of m Gaussian kernels with bandwidths ranging from .01 to 10. The number of bandwidths is selected so that the number of decision variables (displayed above the figure) match in the Gaussian and in the TK kernel case.

Gaussian kernel is globally positive, it is easy to see that for $\mathcal{K} = \mathcal{K}_G^\gamma$ the minimum objective values of Optimization Problems (4.19) and (4.20) are lower bounded by 0.5 and 1 respectively, irrespective of the choice of bandwidths, γ_i and number of data points. In Figs. 4.2(a) and 4.2(b) we numerically show the change in the objective value of Optimization Problems (4.19) and (4.20) for the optimal Gaussian, GPK, and TK kernels as we increase the complexity of the kernel function. For the TK and GPK kernel functions, we increase the complexity of the kernel function by increasing the degree of the monomial basis while scaling the x-axis to ensure equivalent computational complexity.

The results demonstrate that, as expected, the Gaussian kernel saturates with an objective value significantly larger than the lower bound of 0.5 for the 1-norm and exactly at 1 for the ∞-norm (the projected lower bound). Meanwhile, as the degree

increases, both the GPK and TK kernels are able to approximate the kernel matrix arbitrarily well, with almost no error at degree $d = 7$. Furthermore, the TK kernels converge somewhat faster.

4.4 SDP formulation of the TK kernel learning algorithm

Subsection 4.1.2 gave a convex formulation of the kernel learning problem using the convex constraint $k \in \mathcal{K}$. Having now defined the TK class of kernels, we now address specific implementations of the TK kernel learning problem using both an SDP method based on Optimization Problem (4.4) and a method that directly solves the minimax formulation. In both cases, our goal for this section is to define an explicit linear map from the elements of the positive matrix variable, P, to the values of the kernel function $k(x_i, x_j)$.

To construct our mapping, we first create an index of the elements in the basis $Z_d(z, x)$ which is used in $N_T^d(z, x)$ as defined in Eqn. (4.10). Recall $Z_d(z, x)$ is a vector of all monomials of degree d or less of length $q := \binom{d+2n}{d}$. We now specify that the elements of Z_d are ordered, and by default we use lexicographical ordering on the exponents of the variables of the monomials. Specifically, we denote the jth monomial in $Z_d(z, x)$ where $z, x \in \mathbb{R}^n$ as $z^{\gamma_j} x^{\delta_j} := \prod_{i=1}^n z_i^{\gamma_{j,i}} x_i^{\delta_{j,i}}$ where $\gamma_j, \delta_j \in \mathbb{N}^n$ and $\{[\gamma_j, \delta_j]\}_{j=1}^q$ is ordered lexicographically. Note that $\{[\gamma_j, \delta_j]\}_{j=1}^q = \{x \in \mathbb{N}^{2n} : \|x\|_1 \leq d\}$. Using this notation, we have the following representation of the TK kernel k.

Corollary 28. *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a, b]$, and $d \in \mathbb{N}$ we define the finite set $D_d := \{(\gamma, \delta) \in \mathbb{N}^{2n} : \|(\gamma, \delta)\|_1 \leq d\}$. Let $\{[\gamma_i, \delta_i]\}_{i=1}^q \subseteq D_d$ be some ordering of D_d and define $Z_d(z, x)_j = z^{\gamma_j} x^{\delta_j}$. Now let k be as defined in Eqn. (4.7) for some $P > 0$ and where N is as defined in Eqn. (4.10). If we partition $P = \begin{bmatrix} Q & R \\ R^T & S \end{bmatrix}$ then*

we have,

$$k(x,y) = \sum_{i,j=1}^{q} Q_{i,j} g_{i,j}(x,y) + R_{i,j} t_{i,j}(x,y) + R_{i,j}^T t_{i,j}(y,x) + S_{i,j} h_{i,j}(x,y)$$

where $g_{i,j}, t_{i,j}, h_{i,j} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ *are defined as*

$$\begin{aligned} g_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(p^*(x,y), b, \gamma_i + \gamma_j + \mathbf{1}) \\ t_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(x, b, \gamma_i + \gamma_j + \mathbf{1}) - g_{i,j}(x,y) \\ h_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(a, b, \gamma_i + \gamma_j + \mathbf{1}) - g_{i,j}(x,y) - t_{i,j}(x,y) - t_{i,j}(y,x), \end{aligned} \tag{4.22}$$

where $\mathbf{1} \in \mathbb{N}^n$ *is the vector of ones,* $p^* : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ *is defined elementwise as* $p^*(x,y)_i = \max\{x_i, y_i\}$*, and* $T : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{N}^n \to \mathbb{R}$ *is defined as*

$$T(x,y,\zeta) = \prod_{j=1}^{n} \left(\frac{y_j^{\zeta_j}}{\zeta_j} - \frac{x_j^{\zeta_j}}{\zeta_j} \right).$$

Proof. The proof follows from Theorem 22. □

Using a linear map from the elements of P to the value of $k(x,y)$, we may now write the SDP version of the TK kernel learning problem as follows.

$$\min_{t\in\mathbb{R},\ \gamma\in\mathbb{R},\ \nu\in\mathbb{R}^m,\ \delta\in\mathbb{R}^m,\ Q,R,S\in\mathbb{R}^{q\times q}} \quad t \tag{4.23}$$

$$\text{subject to: } \begin{pmatrix} G(P) & e + \nu - \delta + \gamma y \\ (e + \nu - \delta + \gamma y)^T & t - \frac{2}{m\lambda}\delta^T e \end{pmatrix} \geq 0,$$

$$\nu \geq 0, \qquad \delta \geq 0, \qquad P = \begin{bmatrix} Q & R \\ R^T & S \end{bmatrix} > 0, \qquad \operatorname{trace}(P) \leq 1,$$

$$G_{k,l}(P) = y_k y_l \sum_{i,j=1}^{q} Q_{i,j} g_{i,j}(x_k, x_l) + R_{i,j} t_{i,j}(x_k, x_l) + R_{i,j}^T t_{i,j}(x_l, x_k) + S_{i,j} h_{i,j}(x_k, x_l)$$

Optimization Problem (4.23), then, is an SDP and can, therefore, be solved efficiently using standard SDP solvers such as MOSEK [4]. Note that we use the trace constraint to ensure the kernel function is bounded.

Typically SDP problems require roughly p^2n^2 number of operations, where p is the number of decision variables and n is the dimension of the SDP constraint (See [44]). The number of decision variables in (4.23) is moderate, increasingly linearly in the number of training data points and the number of elements of P. However, this optimization problem has a semi-definite matrix constraint whose dimension is linear in m, the number of training data. As we will see in Section 4.6, the increase in training data increases n and limits the amount of training data that can be processed using Optimization Problem (4.23). To improve the scalability of the algorithm, we consider a minimax formulation.

4.5 Minimax formulation of the TK kernel learning algorithm

In this section, we formulate the KL optimization problem as a minimax saddle point problem for classification and regression. This formulation enables a decomposition into convex primal and dual sub-problems, $OPT\_A(P)$ and $OPT\_P(\alpha)$ with no duality gap. We then consider the Frank-Wolfe algorithm and show using Danskin's Theorem that the gradient step can be efficiently computed using the primal and dual sub-problems. Finally, we propose efficient algorithms for computing $OPT\_A(P)$ and $OPT\_P(\alpha)$: in the former case using an efficient SMO algorithm for convex QP and in the latter case, using an analytic solution based on the SVD.

4.5.1 Primal-Dual Decomposition

For convenience, we define the feasible sets for the sub-problems as

$$\begin{aligned}
\mathcal{X} &:= \{P \in \mathbb{R}^{q\times q} \ : \ \text{trace}(P) = q, \ P > 0\} \\
\mathcal{Y}_c &:= \{\alpha \in \mathbb{R}^m \ : \ \sum\nolimits_{i=1}^{m} \alpha_i y_i = 0, \ 0 \le \alpha_i \le C\}, \\
\mathcal{Y}_r &:= \{\alpha \in \mathbb{R}^m \ : \ \sum\nolimits_{i=1}^{m} \alpha_i = 0, \ \alpha_i \in [-C, C]\}.
\end{aligned}$$

In this section, we typically use the generic form $\mathcal{Y}_*$ to refer to either $\mathcal{Y}_c$ or $\mathcal{Y}_r$ depending on whether the algorithm is being applied to the classification or regression problem. To define the objective function we use $\lambda(\alpha, P)$ to indicate

$$\lambda(\alpha, P) := -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j\int_a^b N_T^d(z, x_i)^T P N_T^d(z, y_j)dz, \tag{4.24}$$

where N_T^d are as defined in Eqn. (4.10). Additionally, we have $\kappa_c(\alpha) := \sum_{i=1}^m \alpha_i$ and

$$\kappa_r(\alpha) := -\epsilon\sum\nolimits_{i=1}^{m}|\alpha_i| + \sum\nolimits_{i=1}^{m} y_i\alpha_i.$$

where, again, we use $\kappa_* = \kappa_c$ for classification and $\kappa_* = \kappa_r$ for regression.

The KL optimization problem (OPT) for TK kernels is now defined as the following minimax saddle point optimization problem.

$$OPT_P := \min_{P\in\mathcal{X}}\max_{\alpha\in\mathcal{Y}_*} \quad \lambda(e_*\odot\alpha, P) + \kappa_*(\alpha), \tag{4.25}$$

where $\odot$ indicates elementwise multiplication, $e_c = y$ (vector of labels) for classification, and $e_r = \mathbf{1}_m$ (vector of ones) for regression.

Minimax Duality To find the dual of the KL optimization problem, we formulate two sub-problems:

$$OPT\_A(P) \ := \max_{\alpha\in\mathcal{Y}_*}\lambda(e_*\odot\alpha, P) + \kappa_*(\alpha) \tag{4.26}$$

and

$$OPT\_P(\alpha) := \min_{P\in\mathcal{X}} \quad \lambda(e_*\odot\alpha, P) + \kappa_*(\alpha). \tag{4.27}$$

Now, we have that

$$OPT_P = \min_{P\in\mathcal{X}} \ OPT\_A(P)$$

and its dual is

$$OPT_D = \max_{\alpha\in\mathcal{Y}_*} OPT\_P(\alpha) = \max_{\alpha\in\mathcal{Y}_*}\min_{P\in\mathcal{X}} \quad \lambda(e_*\odot\alpha, P) + \kappa_*(\alpha). \tag{4.28}$$

The following lemma states that there is no duality gap between OPT_P and OPT_D - a property we will use in our termination criterion.

Lemma 29. *$OPT_P = OPT_D$. Furthermore, $\{\alpha^*, P^*\}$ solve OPT_P if and only if $OPT\_P(\alpha^*) = OPT\_A(P^*)$.*

Proof. For any minmax optimization problem with objective function ϕ, we have

$$d^* = \max_{\alpha \in \mathcal{Y}} \min_{P \in \mathcal{X}} \phi(P, \alpha) \leq \min_{P \in \mathcal{X}} \max_{\alpha \in \mathcal{Y}} \phi(P, \alpha) = p^*,$$

and strong duality holds ($p^* - d^* = 0$) if $\mathcal{X}$ and $\mathcal{Y}$ are both convex and one is compact, $\phi(\cdot, \alpha)$ is convex for every $\alpha \in \mathcal{Y}$ and $\phi(P, \cdot)$ is concave for every $P \in \mathcal{X}$, and the function ϕ is continuous [53]. In our case, these conditions hold for both classification and regression where $\phi(P, \alpha) = \lambda(\alpha, P) + \kappa_r(\alpha)$ or $\lambda(\alpha \odot y, P) + \kappa_c(\alpha)$. Hence $OPT_P = OPT_D$. Furthermore, if $\{\alpha^*, P^*\}$ solve OPT_P then

$$OPT\_P(\alpha^*) = \max_{\alpha \in \mathcal{Y}} OPT\_P(\alpha) = \min_{P \in \mathcal{X}} OPT\_A(P) = OPT\_A(P^*).$$

Conversely, suppose $\alpha \in \mathcal{Y}$, $P \in \mathcal{X}$, then

$$\begin{aligned} OPT\_P(\alpha) \leq \max_{\alpha \in \mathcal{Y}} \; OPT\_P(\alpha) &= OPT\_P(\alpha^*) = OPT\_A(P^*) \\ &= \min_{P \in \mathcal{X}} OPT\_A(P) \leq OPT\_A(P). \end{aligned}$$

Hence if $OPT\_A(P) = OPT\_P(\alpha)$, then $OPT\_A(P) = OPT\_A(P^*) = OPT\_P(\alpha^*) = OPT\_P(\alpha)$ and hence P and α solve OPT_A and OPT_P, respectively. □

Finally, we note that $OPT\_A(P)$ is convex with respect to P - a property we will use in Thm. 33.

Lemma 30. *Let $OPT\_A(P)$ be as defined in 4.26. Then, the function $OPT\_A(P)$ is convex with respect to P.*

Algorithm 1: The Frank-Wolfe Algorithm for matrices.

`Initialize` P_0 as any point in $\mathcal{X}$.;

Step 1: $S_k = \arg\min_{S \in \mathcal{X}} \langle \nabla_Q f(Q)|_{Q=P_k}, S \rangle$

Step 2: $\gamma_k = \arg\min_{\gamma \in [0,1]} f(P_k + \gamma(S_k - P_k))$

Step 3: $P_{k+1} = P_k + \gamma_k (S_k - P_k), k = k + 1$, return to step 1.

Proof. First without loss of generality let $e_* \in \mathbb{R}^m$ be a vector of ones and $e_* \odot \alpha = \alpha$. The function $OPT\_A(P) : \mathbb{R}^{n \times n} \to \mathbb{R}$ is convex if and only if for any $X, V \in \mathcal{X}$, $g(t) := OPT\_A(X + tV)$ is convex in t for all $t \in H := \{s \in \mathbb{R} \,|\, X + sV \succeq 0\}$. To prove convexity of $g(t)$ we must prove that,

$$g(\theta t_1 + (1-\theta)t_2) \leq \theta g(t_1) + (1-\theta) g(t_2)$$

for any $\theta \in [0, 1]$ and $t_1, t_2 \in H$. Since $P_1 = X + t_1 V$ and $P_2 = X + t_2 V$ are arbitrary positive semi-definite matrices and $\lambda(\alpha, P)$ is linear with respect to P we have that,

$$\begin{aligned}
g(\theta t_1 + (1-\theta)t_2) &= \max_{\alpha \in \mathcal{Y}_*} \lambda(\alpha, \theta P_1 + (1-\theta) P_2) + \kappa_*(\alpha) \\
&= \max_{\alpha \in \mathcal{Y}_*} \theta \left(\lambda(\alpha, P_1) + \kappa_*(\alpha)\right) + (1-\theta) \left(\lambda(\alpha, P_2) + \kappa_*(\alpha)\right) \\
&\leq \max_{\alpha \in \mathcal{Y}_*} \theta \left(\lambda(\alpha, P_1) + \kappa_*(\alpha)\right) + \max_{\alpha \in \mathcal{Y}_*} (1-\theta) \left(\lambda(\alpha, P_2) + \kappa_*(\alpha)\right) \\
&\leq \theta g(t_1) + (1-\theta) g(t_2).
\end{aligned}$$

□

4.5.2 Primal-Dual Frank-Wolfe Algorithm

For an optimization problem of the form, $\min_{S \in \mathcal{X}} f(S)$, where $\mathcal{X}$ is a convex subset of matrices and $\langle \cdot, \cdot \rangle$ is the Frobenius matrix inner product, the Frank-Wolfe (FW) algorithm is defined as in Algorithm 1. In our case, we have $f(Q) = OPT\_A(Q)$ so

that

$$OPT_P = \min_{P \in \mathcal{X}} OPT\_A(P).$$

Unfortunately, implementation of the FW algorithm requires us to compute $\nabla_Q OPT\_A(Q)|_{Q=P_k}$ at each iteration. Fortunately, as shown in Subsections 4.5.3 and 4.5.4, we may efficiently solve the sub-problems OPT_A and OPT_P. Furthermore, in Theorem 32, we will show that these sub-problems can be used to efficiently compute the gradient $\nabla_Q OPT\_A(Q)|_{Q=P_k}$ - allowing for an efficient implementation of the FW algorithm. Theorem 32 uses Danskin's theorem, shown below as abridged from [10].

Proposition 31 (Danskin's Theorem-[10])**.** *Let $\mathcal{Y} \subset \mathbb{R}^m$ be a compact set, and let $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be continuous such that $\phi(\cdot, \alpha) : \mathcal{X} \to \mathbb{R}$ is convex for each $\alpha \in \mathcal{Y}$. Then if,*

$$\mathcal{Y}_0(P) = \left\{ \bar{\alpha} \;\; | \;\; \phi(P, \bar{\alpha}) = \max_{\alpha \in \mathcal{Y}} \phi(P, \alpha) \right\}.$$

consists of only one unique point, $\bar{\alpha}$, and $\phi(\cdot, \bar{\alpha})$ is differentiable at P then $f(P) = \max_{\alpha \in \mathcal{Y}} \phi(P, \alpha)$ is differentiable at P and

$$\nabla_P f(P) = \nabla_P \phi(P, \bar{\alpha}),$$

where $\nabla_P \phi(P, \bar{\alpha})$ is the vector with coordinates

$$\frac{\partial \phi(P, \bar{\alpha})}{\partial P_i}, \quad i = 1, ..., n.$$

Using Danskin's Theorem we may now efficiently calculate the partial derivative for step 1a of the KL algorithm for TKs.

Lemma 32. *If OPT_A and OPT_P are as defined in Eqns.* (4.26) *and* (4.27)*, then for any $P_k \geq 0$, we have*

$$\arg\min_{S \in \mathcal{X}} \langle \nabla_Q OPT\_A(Q)|_{Q=P_k}, S \rangle = \arg OPT\_P(\arg OPT\_A(P_k)).$$

Proof. For simplicity, we define $D(\alpha)$ as in Eqn. (4.29) such that $\lambda(e_* \odot \alpha, P) := \langle D(\alpha), P\rangle$. Now, since $\lambda(\alpha, P)$ is strictly convex in α, for any $P_k > 0$, $OPT\_A(P_k)$ has a unique solution and hence we have by Danskin's Theorem that

$$\begin{aligned}\arg\min_{S\in\mathcal{X}} \left\langle \nabla_Q OPT\_A(Q)|_{Q=P_k}, S\right\rangle &= \arg\min_{S\in\mathcal{X}} \left\langle \nabla_Q \left[\max_{\alpha\in\mathcal{Y}_*} (\langle D(\alpha), Q\rangle + \kappa_*(\alpha))\right]_{Q=P_k}, S\right\rangle \\ &= \arg\min_{S\in\mathcal{X}} \left\langle \nabla_Q \left[\langle D(\bar{\alpha}), Q\rangle + \kappa_*(\bar{\alpha})\right]_{Q=P_k}, S\right\rangle\end{aligned}$$

where $\bar{\alpha} = \arg OPT\_A(P_k)$. Hence,

$$\begin{aligned}\arg\min_{S\in\mathcal{X}} \left\langle \nabla_Q \left[\langle D(\bar{\alpha}), Q\rangle + \kappa_*(\bar{\alpha})\right]_{Q=P_k}, S\right\rangle &= \arg\min_{S\in\mathcal{X}} \left\langle \nabla_Q \left[\langle D(\bar{\alpha}), Q\rangle\right]_{Q=P_k}, S\right\rangle \\ &= \arg\min_{S\in\mathcal{X}} \left\langle D(\bar{\alpha}), S\right\rangle \\ &= \arg OPT\_P(\bar{\alpha}) \\ &= \arg OPT\_P(\arg OPT\_A(P_k)).\end{aligned}$$

□

We now propose the efficient implementation of the FW algorithm, as defined in Algorithm 2, based on efficient algorithms for computing OPT_A and OPT_P as will be defined in Subsections 4.5.3 and 4.5.4.

In the following theorem, we use convergence properties of the FW algorithm to show that Algorithm 2 has worst-case linear convergence.

Theorem 33. *Algorithm 2 returns iterates P_k and α_k such that, $|\lambda(\alpha_k, P_k) + \kappa_*(\alpha_k) - OPT_P| < O(\frac{1}{k})$.*

Proof. If we define $f = OPT\_A$, then Theorem 32 shows that f is differentiable and, if the P_k satisfy Algorithm 2, that the P_k also satisfy Algorithm 1. In addition, Lemma 30 shows that $f(Q) = OPT\_A(Q)$ is convex in Q. It has been shown in, e.g. [77], that if $\mathcal{X}$ is convex and compact and $f(Q)$ is convex and differentiable on

Algorithm 2: An Efficient FW Algorithm for TKL. Note that the stopping criterion is defined using the duality gap $OPT\_P(\alpha_k) - OPT\_A(P_k) > 0$, which is equivalent to the stopping criterion used in the standard FW algorithm.

`Initialize` $P_0 = I$, $k = 0$, $\alpha_0 = OPT\_A(P_0)$;

while $OPT\_P(\alpha_k) - OPT\_A(P_k) \geq \epsilon$ **do**

Step 1a: $\alpha_k = \arg OPT\_A(P_k)$

Step 1b: $S_k = \arg OPT\_P(\alpha_k)$

Step 2: $\gamma_k = \arg \min_{\gamma \in [0,1]} OPT\_A(P_k + \gamma(S_k - P_k))$

Step 3: $P_{k+1} = P_k + \gamma_k(S_k - P_k)$, $k = k + 1$

end while

$Q \in \mathcal{X}$, then the FW Algorithm produces iterates P_k, such that, $f(P_k) - f(P^*) < O(\frac{1}{k})$ where

$$f(P^*) = \min_{P \in \mathcal{X}} f(P) = \min_{P \in \mathcal{X}} OPT\_A(P) = OPT_P.$$

Finally, we note that

$$\begin{aligned} &\lambda(\alpha_k, P_k) + \kappa_*(\alpha_k) \\ &= \lambda(\arg OPT\_A(P_k), P_k) + \kappa_*(\arg OPT\_A(P_k)) \\ &= \max_{\alpha \in \mathcal{Y}_*} \lambda(\alpha, P_k) + \kappa_*(\alpha) = OPT\_A(P_k) = f(P_k) \end{aligned}$$

which completes the proof. □

In the following subsections, we provide efficient algorithms for computing the sub-problems OPT_A and OPT_P.

4.5.3 Step 1, Part A: Solving $OPT\_A(P)$

For a given $P > 0$, $OPT\_A(P)$ is a convex Quadratic Program (QP). General purpose QP solvers have a worst-case complexity which scales as $O(m^3)$ [173] where,

when applied to OPT_A, m becomes the number of samples. This computational complexity may be improved, however, by noting that OPT_A is compatible with the representation defined in [22] for QPs derived from SVM. In this case, the algorithm in LibSVM [22], can reduce the computational burden somewhat. This improved performance is illustrated in Figure 4.5 where we observe the achieved complexity scales as $O(m^{2.1})$. Note that for the 2-step algorithm proposed in this manuscript, solving the QP in $OPT\_A(P)$ is significantly slower that solving the Singular Value Decomposition (SVD) required for $OPT\_P(\alpha)$, which is defined in the following subsection. However, the achieved complexity of $O(m^{2.1})$ is also significantly faster than solving the SDPs described in [96, 114, 29]. This complexity comparison will be further discussed in Section 4.6.

4.5.4 Step 1, Part B: Solving $OPT\_P(\alpha)$

For a given α, $OPT\_P(\alpha)$ is an SDP. Fortunately, however, this SDP is structured so as to admit an analytic solution using the SVD.

To solve $OPT\_P(\alpha)$ we minimize $\lambda(e_* \odot \alpha, P)$ from Eq. (4.24) which, as per Corollary 28, is linear in P and can be formulated as

$$OPT\_P(\alpha) := \min_{\substack{P\in\mathbb{R}^{q\times q}\\ \text{trace}(P)=q\\ P>0}} \lambda(e_* \odot \alpha, P) := \min_{\substack{P\in\mathbb{R}^{q\times q}\\ \text{trace}(P)=q\\ P>0}} \langle D(\alpha), P\rangle$$

where,

$$D_{i,j}(\alpha) = \sum\nolimits_{k,l=1}^{m} \alpha_k G_{i,j}(x_k, x_l)\alpha_l, \quad G_{i,j}(x,y) := \begin{cases} g_{i,j}(x,y) & \text{if } i \le \frac{q}{2}, j \le \frac{q}{2} \\ t_{i,j}(x,y) & \text{if } i \le \frac{q}{2}, j > \frac{q}{2} \\ t_{i,j}(y,x) & \text{if } i > \frac{q}{2}, j \le \frac{q}{2} \\ h_{i,j}(x,y) & \text{if } i > \frac{q}{2}, j > \frac{q}{2} \end{cases} \tag{4.29}$$

and g, t and h can be found in Corollary 28.

The following theorem gives an analytic solution for OPT_P using the SVD.

