A Probabilistic Modeling Approach to CRISPR-Cas9

by

Jonathan Wilder Lavington

B.S., University of Colorado Boulder, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Master of Science, University of Colorado Boulder, 2018

Department of Applied Mathematics

2018

This thesis entitled:
A Probabilistic Modeling Approach to CRISPR-Cas9
written by Jonathan Wilder Lavington
has been approved for the Department of Applied Mathematics

_____

Prof. Manuel E. Lladser

_____

Prof. Stephen Becker

_____

Prof. William Kleiber

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Lavington, Jonathan Wilder (M.S., Applied Mathematics)

A Probabilistic Modeling Approach to CRISPR-Cas9

Thesis directed by Prof. Manuel E. Lladser

CRISPR-Cas, a particular type of microbial immune response system, has in recent years been modified to make precise changes to an organisms DNA. In the early 2000s scientists discovered through the study of *Streptococcus pyogenes*, that a unique CRISPR locus (Cas9) exhibited specific RNA-guided cleavage near short trinucleotide motifs (PAMs). Further research on Cas9 eventually led researchers to create methods that actively edit genomes through Cas9-dependent cleavage and to manipulate transcription of genes through engineered nuclease-deficient Cas9 (dCas9). These techniques have enabled new avenues for analyzing existing gene functions or engineering new ones, manipulating gene expression, gene therapy, and much more.

While great strides have been made over the last decade, CRISPR is still prone to inaccuracies which often generate sub-optimal editing efficiency or off-target effects. The primary interest of this thesis is the investigation of targeting efficiency concerning changes in the guide RNA (gRNA) composition. While many different factors affect the ability with which a given gRNA can target a DNA sequence, we have focused our research primarily on the formation of the R-loop: the hybrid structure formed when the Cas9/dCas9:gRNA complex binds to a host DNA site.

In our investigation, we have attempted to account for several experimental findings reported in the literature as influential for binding efficiency. These include position dependence, base pair composition dependence, and the effects of runs of consecutive mismatches. Using a Gambler's Ruin Markov model to mimic the process of R-loop formation, we fit our model to experimental data and show that the match/mismatch configuration between the gRNA and the DNA target allows for accurate predictions of R-loop formation in bacteria.

Dedication

This thesis is dedicated to my friends: who always make it hard to keep a straight face, my mother: who taught me how to ski, and to my father: who never forgot to read the Hobbit to me before bedtime, and without whom, none of this would be possible.

Acknowledgements

I would first like to thank my research advisor, Manuel Lladser, for working with me on this amazing project over the last two years. His insight into the problems that we faced every week and his ability to organize my sometimes scattered approach to research has been integral to the success of this project. Most importantly however, he has helped me learn how to step back from the problem at hand to see how it fits into the big picture.

Secondly, I would like to thank Katia Tarasava for providing us not only the data for testing and experimenting with, but also for providing a grounding biological perspective to the model presented below. Without her expertise and insight into the biochemical processes within CRISPR-Cas, our model would lack the principle approach that we set out to achieve.

I would also like to thank Anne Dougherty for funding this research through the National Science Foundation EXTREEMS grant DMS 1407340, and for helping me figure it all out as my academic advisor. Moreover, seeing her leave the ec after 8 pm, or come into work on a Saturday, has helped me focus my efforts, and appreciate how hard the APPM faculty work. This also extends to all the professors and instructors who have found great answers to my sometimes ill-posed questions over the years. Specifically, I'd like to thank: William Kleiber, Adam Norris, Chris Ketelsen, Jem Corcoran, Stephen Becker, and Sujeet Bhat.

Finally, I would like to thank Lily, all my friends, and my family, for somehow putting up with me talking about this project for the last two years. Without them, I would have lost my mind a long time ago.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

This chapter gives a brief overview of what CRISPR-Cas is, how it is used, and what we are interested in modeling. Even more importantly however, it reviews the over-arching biological mechanism of CRISPR-Cas systems. By getting a good understanding of this process we can build an intuition for why the model proposed makes sense. In this chapter we also review the specific application that we are considering, and briefly mention other applications that our model might be extended to in the future.

1.1 What is CRISPR-Cas?

Recently a new genetic engineering technology known as CRISPR-Cas was developed, and has replaced many traditional techniques. CRISPR stands for "Clustered Regularly Interspaced Palindromic Repeats", and Cas for "CRISPR-associated nuclease". Broadly speaking, CRISPR-Cas works by forming a complex between a Cas nuclease (DNA cutting enzyme) and a specialized RNA sequence, called the guide RNA (gRNA). The gRNA contains a common repeat structure that Cas recognizes and binds to and a variable spacer region that contains a unique sequence of $\approx$ 20-30 nucleotides. This complex searches the host genome for a sequence complimentary, or nearly complimentary to the gRNA spacer, and cleaves it. The DNA site cleaved by the gRNA-Cas complex is generally associated with a gene that a researcher would like to mutate in some way. This process whereby a Cas enzyme (such as Cas9) attaches to and cleaves a specific host DNA site is displayed by Figure 1.1.