Theorem 34. *For a given α, denote symmetric $D_\alpha := D(\alpha) \in \mathbb{R}^{q\times q}$ as defined in Eqn.* (4.29) *and let $D_\alpha = V\Sigma V^T$ be its SVD. Let v be the right singular vector corresponding to the minimum singular value of D_α. Then $P^* = qvv^T$ solves $OPT\_P(\alpha)$.*

Proof. Recall $OPT\_P(\alpha)$ has the form

$$\min_{P\in\mathbb{R}^{q\times q}} \langle D_\alpha, P\rangle \quad \text{s.t. } P \geq 0,\ \text{trace}(P) = q.$$

Denote the minimum singular value of D_α as $\sigma_{\min}(D_\alpha)$. Then for any feasible $P \in \mathcal{X}$, by [54] we have

$$\langle D_\alpha, P\rangle \geq \sigma_{\min}(D_\alpha)\text{trace}(P) = \sigma_{\min}(D_\alpha)q.$$

Now consider $P = qvv^T \in \mathbb{R}^{q\times q}$. P is feasible since $P \geq 0$, and $\text{trace}(P) = q$. Furthermore,

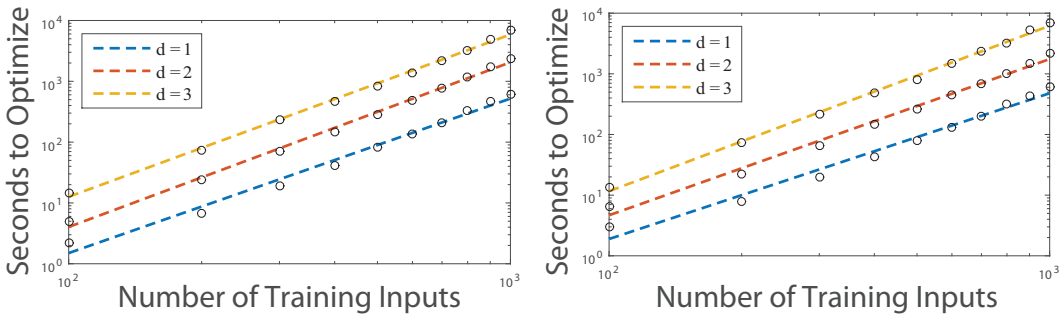
$$\begin{aligned}\langle D_\alpha, P\rangle &= q\,\text{trace}(V\Sigma V^T vv^T) = q\,\text{trace}(v^T V\Sigma V^T v)\\ &= q\,\sigma_{\min}(D_\alpha)\end{aligned}$$

as desired. □

Note that the size, q, of D_α in $OPT\_P(\alpha)$ scales with the number of features, but not the number of samples (m). As a result, we observe that the OPT_P step of Algorithm 2 is significantly faster than the OPT_A step.

4.6 Implementation and complexity analysis

In this section we first analyze the complexity of Optimization Problem (4.23) with respect to the number of training points as well as the selected degree of the TK - $\mathcal{K}_T^d$. We then perform the same analysis on Optimization Problem (4.25) with respect to the number of training points and the number of random matrices selected.



(a) Complexity scaling for identification of circle

(b) Complexity scaling for identification of spiral

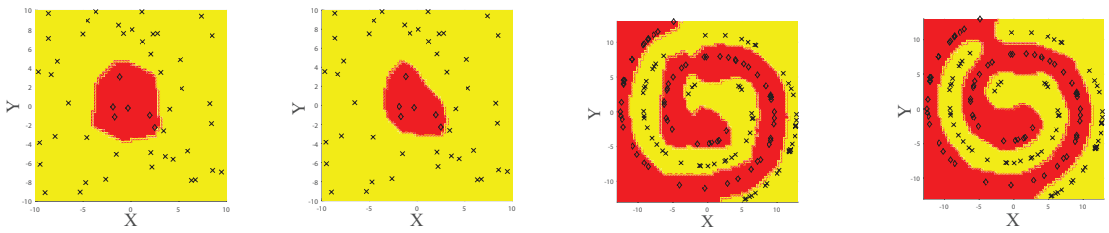
Figure 4.3: Log-Log plot of computation time vs number of training data for 2-feature kernel learning.

Analysis of the SDP Approach: In Optimization Problem (4.23) the constraint that the kernel be a positive TK kernel can be expressed as an LMI constraint with variables P_{ij}. Using Optimization Problem (4.23), if $P \in \mathbb{R}^{q\times q}$, and m is the number of training data, with a Mosek implementation, we find experimentally that the complexity of the resulting SDP scales as approximately $m^{2.6}+q^{1.9}$ as can be seen in Fig. 4.3 and is similar to the complexity of other methods such as the hyperkernel approach [120]. These scaling results are for training data randomly generated by two synthetic 2-feature example problems (circle and spiral - See Fig. 4.4) for degrees $d = 1$, 2, 3 and where d defines the length of Z_d (and hence q) which is the vector of all monomials in 2 variables of degree d or less.

Note that the length of Z_d scales with the degree and number of features, n, as $q = \frac{(n+d-1)!}{n!d!}$. For a large number of features and a high degree, the size of Z_d will become unmanageably large. Note, however, that, as indicated in Section 4.3, even when $d = 0$, every TK kernel is universal.

Analysis of the Minimax Approach:

In Figures 4.5, we plot the computation time of the FW TKL algorithm for both



(a) Circle data set classified with [TK] (n=50)

(b) Circle data set classified with [SimpleMKL] (n=50)

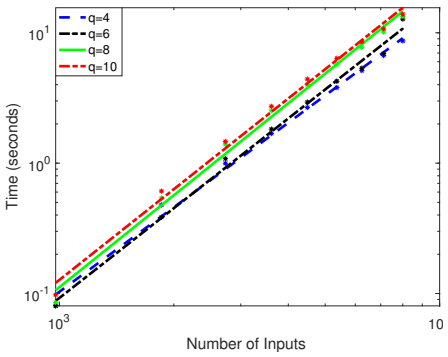
(c) Spiral data set classified with [TK] (n=150)

(d) Spiral data set classified with [SimpleMKL] (n=150)

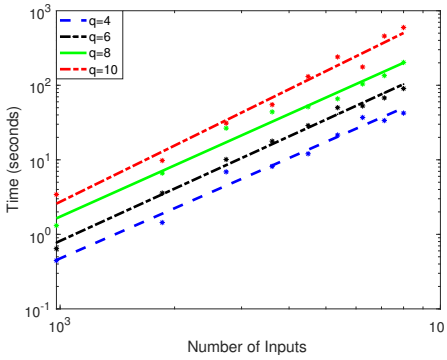
Figure 4.4: a

s Compared with [SimpleMKL] for n Training Data.]Discriminant surface for circle and spiral separator using method [TK] as compared with [SimpleMKL] for n training data.

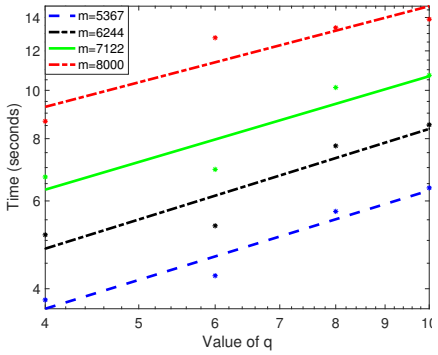
classification and regression on a desktop PC with an Intel i7-5960X CPU at 3.00 GHz and 128 Gb of RAM as a function of m and q, where m is the number of samples used to learn the TK kernel function and the size of P as $q \times q$ (so that q is a function of the number of features and the degree of the monomial basis Z_d). The data set for these plots is Combined Cycle Power Plant (CCPP) from [164, 86], containing 4 features and $m = 9568$ samples. In the case of classification, labels with value greater than or equal to the median of the output were relabeled as 1, and those less than the median were relabeled as -1. To enable comparison with SimpleMKL, we use an identical stopping criterion of 10^{-2}. Figures 4.5(a-d) demonstrate that the complexity of Algorithm 2 scales as approximately $O(m^{2.28}q^{0.57})$ for classification and $O(m^{2.34}q^{2.40})$ for regression. These results are significantly lower with respect to m than the value of $O(m^{2.6}q^{1.9})$ reported in [29] for binary classification using the SDP implementation. Aside from improved scalability, the overall time required for Algorithm 2 is significantly reduced when compared with the SDP algorithm



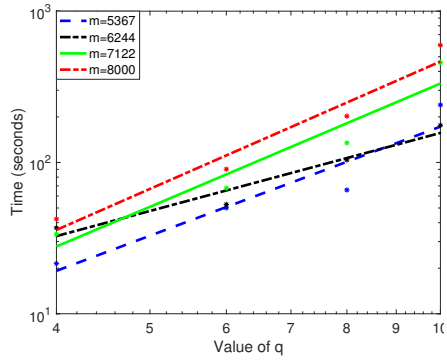
(a) Numerical complexity analysis of TKL for classification versus m.



(b) Numerical complexity analysis of TKL for regression versus m.



(c) Numerical complexity analysis of TKL for classification versus q.



(d) Numerical complexity analysis of TKL for regression versus q.

Figure 4.5: In (a) and (b) we find log scale plots of the time taken to execute FW TKL for $P \in \mathbb{R}^{q \times q}$. The line of best linear fit is included for reference. In (c) and (d) we find log scale plots of the time taken to optimize TKL as a function of q for four different values of m.

in [29], improving by two orders of magnitude in some cases. This is illustrated for classification using four data sets in Table 4.1. This improved complexity is likely due to the lower overhead associated with QP and the SVD.

Table 4.1: We report the mean computation time (in seconds), along with standard deviation, for 30 trials comparing the SDP algorithm in [29] and Algorithm 2. All tests are run on an Intel i7-5960X CPU at 3.00 GHz with 128 Gb of RAM.

Method	Liver	Cancer	Heart	Pima
SDP	95.75 ± 2.68	636.17 ± 25.43	221.67 ± 29.63	1211.66 ± 27.01
Algorithm 2	0.12 ± 0.03	0.41 ± 0.23	4.71 ± 1.15	0.80 ± 0.36

4.7 Accuracy of the New TK Kernel Learning Algorithm for Regression

In this section, we compare the accuracy of the classification and regression solutions obtained from the FW TKL algorithm to a selection of several other state of the art methods. Specifically, we use the following implementations of these algorithms.
[TKL] Algorithm 2 with $d = 1$, $\epsilon = .1$ and we scale the data so that $x_i \in [0, 1]^n$, and then select $[a, b] = [0 - \delta, 1 + \delta]^n$, where $\delta \geq 0$ and C are chosen by 2-fold cross-validation;
[SMKL] SimpleMKL as implemented in [134] with a standard selection of Gaussian and polynomial kernels with bandwidths arbitrarily chosen between .5 and 10 and polynomial degrees one through three - yielding approximately $13(n+1)$ kernels. We set $\epsilon = .1$ as in TKL and C is chosen by 2-fold cross-validation;
[NNet] A neural network with 3 hidden layers of size 50 using MATLABs (`patternnet` for classification and `feedforwardnet` for regression) implementation and stopped learning after the error in a validation set decreased sequentially 50 times.
[RF] The Random Forest algorithm [16] as implemented on the scikit-learn python toolbox [125] for classification and regression. We select between 50 and 650 trees (in 50 tree intervals) using 2-fold cross-validation.
[XGBoost] The XGBoost algorithm [24] for classification and regression. We select between 50 and 650 trees (in 50 tree intervals) using 2-fold cross-validation;

[**AMKL**] The AverageMKL implementation from the MKLpy python package [100], averages a standard selection of Gaussian and polynomial kernels;
[**PWMK**] The PWMK implementation from the MKLpy python package [100], which uses a heuristic based on individual kernel performance from [159] to learn the weights of a standard selection of Gaussian and polynomial kernels;
[**CKA**] The CKA implementation from the MKLpy python package [100], uses the centered kernel alignment optimization in closed form from [38] to learn the weights of a standard selection of Gaussian and polynomial kernels.

Six classification and six regression datasets were chosen arbitrarily from [46, 22] to contain a variety of number of features and number of samples. No other datasets were tested for relative performance and datasets were not "pre-screened". In both classification and regression, our accuracy metric uses 5 random divisions of the data into test sets (m_t samples $\cong$ 20% of data) and training sets (m samples $\cong$ 80% of data). Individual test results and their standard deviations are reported in Tables 4.2 and 4.3 for classification and regression respectively.

Regression analysis Out of the six different regression data sets the proposed algorithm (TKL) scored above average on five of the data sets, an improvement over all other algorithms but XGBoost which also scored above average on five of the data sets. The average percent difference between the average MSE and the MSE for PMKL was -23.16% which was better than all other tested algorithms including XGBoost. Unfortunately, the cost of this increased accuracy is an increased computation time. The average percent difference between the average computation time and the time for PMKL was 19.49%. However, TKL is also the top performing algorithm for three of the six data sets.

Classification analysis Out of the six different classification data sets the proposed algorithm (TKL) scored above average on all of the data sets, an improvement over all other algorithms. The average percent difference between the average TSA and the TSA for TKL was 6.77% which was close to the top score of 6.84% by the AMKL algorithm. The PWMK algorithm had an average percent difference of 8.52%, but failed to converge on one of the data sets. Like in the regression case, the cost of this increased accuracy is an increased computation time. The average percent difference between the average computation time and the time for PMKL was 68.41%. However, TKL is also the top performing algorithm for two of these data sets.

To further illustrate the importance of density property and the TKL framework for practical regression problems, we used elevation data from [8] to learn a degree 2 TK kernel and associated SVM predictor representing the surface of the Grand Canyon in Arizona. This data set is particularly challenging due to the variety of geographical features. The result from the TKL algorithm can be seen in Figure 4.6(d) where we see that the regression surface visually resembles a photograph of this terrain, avoiding the artifacts present in the Gaussian-based method.

Next we use the TKL algorithm and a selection of other machine learning algorithms to analyze an immune dataset.

4.8 Application of the TK Kernel Learning Algorithm to an Immune State Identification Problem

The immune response is a dynamic process by which the body determines whether an antigen is self or nonself. The state of this dynamic process is defined by the relative balance and population of inflammatory and regulatory actors which comprise this decision making process. The goal of immunotherapy as applied to, e.g. Rheumatoid Arthritis (RA), then, is to bias the immune state in favor of the regulatory actors -

thereby shutting down autoimmune pathways in the response. While there are several known approaches to immunotherapy, the effectiveness of the therapy will depend on how this intervention alters the evolution of this state. Unfortunately, this process is determined not only by the dynamics of the process, but the state of the system at the time of intervention - a state which is difficult if not impossible to determine prior to application of the therapy.

In this section we use feature selection algorithms, in combination with the TK kernel learning algorithm and other machine learning algorithms, to identify an immune state using a data-driven approach. We initially define our immunological dataset obtained from ongoing trials of RA immunotherapy. Then we define a set of machine learning algorithms which use a given subset of data observables to identify outputs of interest such as other observables or the RA outcome - as measured by severity of inflammation.

To identify important features from the immunological dataset we propose a metric for suitability of a given set of observables based partially on predictive power of the associated model. The first part of this metric is based on minimality (not prediction), wherein we impose a penalty based on the number of observables in the set (cardinality) in order to reduce experimental and clinical complexity. Second, in order to ensure that relevant immunological data is not lost, we also add a penalty based on the error of the associated model to predict observables from the data not included in the given set. Third, to measure efficacy of the prediction, we impose a penalty based on the error in prediction of RA severity - a quantity we refer to as the "disease state".

Finally we use a variety of feature selection algorithms to determine the set of observables which are optimally suited using the suitability metric described above. We then report the results of applying the resulting algorithms to our dataset after

applying different weights to the three parts of the suitability metric. We define the optimal sets returned by each case as "immune states" and analyze the immune cells that were selected by the feature selection algorithms in each case.

4.8.1 The Immune Dataset

We propose a methodology for identifying observable measures for immune system health from measured features of the immune system. To better illustrate this methodology, we consider the approach as applied to a particularly rich dataset obtained from an ongoing series of experiments involving the use of biomaterials-based particles [107] containing metabolites that promote self tolerance in intermediate/late stage RA in a DBA/1j mouse model which develops severe arthritis when immunized with bovine collagen type 2 (bc2) autoantigen. In this experimental series, the particles were synthesized either with or without autoantigen bc2 - a strategy designed to determine if the particles can generate antigen-specific anti-inflammatory response. An overview of the experimental procedure is provided in Fig. 4.7. The chronology of the experiment is listed here in detail. The data collection used for model generation occurs exclusively on either day 62 or 70.

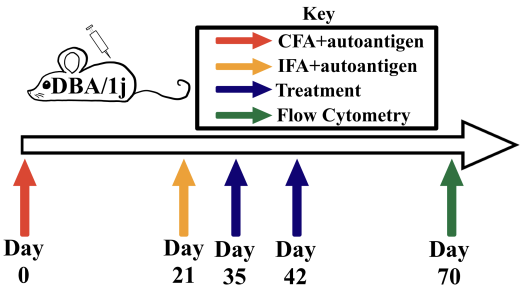


Figure 4.7: A graphical description of the experimental procedure of inducing and treating RA in mice. The first two steps induce RA, the next two steps is the application of the treatment and the final step is the data generation using flow cytometry. CFA = complete Freund's adjuvant, IFA = incomplete Freund's adjuvant.

Day 0 and 21: RA was induced in mice to generate an autoimmune response for the development of severe polyarthritis. On day 35, the mice were divided into 3

groups, each receiving a distinct therapeutic regimen.

Group 0 - Days 35/42: The control group consists of 5 control mice, each receiving two subcutaneous injections of phosphate buffered saline (PBS) near the hind legs on days 35 and 42.

Group 1 - Days 35/42: Treatment group 1 consists of 5 mice. Each mouse receives two injections of 0.5 mg of biomaterials-based particles without embedded autoantigen bc2 near the hind legs on days 35 and 42.

Group 2 - Days 35/42: Treatment group 2 consists of 8 mice. Each mouse receives two injections of 0.5 mg of biomaterials-based particles with embedded autoantigen bc2 near the hind legs on days 35 and 42.

Measurements Taken on Days 62/70: Paw thickness measurements are used to determine arthritic scores for all mice and the end of study paw measurements were obtained either on day 62 or 70, and are defined on the interval $[0, 5]$. Furthermore, flow cytometry was performed on cells collected from the popliteal lymph node, cervical lymph node and spleen of each mouse on day 62 or 70. The flow cytometry procedure measured values for CD4 (T helper (Th) cell marker), CD8 (cytotoxic T (Tc) cell marker), Ki67 (proliferation), CD25 (activation), Foxp3 (regulatory T (Treg) cell transcription factor (TF)), Tbet (Th1/Tc1 TF), GATA3 (Th2/Tc2 TF), RORyT (Th17/Tc17 TF), CD44 (effector memory marker), CD62L (central memory marker), and a tetramer that is specific to the autoantigen. Based on this staining, we identified 41 different combinations of markers which might be used to classify the phenotype of a T cell and determined the percentage of either CD4 or CD8 T cells

presenting the associated combination of markers.

Summary of Associated Dataset: The data consists of 84 samples, based on 18 mice, each sample is associated with a mouse and sample location. All samples are taken on day 62/70, and each sample consists of 43 features and one label. The first two features of each sample indicate group number (0-2) and sample location (1-3). The remaining 41 features defining the percentage (0-100) of the CD4/CD8 population exhibiting the associated combination of markers. The label for each sample is the arthritic score (0-5).

Based on this data we next propose several methods of machine learning to construct predictive models which use subsets of the features to predict both the label (disease progression) and discarded features. For generating these models, all features are scaled to the interval $[0, 1]$.

4.8.2 Selected Machine Learning Algorithms

We define six machine learning algorithms that will be used in combination with a feature selection algorithm to determine the immune state. In each case below, we assume the data set contains m samples, $\{x_i, y_i\}_{i=1}^m$, each with n features, $x_i \in \mathbb{R}^n$ and a label $y_i \in \mathbb{R}$.

Regularized Linear Regression (LR) The regularized linear regression algorithm returns a predictive model $y = f(x) = w^T x + b$, where w solves the following optimization problem.

$$\min_{w \in \mathbb{R}^n} \quad \sum_{i=1}^{m} (y_i - w^T x_i - b)^2 + \alpha_2 ||w||^2 + \alpha_1 ||w||.$$

In this case, $\alpha_1 \geq 0$ and $\alpha_2 \geq 0$ are the regularization parameters. Linear regression has the advantage of low computational complexity. However, the resulting predictor

is linear and if the underlying physical process is nonlinear, accuracy of the predictive model will be poor.

ϵ-loss Support Vector Regression (SVR) The support vector regression problem (Optimization Problem (2.15)) uses a predictive model of the form $f(x) = \sum_{i=1}^{m} \alpha_i k(x, x_i)$ where $\alpha \in \mathbb{R}^m$ is the decision variable and k is a user selected positive kernel function. The objective function penalizes any points where $|f(x_i) - y_i| \geq \epsilon$, where ϵ is a tuning parameter by a regularization parameter C. SVR can generate accurate nonlinear predictive models for appropriate choice of k. However, the selection of the kernel heavily influences the resulting accuracy and this process of selection is difficult to automate.

Kernel Learning (TKL) Kernel learning algorithms improve on the SVR problem by automating the search for a kernel function. We have shown that the class of Tessellated Kernel functions have the properties of universality, density, and tractability - meaning the resulting algorithms are rather accurate and generalize well to new data. The regularization parameters in this case are the ϵ and C as defined above for SVR.

Decision Tree Algorithms Decision trees are composed of a series of conditional statements that branch in a "tree" like manner. We say the "depth" of a decision tree is how many conditional statements appear in a branch before leading to a label denoted the "leaf". Both the depth of the decision trees and the maximum number of leaves are regularization parameters that can be modified by the user. Decision trees are often weak predictors alone and in this paper we use ensemble (random forest) or boosting (boosted trees) methods to increase predictive performance. These algorithms are defined as follows.

- **Random Forest:** The random forest algorithm is an ensemble machine learning method based on a combination of decision trees. Ensemble methods use a combination of predictive models (trees) that individually have poor generalization but when used in combination can have significantly improved predictions. The number of decision trees combined in the random forest algorithm can be used as a regularization parameter.

- **Boosted Trees:** Gradient boosting is another machine learning method also based on a combination of decision trees. In the boosted algorithm trees are added to the predictive model sequentially, and each additional tree is fit to the current residuals of the model. A "learning rate" is a weight applied to the addition of each decision tree, and is often used as a regularization parameter. Small learning rates tend to improve the generalization of the predictive models.

Next we will focus on a metric we may use to identify the observables which are most suitable to the task of predicting self vs nonself determination in autoimmune disease.

4.8.3 Quantifying Suitability of a Given Set of Observables

To identify a set of observables for predicting self vs nonself determination we rigorously define a metric for suitability in order to select the observables which lead to superior predictive models.

First, for the sake of generality, we define the algorithm, OPT, which we use as a placeholder for the machine learning algorithms described previously.

Definition of OPT : Given a dataset $\{x_i, y_i\}_{i=1}^m \subset \mathbb{R}^n \times \mathbb{R}^q$, $OPT(\{x_i, y_i\}_{i=1}^m)$, returns a predictive function, $f = \text{arg}OPT(\{x_i, y_i\}_{i=1}^m)$, where $f : \mathbb{R}^n \to \mathbb{R}^q$.

In cases where the machine learning algorithm returns a function with a single output we train the machine learning algorithm q times, once for each output.

Next, given a possible set of feature indices $F := \{1, \cdots, n\}$, we define the set of all subsets of F as $\mathcal{P}(F)$, and the set of all possible subsets of F of length $w \leq n$ as follows.

$$B_w := \{v \in \mathbb{N}^w \mid v \in \mathcal{P}(F)\}$$

For a given selection of features, $b \in B_w$, we denote the associated projection $P_b : \mathbb{R}^n \to \mathbb{R}^w$ so that $(P_b(x))_i = x_{b_i}$ for $x \in \mathbb{R}^n$ and $i = 1, \cdots, w$.

To define a metric we consider three cost/penalty functions, M_1, M_2, and L. The function L is a function of the cardinality of the number of features selected, $L(|b|_C)$. The costs M_1 and M_2, however, measure how well the selection of features can be used to predict the disease state and the discarded features respectively. To accurately evaluate the performance of the predictor a partition of the data must be withheld from the training algorithm, OPT, and used solely for the purpose of testing the performance. For a given set of data, these metrics will vary depending on which data points are used for training OPT and which are used to evaluate its performance. To explicitly account for the effect of choice in partitioning of data samples, we now define the set of samples $S := \{1, \cdots, m\}$, and the set of partitions of S as $\mathcal{P}(S)$. As for features, we denote the set of sample partitions of length r as

$$S_r := \{v \in \mathbb{N}^r \mid v \in \mathcal{P}(S)\}$$

and for a given selection of samples, $g \in S_r$, we denote the associated projected data set as $\mathcal{P}_g(X) := \{x_i \in X,\ i \in g\}$.

Therefore, the costs M_1 and M_2 are a function of the feature partition, b, and the training partition, $g \in S_r \in \mathcal{P}(S)$, so that $M_1(b, g)$ and $M_2(b, g)$ are the Root Mean Square Error (MSE) of predicting the test partition (S/g). Specifically, let

$R(f,x,y) = \sqrt{\frac{1}{m-r}\sum_{i\in S/g}|f(x_i)-y_i|_2^2}$ and we have

$$M_1(b,g) = R(f_{b,g}, P_b(x), y) \qquad M_2(b,g) = \sum_{j\in F/b} R(d_{b,g}^{(j)}, P_b(x), P_j(x))$$

$$f_{b,g} = \text{arg}OPT(\{P_b(x_{g_i}), y_{g_i}\}_{i=1}^r) \qquad d_{b,g}^{(j)} = \text{arg}OPT(\{P_b(x_{g_i}), P_j(x_{g_i})\}_{i=1}^r))$$

In the ideal case, we would average these costs over all possible partitions of the data set to give an estimate of the predictive power of $b \in B_w$. However, such an approach would result in very large computational overhead. Therefore, we use the k-fold cross validation approach, wherein we divide the samples into k training partitions of size $\frac{m(k-1)}{k}$, which we label as $g(i) \in S_{\frac{m(k-1)}{k}}$ for $i = 1, \cdots, k$. Then the average cost of the feature partition b over the k sample partitions is

$$J(b) = \frac{1}{k}\sum_{i=1}^{k} J'(b, g(i)).$$

where

$$J'(b,g) := \beta_1 M_1(b,g) + \beta_2 M_2(b,g) + L(|b|_C) \tag{4.30}$$

and where $\beta_1, \beta_2 \geq 0$ are given weights.

First, we let $\beta_1 = 1$ and $\beta_2 = L(w) = 0$ - a case we denoted as the Minimal Disease State (MDS). In this case, we are only concerned with predicting the progression of the disease and are not concerned with predicting non-selected features or with the number of features selected. Therefore, the number of features selected for each algorithm is the number of features which generated the smallest $J'(b,g)$. Second, we let $\beta_1 = 0$ and $\beta_2 = 1$ and

$$L(w) = \begin{cases} 0 & \text{for } w \leq 10 \\ \infty & \text{for } w > 10. \end{cases}$$

In this case, we ignore the disease state and are only concerned with reducing the number of features while retaining the ability to reconstruct discarded features - a

case we denote as the Minimal Immune State (MIS). Finally, we let $\beta_1 = \beta_2 = 1$ and $L(w)$ as defined for the MIS. We denote this final case as the Minimal Immune and Disease State (MIDS).

Now that we have defined the metric of suitability as a function of the partition, $b \in B_w$ we will define the following combinatoric optimization problem.

$$\min_{b \in B_w, w \in \mathbb{N}} \quad J(b) \tag{4.31}$$

To perform feature selection to solve Optimization Problem (4.31) we will use a Sequential Feature Selection (SFS) algorithm as described in [20]. This SFS algorithm begins with $b := \emptyset$, and iteratively selects a locally optimal feature (with respect to the objective function of Optimization Problem (4.31)) at each step.

Clearly, the effectiveness of Feature Selection depends on the ML algorithm (OPT) used to generate the predictive model. Therefore, in the Results subsection, we test all the machine learning algorithms proposed herein. Unfortunately, the accuracy of the predictive model is influenced by user-selected parameters within the algorithm. For reproducibility, we list here the selections for these parameter values.

Linear Regression: We test all 16 combinations of $\alpha_1 \in [0, 0.1, 1, 5]$ and $\alpha_2 \in [0, 0.1, 1, 5]$ and select the choice yielding the highest metric (J).

TKL: We use the default TK kernel parameters and test $\epsilon = .005$, and $C \in [.01, .1, .3, .5, 1]$ and select the choice yielding the highest metric (J).

SVR: We test all combinations of $\epsilon = .1$, $C \in [1, 5, 10]$ and 3 kernel functions (linear, RBF, or 3rd degree polynomial) and select the choice yielding the highest metric (J). For the RBF kernel the features are normalized by their variance and a bandwidth of $\frac{1}{n}$ is selected.

Random Forest We test 9 combinations of number of trees ($\mathrm{n_{trees}} \in [50, 100, 150]$) and the maximum tree depth of ($\mathrm{max_{depth}} \in [5, 10, 20]$) and select the choice yielding

the highest metric (J).

Boosted Trees We test 15 combinations of number of trees ($n_{trees} \in [50, 100, 150, 250]$) and learning rate ($LR \in [0.01, 0.1, 0.5]$) and select the choice yielding the highest metric (J).

Alternative feature selection algorithms are used as a baseline by which we may compare the wrapper method. We use three filter methods and one embedded method in the analysis as follows.

Filter Methods

Given a set of data, filter methods use a rating function to rank each features relative "importance". After the features have been ranked, the user may select w features to be kept and the remaining features will be discarded.

Mutual Information (MI) The Mutual Information criteria [7] is a statistical function of two random variables that describes the amount of information contained in one random variable relative to the other.

Analysis of Variance (ANOVA) The ANOVA method [92] is a commonly used method for analyzing variable dependencies. The *F-test* is used to estimate the features importance.

Principle component analysis (PCA) This method approximates the data with linear manifolds [148]. The main methods used to perform PCA are based on the singular value decomposition and diagonalization of the correlation matrix. We calculate the importance based on the first 3 eigenvectors.

In all cases, once a set of features has been selected, suitability (J) is determined using each of the ML algorithms and we report the minimum of these values.

Embedded Methods

Embedded FS methods attempt to embed the process of feature selection directly into the model generation process - typically adding a cost for inclusion of a particular feature in the model. These methods have been used for a similar application in gene expression as in [68]. For this analysis, only a single embedded method was considered.

Mean Decrease in Impurity (RF) The Gini Importance or Mean Decrease in Impurity [17] is an embedded method for the Random forest algorithm. It calculates the importance of features as the mean of the number of splits (over all trees) that include this feature, weighted by the probability of reaching this node.

4.8.4 Results

Here we define three immune states generated by varying the weights of the metric. These three immune states are lower dimensional subsets of the data which can be used to either predict the progression of RA, reconstruct the full set of T cell markers and populations, or perform both tasks simultaneously.

In all cases the SFS algorithms consistently outperformed the filter and embedded methods. For this reason we focus the analysis on the SFS algorithms and the frequency by which the SFS algorithms selected populations of immune system cells.

Case 1: Features for Predicting Disease Progression (MDS)

First we consider the problem of selecting features that are optimal for predicting the disease progression.

Most Important Features Using the SFS Algorithms

In Fig. 4.8 we show the observables that were selected by each of the proposed algorithms. Counting the number of times a feature was selected by the proposed SFS

methods, the following features were chosen by at least three of the methods.

(1) *CD4+GATA3+CD44+CD62(Lo)* (3 times)

(2) *CD4+GATA3+Ki67+* (3 times)

(3) *CD4+Foxp3+CD25+* (3 times)

(4) *CD4+Foxp3+CD25+Ki67+Autoantigen* (3 times)

(5) *CD4+Tbet+* (3 times)

Among the cytotoxic cells, the algorithms were most consistent, with all five of the SFS algorithms selecting one feature in common.

(6) *CD8+Ki67+* (4 times)

(7) *CD8+GATA3+* (3 times)

(8) *CD8+Tbet+* (3 times)

This group of cells consists of cytotoxic (6,7,8), Th memory (1), Th (2,5), and Treg (3,4) T cell sub-populations. The *location* feature (origin of the tested cells), was selected only once by an SFS based algorithm. This implies that there is significant uniformity in the disease state among the lymph nodes and spleen.

In addition, only one selected feature (4) was autoantigen specific - indicating that most of these T cell markers may be correlated to, not just the progression of RA, but the progression of other similar autoimmune diseases as well.

Finally, the MDS consisted of biomarkers that were not data-rich but contained general classes of T cells rather than the more specific sub-populations selected in the MIS and MIDS cases. This suggests that the disease state is caused by a large

irregularity in general populations of T cells rather than irregularities in a few smaller specific populations.

In this case we do not include the treatment as a possible feature, since we are primarily interested in the prediction of the disease state using sub-populations of T cells as opposed to the already known correlation between treatment and disease state. In the next two cases treatment is considered a feature.

Case 2: Features for Reconstructing Discarded Features (MIS)

Next we consider the problem of selecting features that are optimal for reconstructing discarded features to determine a Minimal Immune State.

Most Important Features

In Fig. 4.9 we show the features that were selected by each of the proposed algorithms. Unlike in the previous subsection, there was less of an agreement among the high-performing SFS algorithms as to the most significant features. For MIS only 6 different features were selected by at least three algorithms. First, if we consider markers specific to helper and regulatory cells, and counting the number of times a feature was selected by the SFS methods (each method selected 10 features), the following features were each chosen by at least half of the algorithms.

(1) *CD4+GATA3+CD44+CD62L(Lo)* (4 times)

(2) *CD4+Tbet+Ki67+* (4 times)

(3) *CD4+GATA3+Ki67+Autoantigen* (3 times)

(4) *CD4+Tbet+Autoantigen* (3 times)

We note that two of the selected features are autoantigen specific as opposed to the single autoantigen specific feature selected for cells in MDS.

Among the cytotoxic cells, the algorithms were less consistent, with only three of the SFS algorithms selecting similar sub-populations.

(5) *CD8+Ki67* (3 times)

(6) *CD8+GATA3+CD44+CD62(Lo)* (3 times)

We note that the central memory T cells (CD62L) appear in both the helper populations and the cytotoxic cell populations. No regulatory cells were consistently selected by all five of the top performing algorithms, suggesting that no specific regulatory cell type was most important for reconstructing the immune state.

In this case, data-rich biomarkers (those containing multiple markers), were selected slightly more often when compared to MDS. The average number of markers in the selected features is 3.33 in this case compared to 2.875 in the MDS case. This may be due to the fact that estimating the entire immune state is significantly more difficult then simply estimating the disease state alone, and more data-rich markers may therefore be necessary.

Of particular note is the fact that the *location* feature (origin of the tested cells) and the *treatment* feature (which treatment was applied) were both selected by almost every algorithm. This implies that the immune state is not uniform across the lymph nodes and spleen, and that the specific treatment given to the mice has a large impact on the immune state.

Case 3: Features for Disease Progression and Reconstruction (MIDS)

Next we consider the problem of selecting features that are optimal for predicting a combination of the MIS and MDS objectives, the Minimal Immune and Disease State.

Most Important Features Using the SFS Algorithms

In Fig. 4.10 we show the features that were selected by each of the proposed algorithms. If we consider markers specific to helper and regulatory cells, the following features were each chosen by at least three of the five SFS algorithms.

(1) *CD4+GATA3+CD44+CD62L(Lo)* (4 times)

(2) *CD4+Tbet+Ki67+* (4 times)

(3) *CD4+Tbet+Autoantigen* (4 times)

(4) *CD4+GATA3+Ki67+Autoantigen* (3 times)

As in the MIS case two of the selected features are autoantigen specific, however no cytotoxic cells were consistently selected by at least three of the SFS algorithms.

Similar to the MIS case, no regulatory cells were consistently selected by all five of the top performing algorithms. Just as in the case of the MIS the *location* feature (origin of the tested cells) were selected by almost every algorithm and the *treatment* attribute was likewise selected by every algorithm.

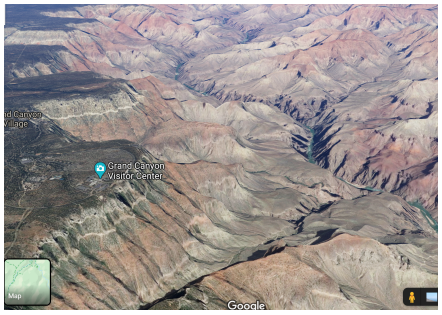
However, in this case the algorithms were least consistent on selecting cytotoxic cells, implying that no single cytotoxic cell type was consistently selected to both reconstruct the immune system and determine the disease progression. This would imply, therefore, that the Th cell populations are more essential to both predicting the disease progression and reconstructing the immune state.

We note that the memory T cell sub-population *CD4+GATA3+CD44+CD62L(Lo)* was selected in all three cases. It is clear that this sub-population is significant to both the immune and disease states.

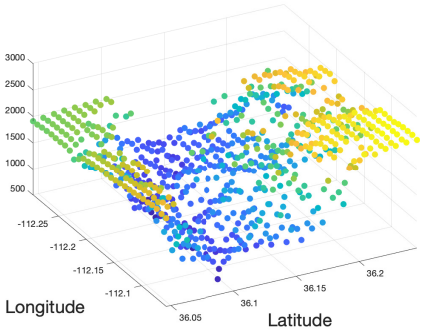
4.9 Conclusion

We have proposed an efficient algorithm for TK kernel learning based on a primal-dual decomposition combined with a FW type algorithm. The set of TK kernels is tractable, dense, and universal, implying that KL algorithms based on TK kernels are more robust than existing machine learning algorithms, an assertion supported by numerical testing on 6 relatively large and randomly selected datasets.

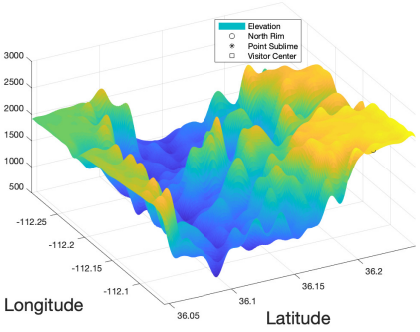
We also considered the use of the KL learning problem in the identification of three different states of the immune system of increasing complexity. Specifically, we used a set of mouse-model experiments to obtain a robust dataset of T cell markers and populations at the end stage of a proposed immunotherapy treatment. The first state is the disease state, which is important for tracking the disease progression and predicting the effectiveness of treatments. Next is the immune state which is the minimal number of sub-populations needed to reconstruct the remaining discarded features. Finally we find the combined overall immune state for predicting both the disease state and reconstructing the discarded features. From these experiments we were able to determine that the *CD4+GATA3+CD44+CD62L(Lo)* memory T cell sub-population is significant to both the immune state and disease state of mice with RA - and have developed a set of T cell sub-populations important for tracking the disease progression and outcome of immunotherapy.



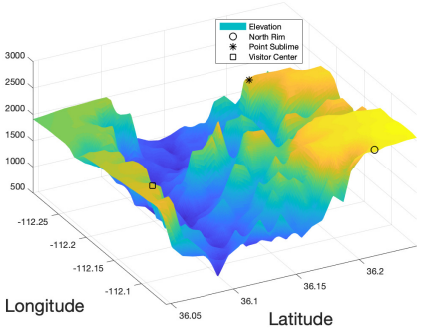
(a) An image from Google Maps of a section of the Grand Canyon corresponding to (36.04, -112.05) latitude and (36.25, -112.3) longitude.



(b) Elevation data ($m = 750$) from [8] for a section of the Grand Canyon between (36.04, -112.05) latitude and (36.25, -112.3) longitude.



(c) Predictor using a hand-tuned Gaussian kernel trained on the elevation data in (b). The Gaussian predictor poorly represents the sharp edge at the north and south rim.



(d) Predictor from Algorithm 2 trained on the elevation data in (b). The TK predictor accurately represents the north and south rims of the canyon.

Figure 4.6: Subfigure (a) shows a 3D representation of the section of the Grand Canyon to be fitted. In (b) we plot elevation data of this section of the Grand Canyon. In (c) we plot the predictor for a hand-tuned Gaussian kernel. In (d) we plot the predictor from Algorithm 2 for $d = 2$.

Table 4.2: All classifications tests are run on a desktop with Intel i7-4960X CPU at 3.60 GHz and with 64 GB of RAM. N/A indicates that the algorithm was stopped after 24 hours without a solution.

Dataset	Method	Accuracy (%)	Time (s)	Dataset	Method	Accuracy (%)	Time (s)
Abalone	TKL	84.61 ± 1.60	17.63 ± 3.77	Hill Valley	TKL	86.70 ± 5.49	86.78 ± 48.18
$n = 8$	SMKL	83.13 ± 1.06	350.41 ± 175.15	$n = 100$	SMKL	51.23 ± 3.55	2.81 ± 2.83
$m = 4000$	NNet	84.70 ± 1.82	4.68 ± 0.64	$m = 1000$	NNet	70.00 ± 4.79	3.79 ± 1.75
$m_t = 677$	RF	84.11 ± 1.33	0.98 ± 0.21	$m_t = 212$	RF	56.04 ± 3.27	0.75 ± 0.33
	XGBoost	82.69 ± 1.06	0.20 ± 0.06		XGBoost	55.66 ± 2.37	0.58 ± 0.34
	AMKL	84.64 ± 1.01	0.95 ± 0.07		AMKL	94.71 ± 1.72	5.50 ± 3.84
	PWMK	84.64 ± 1.01	3.13 ± 0.12		PWMK	94.34 ± 1.69	13.10 ± 5.19
	CKA	65.05 ± 0.76	21.43 ± 0.32		CKA	47.92 ± 0.57	0.50 ± 0.08
Transfusion	TKL	77.84 ± 3.89	0.25 ± 0.08	Shill Bid	TKL	99.76 ± 0.08	23.66 ± 2.63
$n = 4$	SMKL	76.62 ± 4.79	2.44 ± 3.08	$n = 9$	SMKL	97.71 ± 0.32	81.04 ± 13.11
$m = 600$	NNet	78.78 ± 3.26	1.01 ± 0.47	$m = 5000$	NNet	98.64 ± 0.86	3.56 ± .60
$m_t = 148$	RF	75.00 ± 3.58	0.54 ± 0.24	$m_t = 1321$	RF	99.35 ± 0.14	0.78 ± 0.36
	XGBoost	73.92 ± 3.95	0.13 ± 0.11		XGBoost	99.61 ± 0.06	0.13 ± 0.04
	AMKL	74.46 ± 1.50	766.90 ± 315.49		AMKL	99.72 ± 0.10	1.24 ± 0.04
	PWMK	N/A	N/A		PWMK	99.72 ± 0.10	3.03 ± 0.04
	CKA	76.35 ± 4.27	0.18 ± 0.03		CKA	99.65 ± 0.21	55.82 ± 0.93
German	TKL	75.80 ± 1.89	58.78 ± 36.08	FourClass	TKL	99.77 ± 0.32	0.13 ± 0.01
$n = 24$	SMKL	74.30 ± 3.55	17.78 ± 4.79	$n = 2$	SMKL	94.53 ± 12.22	0.85 ± 0.48
$m = 800$	NNet	72.70 ± 3.98	0.61 ± 0.05	$m = 690$	NNet	100.00 ± 0.00	0.53 ± 0.03
$m_t = 200$	RF	74.90 ± 1.35	0.64 ± 0.28	$m_t = 172$	RF	99.30 ± 0.44	0.68 ± 0.53
	XGBoost	72.40 ± 2.89	0.10 ± 0.03		XGBoost	98.95 ± 0.44	0.04 ± 0.00
	AMKL	70.80 ± 1.47	2.88 ± 0.38		AMKL	100.00 ± 0.00	1.39 ± 0.03
	PWMK	70.70 ± 1.50	907.00 ± 77.58		PWMK	100.00 ± 0.00	990.08 ± 60.16
	CKA	68.50 ± 1.58	0.25 ± 0.03		CKA	66.16 ± 3.44	0.16 ± 0.03

Table 4.3: All regression tests are run on a desktop with Intel i7-5960X CPU at 3.00 GHz and with 128 Gb of RAM.

Dataset	Method	Error	Time (s)	Dataset	Method	Error	Time (s)
Gas Turbine	TKL	0.23 ± 0.01	13580 ± 2060	CCPP	TKL	10.57 ± 0.82	626.76 ± 456.05
$n = 11$	SMKL	N/A	N/A	$n = 4$	SMKL	13.93 ± 0.78	13732 ± 1490
$m = 30000$	NNet	0.27 ± 0.03	1172 ± 100	$m = 8000$	NNet	15.20 ± 1.00	305.71 ± 9.25
$m_t = 6733$	RF	0.38 ± 0.02	16.44 ± 0.57	$m_t = 1568$	RF	10.75 ± 0.70	1.65 ± 0.19
	XGBoost	0.33 ± 0.005	49.46 ± 1.93		XGBoost	8.98 ± 0.81	5.47 ± 2.73
Airfoil	TKL	1.41 ± 0.44	49.87 ± 4.29	CA	TKL	0.012 ± 0.0003	1502.4 ± 2154.2
$n = 5$	SMKL	4.33 ± 0.79	617.82 ± 161.63	$n = 8$	SMKL	N/A	N/A
$m = 1300$	NNet	6.06 ± 3.84	211.86 ± 41.04	$m = 16500$	NNet	0.0113 ± 0.0004	914.3 ± 95.9
$m_t = 203$	RF	2.36 ± 0.42	0.91 ± 0.20	$m_t = 4140$	RF	0.0096 ± 0.0003	5.28 ± 3.13
	XGBoost	1.51 ± 0.40	2.59 ± 0.06		XGBoost	0.0092 ± 0.0002	5.28 ± 3.13
Space	TKL	0.013 ± 0.001	121.81 ± 49.22	Boston Housing	TKL	10.36 ± 5.80	63.05 ± 2.90
$n = 12$	SMKL	0.019 ± 0.005	3384 ± 589	$n = 13$	SMKL	15.46 ± 11.49	10.39 ± 0.89
$m = 6550$	NNet	0.014 ± 0.004	209.70 ± 37.37	$m = 404$	NNet	50.90 ± 44.19	79.16 ± 42.80
$m_t = 1642$	RF	0.017 ± 0.003	1.06 ± 0.27	$m_t = 102$	RF	10.27 ± 5.70	0.68 ± 0.40
	XGBoost	0.015 ± 0.002	0.32 ± 0.02		XGBoost	9.40 ± 4.17	0.14 ± 0.06

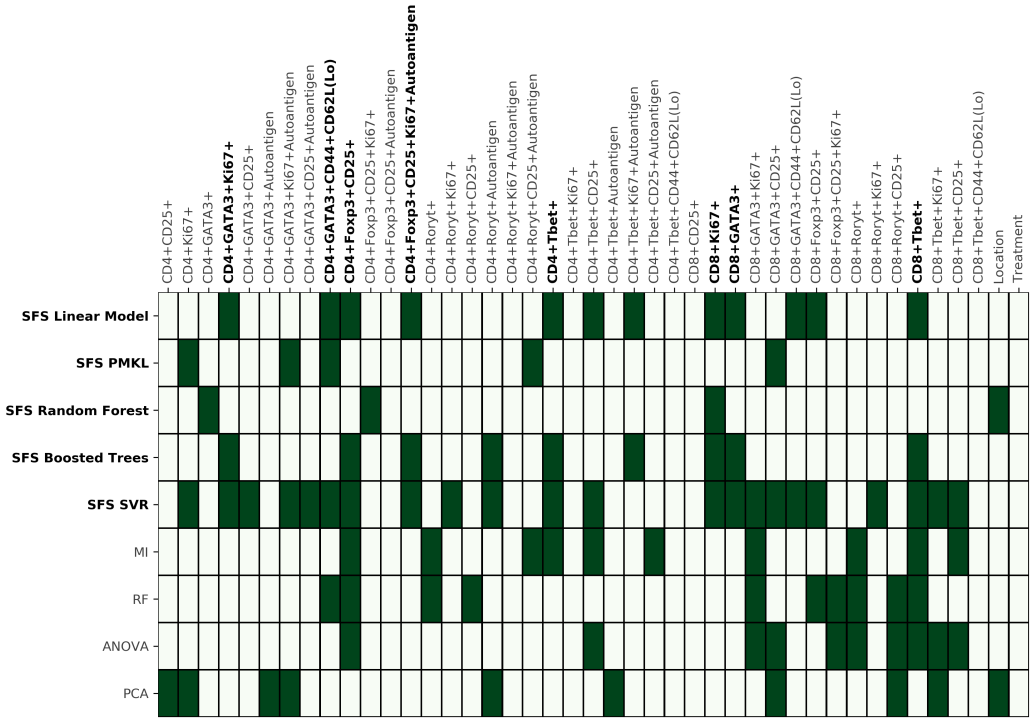


Figure 4.8: The green squares indicate that the feature selection method (left) selected the feature (top). The methods are ordered from lowest objective function, $J(b)$, at the top to greatest objective at the bottom. The SFS methods and the features most commonly selected by those methods are bolded.

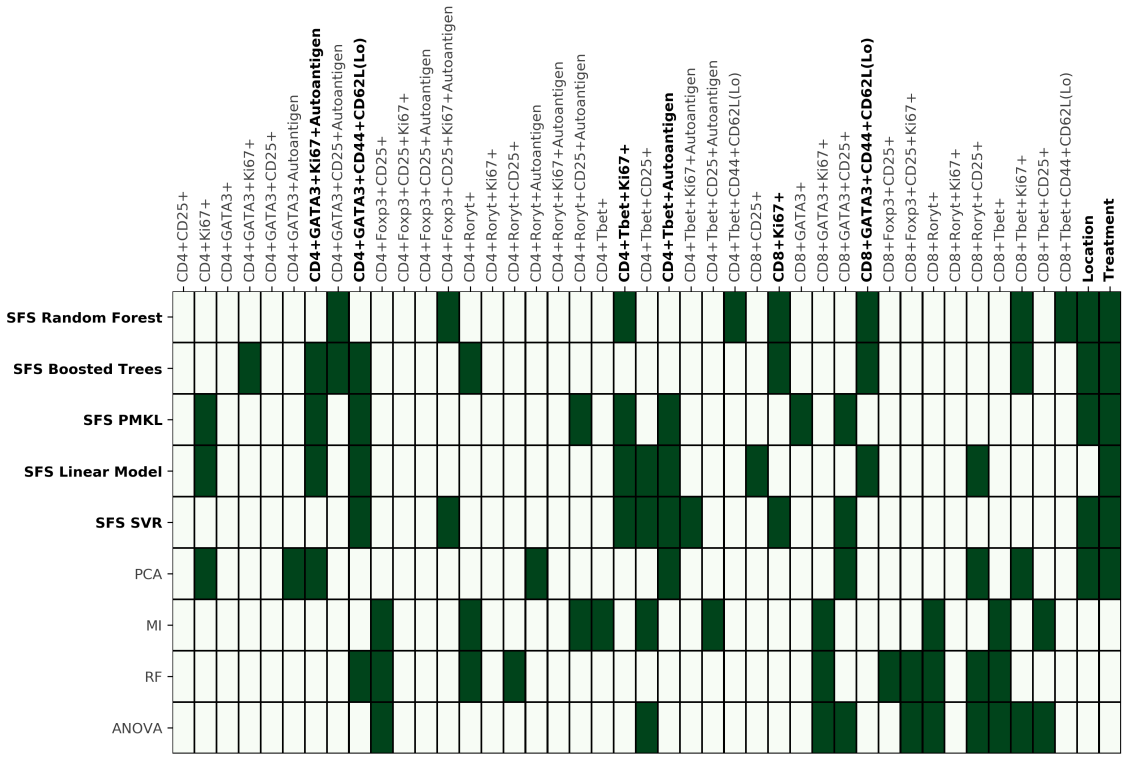


Figure 4.9: The green squares indicate that the feature selection method (left) selected the feature (top). The methods are ordered from lowest objective function, $J(b)$, at the top to greatest objective at the bottom. The SFS methods and the features most commonly selected by those methods are bolded.