The Cas enzyme is special because of its ability to bind to specific sites along a genome that are located next to short sequences of nucleotides, known as Protospacer Adjacent Motifs (PAMs). These short sequences are usually 3 or more nucleotides long, and occur commonly on the genome, making almost any site available for targeting by the Cas enzyme. After identifying one of these sites, the gRNA-Cas complex is able to bind to, and cut the DNA sequences complimentary to the gRNA spacer. This cutting then initiates natural forms of DNA repair such as non-homologous end joining (NHEJ), or homology directed repair (HDR). In the first case, the base pairs at the two endpoints where the DNA was cut (called double-strand breaks or DSBs) are attached to each other via naturally occurring processes in the cell. In the second case, the cell will be given template DNA containing sequences homologous to the region where the DNA was cut. This allows scientists to actually edit the hosts genome DNA where the Cas has cleaved it. Through each of these techniques, and others not mentioned here, researchers have proven CRISPR's ability to actively modify genomes.

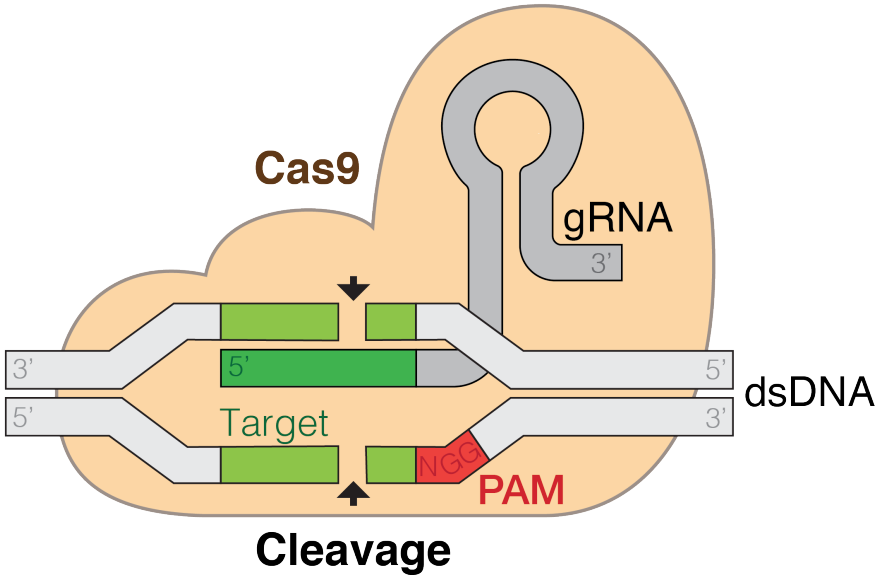

Figure 1.1: Representation of active Cas9-gRNA cleaving a double stranded DNA (dsDNA) at a target site. Figure from reference [16].

1.2 Advantages of CRISPR-Cas

CRISPR-Cas methods hold advantages over previously developed genetic engineering techniques due to high efficiency, decreased cost, relatively simple implementation, and improved targeting specificity [33]. The last being one of the most important, as the ability to modify a specific target site on a host genome without changing anything else has historically proven difficult, especially in human cells. Even more importantly, unlike many of the former approaches to gene editing that require expensive and difficult manipulations to engineer enzymes to target certain sequences (such as ZFNs and TALENs), CRISPR-Cas systems are highly versatile [8]. This is because in CRISPR-Cas, targeting specificity is determined primarily through the complementary gRNA spacer sequence that directs the enzyme to the target site [8]. While there are still drawbacks to using this system, many of these issues are rapidly being addressed as CRISPR-Cas systems become more widely adopted [33]. Researchers around the globe have already begun to apply this approach to a variety of applications in biotechnology, including materials and biofuels production, biomedical applications, gene repression and activation, gene insertion, deletion or replacement, base editing, and much more [33].