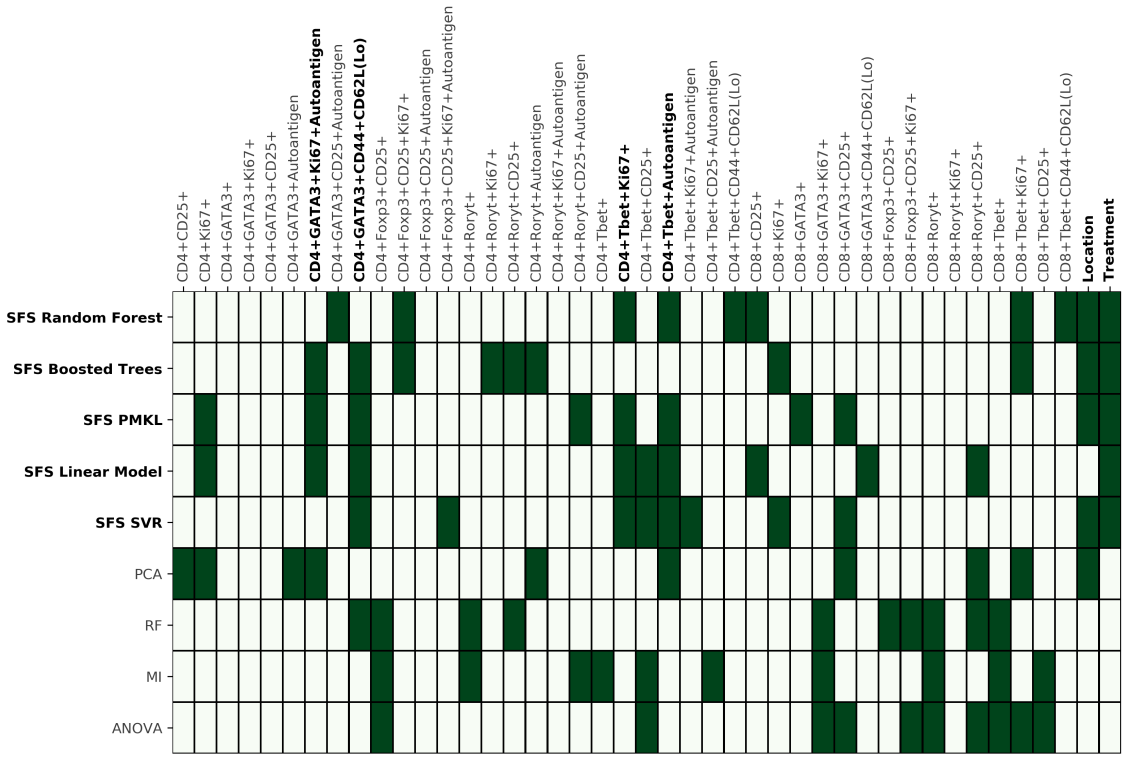


Figure 4.10: The green squares indicate that the feature selection method (left) selected the feature (top). The methods are ordered from lowest objective function, $J(b)$, at the top to greatest objective at the bottom. The SFS methods and the features most commonly selected by those methods are bolded.

Chapter 5

PREDICTIVE MODELING WITH CONSTRAINED POLYNOMIALS USING SUM-OF-SQUARES PROGRAMMING

A number of immunotherapies have been developed to modulate the immune response when the immune system fails to recognize and eliminate a threat such as cancer, or identifies self antigens as a threat as in Rheumatoid Arthritis. While immunotherapies are promising, for example as cancer treatments, the question that arises is how to determine an optimal immunotherapy for a patient given the current population of immune system cells [79]. Unfortunately, it is not always clear how exactly an immunotherapy will affect the immune system cells, nor how the current immune system cells can affect the immunotherapies effectiveness. Because the interactions between immunotherapies and immune system cells are unclear, generating physics based models, therefore, is not always tractable or efficient.

Rather than relying on physics based models of the immune system, in this chapter we look at data-driven methods for calculating the region of attraction of a system with unknown dynamics. Finding the Region Of Attraction (ROA) of nonlinear Ordinary Differential Equations (ODEs) is a well-studied and important problem. For instance, estimates of the region of attraction have been used in cases such as verifying and validating flight control [19] and analyzing cancer dormancy equilibria [108]. Unfortunately, in many real-world cases the vector field defining the ODE may not be known a priori and there exist few methods which can estimate the ROA. In this chapter we consider systems of the form

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0, \tag{5.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is the vector field and $x_0 \in \mathbb{R}^n$ is the initial condition.

Lyapunov functions and their predictive models, by design, must be globally positive. In Section 5.1 we first consider the problem of generating predictive models that are constrained to be globally positive or positive over a semialgebraic set using polynomial functions. Polynomial functions have been used to model system dynamics for system identification [23], and have been used to generate predictive models using, for instance, least squares regression [63]. In chapter 1 we showed that the set of Sum-of-Squares polynomials have been used in toolboxes such as SOStools [130] to solve important problems in control systems such as finding a Lyapunov function for a system with known dynamics. In this chapter we show that the class of sum-of-squares polynomials can also be used to model Lyapunov functions given only measured trajectory data of an unknown system.

We assume the vector field is unknown, but trajectory data is available. Specifically, define $g(x_0, t)$ to be the solution map of Eq. (5.1), where $g(x, 0) = x(0)$ and $\frac{d}{dt}g(x, t) = f(g(x, t))$ for all $x \in \mathbb{R}^n$ and $t \geq 0$. Then we assume trajectory data is available in the form of $g(x_i, k\Delta t)$ for $k = 1, \ldots, m$ and $i = 1, \ldots, m$. The question is then how to use this data to estimate the region of attraction.

In Section 5.2 we consider generating estimates of a converse Lyapunov function,

$$V(x) = \int_0^\infty \|g(x, t)\|^2 \, dt, \tag{5.2}$$

by observing and integrating trajectories $g(x_i, t)$ over possibly multiple initial conditions x_i. We then search for an optimal sum-of-squares polynomial by solving the least absolute deviations optimization problem,

$$\min_{h \in H} \quad \sum_{i=1}^{m} |h(x_i) - y_i| \,, \tag{5.3}$$

where H is the convex cone of sum-of-squares polynomials, $x_i \in \mathbb{R}^n$ from $i = 1, ..., m$ to be the given initial conditions and $y_i = V(x_i)$. We briefly note that there has been

some work on using trajectory data to fit Lyapunov functions for purposes other than estimating the region of attraction [89, 118]. However, these results provide no labeling (i.e. the true or estimated value of $V(x_i)$ is unknown) and hence cannot be used to estimate stability regions.

Upon obtaining the optimal Lyapunov function $V^* \in H$, we estimate the region of attraction as a maximal level set of the Lyapunov function. We define the γ level set as $V_\gamma^* = \{ x \mid \quad V^*(x) \leq \gamma\}$ and, after selection of γ, use V_γ^* as the estimate of the ROA.

Finally we model the Lyapunov function of the dynamics between cancer cell populations, and immune system cell populations for a range of immunotherapy treatments. We use simulated trajectory data of the population of tumor cells throughout time from various initial values of the immune system cells and a range of immunotherapy treatments. We assume in this case that the effect of the variation in patient dynamics is negligible when compared to the differences in the initial conditions of the immune system cells. Therefore the learned Lyapunov function is an average estimate among a collection of different patients.

We show that the generated model can be used to predict whether an immunotherapy treatment (within the given range) will lead to complete tumor elimination with greater than 90% accuracy over a subset of initial conditions even when patient dynamics differ. The prediction requires only an initial measurement of the populations of the tumor cells, some immune system cells and a cytokine. We then show that finding an optimal immunotherapy treatment can be formulated as a global polynomial optimization problem using the ROA model as a constraint.

5.1 Generating Optimal Predictive Models with Sum-of-Squares Polynomial Functions

In this section we consider the problem of generating polynomial models that are constrained to be globally positive or positive over a set.

Recall the semialgebraic set $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0,\ i = 1, \ldots, l\}$, where $g_i \in \mathbb{R}[x]$, $S \neq \emptyset$ and S is compact. We can enforce positivity of a polynomial p on S as follows,

$$\begin{aligned} \min_{\substack{p \in \mathbb{R}_k[x] \\ \sigma_i \in \mathbb{S}_k[x]\ \forall i=0,\ldots,l}} \quad & C(p) \\ \text{such that:} \quad & p = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i, \end{aligned} \tag{5.4}$$

where C is an objective function that is being minimized and the g_i are the polynomials in the semialgebraic set S. Let c be the decision variables that parametrize p, then if $C(p)$ is linear or quadratic with respect to the vector c, the problem can be efficiently solved as a semi-definite program using interior point methods such as those in [2] and a suitable solver such as Mosek [4] or SeDuMi [153].

5.1.1 Fitting Sum-of-Squares Polynomials to Data

Given data, $\{x_i \in \mathbb{R}^n, y_i \in \mathbb{R}\}_{i=1}^m$ one wants to find the function, $p(x)$ that best maps the x_i to the corresponding y_i. Two metrics used to select the model which best maps inputs to outputs are the least squares and least absolute deviations metric. The least squares and least absolute deviations regression objectives respectively are,

$$C(p) = \sum_{i=1}^{m} (p(x_i) - y_i)^2 \quad \text{and,} \quad C(p) = \sum_{i=1}^{m} |p(x_i) - y_i|. \tag{5.5}$$

The solution to Optimization Problem 5.4 using one of the objective functions in Eq. (5.5), is the degree k polynomial function that is positive on the set S and best maps the inputs to the outputs with respect to the given data and selected metric.

Least Absolute Deviations

Here we consider the general problem of how we may fit a SOS function, say $p : \mathbb{R}^n \to \mathbb{R}^+$ that is guaranteed to be positive on a semialgebraic set S and that maps from x_i to y_i with minimal error. Perhaps the simplest method of function fitting is to minimize some variation on $\sum_i \|p(x_i) - y_i\|$. We first select a computationally inexpensive approach of least absolute deviations, defined as

$$\min_{\substack{p\in\mathbb{R}_d[x] \\ \sigma_i\in\mathbb{S}_d[x] \ \forall \ i=1,\ldots,l}} \sum_{i=1}^{m} |p(x_i) - y_i| \,. \tag{5.6}$$

$$\text{such that:} \quad p(x) = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i,$$

where $p(x)$ is thus positive on $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0, \ i = 1, \ldots, l\}$.

While the objective function is still not linear, we may define a dummy variable $\gamma \in \mathbb{R}^m$ as well as $2m$ constraints to obtain the following optimization problem,

$$\min_{\substack{\gamma\in\mathbb{R}^m \\ p\in\mathbb{R}_d[x] \\ \sigma_i\in\mathbb{S}_d[x] \ \forall \ i=1,\ldots,l}} \quad \sum_{i=1}^{m} \gamma_i \tag{5.7}$$

$$\text{such that:} \quad p(x) = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i,$$

$$p(x_i) - y_i \geq \gamma_i, \qquad y_i - p(x_i) \geq \gamma_i, \ \forall \ i = 1, \ldots, m,$$

which is equivalent to optimization problem (5.6).

If we substitute the parameterization $\sigma_j(x) = Z_d^T(x_i) P_j Z_d(x_i)$ in optimization

problem (5.7), we have

$$\min_{\substack{\gamma\in\mathbb{R}^m \\ P_i\in\mathbb{R}^{q\times q}\ \forall\ i=0,\ldots,l}} \quad \sum_{i=1}^{m}\gamma_i \tag{5.8}$$

$$\text{such that: } Z_d^T(x_i)P_0Z_d(x_i) + \sum_{j=1}^{l} g_j(x_i)Z_d^T(x_i)P_jZ_d(x_i) - y_i \geq \gamma_i\ \forall\ i=1,\ldots,m,$$

$$y_i - Z_d^T(x_i)P_0Z_d(x_i) - \sum_{j=1}^{l} g_j(x_i)Z_d^T(x_i)P_jZ_d(x_i) \geq \gamma_i\ \forall\ i=1,\ldots,m.$$

Since the value of $Z_d^T(x_i)P_jZ_d(x_i)$ is linear with respect to the elements of P, this optimization problem then has $2m$ linear constraints, j semidefinite matrix decision variables of size $q \times q$ and a linear objective function. This problem can then be efficiently solved as a semi-definite program using interior point methods such as those in [2] and a suitable solver such as mosek [4] or SeDuMi [153].

Least Squares

We define the least squares version of the constrained polynomial optimization problem as,

$$\min_{\substack{p\in\mathbb{R}_d[x] \\ \sigma_i\in\mathbb{S}_d[x]\ \forall\ i=1,\ldots,l}} \sum_{i=1}^{m}\left(p(x_i)-y_i\right)^2 \tag{5.9}$$

$$\text{such that: } \quad p(x) = \sigma_0 + \sum_{i=1}^{l}\sigma_i g_i,$$

where $p(x)$ is thus positive on $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0,\ i=1,\ldots,l\}$.

While the objective function is not linear, we may define a dummy variable $\gamma \in \mathbb{R}^m$ as well as $2m$ constraints to obtain the following optimization problem,

$$\min_{\substack{\gamma\in\mathbb{R}^m \\ p\in\mathbb{R}_d[x] \\ \sigma_i\in\mathbb{S}_d[x]\ \forall\ i=1,\ldots,l}} \quad \sum_{i=1}^{m}\gamma_i \tag{5.10}$$

$$\text{such that: } \quad p(x) = \sigma_0 + \sum_{i=1}^{l}\sigma_i g_i, \quad \begin{bmatrix} \gamma_i & p(x_i)-y_i \\ p(x_i)-y_i & 1 \end{bmatrix} \succeq 0\ \forall\ i=1,\ldots,m,$$

which is equivalent to optimization problem (5.9) since the LMI constraint enforces $\gamma_i \geq (p(x_i) - y_i)^2$.

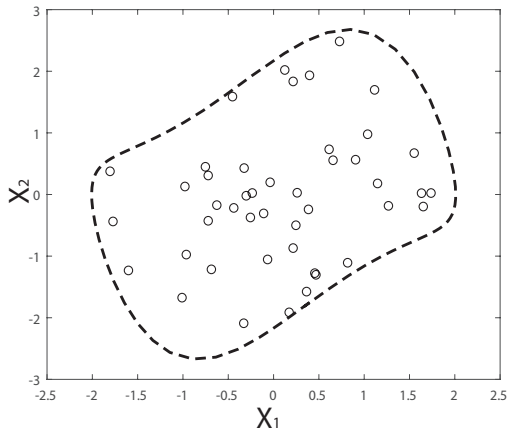
This optimization problem can be cast as an LMI and solved using semi-definite programming. If we replace $\sigma_i \in \mathbb{S}[x]$ with $\sigma_i(x) = Z_d(x)^T P_i Z_d(x)$ for $P_i \geq 0$ in optimization problem (5.10), we have

$$\begin{aligned} \min_{\substack{\gamma \in \mathbb{R}^m \\ P_i \in \mathbb{R}^{q \times q} \ \forall \ i=0,\ldots,l}} \quad & \sum_{i=1}^{m} \gamma_i \\ \text{such that:} \quad & p(x) = Z_d^T(x_i) P_0 Z_d(x_i) + \sum_{j=1}^{l} g_j(x_i) Z_d^T(x_i) P_j Z_d(x_i) \\ & \begin{bmatrix} \gamma_i & p(x_i) - y_i \\ p(x_i) - y_i & 1 \end{bmatrix} \succeq 0 \ \forall \ i = 1, \ldots, m. \end{aligned} \tag{5.11}$$

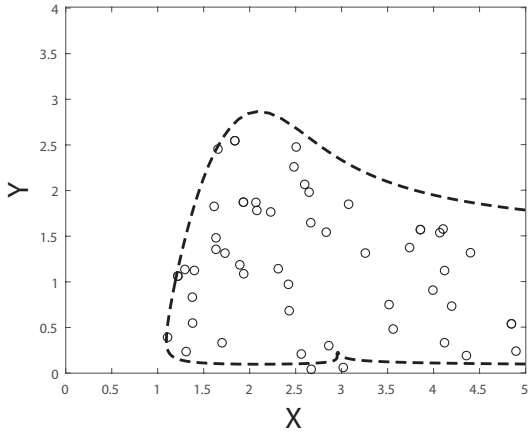
Since the value of $Z_d^T(x_i) P_j Z_d(x_i)$ is linear with respect to the elements of P, this optimization problem then has m 2x2 LMI constraints, j semidefinite matrix decision variables of size $q \times q$ and a linear objective function. This problem can then, like the least absolute deviations version of the optimization problem, be efficiently solved as a semi-definite program using interior point methods.

If $p(x)$ is only required to be globally positive everywhere, as in the next section, then $p = \sigma_0$ and the problem only has one semidefinite matrix decision variable of size $q \times q$. In this case the solution to optimization problem (5.8) and (5.11) returns the optimal function, $p^*(x) = Z_d(x)^T P^* Z_d(x)$.

In the following section we show how these constrained polynomial functions can be used to approximate the Region of Attraction from state measurements of a system with unknown dynamics.



(a) The initial condition data (black circles) used in optimization problem 5.8 for the Van Der Pol Oscillator. The area within the black dotted line is a numerical estimate of the region of attraction of the system using the reverse time dynamics.



(b) The initial condition data (black circles) used in optimization problem 5.8 for the Predator-Prey model. The area within the black dotted line is a numerical estimate of the region of attraction of the system using the reverse time dynamics.

Figure 5.1: Subfigures (a) and (b) show the initial conditions used to generate the data for optimization problem (5.8).

5.2 Data Based Estimation of the Region of Attraction

Recall that we consider systems of the form

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0, \tag{5.12}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is the vector field and $x_0 \in \mathbb{R}^n$ is the initial condition. We assume trajectory data is available in the form of $g(x_i, k\Delta t)$ for $k = 1, \ldots, m$ and $i = 1, \ldots, m$, where $g(x_0, t)$ is the solution map of Eq. (5.12).

5.2.1 Determining the Value of a Converse Lyapunov Function

We use trajectory data to define inputs of the form x_i and associated outputs of the form $y_i = \log(1 + V(x_i))$ where V if the converse Lyapunov function defined in Equation (5.2). If we let $g(x,t)$ be the solution map to the nonlinear ODE, then we define our trajectory data to be of the form of vectors $a(i,j) = g(x_i, j\Delta t)$ for $j = 1, ..., K$ where Δt is the measurement time-step and x_i are the initial conditions used to generate the trajectories. Then

$$V(x_i) = \int_0^{K\Delta t} \|g(x,t)\|^2 \, dt + V(a(i,K)).$$

If we take K sufficiently large, we may assume $a(i,K) \approx 0$. If we likewise assume Δt is small, then we make the approximation

$$V(x_i) = \int_0^{K\Delta t} \|g(x,t)\|^2 \, dt \approx \sum_{j=0}^{K} \|a(i,j)\|^2 \, \Delta t. \tag{5.13}$$

In practice, of course, we use a trapezoidal approximation of this integral. Ideally Δt is constant, however, the trapezoidal approximation of the integral does not require that the trajectory be measured on a specific time step. Thus we may still approximate the converse Lyapunov function in cases where measurements are taken in irregular time steps.

We conclude that a given set of trajectories at K time instants associated with L initial conditions gives us $K \cdot L$ values of $V(x)$.

However, we do not use $V(x_i)$ as our label. This is because $V(x)$ grows very quickly as x approaches the edge of the region of attraction and decreases quickly near the origin, resulting in several orders of magnitude variation. This variation causes performance issues when solving Optimization Problem (5.8) - points of a smaller magnitude have less influence on the value of the optimal function $V^*(x)$. To resolve this issue we instead use data labels $y_i = log(1 + V(x_i))$ as data for the

optimization algorithm where the values of $V(x_i)$ are obtained from Equation (5.13). This means, however, that the actual output from the optimization solver is

$$p^*(x) = Z_d^T(x)P^*Z_d(x) \cong \log(1 + V(x))$$

from whence we may obtain our estimate of the converse function as

$$V^*(x) = 10^{p^*(x)} - 1.$$

Note that $p^*(x) > 0$ if and only if $V(x) > 0$ and $p^*(x) = 0$ if and only if $V(x) = 0$. Furthermore, $\dot{V}(x(t)) \leq 0$ implies $\dot{p}^*(x(t)) \leq 0$ and hence $p^*(x)$ is fitting to a valid Lyapunov function for the system - albeit not the original converse from Section 2.

Estimating the Region of Attraction

Having shown how trajectory data can be used to provide training data to model a Lyapunov function, we now discuss how to use that function to estimate the region of attraction. We denote this fitted Lyapunov function as $V^*(x) = 10^{Z_d^T(x)P^*Z_d(x)} - 1$ which is the optimal globally positive sum-of-squares function returned by either Optimization Problem 5.8 or Optimization Problem 5.11.

Recall from Chapter 1 that the level set of a Lyapunov function,

$$V_\gamma := \{x \in \mathbb{R}^n \mid V(x) \leq \gamma\}$$

can be used as an estimate of the region of attraction if the time derivative of the Lyapunov function is also negative on the level set. However, if V is the converse Lypaunov function, then any state x where V is finite is part of the region of attraction (see [69]).

Therefore if one had the exact converse Lyapunov function (5.2), then by choosing a suitably large γ one could estimate the region of attraction arbitrarily well. In this

case however we have only an estimate, $V^*(x)$, of the Lyapunov function that is optimal with respect to the trajectory data we are given. In areas where we do not have trajectory information $V^*(x)$ is not likely to be accurate, so we must consider a metric with which to select γ.

Consider the largest value of the converse Lyapunov function (5.2) of all the trajectories used to find the optimal $V^*(x)$, which we will denote as $\gamma^* = \max_j\{V^*(x_j)\}$, where $j = 1, ..., m$ and x_j are the initial conditions of the given trajectories. Then we will only consider the level set of our optimal function, $V^*(x)$, that is less than or equal to γ^* since this is the smallest value of the actual converse Lyapunov function (5.2) that should contain all of the trajectory measurements.

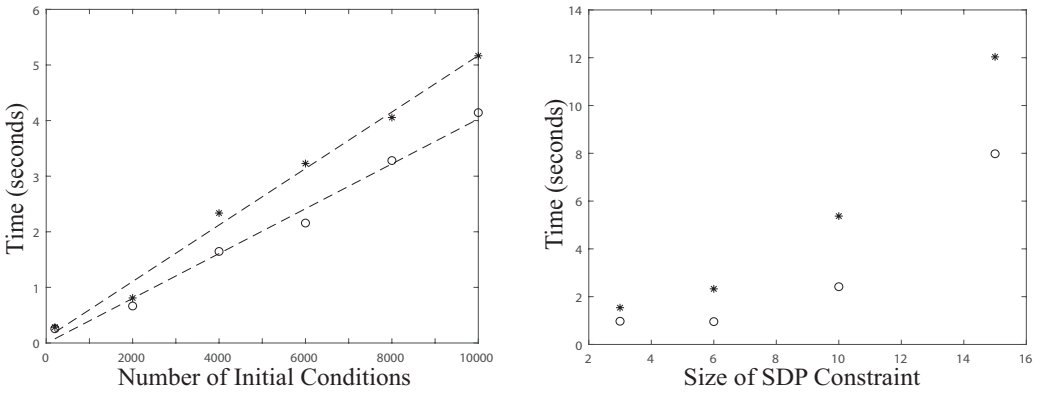
However, it is possible that γ^* may be too large of a value and the estimate of the region of attraction may contain points outside of the true region of attraction S. We will then consider a factor of safety, $0 < \eta \leq 1$, to define a smaller estimate of the region of attraction. We then have that our estimate for the region of attraction is

$$E_{\eta\gamma^*} := \{\ x\ |V^*(x) \leq \eta\gamma^*\},$$

where values of η that are closer to zero result in a smaller, more conservative estimate of the region of attraction when compared to values of η that are closer to one.

In cases where test data is available, one may find the percentage of points that are falsely determined to be within the region of attraction (false positives) and use the proportion of these as a metric for selecting η. For instance, if a more conservative estimate of the region of attraction is needed the value of η can be selected by choosing the largest value of η which has no false positive results on the test set.

Next we perform a series of numerical tests to determine the efficacy of the proposed method.



(a) The computation time of Optimization Problem (5.8) versus the number of data points. The black circles indicate a 2nd order polynomial was used and the black stars are 4th order polynomials. The black dashed line is the line of best fit through these points.

(b) The computation time of Optimization Problem (5.8) versus the size of the semi-definite matrix, $q = \binom{n+d}{n}$. The black circles are indicate the problem was optimized with 2000 data points and the black stars indicate the problem was optimized with 4000 data points.

Figure 5.2: Subfigures (a) and (b) plot the computation time of Optimization Problem (5.8) with respect to the number of data points (a) and the size of the semi-definite matrix (b).

5.3 Numerical Tests

In this section we provide the results from numerical tests on two nonlinear dynamical systems to determine the accuracy of the method and its numerical complexity. Then we perform a deeper analysis of a controlled nonlinear system which has five state variables. In this case we generate a function which, given a control strategy, returns a model of the Lyapunov function of the system dynamics for the control strategy and can be used to estimate if the initial conditions of the patient are within the ROA.

Table 5.1: Test set accuracy of the SOS optimal function on the Van Der Pol Oscillator and the Predator-Prey model data. Accuracy of the Lyapunov function is defined as the sum of the absolute error of the function for each test point divided by the total number of test points.

ODE	d = 2	d = 4	d = 6
Van Der Pol	0.3828	0.3942	0.1708
Predator-Prey	0.6043	0.2850	0.1546
Immune-Dynamics	0.5950	0.3498	1.3991

5.3.1 Numerical Results for Estimating the Region of Attraction

Here we perform numerical testing to estimate the performance of the ROA model on new data. Given new trajectory data that was not used to generate the model we calculate the number of initial conditions correctly identified as being in the region of attraction, falsely labeled as being in the region of attraction (false positives) and those that are falsely labeled as not being in the region of attraction (false negatives), including how the selection of γ affects those values. In addition we examine the effect of the polynomial degree on the accuracy of the ROA estimate and the accuracy of the region of attraction prediction, $V^*(x)$.

Example 1: Our first test system is the Van der Pol oscillator in reverse time, defined as

$$\begin{aligned} \dot{x_1} &= -x_2 \\ \dot{x_2} &= x_1 + x_2(x_1^2 - 1), \end{aligned} \tag{5.14}$$

which has a locally asymptotically stable equilibrium point at $x_1 = x_2 = 0$.

To generate the data set we use $L = 50$ different initial conditions taken from within the region of attraction, between a radius of 1 and 4 from the equilibrium

point $x = 0$. We simulate the trajectory of the nonlinear ODE for 10 seconds with $\Delta t = .1$ and $K = 100$, although we only use the $j = 1$ through $j = 4$ time-steps per trajectory for data generation, resulting in 200 data points of the form x_i, y_i. In addition, we add normally distributed noise scaled as 10^{-2} where the labels are typically in the interval $y_i \in [3, 13]$. Fig. 5.1 shows the initial conditions used to generate the data for optimization problem (5.8).

To evaluate the accuracy of the fit we generated a second test set of trajectory data with 500 initial conditions x_i evenly spread in the region of attraction and calculated the value of the converse Lyapunov function at each point. In Table 5.2 we calculate the average least absolute error of the ROA model on a test set of data. Increasing the degree of the ROA model decreased the error on the test set of data indicating that the degree 6 model performed best. A graphical representation of the region of attraction is presented In Fig. 5.3(a) where the true region of attraction of the system S and the estimated region of attraction for the 6th degree polynomial, $E_{\eta\gamma^*}$ for $\eta = 1$ are shown.

The 6th degree polynomial correctly identified 97.50% of the test set as being within the region of attraction with a 2.48% false negative rate and a 0.02% false positive rate, outperforming the lower degree polynomials in all categories except the false positive rate. In some cases a lower false positive rate may be required; a requirement that can be fulfilled by decreasing the value of η which decreases the false positive rate by returning a smaller estimate of the region of attraction. In Table 5.3 we see that decreasing η shrinks the region of attraction, decreasing both the percent of test data correctly categorized and the number of false positives.

Table 5.2: The percentage of initial conditions that were correctly determined to be within the region of attraction, falsely reported to be within the region of attraction and falsely reported to be outside of the region of attraction by the optimal Lyapunov function obtained from Optimization Problem (5.8).

ODE	Degree	True Pos.	False Pos.	False Neg.
Van Der Pol	2	96.16 %	0.00 %	3.84 %
	4	95.92 %	0.00 %	4.08 %
	6	97.50 %	0.02 %	2.48 %
Predator-Prey	2	61.22 %	35.22 %	3.56 %
	4	93.89 %	1.89 %	4.22 %
	6	96.11 %	2.11 %	1.78 %
Immune-Dynamics	2	90.83 %	1.79 %	9.17 %
	4	96.33 %	4.98 %	3.67 %
	6	81.88%	6.07 %	18.12 %

Example 2: The second test system is a biological model of predator-prey dynamics as described in [61],

$$\begin{aligned} \dot{x} &= x(-(x-\alpha)(x-\beta)-\gamma y) \\ \ddot{y} &= y(-c+x), \end{aligned} \tag{5.15}$$

where $\alpha = 1, \beta = 3, \gamma = 0.5$ and $c = 2.1$. Here α represents the minimum density for successful mating, and β represents the asymptotic carrying capacity. We are interested then in the region of attraction of the point $x = 2.1$, and $y = 1.98$, which is a locally asymptotically stable equilibrium point. The region of attraction of this point is the region over which the predator-prey system will asymptotically converge to a non-zero, desirable, equilibrium point.

To generate the data set we use $L = 50$ different initial conditions taken from within the region of attraction, between a radius of 1 and 4 from the equilibrium point $x = 2.1$, and $y = 1.98$. We simulate the trajectory of the nonlinear ODE for 10 seconds with $\Delta t = .1$ and $K = 100$, although we only use the $j = 1$ through $j = 4$ time-steps per trajectory for data generation, resulting in 200 data points of the form x_i, y_i. In addition, we add normally distributed noise scaled as 10^{-2} where the labels are typically in the interval $y_i \in [3, 13]$. Fig. 5.1 shows the initial conditions used to generate the data for optimization problem (5.8).

In Table 5.2 we calculate the average least absolute error of the ROA model on a test set of data. As in example 1, increasing the degree of the ROA model decreased the error on the test set of data indicating that the degree 6 model performed best. A graphical representation of the region of attraction is presented In Fig. 5.3(b) where the true region of attraction of the system S and the estimated region of attraction for the 6th degree polynomial, $E_{\eta\gamma^*}$ for $\eta = 1$ are shown.

The 2nd degree polynomial model performs nearly 30% worse with respect to the accuracy when compared to example 1. Since the predator-prey model has a more complicated region of attraction than example 1, the 2nd degree polynomial is insufficient to capture this region. In fact we see an increase in accuracy of over 30% from the 2nd degree to the 4th degree sum-of-squares polynomial and the 4th degree polynomial models of example 1 and 2 have similar performance.

The 6th degree sum-of-squares polynomial model performed best, correctly identified 96.11% of the test set as being within the region of attraction with a 2.1% false positive rate and a 1.78% false negative rate. In Table 5.3 we see that decreasing η again decreases the percent of test data correctly categorized, but in this case it causes a significant decrease in the number of false positives. In fact when decreasing η from 1 to 0.8 we have that the false positive rate drops from 2.11% to 0.00%.

5.3.2 Computational Complexity of the Optimization Problem

Using the data simulated using the nonlinear dynamics of the Examples 1 and 2 we will analyze the computational complexity of the optimization problem.

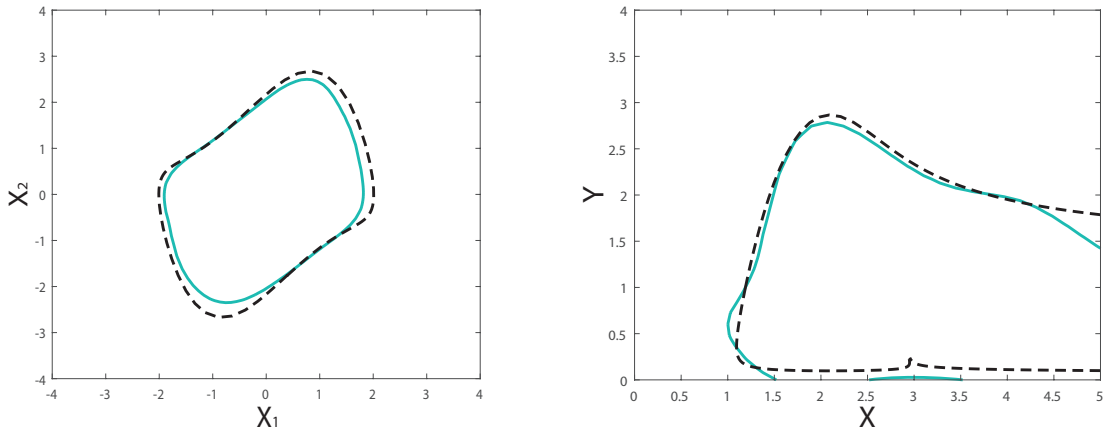
In Fig. 5.2 we plot the computation time for Optimization Problem (5.8) versus: the number of data points in Subfigure (a); and versus the length of the semi-definite matrix $P \in \mathbb{R}^{q(d)\times q(d)}$ where $q = \binom{n+d}{n}$ in Subfigure (b). The best least absolute deviations fit to the complexity data in Subfigure (a) had a slope of .004 and .005 for the degree two polynomial and the degree four polynomial respectively.

Based on the numerical data the complexity of optimization problem (5.8) scales linearly with respect to m, the number of data points used in the optimization problem. With respect to the length of the semi-definite matrix, the computational complexity is sublinear. This implies that modeling problems with large numbers of states or higher degree polynomial models may be computationally expensive.

5.3.3 Modeling the Region of Attraction of a Biological System with Pulsed-Immunotherapy

Here we perform an analysis of simulated trajectories of an immunotherapy for the treatment of cancer. We separate the analysis of this system from Examples 1 and 2 because the ROA model can account for changes in the dynamics due to different control strategies - unlike the previous two uncontrolled system examples.

Rather than modeling a single Lyapunov function, the learned function will return a Lyapunov function for any of the supplied control strategies, of a given form. Then, for a given control strategy, the model of the Lyapunov function can be used to predict whether the initial conditions of the patient are within the ROA assuming the given control strategy is applied.



(a) The area contained within the black dotted line is within the region of attraction of the Van Der Pol Oscillator and the area contained within the blue line is the estimate of the region of attraction defined as $E_{\eta\gamma^*}$ where $\eta = 1$.

(b) The area contained within the black dotted line is within the region of attraction of the predator-prey model and the area contained within the blue line is the estimate of the region of attraction defined as $E_{\eta\gamma^*}$ where $\eta = 1$.

Figure 5.3: Subfigures (a) and (b) show the estimated region of attraction E_{γ^*} versus the true region of attraction identified by observing the trajectories of the system in reverse.

We consider a nonlinear system developed in [170] with five state variables, that was first defined in Chapter 1,

$$\begin{aligned}
\dot{T}(t) &= a_0 T(t)(1 - c_0 T(t)) - \delta_0 \frac{E(t)T(t)}{1 + c_1 B(t)} - \delta_o T(t) V(t), \\
\dot{B}(t) &= a_1 \frac{T(t)^2}{c_2 + T(t)^2} - dB(t), \\
\dot{E}(t) &= \frac{f E(t) T(t)}{1 + c_3 T(t) B(t)} - rE(t) - \delta_0 R(t) E(t) - \delta_1 E(t), \\
\dot{R(t)} &= rE(t) - \delta_1 R(t), \\
\dot{V}(t) &= g(t) - \delta_1 V(t),
\end{aligned} \tag{5.16}$$

where the nominal values of the constants a_0, c_0, δ_0, c_1, a_1, d, c_3, f, and r are as

defined in Table 1 in [170], the constant $c_2 = 7,000$ represents the addition of an anti-TGF-β treatment and $g(t)$ represents a pulsed immunotherapy treatment.

These dynamics define the relationship between tumor size T, TGF-β B, effector immune cells E, regulatory immune cells R and activated tumor-specific cytotoxic T-cells administered as a vaccine V. The effector immune cells and the immune cells added into the system via vaccination act to deplete the number of tumor cells, while the regulatory cells R and amount of TGF-β, B, act to decrease the number of effector immune cells.

As opposed to the previous two examples we are primarily interested only in the final state of one of the variables. In this case we are interested solely in the region of attraction of the tumor dynamics, and while the other variables also reach an equilibrium we are unconcerned with their final values.

Therefore we generate measurements of a converse Lyapunov function as the integral of the tumor state alone, as opposed to the integral of all states as in the previous two examples. We model the input $g(t)$ as a pulsed immunotherapy treatment where the dose is denoted as d_v, the period of treatment as p_v, and the number of treatments as n_v. We model the dynamics for 60 days and assume that the treatment will continue for the entire 60 day period to ensure tumor elimination.

The inputs for the resulting model are the initial values of T, B, E, R, and the dose and number of treatments d_v and n_v. The training data is obtained by uniformly sampling initial conditions for $T(0) \in [0, 10]$, $B(0) \in [0, .5]$, $E(0) \in [50, 300]$, $R(0) \in [0, 50]$, $V(0) = 0$ and $d_v \in [0, 10000]$ and $n_v \in [60, \frac{60}{7}]$. The number of doses given varies between once a day to once a week. In addition we simulate variability in the dynamics of each patient by uniformly drawing the coefficients to be $\pm 10\%$ of the nominal coefficient values.

To generate the data set we use $L = 443$ different initial conditions taken from

Table 5.3: The percentage of initial conditions that were correctly determined to be within the region of attraction, falsely reported to be within the region of attraction and falsely reported to be outside of the region of attraction by the optimal Lyapunov function obtained from Optimization Problem (5.8). All examples are for the degree 6 model of the ROA except for the Immune-dynamics which is a degree 4 model.

ODE	η	True Pos.	False Pos.	False Neg.
Van Der Pol	0.6	94.64 %	0.00 %	5.36 %
	0.8	96.28 %	0.00 %	3.72 %
	1	97.50 %	0.02 %	2.48 %
Predator-Prey	0.8	92.89 %	0.00 %	7.11 %
	0.9	95.89 %	0.67 %	3.44 %
	1	96.11 %	2.11 %	1.78 %
Immune-Dynamics	0.8	56.39 %	0.20 %	43.61 %
	0.9	81.97%	1.18%	18.03%
	1	96.33 %	4.98 %	3.67 %

within the region of attraction and 557 initial conditions taken outside the region of attraction for a total of 1000 intial conditions. We simulate the trajectory of the nonlinear ODE for 60 days with $\Delta t = 1$ day and $K = 60$.

To select an optimal model of the Lyapunov function using points outside of the region of attraction, we will add one additional constraint to optimization problem (5.8) as proposed in [95]. Suppose, $\{x_i, y_i\}_{i=1}^{m}$, be the initial conditions and converse Lyapunov function values of initial conditions within the region of attraction, and let $\{z_i\}_{i=1}^{m_o}$ be initial conditions that do not converge and are thus outside the region of attraction. Then the value of the Lyapunov function at values z_i must be greater

than $y_{\max} = \arg\max_i y_i$. Thus we solve the following modified optimization problem to force the Lyapunov function values at z_i to be greater than $y_{\max}$.

$$\begin{aligned} \min_{\substack{\gamma \in \mathbb{R}^m \\ P \in \mathbb{R}^{q \times q}}} \quad & \sum_{i=1}^{m} \gamma_i \\ \text{such that:} \quad & Z_d^T(x_i) P Z_d(x_i) - y_i \geq \gamma_i \ \ \forall\ i = 1, \ldots, m \\ & y_i - Z_d^T(x_i) P Z_d(x_i) \geq \gamma_i \ \ \forall\ i = 1, \ldots, m \\ & Z_d^T(z_i) P Z_d(z_i) - y_{\max} \geq 0 \ \ \forall\ i = 1, \ldots, m_o \end{aligned} \tag{5.17}$$

Because of patient variability, the LAD metric used to fit p to the given data implies that p should be a median estimate of the Lyapunov function for all patients. However, the introduction of the previous constraint enforces the Lyapunov function to overestimate the value of the Lyapunov function anywhere a treatment failed to eliminate a patients tumor - increasing the number of true positives and false negatives.

The Lyapunov function, and thus the ROA estimate, is dependent upon the selected treatment strategy. Therefore given a dosage d_v and number of doses n_v, the resulting polynomial model of the ROA is the semi-algebraic set $S = \{[T, B, E, R] \in \mathbb{R}^4 \mid p(T, B, E, R, d_v, n_v) \leq \alpha\}$ where p is a polynomial returned by the proposed algorithm and α is the value of the selected level set.

In Table 5.2 we calculate the average least absolute error of the ROA model returned by Optimization Problem (5.17) on a test set of 10,000 data points. Unlike examples 1 and 2, increasing the degree of the ROA model only decreased the error on the test set of data for the degree 4 polynomial model. This implies that the degree 6 model overfit to the data and we restrict the analysis to the degree 4 polynomial model.

The 4th degree sum-of-squares polynomial correctly identified 96.53% of the test

set as being within the region of attraction with a 5.37% false positive rate. In Table 5.3 we see that decreasing η again decreases the percent of test data correctly categorized, but causes a significant decrease in the number of false positives. When decreasing η from 1 to 0.8 we have that the false positive rate drops from 5.37% to just 0.20% but unfortunately decreases the accuracy of true positives to just 56.39%. Because of the low rate of false positives, we select a value of $\eta = 1$ and the semi-algebraic set of the estimated ROA, for given n_v and d_v, is then

$$S_1 = \{[T, B, E, R] \in \mathbb{R}^4 \mid p(T, B, E, R, d_v, n_v) \leq 7.4390\}.$$

However, p can be used to do more than just estimate the region of attraction for a given treatment strategy. Instead, suppose we are given the initial conditions, $T(0)$, $B(0)$, $E(0)$ and $R(0)$ of a patient and want to search for a treatment strategy that minimizes the volume of administered immunotherapy but still eliminates the tumor in the patient.

Given a patients initial conditions T_0, B_0, E_0 and R_0 (and the model p), solving the following optimization problem will return a predicted open loop treatment strategy, where x_1 is the dose and x_2 is the number of doses, that leads to complete tumor elimination while minimizing the volume of administered immunotherapy.

$$\begin{aligned} \underset{x \in \mathbb{R}^2}{min} \quad & x_1 x_2 \\ \text{subject to} \quad & 7.4390 - p(T_0, B_0, E_0, R_0, x_1, x_2) \geq 0 \end{aligned} \tag{5.18}$$

Optimization Problem (5.18) is a type of Global Polynomial Optimization (GPO) problem because the objective is bi-linear and p is a polynomial function. In the next chapter we identify a new method for solving such problems. The solution to this problem uses the ROA model to generate a predicted treatment strategy that eliminate the cancer cells while minimizing the amount of treatment administered.

Thus treatment strategies can be selected without any knowledge of the underlying system dynamics.

5.4 Conclusion

We have proposed a new method for fitting a polynomial function to given data using least absolute deviations or least squares while additionally constraining the function to be positive over semi-algebraic sets. We validate this approach by modeling a a nonnegative Lyapunov function of a system with unknown dynamics, and show that the resulting models are constrained and effective.

The model of the Lyapunov function is then used to estimate the region of attraction of the nonlinear ODE given only data on the trajectory of the system over a finite set of initial conditions. This method is independent of any knowledge of the vector field, and the region of attraction can therefore be predicted without requiring the system dynamics to be identified. We ran numerical tests on two systems with two states, and one nonlinear controlled system with four states. We show how using the ROA model can be used to identify an optimal pulsed immunotherapy treatment strategy by solving a GPO problem - but leave further analysis of the system to the next chapter.

Chapter 6

SOLVING GLOBAL POLYNOMIAL OPTIMIZATION PROBLEMS

In Chapter 5 we formulated a global polynomial optimization problem whose solution returns an optimal pulsed immunotherapy dosage and period of treatment, that is based on the populations of immune system cells in the patient. In this Chapter we develop a method for solving such global polynomial optimization problems, and analyze the optimal treatments found by solving the optimization problem from the previous chapter.

Global Polynomial Optimization (GPO) is defined as optimization of the form

$$\begin{aligned} f^* := \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \geq 0 \quad \text{for } i = 1, \cdots, s \\ & h_j(x) = 0 \quad \text{for } j = 1, \cdots, t, \end{aligned} \tag{6.1}$$

where f, g_i, and h_i are real-valued polynomials in decision variables x.

As defined in Eq. 6.1, the GPO problem encompasses many well-studied subclasses including Linear Programming (LP) [88, 85], Quadratic Programming (QP) [106], Integer Programming (IP), Semidefinite Programming (SDP) and Mixed-Integer Nonlinear Programming (MINLP) [99]. Because of its generalized form, almost any optimization problem can be cast or approximately cast as a GPO, including certain NP-hard problems from economic dispatch [122], optimal power flow [64] and optimal decentralized control [101]. As applied to control theory, GPO can be used for stability analysis of polynomial dynamical systems by, e.g., verifying polytopic invariants as in [110].

More information on the GPO problem can be found in Chapter 2, including other special cases where alternative methods can be used to solve this problem. In this section however, we propose a polynomial-time algorithm for using SOS estimates of the Greatest Lower Bound (GLB) to extract arbitrarily accurate approximate solutions. Specifically, define the feasible set

$$S := \{x \in \mathbb{R}^n \,:\, g_i(x) \geq 0,\ h_j(x) = 0\}. \tag{6.2}$$

The Greatest Lower Bound (GLB) problem associated to the GPO Problem (6.1) is defined as

$$\begin{aligned} \lambda^* :=& \max_{\lambda \in \mathbb{R}} \quad \lambda \\ & \text{subject to} \quad f(x) - \lambda > 0, \quad \forall x \in S. \end{aligned} \tag{6.3}$$

The GLB and GPO problems are closely related in that $\lambda^* = f^* = f(x^*)$, but are not equivalent in that the GLB problem does not find x^*. Our approach is based on the observation that while the GLB problem does not return x^*, it can be used as a selection criteria in a bisection algorithm - See Section 6.3. This means that an algorithm which solves the GLB problem with complexity $O(k)$ can be combined with bisection to find a point $x \in \mathbb{R}^n$ such that $|x - x^*| \leq \epsilon$ for any ϵ and the resulting complexity is of order $O(kn \log(\frac{r}{\epsilon}))$ where r is the radius of a hyper-sphere containing all feasible points.

In Section 6.4 we propose a sequence of algorithms, E_k, and show that they will return in polynomial-time a point $x_k \in \mathbb{R}^n$ that is sub-optimal to the GPO problem in the following sense. If x_k is the sequence of proposed solutions produced by the sequence of algorithms E_k, we show that if the feasible set, S, of Problem (6.1) is bounded, then there exist a sequence of feasible points $y_k \in S$ such that $\lim_{k\to\infty} ||x_k - y_k|| = 0$ and $\lim_{k\to\infty} f(y_k) = \lim_{k\to\infty} f(x_k) = f^*$, where f^* is the objective value of the GPO problem.

In Section 6.5, we illustrate the effectiveness of the proposed algorithm by applying it to an example problem wherein the existing moment based approach fails to extract a solution. We then use the algorithm to solve Optimization Problem (5.18) from Chapter (5) to demonstrate how an optimal pulsed immunotherapy treatment can be identified even without knowledge of the underlying system dynamics.

6.1 Problem Statement

In this chapter, we consider simplified GPO problems of the form:

$$\begin{aligned} f^* := \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{such that:} \quad & g_i(x) \geq 0 \qquad \text{for } i = 0, \cdots, l \end{aligned} \tag{6.4}$$

where $f, g_i \in \mathbb{R}[x]$. The class of problems in (6.4) is equivalent to that in (6.1), where we have simply replaced every $h_i(x) = 0$ constraint with some $g_1(x) = h(x) \geq 0$ and $g_2(x) = h(x) \leq 0$. For every problem of Form (6.4), we define the associated feasible set $S := \{x \in \mathbb{R}^n \, : \, g_i(x) \geq 0\}$.

First we assume $S \neq \emptyset$, otherwise there would be no feasible point and therefore no optimal point. Note that given g_i, one may use SOS optimization combined with Positivstellensatz results in [152] to determine feasibility of S.

Proposed Algorithm In this subsection, we propose a GLB and Branch and Bound-based algorithm which, for any given $\epsilon > 0$, will return some $x \in \mathbb{R}^n$ for which there exists a point $x \in S$ such that:

$$f(x) - f^* \leq \epsilon \tag{6.5}$$

Before defining this algorithm, however, in the following section, we describe some background on the dual SOS and Moment algorithms for generating approximate solutions of the GLB Problem.

6.2 SOS approach to solving the GLB problem

In this section, we briefly describe the use of SOS programming to define a hierarchy of GLB problems. Recall from Chapter 2 that the quadratic module is defined as follows.

Definition 35. *Given a finite collection of polynomials $g_i \in \mathbb{R}[x]$, we define the quadratic module as*

$$M := \{p \mid p = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i \quad \sigma_i \in \mathbb{S}[x]\},$$

and the degree-k bounded quadratic module as

$$M^{(k)} := \{p \mid p = \sigma_0 + \sum_{i=1}^{l} \sigma_i g_i \quad \sigma_i \in \mathbb{S}_k[x]\}.$$

Consider the GPO Problem (6.4) where $\{g_i\}$ is an Archimedean representation of the feasible set, S, with associated quadratic module M. We now define the degree-unbounded version of the SOS GLB problem.

$$\begin{aligned} f^* = \lambda^* := \max_{\lambda \in \mathbb{R}} \quad & \lambda \\ \text{such that:} \quad & f(x) - \lambda \in M. \end{aligned} \tag{6.6}$$

Since M is Archimedian, it follows that $\lambda^* = f^*$ (where f^* is as defined in (6.4)). Although Problem (6.6) is convex, for practical implementation we must restrict the degree of the SOS polynomials which parameterize M - meaning, we must restrict ourselves to optimization on $M^{(k)}$. This defines a new sequence of GLB problems as

$$\begin{aligned} p_k^* := \quad \max_{\lambda \in \mathbb{R}} \quad & \lambda \\ \text{such that:} \quad & f(x) - \lambda \in M^{(k)}. \end{aligned} \tag{6.7}$$

Clearly, $p_i^* \leq p_j^* \leq \lambda^*$ for any $i < j$. Additionally, it was shown in [123] that $\lim_{k \to \infty} p_k^* = p^*$. Furthermore, it was shown in [141, 116] that bounds on the convergence rate of

$p_k^* \to \lambda^*$ exist as a function of g_i, f and k. Finally, the computational complexity of p_k is equivalent to that of a semidefinite program with order $(l+1)\Lambda(\lceil \frac{k}{2} \rceil)^2$ scalar variables, where l is the number of g_i in Problem (6.4) and $\Lambda(d) := \binom{d+n}{d}$.

6.3 Solving the GPO Problem using the ideal Branch and Bound

In this section we show that, given an algorithm that returns the exact solution to the GLB problem, we can design an algorithm that returns $x \in \mathbb{R}^n$ such that $|x - x^*| \leq \epsilon$, using $\frac{2}{\log 2} n \log(\frac{r}{\epsilon})$ computations of the GLB for any desired accuracy ϵ.