1.3 A Biological Perspective of CRISPR-Cas Systems

CRISPR-Cas technologies were developed from studying natural adaptive immune system found in some bacteria. These systems evolved in response to invasion by viruses that inject foreign genetic material in the form of DNA or RNA into bacterial cells to exploit their machinery and produce more viruses. CRISPR immunity gets around this by effectively creating specialized molecules that can recognize and destroy foreign DNA that the cell has seen in the past similarly to how antibodies work. Five primary types of CRISPR-Cas systems have thus far been discovered; this thesis however focuses specifically on the type II system which is characterized by the Cas9 nuclease. In Cas9-based systems the process of developing immunity generally occurs in three separate stages, as show in Figure 1.2.

In the first stage, called acquisition, short DNA sequences (called protospacers) are selected from foreign nucleic acids and integrated into a specific CRISPR locus on the host genome. In the next step, bio-genesis, the CRISPR locus, which contains multiple gRNA spacers and repeats, is transcribed into a precursor RNA, which is then processed into mature gRNAs. The gRNA molecules then bind the Cas enzyme to form targeting complexes that contain a unique spacer sequence that is complementary to the foreign target DNA. In the last step, called interference, Cas nuclease is recruited to the target DNA site via complimentary base pairing to the gRNA where it cleaves the foreign DNA. The process of foreign DNA recognition depends on the Cas nuclease searching the genome for specific sequences complementary to the gRNA. To prevent cleavage of the host CRISPR locus which contains the same sequences, Cas interference also requires the presence of short sequences called protospacer adjacent motifs (PAMs), which are not present in the CRISPR locus. By targeting and then destroying these foreign sequences, the bacteria are able to avoid transcribing DNA from viruses, making its genome resistant to viral invasion. It should also be noted that the gRNA can be expressed in different forms: as synthetic single-guide RNA (sgRNA) or the naturally occurring CRISPR RNA (crRNA) that has to pair with a trans-activating CRISPR RNA (tracrRNA). The only practical difference between these two targeting sequences is that sgRNA is a synthetic hybrid where the two parts (crRNA and tracrRNA) are combined into one molecule for ease of expression. While not prominently discussed, it is important to note when reading through the general literature.

1.4 dCas9 Mediated Gene Repression

This thesis addresses a specific application of CRISPR-Cas technology used to modify gene expression in bacteria, by blocking the RNA polymerase (RNAP) enzyme from transcribing a gene of interest. In this application, the nuclease activity of Cas9 has been abolished via mutagenesis. The resulting enzyme is called nuclease-deficient Cas9 (dCas9). This allows the gRNA-dCas9 complex to bind to the gene of interest without cleaving it. Since the RNA polymerase cannot access the blocked gene, this gene will be transcribed less often (or not at all), thereby reducing