The Ideal Branch and Bound Algorithm

At every iteration, we have an active hyper-rectangle $A(j) = [a, b]$;

1. Initialize the algorithm $A(0) = [-rI, rI]$, $j = 0$;
2. Bisect $A(j) = [a, b] = [a', b'] \cup [a'', b''] = A_1 \cup A_2$;
3. Compute the Greatest Lower Bound of

$$\begin{aligned} \lambda_i^* := \max_{\lambda \in \mathbb{R}} \quad & \lambda \\ \text{subject to} \quad & f(x) - \lambda > 0 \quad , \forall x \in S \cap A_i; \end{aligned} \tag{6.8}$$

4. If $\lambda_1^* > \lambda_2^*$, set $A(j+1) = A_1$, otherwise $A(j+1) = A_2$;
5. Set $j = j + 1$ and go to step 2;

At termination, we may choose any $x \in A$, which will be accurate within $|x - x^*| \leq r 2^{-j/n}$.

Let us examine these steps in more detail.

Initialize the algorithm Since the set S is compact, there exists some $r > 0$ such that $S \subset B_r(r)$. We may then initialize $A = [-r\mathbf{1}, r\mathbf{1}]$, where $\mathbf{1}$ is the vector of all 1's.

Bisect Bisection of the hypercube occurs along the longest edge. Thus, after n iterations, we are guaranteed a two-fold increase in accuracy. As a result, the largest edge of the hypercube diminishes as $2^{-k/n}$, and we need to compute the GLB $2n\log(\frac{r}{\epsilon})$ times to attain the desired accuracy.

Compute the Greatest Lower bound We assume that our solution to the GLB problem is exact. In this case, we are guaranteed that an optimizing x will always lie in A_i.

Complexity To attain a desired accuracy of ϵ requires $\frac{n\log(\frac{r}{\epsilon})}{\log(2)}$ number of bisections, since the largest edge of the hypercube generates a two fold decrease in the error for every n bisections. This means that the GLB must be computed $\frac{2n\log(\frac{r}{\epsilon})}{\log(2)}$ times and if the GLB is of complexity $O(k)$, then the complexity of the ideal algorithm must be $O(c_1 kn\log(\frac{r}{\epsilon}))$ where $c_1 = \frac{2}{\log(2)}$.

As an example, consider the simple two variate optimization problem,

$$\begin{aligned} &\min_{x,y\in\mathbb{R}} \quad y \\ &\text{subject to} \quad x+5\geq 0, \qquad xy-10\geq 0, \\ &\qquad\qquad 15-x-y\geq 0, \quad x^2+3y^2-180=0. \end{aligned} \tag{6.9}$$

Conceptually, the first eight iterations of the ideal branch and bound algorithm for this two variate optimization problem are displayed in Figure 6.1. In this case, the algorithm returns a sequence of nested rectangles $A(k)$. Let $A_1(k)$ and $A_2(k)$ be the bisection of $A(k)$ and $\lambda^*(A)$ be the greatest lower bound over domain A. If the optimal feasible point x^* lies in the rectangle $A(k)$, and $i(k) = \arg\min\{\lambda^*(A_1(k)), \lambda^*(A_2(k))\}$, then x^* is guaranteed to lie in $A_{i(k)}$. Therefore, by induction, at every iteration k, x^* is guaranteed to lie in domain $A(k) = A_{i(k)}$. Figure 6.1 illustrates the rectangle discarded at each iteration, the optimal point x_1^* (which lies in A_k at each iteration) and the associated optimal objective values $\lambda^*(A_i(k))$.

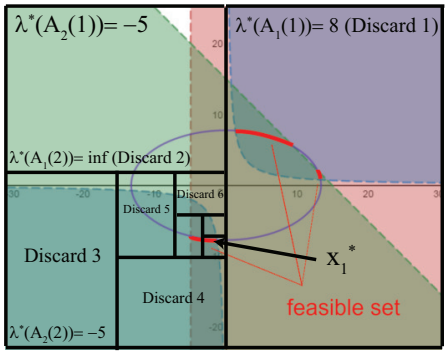


Figure 6.1: The ideal Branch and Bound Algorithm applied to the two-variate Optimization Problem (6.9). Note that every iteration discards half of the active hyper-rectangle.

6.4 Modified Branch and Bound Algorithm

Here, we present a slightly modified branch and bound algorithm that combined with SOS/Moment relaxations, can approximate the solution to the GPO problem to any desired accuracy, in a certain sense.

The Modified Branch and Bound Algorithm At every iteration, we have an active hyper-rectangle $A = [a, b]$, a set of feasible rectangles $Z = \{[a_i, b_i]\}_i$ each with associated GLB λ_i.

1. Initialize the algorithm
2. Bisect $A = [a, b] = [a', b'] \cup [a'', b''] = A_1 \cup A_2$
3. Compute the Greatest Lower Bound of

$$\begin{aligned} \lambda_i^* := \max_{\lambda \in \mathbb{R}} \quad & \lambda \\ \text{such that:} \quad & f(x) - \lambda > 0 \quad , \forall x \in S \cap A_i. \end{aligned} \tag{6.10}$$

4. Find a locally optimal solution to

$$x_i^* := \min_{x \in \mathbb{R}^n} \quad f(x) \tag{6.11}$$
$$\text{such that:} \quad x \in S \cap A_i, \quad g_j(x) \geq 0 \qquad \text{for } j = 0, \cdots, l$$

5. If $\lambda_i^* \leq f(x_i^*) - \epsilon$, add A_i to Z and λ_i to Λ; otherwise add x_i^* to P.

6. Discard any Z_j where $\lambda_j^* > \min_{p \in P} f(p)$.

7. If Z is empty stop; otherwise set $A = Z_i$ where Z_i is the smallest element of Z and goto 2.

At termination, the globally optimal point is $\arg\min_{p \in P} f(p)$, which must be accurate within ϵ even if the local solvers were unable to find a solution.

6.4.1 The GLB and GD Subroutines

Consider GPO Problem (6.4) and suppose the corresponding feasible set $S := \{x \in \mathbb{R}^n : g_i(x) \geq 0\}$, is nonempty and compact with $S \subset C(a, b)$, for some $a, b \in \mathbb{R}^n$ with associated Archimedean quadratic module M.

Given $A = [a, b]$, define the polynomials $w_i(x) := (b_i - x_i)(x_i - a_i)$. These polynomials are then used to define the modified feasible set $S \cap A$ as

$$S_{ab} := \{x \in \mathbb{R}^n : g_i(x) \geq 0, \ \forall i : 1 \leq i \leq s, w_j(x) \geq 0, \ \forall j : \ 1 \leq j \leq n\}, \tag{6.12}$$

Before defining the main sequential algorithm E_k, we will define the k^{th}-order SOS/Moment GLB subroutine, denoted B_k, which calculates the GLB in Step (3) of the Modified Branch and Bound Algorithm, and the Sequential Quadratic Programming subroutine which calculates a locally optimal point in Step (4) of the Modified Branch and Bound Algorithm.

GLB Subroutine $\lambda^*_{\mathbf{k}} = \mathbf{B_k}[\mathbf{a}, \mathbf{b}]$

The corresponding modified degree-k bounded quadratic module for the hyper-rectangle A is defined as,

$$M^{(k)}_{a\,b} := \Big\{ p \,:\, p = \sum_{i=0}^{s} \sigma_i g_i + \sum_{i=s+1}^{s+n} \sigma_i w_i, \quad \sigma_i \in \Sigma_S, \deg(\sigma_i g_i) \leq k, \;\; \deg(\sigma_i w_i) \leq k \Big\}, \tag{6.13}$$

where $g_0(x) = 1$. This allows us to formulate and solve the modified k-th order SOS GLB problem

$$\begin{aligned} \lambda^*_k := \quad & \max_{\lambda \in \mathbb{R},\ \sigma_i \in \Sigma_S \ \forall i = 1 \ldots s+n} \quad \lambda \\ & \text{such that:} \quad f(x) - \lambda \in M^{(k)}_{a\,b} \end{aligned} \tag{6.14}$$

and the corresponding dual GLB moment problem. The subroutine returns the value λ^*_k such that $|f^* - \lambda^*_k| \cong \frac{c_2}{\sqrt[c_1]{\log(k)}}$.

The Sequential Quadratic Programming Subroutine $\mathbf{x^*_k} = \mathbf{G}[\mathbf{a}, \mathbf{b}]$

We may solve the original GPO problem, with the additional constraint that, $x \in A_i$, using the following optimization problem,

$$G_{a\,b} := \min_{x \in \mathbb{R}^n} \quad f(x) \tag{6.15}$$

$$\text{such that:} \quad x \in S \cap A_i, \;\; g_j(x) \geq 0 \qquad \text{for } j = 0, \cdots, l. \tag{6.16}$$

Since the derivatives of polynomial functions are smooth we may use algorithms such as Sequential Quadratic Programming (SQP) to find a locally optimal solution using MATLABs `fmincon` application of SQP. When the locally optimal solution returned by the SQP is within ϵ of the GLB returned by the GLB subroutine, then x^* is a globally optimal solution with ϵ tolerance to the GPO problem on the hyper-rectangle A_i. The locally optimal points returned by the SQP subroutine can also be used to discard hyper-rectangles if the GLB on those hyper-rectangles are greater than $f(x^*)$,

since a globally optimal solution could not possibly exist on that hyper-rectangular set.

6.4.2 Formal Definition of the Modified Branch and Bound Algorithm, E_k

We now define a sequence of Algorithms E_k such that for any $k \in \mathbb{N}$, E_k takes GPO Problem (6.4) and returns an estimated feasible point x^*.

The Sequence of Algorithms E_k:

In the following, we use the notation $a \leftarrow b$ to indicate that the algorithm takes value b and assigns it to a. The parameter l represents the number of branch and bound loops and in Theorem 38 is set by the desired accuracy as $l > n \log_2(\frac{L\sqrt{n}}{\epsilon})$ where n is the number of variables and L is a bound on the radius of the feasible set.

The inputs to the following algorithm E_k are the functions $\{g_i\}$ and f, the initial hyper-rectangle such that $S \subset [a, b]$, and the design parameters ϵ and l. The output is the estimated feasible point, x.

Algorithm E_k :

`input:` $\epsilon \in \mathbb{R}^+$, $l \in \mathbb{N}$, $a, b \in \mathbb{R}^n$, $f, g_1, \ldots, g_s \in \mathbb{R}[x]$.

`output:` $x \in \mathbb{R}^n$ (as an ϵ accurate solution to GPO Problem (6.4)).

`Initialize:`

$Z_0 = [a, b]$; $\Lambda(0) \leftarrow B_k(a(0), b(0))$; $m \leftarrow 0$;

`While` $(m < l) : \{$

$$a^*, b^* \leftarrow \arg\min_{[a,b] \in Z} \prod_{i=1}^{n} b_i - a_i;$$

$$r^* \leftarrow \arg\max_{j \in \{1,\ldots,n\}} (b_j^* - a_j^*); \quad a^{(1)} \leftarrow a^*; \quad b^{(2)} \leftarrow b^*;$$

`For` r `from 1 to` n :

$$b_r^{(1)} \leftarrow \begin{cases} \frac{b_r^* + a_r^*}{2} & \text{if } r = r^* \\ b_r^* & \text{otherwise} \end{cases}; \quad a_r^{(2)} \leftarrow \begin{cases} \frac{b_r^* + a_r^*}{2} & \text{if } r = r^* \\ a_r^* & \text{otherwise,} \end{cases};$$

Compute: $\lambda_1 \leftarrow B_k(a^{(1)}, b^{(1)}); \quad \lambda_2 \leftarrow B_k(a^{(2)}, b^{(2)}); \quad p_1 \leftarrow G(a^{(1)}, b^{(1)});$

$p_2 \leftarrow G(a^{(2)}, b^{(2)}); \quad m \leftarrow m + 1;$

If: $\lambda_i \geq f(p_i) - \epsilon$ then add p_i to P;

Else: Add $[a^{(i)}, b^{(i)}]$ to Z and λ_i to Λ and $\dfrac{a^{(i)} + b^{(i)}}{2}$ to M;

Discard: $Z_j,\ M_j$ and $\Lambda(j)$ where $\Lambda(j) > \min\limits_{p \in P} f(p)$;

`Return` $x := \arg\min_{p \in P \cup M} f(p)$;

If the SQP algorithm fails to find a locally optimal solution within ϵ of the GLB, the E_k algorithm can still return a point with guaranteed accuracy. In the following section we will discuss the complexity and accuracy of the sequence of Algorithms E_k.

6.4.3 Convergence and Complexity of E_k

In this subsection, we first show that for any $k \in \mathbb{N}$, the greatest lower bounds obtained by the subroutine B_k increase at each iteration of the Branch and Bound loop. Next we show that for any desired accuracy, there exists a sufficiently large k, such that Algorithm E_k returns a proposed solution with that accuracy. Since there are no guarantees on the local solution found using Sequential Quadratic Programming subroutine we assume the algorithm continues for at least l iterations and provide bounds on the solution returned by E_k in the worst case scenario where the SQP algorithm does not return any feasible points.

The following lemma gives an algebraic property of the polynomials of the form $w(x) = (x - a_i)(b_i - x)$ which are used to define the augmented feasible set S_{ab}.

Lemma 36. *Let $a \leq c < d \leq b \in \mathbb{R}$, $g := (x-a)(b-x)$ and $h := (x-c)(d-x)$. Then, there exist α, β and $\gamma \in \mathbb{R}$, such that*

$$g(x) = \alpha h(x) + \beta(x+\gamma)^2\,, \quad \alpha, \beta \geq 0$$

Proof. Without loss of generality, one can assume that $a = 0$ (consider the change of variable $z := x - a$). Now let $p^2 := c$, $q^2 := d - c$, and $r^2 := b - d$. First, we consider the case where $p^2, r^2 \neq 0$. This leads to two sub-cases:

Case 1 : $r^2 \neq p^2$. Let

$$\gamma = \frac{p^4 + p^2q^2 - \sqrt{p^2r^2(p^2+q^2)(q^2+r^2)}}{r^2 - p^2}\,,\ \beta = \frac{p^4 + p^2q^2}{\gamma^2 - p^4 - p^2q^2}\,,$$

and $\alpha = \beta + 1$. Verifying the equality $g(x) = \alpha h(x) + \beta(x+\gamma)^2$ is straightforward. To show that $\beta, \alpha \geq 0$, we use the following.

$$\beta \geq 0 \iff \gamma^2 > p^4 + p^2q^2$$

$$\iff \left(p^4 + p^2q^2 - \sqrt{p^2r^2(p^2+q^2)(q^2+r^2)}\right)^2 > (p^4 + p^2q^2)(r^2 - p^2)^2$$

$$\iff \underbrace{(p^4 + p^2q^2)^2 + p^2r^2(p^2+q^2)(q^2+r^2) - (p^4 + p^2q^2)(r^2 - p^2)^2}_{L} >$$

$$\underbrace{2(p^4 + p^2q^2)\sqrt{p^2r^2(p^2+q^2)(q^2+r^2)}}_{U} \iff \begin{cases} L > 0 \\ L^2 > U^2 \end{cases}$$

After simplification we have:

$$L^2 - U^2 = p^4q^4(p^2+q^2)^2(p^2-r^2)^2 > 0, \quad \text{and}$$

$$L = p^2\ \left(p^2+q^2\right)\ \left(p^2\,q^2 + 2\,p^2\,r^2 + q^2\,r^2\right) > 0$$

which completes the proof for Case 1.

Case 2 : $r^2 = p^2$. In this case, let

$$\gamma = -\frac{2p^2+q^2}{2}\,,\ \beta = \frac{4p^2(p^2+q^2)}{q^4}\,,\ \alpha = \beta + 1$$

Equality and positivity for this case can then be easily verified. Now, suppose $r^2 = p^2 = 0$. In this case, simply set $\beta = 0$, $\alpha = 1$. If $p^2 = 0, r^2 \neq 0$, set $\beta = \frac{b}{d} - 1$, $\alpha = \frac{b}{d}$, $\gamma = 0$. The case $p^2 \neq 0$, $r^2 = 0$ is similar to $p^2 = 0$, $r^2 \neq 0$,through the change of variable $z = b - x$. □

In the following lemma, we use Lemma 36 to show that for any $a_1,\ a_2,\ b_1,\ b_2 \in \mathbb{R}^n$ such that $a_1 \leq a_2 < b_2 \leq b_1 \in \mathbb{R}^n$, the feasible set of the SOS problem solved in Subroutine $B_k(a_1, b_1)$ is contained in that of Subroutine $B_k(a_2, b_2)$.

Lemma 37. *For any $k \in \mathbb{N}$ and $a \leq b \in \mathbb{R}^n$, let $M^{(k)}_{a\,b}$ be the modified degree-k bounded quadratic module associated to polynomials $g_1, \ldots, g_s$, as defined in* (6.13)*. If $\gamma \leq \alpha < \beta \leq \delta \in \mathbb{R}^n$, then $M^{(k)}_{\gamma\,\delta} \subset M^{(k)}_{\alpha\,\beta}$, for all $k \geq 2$.*

Proof. For any $j = 1, \ldots, n$, let $w_{j,1}(x) := (\beta_j - x_j)(x_j - \alpha_j)$, and $w_{j,2}(x) := (\delta_j - x_j)(x_j - \gamma_j)$. Since $\gamma_j \leq \alpha_j < \beta_j \leq \delta_j$, then it is followed from Lemma 36 that there exist $p_j, q_j, r_j \in \mathbb{R}$ such that

$$w_{j,2}(x) = p_j^2\ w_{j,1}(x)\ +\ q_j^2\ (x_j + r_j)^2.$$

Now, if $h \in M^{(k)}_{\gamma\,\delta}$, we will show that $h \in M^{(k)}_{\alpha\,\beta}$. By definition, there exist $\sigma_i, \omega_{j,2} \in \Sigma_S$ such that $h = \sum_{i=0}^{s} \sigma_i g_i + \sum_{j=1}^{n} \omega_j w_{j,2}$, where $g_0(x) = 1$. Hence, we can plug in the expression for $w_{j,2}$ to get

$$\begin{aligned} h &= \sum_{i=0}^{s} \sigma_i \cdot g_i + \sum_{j=1}^{n} \omega_j \cdot (p_j^2 \cdot w_{j,1} + q_j^2 \cdot (x_j + r_j)^2) \\ &= \underbrace{(\sigma_0 + \sum_{j=1}^{n} q_j^2 \cdot \omega_j \cdot (x_j + r_j)^2)}_{\sigma_{0\text{ new}}} + \sum_{i=1}^{s} \sigma_i \cdot g_i + \sum_{j=1}^{n} \underbrace{p_j^2\, \omega_j}_{\omega_{j\text{ new}}} \cdot w_{j,1}. \end{aligned}$$

Clearly $\sigma_{0\text{ new}}, \omega_{j\text{ new}} \in \Sigma_S$. Furthermore, since $k \geq 2$, $\deg(\sigma_{0\text{ new}}) \leq k$, and $\deg(\omega_{j\text{ new}} \cdot w_{j,1}) \leq k$ which implies that $h \in M^{(k)}_{\alpha\,\beta}$. □

Now suppose $\{g_i\}$ all have degree d or less. Then for any $k \geq d+2$ and for any hyper-rectangles $C(c,d) \subset C(a,b)$, if $\lambda_{(a,b)}$ and $\lambda_{(c,d)}$ are the solutions obtained by Subroutines $B_k(a,b)$ and $B_k(c,d)$ applied to GPO Problem (6.4), then Lemma 37 shows that $\lambda_{(a,b)} \leq \lambda_{(c,d)}$.

Now, for a fixed $k \in \mathbb{N}$, let ϵ and l be the design parameters of Algorithm E_k applied to GPO Problem (6.1). For $m = 0, \ldots, l$, let $(\lambda^*)_m := \lambda(j^*)$, where j^* is as we defined in iteration m of the loop in Algorithm E_k. Using Lemma 37, it is straightforward to show that $(\lambda^*)_m \leq (\lambda^*)_{m+1}$ for $m \leq l-1$.

In the next theorem, we will show that for any given $\epsilon > 0$, there exist $k \in \mathbb{N}$ such that Algorithm E_k applied to GPO Problem (6.4) will provide a point $x \in \mathbb{R}^n$ satisfying (6.5).

Theorem 38. *Suppose GPO Problem* (6.4) *has a nonempty and compact feasible set* S. *Choose* $a, b \in \mathbb{R}^n$ *such that* $S \subset C(a,b)$. *For any desired accuracy,* $0 < \epsilon < 1$, *let* $l > n \log_2(\frac{L\sqrt{n}}{\epsilon})$ *where* $L = \max_i b_i - a_i$. *Then there exists a* $k \in \mathbb{N}$ *such that if* $x = E_k(\epsilon, l, a, b, f, g_i)$, *then there exists a feasible point* $y \in S$ *such that* $f(y) - f^* \leq \epsilon$ *and* $\|y - x\| < \epsilon$, *where* f^* *is the objective value of the GPO Problem* (6.4).

Proof. Define $\mathcal{P}$ to be the set of all possible hyper-rectangles generated by the branching loop of Algorithm E_k (for any k) with number of branches bounded by l. The vertices of all elements of $\mathcal{P}$ clearly lie on a grid with spacings $\frac{|a_i,b_i|}{2^l}$. Therefore, the cardinality $|\mathcal{P}|$ is finite and bounded as a function of l, a and b. It has been shown that for any $C_\alpha := C(e,f) \in \mathcal{P}$, there exists a $k_\alpha \in \mathbb{N}$ such that for any $k' \geq k_\alpha$, the solution of Subroutine $B_{k'}(e,f)$ is accurate with the error tolerance ϵ. Now define $k := \max\{k_\alpha \mid C_\alpha \in \mathcal{P}\}$.

We will now show that Algorithm E_k returns a point x with the desired accuracy. First, we show that Algorithm E_k generates exactly l nested hyper-rectangles. The

proof is by induction on m.

For $m = 0, \ldots, l-1$, let $(a)_m := a(j^*)$, $(b)_m := b(j^*)$, $(C)_m := C((a)_m, (b)_m)$, $(\lambda)_m := B_k((a)_m, (b)_m)$, $(a^{(1)})_m := a^{(1)}$, $(b^{(1)})_m := b^{(1)}$, $(a^{(2)})_m := a^{(2)}$, $(b^{(2)})_m := b^{(2)}$, $(C^{(1)})_m := C((a^{(1)})_m, (b^{(1)})_m)$, $(C^{(2)})_m := C((a^{(2)})_m, (b^{(2)})_m)$ and $(\lambda^*)_m := \lambda(j^*)$ where j^*, $\tilde{a}$, $\tilde{b}$, $\hat{a}$ and $\hat{b}$ are defined as in iteration m of Algorithm E_k.

We use induction on m to show that for all $m \leq l$:

$$(C)_m \subset (C)_{m-1}.$$

The base case $m = 0$ is trivial. For the inductive step, first note that $(\lambda^*)_m \leq f^*$ for all $m \leq l$ and $(\lambda^*)_1 \leq \cdots \leq (\lambda^*)_l$. The latter is obtained from Lemma 37 and the former is because at each iteration, $S \subset \bigcup_{i=0}^{m} C(a(i), b(i))$.

The assumption that the solution of Subroutine $B_{k'}(e, f)$ is accurate within ϵ, implies that

$$B_k((a)_m, (b)_m) \leq (\lambda^*)_m + \epsilon. \tag{6.17}$$

Now we will show that again at iteration $m+1$ one of the hyper-rectangles $(C^{(1)})_m$ and $(C^{(2)})_m$ must also satisfy Eqn. 6.17. Suppose this is not true. Then we can write

$$\begin{aligned} (\lambda^*)_{m+1} &< B_k((\tilde{a})_m, (\tilde{b})_m) - \epsilon, \\ (\lambda^*)_{m+1} &< B_k((\hat{a})_m, (\hat{b})_m) - \epsilon. \end{aligned} \tag{6.18}$$

Now, since $(\lambda^*)_{m+1} \geq (\lambda^*)_m$, Eq. (6.18) implies

$$\begin{aligned} (\lambda^*)_m &< B_k((\tilde{a})_m, (\tilde{b})_m) - \epsilon, \\ (\lambda^*)_m &< B_k((\hat{a})_m, (\hat{b})_m) - \epsilon. \end{aligned} \tag{6.19}$$

Using Eq. (6.19) and Eq. (6.17) one can write

$$\begin{aligned} B_k((a)_m, (b)_m) &< B_k((\tilde{b})_m, (\tilde{b})_m) - \epsilon, \\ B_k((a)_m, (b)_m) &< B_k((\hat{a})_m, (\hat{b})_m) - \epsilon. \end{aligned} \tag{6.20}$$

This contradicts the fact that all $B_k((\tilde{a})_m, (\tilde{b})_m)$, $B_k((\hat{a})_m, (\hat{b})_m)$ and $B_k((a)_m, (b)_m)$ have accuracy higher than ϵ. Therefore, it is clear that either $(C^{(1)})_m$ or $(C^{(2)})_m$ will be bisected at iteration $m+1$. This fact, together with the induction hypothesis which certifies that $(C)_m$ possesses the smallest volume between all the hyper-rectangles obtained up to that iteration, guaranteeing that either $(C^{(1)})_m$ or $(C^{(2)})_m$, will be branched at the next iteration. Therefore, the algorithm will generate l nested hyper-rectangles. Now, $(\lambda^*)_0 \in [f^* - \epsilon, f^*]$ implies that

$$(\lambda^*)_m \in [f^* - \epsilon, f^*], \quad \text{for all } m = 1, \ldots, l. \tag{6.21}$$

Eq. (6.17) and Eq.(6.21) together with the fact that $(\lambda)_m \geq (\lambda^*)_m$ imply

$$(\lambda)_m \in [f^* - \epsilon, f^* + \epsilon], \ \forall\, m = 1, \ldots, l. \tag{6.22}$$

Finally, as a special case $m = l$, one can write:

$$f^* - \epsilon \leq B_k((a)_l, (b)_l) \leq f^* + \epsilon.$$

Now, note that the ϵ-accuracy of $B_k((a)_l, (b)_l)$ implies that $(C)_l$ is feasible. It also can be implied that $(C)_l \cap S$ contains y such that $f(y) \geq B_k((a)_l, (b)_l) \geq f(y) - \epsilon$. Therefore, $|f(y) - f^*| \leq \epsilon$.

Finally, if x is the point return by Algorithm E_k, then based on the definition of l, it is implied that after the last iteration $m = l - 1$, the largest diagonal of the branched hyper-rectangle is less than ϵ, hence $\|y - x\|_2 \leq \epsilon$, as desired. □

Theorem 38 ensures that for any accuracy $\epsilon > 0$ there exists a $k \in \mathbb{N}$ such that Algorithm E_k returns ϵ-approximate solutions to the GPO problem with a logarithmic bound on the number of branching loops. The following corollary shows that these ϵ-approximate solutions can themselves approximately satisfy the constraints of the original GPO as follows.

Corollary 39. *Let GPO Problem* (6.4) *have nonempty and compact feasible set that is contained in* $C(a,b)$ *for some* $a,b \in \mathbb{R}^n$. *For any given* $\delta > 0$, *there exists* $\epsilon > 0$ *such that if* ϵ *and* $x = E(\epsilon, l, a, b, f, g_i)$ *satisfy the conditions in Theorem 38, then*

$$|f(x) - f^*| \leq \delta \text{ and } \quad g_i(x) \geq -\delta, \ \forall i = 1, \ldots, s. \tag{6.23}$$

Proof. Let L be such that any polynomial $h \in \{f, g_1, \ldots, g_s\}$ satisfies $|h(c) - h(d)| \leq L|c - d|_2, \ \forall c, d \in C(a,b)$. (Existence of L follows from the Lipschitz continuity of polynomials on compact sets.) Choose ϵ such that $\epsilon \leq \delta/L$. Let ϵ and x satisfy the conditions in Thoerem 38. It is straightforward to show that x satisfies Eq. (6.23). □

Unfortunately, of course, Theorem 38 does not provide a bound on the size of k (although the proof implies an exponential bound). Fortunately, the SQP step of algorithm E_k often finds the optimal point after just a few iterations of the algorithm, implying that even if k is not large enough, an ϵ-approximate solution can still be found.

Table 6.1: A comparison of starting conditions of interior point method and the resulting number of suboptimal (OBV $< f(\hat{x_2})$) or failed solutions over 50 trials.

Max Perturbation	Suboptimal	No Solution	Trials
± 1	8	1	50
± 2	16	0	50
± 5	24	2	50
± 10	36	2	50

6.5 Numerical Results

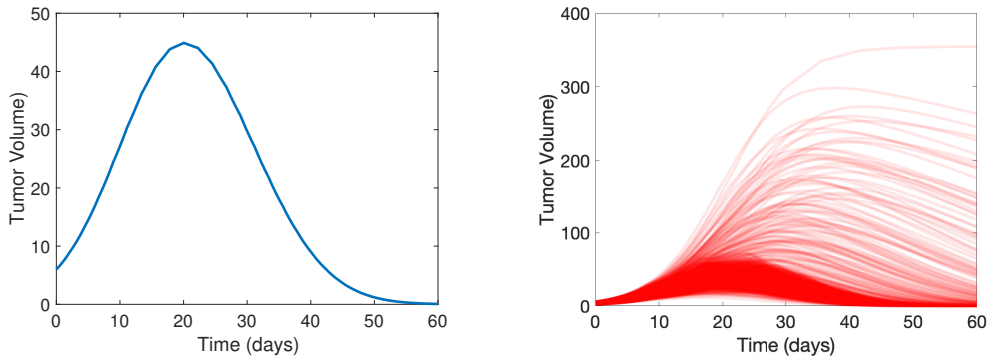
Here we consider two applications of the GPO algorithm proposed in this chapter. In the first example we illustrate the importance of using the GPO method over local solvers. In the second example we analyze the region of attraction model for a pulsed immunotherapy cancer treatment that was generated in Chapter 5 to select an optimal treatment strategy.

Example 1: Consider the following GPO problem.

$$\begin{aligned}
\underset{x\in\mathbb{R}^6}{min} \quad & f(x) = 7x_1x_5^3 + 6x_1x_5^2x_6 + 9x_2x_4^3 + 4x_2x_4x_5 + \\
& \qquad 3x_2x_5x_6 + x_3x_4x_5 \\
\text{subject to} \quad & g_1(x) = 100 - (x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2) \geq 0 \\
& g_2(x) = x_1^3 + x_2^2x_4 + x_3x_5^2 \geq 0 \\
& g_3(x) = x_2^2x_1 + x_5^3 + x_4x_1x_2 \geq 0 \\
& h_1(x) = x_1 + x_2^2 - x_3^2 + x_4x_5 = 0 \\
& h_2(x) = x_5x_1 - x_4^2 = 0.
\end{aligned}$$

In this example we have 6 variables, an objective function of degree 4 and several equality and inequality constraints of degree 4 or less. Polynomial optimization problems similar to this example are important in economic dispatch models [122] or optimal power flow [64] and since the ideal generated by equality constraints in this case are not zero dimensional, the Moment approach to extracting solutions fails. We applied Algorithm E_5 to this problem with parameters $\epsilon = 0.005$ and $l = 200$, using Sedumi to solve the SDPs associated with the SOS and Moment problems.

The result is the point $\hat{x} = [5.1274,\ 3.9372,\ 0.8043,\ -4.6793,\ 4.2704,\ -4.1748]$, where all inequalities are feasible and the equality constraints h_1 and h_2 have errors



(a) The simulated tumor volume of the example problem using the optimal pulsed immunotherapy treatment generated by solving Optimization Problem (6.24).

(b) The simulated tumor volume of 1000 randomized initial conditions using the optimal pulsed immunotherapy treatment generated by solving Optimization Problem (6.24) for each problem set of initial conditions.

Figure 6.2: Subfigures (a) and (b) plot simulations of tumor dynamics using the optimal pulsed immunotherapy treatments found using the GPO algorithm.

of 1.015×10^{-9} and 0.019×10^{-9} respectively. Furthermore, the objective value is $f(\hat{x}) = -3719$ which closely tracks the GLB value of -3718.94.

To illustrate the importance of using this branch and bound technique over running the SQP algorithm with random initial guesses, we ran several batteries of tests, successively decreasing the accuracy of the initial guess. Using $\hat{x}$ as the centroid, we proposed 50 randomly distributed initial guesses within a radius of $1, 2, 5$ and 10. These results are listed in Table 6.1 and indicate the increasing number of initial guesses which resulted in either local optima or no feasible solution - ultimately at 76% for maximum radius 10.

Example 2: In this example we use the GPO solver to identify optimal treatments using the ROA model of tumor-immune dynamics from Chapter 5. The system dy-

namics are detailed in Eq. (1.7) in Chapter 1.

These dynamics define the relationship between tumor size T, TGF-β B, effector immune cells E, regulatory immune cells R and activated tumor-specific cytotoxic T-cells administered as a vaccine V. In Chapter 5 we found an optimal data-driven model of the region of attraction for a given dosage d_v and number of doses n_v defined as the following semi-algebraic set,

$$S_1 := \{[T, B, E, R] \in \mathbb{R}^4 \mid p(T, B, E, R, d_v, n_v) \leq 7.4390\}.$$

To demonstrate how the combination of the ROA model and the GPO solution can identify optimal data-driven controllers for pulsed immunotherapy treatment we simulate a test case. Assume we have a patient with nominal system parameters and the following initial conditions, $T(0) = 5$, $B(0) = 0.1$, $E(0) = 100$, $R(0) = 50$. Using these initial conditions we can define a semialgebraic set of all treatment parameters, d_v and n_v which are predicted to eliminate the tumor in 60 days. This set is defined as,

$$R_1 := \{[d_v, n_v] \in \mathbb{R}^2 \mid 3.9052 \cdot 10^{-15} d_v^4 + 1.8477 \cdot 10^{-12} d_v^3 n_v + 7.3554 \cdot 10^{-10} d_v^2 n_v^2 + 2.8338 \cdot 10^{-7} d_v n_v^3 + 1.1894 \cdot 10^{-4} n_v^4 - 1.5312 \cdot 10^{-10} d_v^3 - 7.3767 \cdot 10^{-8} d_v^2 n_v - 2.9969 \cdot 10^{-5} d_v n_v^2 - 1.1793 \cdot 10^{-2} n_v^3 + 2.4715 \cdot 10^{-6} d_v^2 + 1.1979 \cdot 10^{-3} d_v n_v + 0.4704 n_v^2 - 0.020609 d_v - 9.3673 n_v + 92.1062 \leq 7.4390\}$$

and assuming the model is accurate we may solve the following optimization problem to identify an optimal treatment strategy for this patient.

$$\begin{aligned} \underset{x \in \mathbb{R}^2}{min} \quad & d_v n_v \\ \text{subject to} \quad & [d_v, n_v] \in R_1. \end{aligned} \tag{6.24}$$

We determined the optimal data-driven pulsed immunotherapy treatment of approximately 2000 cells injected at a period of every 2.25 days. After simulation it was

confirmed that this pulsed immunotherapy treatment does successfully eliminate the tumor cells within 60 days. We show the population of tumor cells throughout time for this nominal case in Fig. 6.2(a).

Since this data-driven modeling of the ROA is not exact we next test the robustness of the optimal solutions. To test the robustness of the optimal treatments we randomly generate 1,000 different patients whose dynamic constants vary within $\pm 10\%$ and whose initial conditions are within $T(0) \in [2, 6], B(0) \in [0.05, 0.15], E(0) \in [90, 110], R(0) \in [40, 60]$. We found that 75.00% of the optimal treatments found using Optimization Problem (6.24) eliminated the tumor within 60 day. In Fig. 6.2(b) we plot the volume of tumor cells throughout time after applying the optimal treatment.

To determine how close to optimal the treatments are we decreased the dosage by 10% and found that only 35.69% of the treatments led to tumor elimination within 60 days. Likewise only 34.68% of treatments eliminated the tumor when the period of treatment was increased by 10% of the optimal value. When decreasing the dosage by 10% and increasing the period by 10% only 3.85% of the cases eliminated the tumor within 60 days, implying that over 71% of the treatments eliminated the cancer cells and were within 10% of the optimal dose and period.

6.6 Conclusion

We have proposed a sequence of Algorithms $E_k,\ k \in \mathbb{N}$ to extract solutions to the GPO problem based on a combination of Branch and Bound and SOS/Moment relaxations. The computational-complexity of Algorithm E_k is polynomial in k, polynomial in the number of constraints and linear in the number of branches l. Additionally, for any scalar $\epsilon > 0$, there exist $k \in \mathbb{N}$ such that Algorithms E_k, in $n \log(\frac{r}{\epsilon})$ number of iterations, returns a point that is within the ϵ-distance of a feasible and ϵ-suboptimal point. For a fixed degree of semidefinite relaxations, our numerical case study demon-

strates convergence to an ϵ-suboptimal point returned by the E_k algorithm.

Additionally we were able to use the GPO algorithm in conjunction with a ROA model from Chapter 5 to select optimal pulsed immunotherapy treatments for the treatment of cancer. This method did not require any system models to be known, instead relying on trajectory measurements to build a semialgebraic set used to model the region of attraction, and the GPO algorithm to select an optimal point contained within the model. Applying such a model in practice would require only an initial measurement of the tumor volume, levels of TGF-β, and amounts of effector and regulatory immune cells.

Chapter 7

CONCLUSION

Motivated by three problems to generate better predictive models of the immune system, we formulated convex optimization problems to generate improved data-driven models of physical processes. These methods were applied to determine cellular characteristics that differ most between healthy patients and those with rheumatoid arthritis; identify populations of immune system cells that are most correlated to the severity of rheumatoid arthritis in mice; and select an optimal dosage and period of an immunotherapy treatment for cancer based upon the size of the tumor, initial populations of effector and regulatory cell types, and the amount of a cytokine signal in the tumor environment.

In this chapter we present concluding remarks for each problem.

Problem 1: Generating Models of the Distribution of Measured Data

In Chapter 3 we propose convex optimization problems to select optimal parameter of Sliced Distribution PDFs to model the density of a random variable. Unlike other sets of distributions, SD PDFs are dense in the set of all bounded PDFs in $L_1(\Delta)$. This implies that we can use SDs to model the PDF of a random variable while making only a few minor assumptions on the physical process which generated the data. Additionally, we formulated convex optimization problems to select optimal parameters for the SDs, so the globally optimal member of the degree bounded set of distributions is guaranteed to be found, unlike other sets of PDFs such as Gaussian mixture models.

The performance of SDs is compared to multivariate normals and Guassian mixture models using two metrics to demonstrate that the proposed methods have supe-

rior performance with respect to modeling the distribution of random variables. The first metric is based on the likelihood of the models to generate a test partition of the data, while the second is based on the volume of the level sets of the PDFs that most tightly contain the test partition of the data. In both cases methods using the proposed SDs perform better than the other tested distributions, though this improved performance comes at an increase in computation time.

Finally we show that SDs may be used to model the variation in immune system cell characteristics measured from mice with RA as well as healthy mice. By comparing the models generated for both groups, we identified a set of immune system characteristics that differed most between healthy and diseased mice. These "immune features" could be used to perfectly differentiate between mass cytometry datasets taken from diseased and healthy mice, illustrating that the PDF models captured essential features of the immune system that differ after the onset of RA.

Problem 2: Generating Optimal Machine Learning Algorithms

In Chapter 4 we proposed an efficient kernel learning algorithm which met three criteria to ensure the resulting predictors are robust. Specifically we introduced an efficient TK kernel learning algorithm based on a FW type algorithm which simultaneously selects an optimal TK kernel function and generates a predictor using that kernel function. While the average computation time of the TK kernel learning algorithm is larger than the other methods, the set of TK kernels is tractable, dense, and universal, implying that KL algorithms based on TK kernels are more robust than existing machine learning algorithms. This assertion is supported by numerical testing on 12 moderately sized and randomly selected datasets, which yielded increases in the average accuracy of the TK kernel learning algorithm over other state-of-the-art alternatives.

We also considered the use of the KL learning problem in the identification of three different states of the immune system of increasing complexity. Specifically, we used a set of mouse-model experiments to obtain a robust dataset of T cell markers and populations at the end stage of a proposed immunotherapy treatment. From these experiments we were able to determine that the *CD4+GATA3+CD44+CD62L(Lo)* memory T cell sub-population is a significant population in that it is identified as an essential component of the immune system and an essential component for estimating the disease severity of mice with RA. Other immune system cell populations were also identified that are highly correlated with, for example, estimating the disease severity, or estimating the populations of other immune system cells.

Problem 3: Generating Constrained Predictive Models and Identifying Optimal Treatments

In Chapter 5 we solve the problem of generating predictive models that are constrained to be, for example, globally positive. We formulate a convex method to optimize the parameters of a polynomial function to fit given data with respect to the least squares or least absolute deviations metrics and constraint the function to be either globally positive, and thus a sum-of-squares polynomial, or to be positive over a semi-algebraic set.

This method was used to generate predictive models of a converse Lyapunov function from trajectory data taken from a dynamical system. This dynamical system models the relationship between tumor cells, immune system cells, and cytokines for a given immunotherapy treatment. We assume that variability in the dynamical systems for different patients has a minor effect on the ROA, and that different patient trajectories can be combined to model the average Lyapunov function of a group of patients. Specifically, given trajectories from patients given pulsed immunotherapy

treatments of different periods and dosages, we fit a function that generates a prediction of the region of attraction of the system for a given period and dosage of therapy.

Given the initial conditions of a patients immune system, the model can also be used to predict if a given dosage and period of treatment will lead to elimination of the tumor. Fortunately, for a set of initial conditions, the model of the dosage and period of treatment that leads to complete tumor elimination in 60 days is a semialgebraic set. Therefore, given the initial conditions of a patients immune system, we may solve a global polynomial optimization problem to select a treatment that is predicted to eliminate the tumor within 60 days while minimizing the total dosage of treatment.

Next in Chapter 6 we proposed a sequence of Algorithms E_k, $k \in \mathbb{N}$ to extract solutions to GPO problems based on a combination of Branch and Bound and SOS/Moment relaxations. The computational-complexity of Algorithm E_k is polynomial in k, polynomial in the number of constraints and linear in the number of branches l. Additionally, for any scalar $\epsilon > 0$, there exist a $k \in \mathbb{N}$ such that Algorithms E_k, in $O(\log(1/\epsilon))$ number of iterations, returns a point that is within the ϵ-distance of a feasible and ϵ-suboptimal point. For a fixed degree of semidefinite relaxations, our numerical case study demonstrated convergence to an ϵ-suboptimal point returned by the E_k algorithm.

This GPO algorithm was applied to solve the GPO problem formulated in Chapter 5 to select an optimal dosage and period of a pulsed immunotherapy treatment to eliminate the tumor cells. This method thus relies only on trajectory measurements to generate a semialgebraic set that estimates the region of attraction. This region of attraction estimate is then used to generate a GPO problem which is solved with the GPO algorithm to select an optimal dosage and treatment period. The simulations

of the immunotherapy model show that over 84% of the treatments returned by the GPO algorithm are within 10% of the optimal treatment strategy, implying that even though the dynamical systems of the patient differed slightly, the ROA estimate was still able to find effective treatments for a majority of patients.

REFERENCES

[1] A. Ahmadi, M. Krstic, and P. Parrilo. A globally asymptotically stable polynomial vector field with no polynomial lyapunov function. In *Decision and Control and European Control Conference*, pages 7579–7580, 2011.

[2] F. Alizadeh, J. Haeberly, and M. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 8(3), 1998.

[3] F. Alizadeh, J.-P. Haeberly, and M. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 8(3):746–768, 1998.

[4] M. ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*, 2015.

[5] J. Arciero, T. Jackson, and D. Kirschner. A mathematical model of tumor-immune evasion and sirna treatment. *Discrete & Continuous Dynamical Systems-B*, 4(1):39, 2004.

[6] D. C. Bailey. Not normal: the uncertainties of scientific measurements. *Royal Society open science*, 4, 2017.

[7] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE transactions on neural networks*, 5(4), 1994.

[8] J. Becker, D. Sandwell, W. Smith, J. Braud, B. Binder, J. Depner, D. Fabre, J. Factor, S. Ingalls, S. Kim, et al. Global bathymetry and elevation data at 30 arc seconds resolution: Srtm30_plus. *Marine Geodesy*, 32(4):355–371, 2009.

[9] F. Berkenkamp, R. Moriconi, A. Schoellig, and A. Krause. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In *Proceedings on Decision and Control*, 2016.

[10] D. Bertsekas. *Nonlinear programming*. Athena Scientific, 2 edition, 1998.

[11] C. M. Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[12] R. Bono, M. J. Blanca, J. Arnau, and J. Gómez-Benito. Non-normal distributions commonly used in health, education, and social sciences: a systematic review. *Frontiers in psychology*, 8, 2017.

[13] K. M. Borgwardt, A. Gretton, M. J. Rasch, H. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 2006.

[14] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, 1992.

[15] M. Branicky. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on Automatic Control*, 43(4), 1998.

[16] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2004.

[17] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone. Classification and regression trees. *Biometrics*, 40(3), 1984.

[18] P. Brodin and M. Davis. Human immune system variation. *Nature reviews immunology*, 17(1):21–29, 2017.

[19] A. Chakraborty, P. Seiler, and G. Balas. Nonlinear region of attraction analysis for flight control verification and validation. *Control Engineering Practice*, 19 (4):335–345, 2011.

[20] G. Chandrashekar and F. Sahin. A survey on feature selection methods. *Computers and Electrical Engineering*, 40(1), 2014.

[21] C. Chang, Y.-C. Chen, H.-M. Chen, N.-S. Yang, and W.-C. Yang. Natural cures for type 1 diabetes: a review of phytochemicals, biological actions, and clinical potential. *Current Medicinal Chemistry*, 20(7):899–907, 2013.

[22] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

[23] S. Chen and S. Billings. Representations of non-linear systems: the narmax model. *International journal of control*, 49(3), 1989.

[24] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[25] G. Chesi. Lmi techniques for optimization over polynomials in control: a survey. *IEEE Transactions on Automatic Control*, 2010.

[26] H.-D. Chiang, M. Hirsch, and F. Wu. Stability regions of nonlinear autonomous dynamical systems. *IEEE Transactions on Automatic Control*, 33(1):16–27, 1988.

[27] B. Colbert and M. Peet. Using trajectory measurements to estimate the region of attraction of nonlinear systems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2341–2347, 2018.

[28] B. Colbert and M. Peet. Using sdp to parameterize universal kernel functions. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4622–4629, 2019.

[29] B. Colbert and M. Peet. A convex parametrization of a new class of universal kernel functions. *Journal of Machine Learning Research*, 21(45):1–29, 2020. URL `http://jmlr.org/papers/v21/19-594.html`.

[30] B. Colbert and M. Peet. A new algorithm for tessellated kernel learning, 2020.

[31] B. Colbert, H. Mohammadi, and M. Peet. Combining sos with branch and bound to isolate global solutions of polynomial optimization problems. In *2018 Annual American Control Conference (ACC)*, pages 2190–2197, 2018.

[32] B. Colbert, L. Crespo, and M. Peet. A sum of squares optimization approach to uncertainty quantification. In *2019 American Control Conference (ACC)*, pages 5378–5384, 2019.

[33] B. Colbert, L. Crespo, and M. Peet. A convex optimization approach to improving suboptimal hyperparameters of sliced normal distributions. In *2020 American Control Conference (ACC)*, pages 4478–4483, 2020.

[34] M. Collins and N. Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems*, volume 14, 2002.

[35] W. Cook, T. Koch, D. E. Steffy, and K. Wolter. A hybrid branch-and-bound approach for exact rational mixed-integer programming. *Mathematical Programming Computation*, 5(3), 2013.

[36] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3), 1995.

[37] C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In *Advances in Neural Information Processing Systems*, volume 22, 2009.

[38] C. Cortes, M. Mohri, and A. Rostamizadeh. Two-stage learning kernel algorithms. *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[39] C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13(28): 795–828, 2012.

[40] L. Crespo, B. Colbert, T. Slagel, and S. Kenny. Robust estimation of sliced-exponential distributions. In *Proceedings of the IEEE Conference on Decision and Control*, 2021.

[41] L. G. Crespo, B. K. Colbert, S. P. Kenny, and D. P. Giesy. On the quantification of aleatory and epistemic uncertainty using sliced-normal distributions. *Systems & Control Letters*, 2019.

[42] M. Dash and H. Liu. Feature selection for classification. *Intelligent data analysis*, 1, 1997.

[43] B. DeVolder, J. Glimm, J. Grove, Y. Kang, Y. Lee, K. Pao, D. Sharp, and K. Ye. Uncertainty quantification for multiscale simulations. *J. Fluids Eng.*, 124(1), 2002.

[44] A. C. Doherty, P. A. Parrilo, and F. M. Spedalieri. Complete family of separability criteria. *Physical Review A*, 2004.

[45] M. Doherty and M. Robertson. Some early trends in immunology. *TRENDS in Immunology*, 25(12):623–631, 2004.

[46] D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

[47] G.-R. Duan and H.-H. Yu. *LMIs in control systems: analysis, design and applications*. CRC Press, 1 edition, 2013.

[48] J. Egozcue, J. Díaz-Barrero, and V. Pawlowsky-Glahn. Hilbert space of probability density functions based on aitchison geometry. *Acta Mathematica Sinica*, 22(4):1175–1182, 2006.

[49] H. El-Gabalawy and P. Lipsky. Why do we not have a cure for rheumatoid arthritis? *Arthritis Research & Therapy*, 4(3):1–5, 2002.

[50] H. El-Samad, S. Prajna, A. Papachristodoulou, M. Khammash, and J. Doyle. Model validation and robust stability analysis of the bacterial heat shock response using sostools. *IEEE Conference on Decision and Control*, 2003.

[51] S. R. Eliason. *Maximum likelihood estimation: Logic and practice*. Sage, 1993.

[52] E. Eskin, J. Weston, W. Noble, and C. Leslie. Mismatch string kernels for svm protein classification. In *Advances in Neural Information Processing Systems*, volume 15, 2003.

[53] K. Fan. Minimax theorems. *Proceedings of the National Academy of Sciences of the United States of America*, 39(1):42, 1953.

[54] Y. Fang, K. Loparo, and X. Feng. Inequalities for the trace of matrix product. *IEEE Transactions on Automatic Control*, 39(12):2489–2490, 1994.

[55] R. Fisher. Contributions to mathematical statistics. *Biometrika*, 1950.

[56] M. F. Flajnik and M. Kasahara. Origin and evolution of the adaptive immune system: genetic events and selective pressures. *Nature Reviews Genetics*, 11(1): 47–59, 2010.

[57] G. B. Folland. *Real analysis: modern techniques and their applications*, volume 40. John Wiley & Sons, 1999.

[58] K. Gai, G. Chen, and C.-S. Zhang. Learning kernels with radiuses of minimum enclosing balls. In *Advances in Neural Information Processing Systems*, volume 23, 2010.

[59] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning Theory and Kernel Machines*, pages 129–143. Springer, 2003.

[60] M. Gasca and T. Sauer. On the history of multivariate polynomial interpolation. In *Numerical Analysis: Historical Developments in the 20th Century*, pages 135–147. Elsevier, 2001.