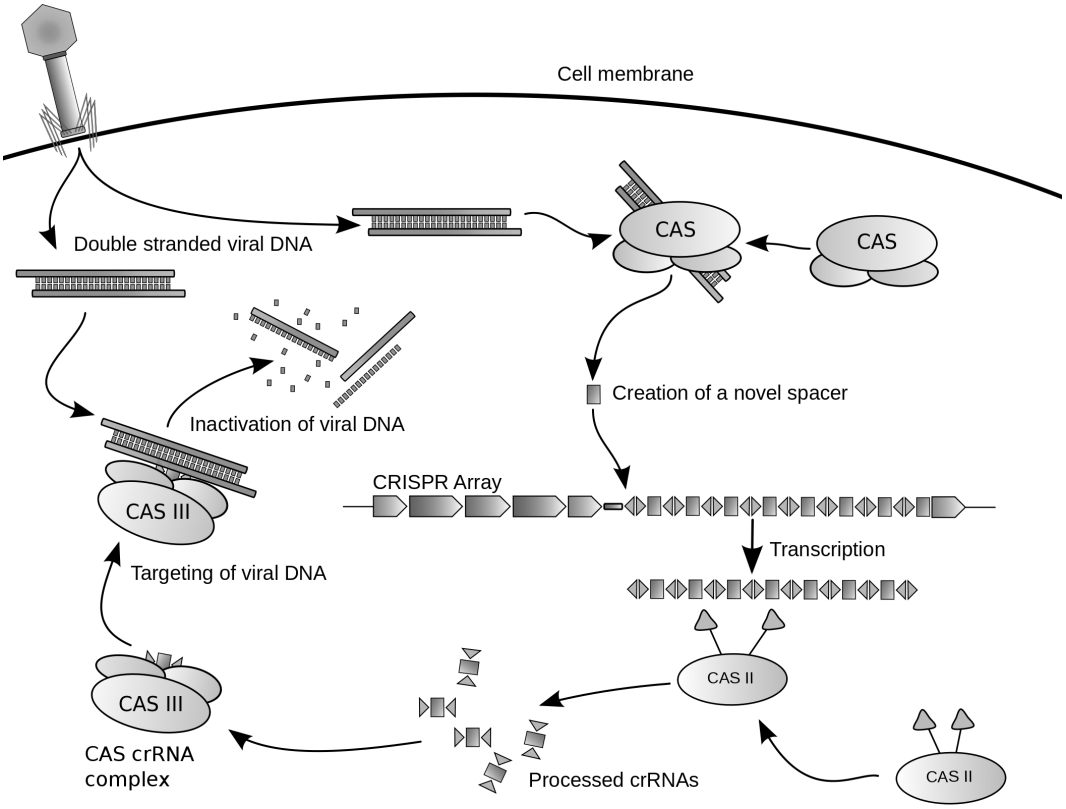

Figure 1.2: Visualization of the three stages of adaptive immunity via CRISPR-Cas within a bacterial cell. Figure from reference [11]

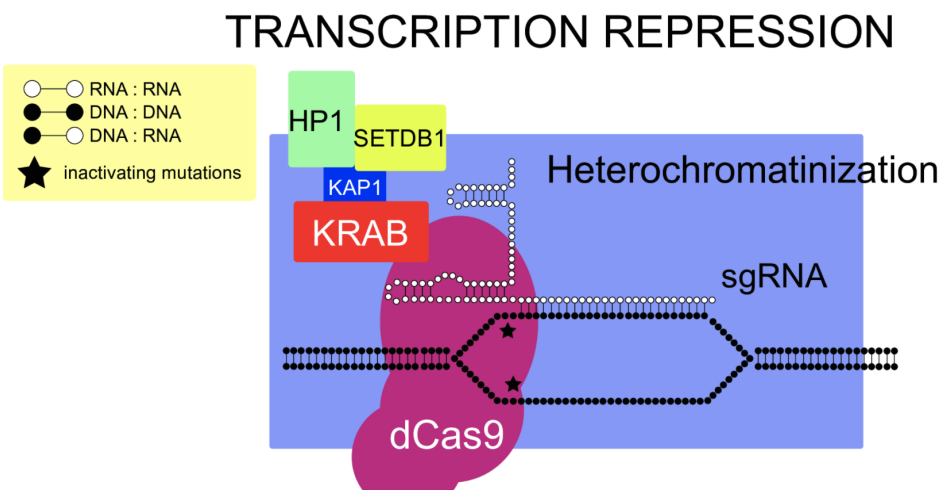

Figure 1.3: Transcriptional repression via effector domain. Figure from [30].

its expression. Importantly, it is possible to affect the level of gene repression created by these binding events modulating the strength of interaction between the gRNA and the target DNA by incorporating mismatches into the gRNA sequence [35]. For a visualization of this process, see Figure 1.3.

To accomplish this process, the dCas9 enzyme and the gRNA must be incorporated into the cell. Inside the cell, these two components bind to form what is referred to as the intermediate complex. This complex then spends time shifting its conformation until stabilizing, a process referred to as isomerization. Once isomerized, the complex is described as active. The active complex will continuously attempt to bind the host DNA at sites containing protospacer adjacent motifs (PAMs). Upon reaching any one of these sites, the complex will attempt to form a gRNA-DNA hybrid. This hybrid is referred to as the R-loop, and is the combination of the dCas9 complex, a bound (or partially bound) gRNA, and the complimentary single-stranded host DNA site. Upon successful R-loop formation, the dCas9 enzyme will either dissociate from the site or remain bound, depending on the strength of the interaction and environmental factors such as temperature and local DNA super-coiling [8, 13, 37, 35]. Interestingly enough this technique can be modified so as to increase transcriptional activity instead of decreasing it (known as transitional activation) via incorporation of activator domains, as displayed in Figure 1.3.

1.5 Thesis Overview

This thesis proposes a new probabilistic model of the formation of R-loops inspired by features of this process described in the literature [8, 13, 37]. The most central of these is that R-loop formation occurs in a sequential, stepwise fashion. This motivates to model this process as a discrete state-space Markov chain, where each state represents the extent to which complementary bases between the host genome and the gRNA have successfully bound.

Although other models of R-loop formation in the literature have relied solely on statistical dynamics [8, 37] or machine learning [35, 1], the granular nature of the new model facilitates the investigation of previously untestable hypothesis. In particular, this thesis draws various conclusions about both novel hypotheses regarding dCas9 systems, as well as other contentions in the current literature.

Using the new model, we can fit and predict with reasonable accuracy normalized fluorescence data provided by K. Tarasava of the Gill Lab at the University of Colorado - Boulder. In turn, by targeting sequences that are not fully complimentary, this work shows that it is possible to precisely manipulate host gene expression for virtually any gene, lending this process to a wide variety of potential industrial and bio-medical applications [33, 34, 8]. While the primary topic of this thesis is to investigate R-loop formation, and much of the content of our hypothesis is in regards to individual sequence composition, to account for competitive off-target binding, a model for DNA integration has been adapted from earlier work [8].