[61] M. Gatto and S. Rinaldi. Stability analysis of predator-prey models via the liapunov method. *Bulletin of mathematical biology*, 39(3), 1977.

[62] R. Genesio, M. Tartaglia, and A. Vicino. On the estimation of asymptotic stability regions: State of the art and new proposals. *IEEE Transactions on Automatic Control*, 30(8):747–755, 1985.

[63] J. Gergonne. The application of the method of least squares to the interpolation of sequences. *Historia Mathematica*, 1(4), 1974.

[64] B. Ghaddar, J. Marecek, and M. Mevissen. Optimal power flow as a polynomial optimization problem. *IEEE Transactions on Power Systems*, 31(1), 2016.

[65] M. Gönen and E. Alpaydin. Localized multiple kernel learning. In *Proceedings of the International Conference on Machine learning*, page 352359, 2008.

[66] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12(64):2211–2268, 2011.

[67] L. Gu. Multivariate gaussian distribution. Technical report, CMU, 2008.

[68] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46, 2004.

[69] W. Hahn. The converse of the stability theorems. In *Stability of Motion*. Springer, 1967.

[70] F. Hamidi, H. Jerbi, W. Aggoune, M. Djemai, and M. N. Abdkrim. Enlarging region of attraction via LMI-based approach and Genetic Algorithm. In *Communications, Computing and Control Applications*, 2011.

[71] D. Haussler. Convolution kernels on discrete structures. Technical report, University of California in Santa Cruz, 1999.

[72] E. Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271, 1909.

[73] D. Henrion and J.-B. Lasserre. *Positive Polynomials in Control*, chapter Detecting Global Optimality and Extracting Solutions in GloptiPoly, pages 293–310. Springer, 2005.

[74] F. S. Hodi, S. J. O'Day, D. F. McDermott, R. W. Weber, J. A. Sosman, J. B. Haanen, R. Gonzalez, C. Robert, D. Schadendorf, J. C. Hassel, et al. Improved survival with ipilimumab in patients with metastatic melanoma. *New England Journal of Medicine*, 363(8):711–723, 2010.

[75] T. Höfer, O. Krichevsky, and G. Altan-Bonnet. Competition for il-2 between regulatory and effector t cells to chisel immune responses. *Frontiers in immunology*, 3:268, 2012.

[76] J. Hu, A. Perer, and F. Wang. Data driven analytics for personalized healthcare. In *Healthcare Information Management Systems*. Springer, 2016.

[77] M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th international conference on machine learning*, 2013.

[78] A. Jain, S. Vishwanathan, and M. Varma. SPF-GMKL: generalized multiple kernel learning with a million kernels. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*, pages 750–758, 2012.

[79] L. Jeanbart and M. Swartz. Engineering opportunities in cancer immunotherapy. *Proceedings of the National Academy of Sciences*, 112(47):14467–14472, 2015.

[80] P. Johansen, T. Storni, L. Rettig, Z. Qiu, A. Der-Sarkissian, K. Smith, V. Manolova, K. Lang, G. Senti, B. Müllhaupt, et al. Antigen kinetics determines immune reactivity. *Proceedings of the National Academy of Sciences*, 105(13):5189–5194, 2008.

[81] M. Johansson and A. Rantzer. Computation of piecewise quadratic lyapunov functions for hybrid systems. *IEEE Transactions on Automatic Control*, 43(4), 1998.

[82] M. Jones, H. Mohammadi, and M. Peet. Estimating the region of attraction using polynomial optimization: A converse Lyapunov result. In *Proceedings of the IEEE Conference on Decision and Control*, 2017.

[83] M. Kachuee, M. M. Kiani, H. Mohammadzade, and M. Shabany. Cuff-less high-accuracy calibration-free blood pressure estimation using pulse transit time. In *2015 IEEE international symposium on circuits and systems (ISCAS)*, 2015.

[84] A. Kann and J. P. Weyant. Approaches for performing uncertainty analysis in large-scale energy/economic policy models. *Environmental Modeling & Assessment*, 5, 2000.

[85] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4, 1984.

[86] H. Kaya, P. Tüfekci, and F. Gürgen. Local and global learning methods for predicting power of a combined gas & steam turbine. In *Proceedings of the international conference on emerging trends in computer and electronics engineering*, pages 13–18, 2012.

[87] D. Ke, C. Chung, and Y. Sun. A novel probabilistic optimal power flow model with uncertain wind power generation described by customized gaussian mixture model. *IEEE Transactions on Sustainable Energy*, 2015.

[88] L. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1), 1980.

[89] S. Khansari-Zadeh and A. Billard. Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions. *Robotics and Autonomous Systems*, 62(6), 2014.

[90] H.-J. Kim, B. Verbinnen, X. Tang, L. Lu, and H. Cantor. Inhibition of follicular t-helper cells by cd8+ regulatory t cells is essential for self tolerance. *Nature*, 467(7313):328–332, 2010.

[91] P. Kim, P. Lee, and D. Levy. Emergent group dynamics governed by regulatory cells produce a robust primary t cell response. *Bulletin of mathematical biology*, 72(3):611–644, 2010.

[92] M. Kumar, N. K. Rath, A. Swain, and S. Rath. Feature selection and classification of microarray data using mapreduce based anova and k-nearest neighbor. *Procedia Computer Science*, 54, 2015.

[93] Z. Kvatadze and T. Shervashidze. On the accuracy of kraft upper bound for the l1-distance between gaussian densities in rk. *Georgian Mathematical Journal*, 12, 2005.

[94] O. Laerum and T. Farsund. Clinical application of flow cytometry: a review. *Cytometry: The Journal of the International Society for Analytical Cytology*, 2 (1):1–13, 1981.

[95] B. Lai, T. Cunis, and L. Burlion. Nonlinear trajectory based region of attraction estimation for aircraft dynamics analysis. In *AIAA Scitech 2021 Forum*, 2021.

[96] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 2004.

[97] J. B. Lasserre. A sum of squares approximation of nonnegative polynomials. *SIAM review*, 49(4), 2007.

[98] J. B. Lasserre. *Moments, positive polynomials and their applications*, volume 1. World Scientific, 2009.

[99] M. Laurent. *Emerging Applications of Algebraic Geometry*, chapter Sums of Squares, Moment Matrices and Optimization Over Polynomials. Springer, 2009.

[100] I. Lauriola and F. Aiolli. Mklpy: a python-based framework for multiple kernel learning. *arXiv preprint arXiv:2007.09982*, 2020.

[101] J. Lavaei. Optimal decentralized control problem as a rank-constrained optimization. In *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*, pages 39–45, 2013.

[102] E. Levin, K. Serrano, and D. Devine. Standardization of cd 62p measurement: results of an international comparative study. *Vox sanguinis*, 105(1):38–46, 2013.

[103] Y. Liu. Big data and predictive business analytics. *The Journal of Business Forecasting*, 33(4), 2014.

[104] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2002.

[105] F. E. Lund and T. D. Randall. Effector and regulatory b cells: modulators of cd4+ t cell immunity. *Nature Reviews Immunology*, 10(4):236–247, 2010.

[106] Z.-Q. Luo and S. Zhang. A semidefinite relaxation scheme for multivariate quartic polynomial optimization with quadratic constraints. *SIAM Journal on Optimization*, 20(4), 2010.

[107] J. Mangal, S. Inamdar, Y. Yang, S. Dutta, M. Wankhede, X. Shi, H. Gu, M. Green, K. Rege, M. Curtis, et al. Metabolite releasing polymers control dendritic cell function by modulating their energy metabolism. *Journal of Materials Chemistry B*, 8(24), 2020.

[108] A. Merola, C. Cosentino, and F. Amato. An insight into tumor dormancy equilibrium via the analysis of its domain of attraction. *Biomedical Signal Processing and Control*, 3(3):212–219, 2008.

[109] C. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 2006.

[110] B. Mohamed and G. Antoine. Computation of polytopic invariants for polynomial dynamical systems using linear programming. *Automatica*, 48(12), 2012.

[111] T. S. Motzkin. The arithmetic-geometric inequality. *Inequalities (Proc. Sympos. Wright-Patterson Air Force Base, Ohio, 1965)*, 1967.

[112] K. Murphy and C. Weaver. *Janeway's immunobiology*. Garland science, 2016.

[113] I. Myung. Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology*, 2003.

[114] K. Ni, S. Kumar, and T. Nguyen. Learning the kernel matrix for superresolution. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, pages 441–446, 2006.

[115] J. Nie. Optimality conditions and finite convergence of lasserre's hierarchy. *Mathematical Programming*, 146(1), 2013.

[116] J. Nie and M. Schweighofer. On the complexity of putinar's positivstellensatz. *Journal of Complexity*, 23(1), 2007.

[117] H. Norma, M. Mendy, J. Janean, D. Agnes, B. Brenda, H. Honey, E. Carrie, J. Mary, H. Robin, B. Marsha, L. Peggy, and L. Terri. *RN Adult Medical Surgical Nursing Review Module*. ati Nursing Education, 10 edition, 2016.

[118] N. Noroozi, P. Karimaghaee, F. Safaei, and H. Javadi. Generation of lyapunov functions by neural networks. In *Proceedings of the World Congress on Engineering*, 2008.

[119] M. D. of Transportation. Metro interstate traffic volume data set, 2019. URL `https://archive.ics.uci.edu/ml/datasets/Metro+Interstate+Traffic+Volume`.

[120] C. S. Ong, A. J. Smola, and R. C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 2005.

[121] F. Paillard. Commentary: Immunosuppression mediated by tumor cells: A challenge for immunotherapeutic approaches. *Human gene therapy*, 11(5):657–658, 2000.

[122] J. B. Park, Y. W. Jeong, J. R. Shin, and K. Y. Lee. An improved particle swarm optimization for nonconvex economic dispatch problems. *IEEE Transactions on Power Systems*, 25(1), 2010.

[123] P. A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

[124] A. Patt, R. J. Klein, and A. de la Vega-Leinert. Taking the uncertainty in climate-change vulnerability assessment seriously. *Comptes Rendus Geoscience*, 337, 2005.

[125] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[126] M. Peet, P. Kim, and P. Lee. Biological circuit models of immune regulatory response: A decentralized control system. In *2011 50th IEEE Conference on Decision and Control*, pages 3020–3025, 2011.

[127] M. M. Peet, A. Papachristodoulou, and S. Lall. Positive forms and stability of linear time-delay systems. *SIAM Journal on Control and Optimization*, 2009.

[128] D. B. Percival and P. Guttorp. Long-memory processes, the allan variance and wavelets. In *Wavelet analysis and its applications*, volume 4. Elsevier, 1994.

[129] S. Prajna, P. Antonis, and P. Pablo. Introducing sostools: A general purpose sum of squares programming solver. In *Proceedings of the IEEE Conference on Decision and Control*, 2002.

[130] S. Prajna, A. Papachristodoulou, and P. A. Parrilo. Introducing sostools: A general purpose sum of squares programming solver. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 1, pages 741–746, 2002.

[131] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. SOSTOOLS and its control applications. In *Positive polynomials in control*. Springer, 2005.

[132] S. Qiu and T. Lane. Multiple kernel learning for support vector regression. *Computer Science Department, The University of New Mexico, Albuquerque, NM, USA, Tech. Rep*, 2005.

[133] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[134] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 2008.

[135] B. Recht. *Convex Modeling with Priors*. PhD thesis, Massachusetts Institute of Technology, 2006.

[136] S. Rosenberg, N. Restifo, J. Yang, R. Morgan, and M. Dudley. Adoptive cell transfer: a clinical path to effective cancer immunotherapy. *Nature Reviews Cancer*, 8(4):299–308, 2008.

[137] I. Safran and O. Shamir. Spurious local minima are common in two-layer ReLU neural networks. In *International Conference on Machine Learning*, 2018.

[138] S. Sakaguchi, M. Miyara, C. M. Costantino, and D. A. Hafler. Foxp3+ regulatory t cells in the human immune system. *Nature Reviews Immunology*, 10(7): 490–500, 2010.

[139] A. Scheffold, J. Hühn, and T. Höfer. Regulation of cd4+ cd25+ regulatory t cell activity: it takes (il-) two to tango. *European journal of immunology*, 35 (5):1336–1341, 2005.

[140] B. Schölkopf, A. J. Smola, and F. Bach. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[141] M. Schweighofer. Optimization of polynomials on compact semialgebraic sets. *SIAM Journal on Optimization*, 15(3), 2005.

[142] H. Shapiro. Multistation multiparameter flow cytometry: a critical review and rationale. *Cytometry: The Journal of the International Society for Analytical Cytology*, 3(4):227–243, 1983.

[143] B. Shekhtman. Why piecewise linear functions are dense in C [0, 1]. *Journal of Approximation Theory*, 1982.

[144] H. D. Sherali and C. H. Tuncbile. New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems. *Operations Research Letters*, 21(1), 1997.

[145] N. Z. Shor. Quadratic optimization problems. *Soviet Journal of Computer and Systems Sciences*, 25(6), 1987.

[146] A. M. Silverstein. *A history of immunology*. Academic Press, 2009.

[147] A. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 2004.

[148] F. Song, Z. Guo, and D. Mei. Feature selection using principal component analysis. *2010 International Conference on System Science, Engineering Design and Manufacturing Informatization*, 1, 2010.

[149] S. Ć. Sonnenburg, S. Henschel, C. Widmer, J. Behr, A. Zien, F. d. Bona, A. Binder, C. Gehl, V. Franc, et al. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 2010.

[150] C. Starr, R. Taggart, C. Evers, and L. Starr. *Biology: The unity and diversity of life*. Cengage Learning, 2015.

[151] D. E. Steffy and K. Wolter. Valid linear programming bounds for exact mixed-integer programming. *INFORMS Journal on Computing*, 25(2), 2013.

[152] G. Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2), 1974.

[153] J. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11, 1999.

[154] B. Sturmfels. *Solving systems of polynomial equations*. American Mathematical Society, 2002.

[155] N. Subrahmanya and Y. Shin. Sparse multiple kernel learning for signal processing applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[156] T. Sullivan. *Introduction to uncertainty quantification*. Springer, 2015.

[157] H. Sun. Mercer theorem for RKHS on noncompact sets. *Journal of Complexity*, 2005.

[158] H. Tam, M. Melo, M. Kang, J. Pelet, V. Ruda, M. Foley, J. Hu, S. Kumari, J. Crampton, A. Baldeon, et al. Sustained antigen availability during germinal center initiation enhances antibody responses to vaccination. *Proceedings of the National Academy of Sciences*, 113(43):E6639–E6648, 2016.

[159] H. Tanabe, B. Ho, C. Nguyen, and S. Kawasaki. Simple but effective methods for combining kernels in computational biology. In *2008 IEEE International Conference on Research, Innovation and Vision for the Future in Computing and Communication Technologies*, 2008.

[160] M. Terabe, E. Ambrosino, S. Takaku, J. O'Konek, D. Venzon, S. Lonning, J. McPherson, and J. Berzofsky. Synergistic enhancement of cd8+ t cell–mediated tumor vaccine efficacy by an anti–transforming growth factor-β monoclonal antibody. *Clinical Cancer Research*, 15(21):6560–6569, 2009.

[161] M. Todd. *Minimum-volume ellipsoids: Theory and algorithms*. SIAM, 2016.

[162] M. J. Todd and E. A. Yıldırım. On khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. *Discrete Applied Mathematics*, 155 (13), 2007.

[163] S. L. Topalian, M. Sznol, D. F. McDermott, H. M. Kluger, R. D. Carvajal, W. H. Sharfman, J. R. Brahmer, D. P. Lawrence, M. B. Atkins, J. D. Powderly, et al. Survival, durable tumor remission, and long-term safety in patients with advanced melanoma receiving nivolumab. *Journal of clinical oncology*, 32(10): 1020, 2014.

[164] P. Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014.

[165] Y. Y. Wan and R. A. Flavell. How diverse–CD4 effector t cells and their functions. *Journal of molecular cell biology*, 1(1):20–36, 2009.

[166] H. Wang, Q. Xiao, and D. Zhou. An approximation theory approach to learning with ℓ_1 regularization. *Journal of Approximation Theory*, 2013.

[167] M. Webster and C.-H. Cho. Analysis of variability and correlation in long-term economic growth rates. *Energy economics*, 28, 2006.

[168] M. D. Webster, M. Babiker, M. Mayer, J. M. Reilly, J. Harnisch, R. Hyman, M. C. Sarofim, and C. Wang. Uncertainty in emissions projections for climate models. *Atmospheric environment*, 36, 2002.

[169] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer. Probabilistic trajectory prediction with gaussian mixture models. In *2012 IEEE Intelligent Vehicles Symposium*, 2012.

[170] S. Wilson and D. Levy. A mathematical model of the enhancement of tumor vaccine efficacy by immunotherapy. *Bulletin of mathematical biology*, 74(7): 1485–1500, 2012.

[171] Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu. Simple and efficient multiple kernel learning by group lasso. In *Proceedings of the 27th international conference on machine learning*, pages 1175–1182, 2010.

[172] H. Yang, Z. Xu, J. Ye, I. King, and M. Lyu. Efficient sparse generalized multiple kernel learning. *IEEE Transactions on neural networks*, 22(3):433–446, 2011.

[173] Y. Ye and E. Tse. An extension of Karmarkar's projective algorithm for convex quadratic programming. *Mathematical programming*, 44(1-3):157–179, 1989.

[174] E. Zanaty and A. Afifi. Support vector machines (SVMs) with universal kernels. *Applied Artificial Intelligence*, 2011.

APPENDIX A

APPENDIX

A.1 NUMERICAL SCALABILITY OF SLICED EXPONENTIALS

In this section we consider the computational complexity of the MLE and WCE optimization problems for SE. We first consider the effect of the number of samples in S that are used to calculate the numerical integration constant. Then we consider the effect of the number and dimension of the training data points and the degree of the SE.

Evaluating the Effect of the Sample Number

The number of samples used to calculate the numerical integration constant in Optimization Problems 3.6 (MLE) and 3.16 (WCE) will have an effect on the optimal λ^* and the computation time of the algorithm. We perform numerical experiments using the Iris data set and the MLE and WCE optimization methods to observe the effect of the number of selected samples, S, on the computation time and objective value of the resulting λ^*.

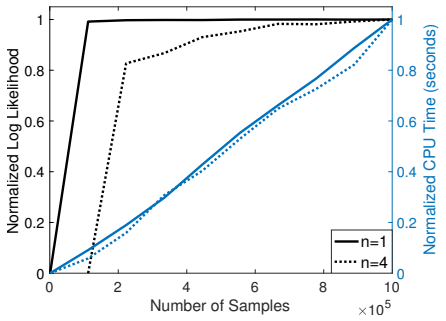
The dimension of the data effects the accuracy of the Monte Carlo integral estimation. To compare the difference between data sets of different dimension we will perform the tests first using only the first dimension of the Iris data set, then compare those results to using the full Iris data set.

The MLE Optimization Problem To study the properties of Optimization Problem (3.6) with respect to the number of samples in S we plot the normalized log likelihood and the normalized computation time of the MLE problem in Fig. A.1(a) averaged over 15 trials. The log likelihood is calculated after the optimal parameter λ^* has been returned by Optimization Problem 3.6 using one million samples and then normalized so that the one dimensional data set can be compared to the four dimensional data set. The computation time is likewise normalized for comparison between the one dimensional and four dimensional cases.

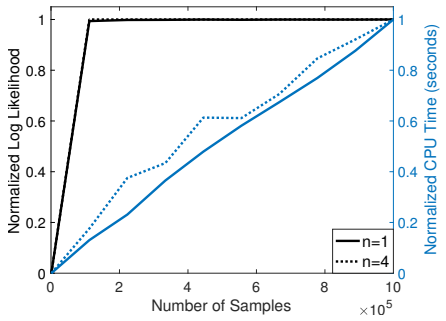
Note that in each case the computation time increases approximately linearly as a function of the number of samples in S. The computation time does not increase linearly with respect to the dimension of the problem, which we explore in the following subsection.

In the one dimensional case the log likelihood does not significantly change after approximately 10^5 samples. On the other hand the log likelihood in the four dimensional case was too small to be accurately computed in the case of 10^4 samples, and required $8 \cdot 10^5$ samples before the log likelihood was accurate. In the one dimensional case as few as 10^5 samples can be used where in the four dimensional case at least eight times as many points are required.

The WCE Optimization Problem To study the properties of Optimization Problem 3.16 with respect to the number of samples in S we plot the normalized



(a) The normalized log likelihood and computation time of a degree 10 SE trained with the first dimension of the Iris dataset (n=1) and a degree 4 SE trained on the full data set (n=4).



(b) The normalized worst likelihood and computation time of a degree 10 WCSE trained with the first dimension of the Iris dataset (n=1) and a degree 4 WCSE trained on the full data set (n=4).

Figure A.1: The normalized log likelihood and computation time to compute SE and WCSE distributions after using one million points to calculate the normalization constant.

worst case likelihood and the normalized computation time of the WCE problem in Fig. A.1(b) averaged over 15 trials. The worst case likelihood is calculated after λ^* has been returned by Optimization Problem 3.16 using one million samples and then normalized so that the one dimensional data set can be compared to the four dimensional data set. The computation time is likewise normalized for comparison between the one dimensional and four dimensional data sets.

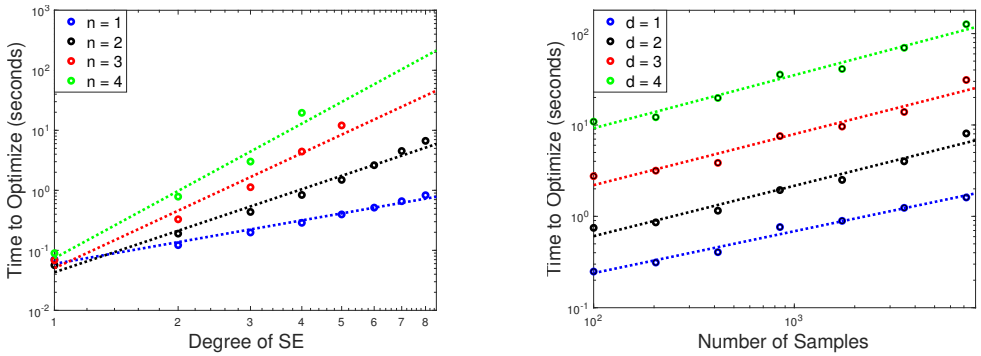
Note that like in the MLE case, the computation time increases approximately linearly as a function of the number of samples in S. However, unlike the MLE case, in both the one dimensional and four dimensional cases the worst case likelihood does not significantly change after approximately 10^5 samples. The λ^* of the WCE problem thus is significantly less affected by the dimension of the problem than the MLE problem.

Computational Complexity of Optimizing SE PDFs

Here we provide a numerical estimation of the complexity of the MLE and WCE optimization problems with respect to the number of training data points, the dimension and the degree of the SE. We first will specify two techniques employed to decrease the computational complexity of solving the MLE and WCE problems.

Implementation To solve the MLE and WCE optimization problems we develop two techniques to decrease the computation time. The first technique automatically selects the number of samples used to numerically compute the integration constant. The second solves the MLE and WCE problems by iteratively solving the problem for smaller degree SEs first.

The Sampling Sub-Routine: To compute λ^* we use a small initial sample of m_c points to calculate the numerical integration constant and a much larger sample of m_0 points to verify that the number of samples was adequate. Using the computed λ^*, if the objective function using m_c samples is within a threshold value of the objective



(a) Computational complexity analysis of the MLE problem, the different colored lines represent data sets of different dimension.

(b) Computational complexity analysis of the WCE problem, the different colored lines represent different degree SEs.

Figure A.2: Computational complexity analysis of both the MLE and WCE optimization problems as the degree or number of training samples is varied.

function using m_0 then λ^* is accepted, otherwise the number of samples is increased by a factor of κ.

In this dissertation we select m_c to be 10^3 samples, m_0 to be 10^6 samples and the threshold for accepting λ^* is if the objective functions are within 5% of each other. Finally we set $\kappa = 10$ so that we increase the number of samples by a whole magnitude if the number of samples was not enough.

Iterative Solutions: To find an optimal degree k SE we apply an iterative procedure, wherein we first find λ^* for a degree $d = 1$ SE. We then use this λ^* to initialize the search for the optimal degree $d + 1$ SE and repeat this process until we have the optimal λ^* of the degree k SE.

Data: In this section we use the data sets, VOS, TV[1], BP, and IR as can be found in [128, 119, 83, 55] in the UCI machine learning or OpenML databases.

The MLE Optimization Problem For the MLE formulation we plot the computational time in Figure A.2(a) as a function of the degree for each of the four data sets in a log-log plot. Empirically the computational complexity is of order $O(d^{1.2})$, $O(d^{2.3})$, $O(d^{3.2})$, and $O(d^{3.7})$ for dimension of 1 through 4 respectively.

The number of data points does not have a consistent effect on the computation time of the MLE optimization problem. For instance, increasing the number of samples can decrease the computation time as the number of data points changes the optimal λ.

The WCE Optimization Problem For the WCE optimization problem the number of data points has a more consistent effect on the computational time than the MLE optimization problem. Therefore, we plot the computational time of the three dimensional data set BP, in Figure A.2(b) where the number of points m are varied

[1]The data set TV originally contains data points with more than two dimensions, however, we extract the time of day and traffic volume from the original data (the real valued variables) to create our two dimensional data set.

along the x axis and we plot separate colors for degrees d between 1 and 4. From this data we have empirically determined the computational complexity is approximately $O(m^{0.54}d^{2.7})$ where m is the number of data points in $\mathcal{D}$ and d is the degree of the SE random variable.

For large numbers of data points the MLE optimization problem will have a significantly faster computation time.