Chapter 2

A System-Wide Model of CRISPR-Cas Activity

This chapter discusses how one can construct a system-wide biophysical model of the CRISPR-dCas9 genetic repression mechanism. The model discussed below was derived from previous sources [8], but was adapted to fit the specific requirements of the problem at hand. This chapter will begin by giving an overview of the biological process, which can be described as a sequence of kinetic reactions that can be more easily modeled if various assumptions are made. Next the explicit model is constructed, and model behavior analyzed following various examples. Finally, a few of the assumptions and quirks of this approach to CRISPR modeling are reviewed for posterity.

2.1 Chemical Kinetics Description

The following kinetic model and much of the structure for the system differential equations used within this thesis is derived from reference [8]. Modifications were then made to various aspects of the system in order to then fit the purposes of this thesis, and more specifically the dCas9 binding system that was investigated. To better understand how this model is formulated, it is easiest to first see how the singular biological process occurs in vivo (within the cell), and then consider how to model it via differential equations.

The first step within most laboratory procedures for dCas9 mediated promoter repression begins with the insertion of the dCas9 enzyme and artificially generated guide RNAs (gRNAs) into a population of cells. To model the molecular concentration of dCas9 and gRNA within the cells—as functions of time—we consider their respective rates of expression (r_{dCas9}, and r_{gRNA}),

order	example	rate
0	$A \longrightarrow P$	constant
1	$A \longrightarrow P$	$\propto A$
2	$A + B \longrightarrow P$	$\propto A \cdot B$

Table 2.1: Examples of chemical reactions of differing orders

rates of deterioration (δ_{dCas9}, and δ_{gRNA}), and the rate at which they will combine to form the next intermediate molecule (k_f). Each of these rates describes a constant term within a chemical reaction that is either zero order, first order, or second order. An example of each of these types of equations is given in 2.1, however what is important to note is that the coefficients within the system of equations do not always represent a direct source of production for new molecules. There are also cases where the rate coefficient for certain molecules becomes less meaningful. One such example is the case where one of the reactants is saturated, and therefore is unlikely to become a limiting reactant within the mixture. For a more through discussion on reaction kinetics, and more organized discussion of what each of these reaction coefficients represent, see [5]. In the case of the equation below r_{dCas9} (as well as r_{gRNA}) represent the rate at which the inserted r_{dCas9} molecules become active within the system. Using these rates, one can describe the change in the number of active molecules (N_{dCas9}, and N_{gRNA}) within the cell over time as follows:

$$\frac{d\,N_{dCas9}}{dt} = r_{dCas9} - \delta_{dCas9} \cdot N_{dCas9} - k_f \cdot N_{dCas9} \cdot N_{gRNA}, \tag{2.1}$$

$$\frac{d\,N_{gRNA}}{dt} = r_{gRNA} - \delta_{gRNA} \cdot N_{gRNA} - k_f \cdot N_{dCas9} \cdot N_{gRNA}. \tag{2.2}$$

These equations describe the process whereby upon reaching maturity, the gRNA and the dCas9 enzyme react creating intermediary complexes within the cell. To describe the number of these over time (N_{Inter}), we consider the rate of deterioration ($\delta_{Complex}$) and isomerization (k_I) of the intermediary complexes. Recall isomerization is the process whereby a molecule shifts between atomic conformations, eventually transforming to its most stable molecular state. For CRISPR-dCas9 systems, this process is required to bind to the host genome [8], and is in fact the rate

limiting step for the binding process [8] (i.e. it is the slowest step of the chemical process and determines the overall rate of the biological reaction). Having identified all sources of depletion of intermediary complexes, we find that the change of N_{Inter} over time via the differential equation:

$$\frac{d\,N_{Inter}}{dt} = k_f \cdot N_{Cas9} \cdot N_{gRNA} - \delta_{Complex} \cdot N_{Inter} - k_I \cdot N_{Inter}. \tag{2.3}$$

We can then note the the rate of deterioration for both N_{Inter}, and $N_{Complex}$ will be the same as they are in fact the same molecules (under different structural conditions, as they are isomers). Finally, the change in the number of active complexes ($N_{Complex}$) over time can then be described by the equation:

$$\frac{d\,N_{Complex}}{dt} = k_I \cdot N_{Inter} - \delta_{Complex} \cdot N_{Complex} - \sum_j r_{binding}(j), \tag{2.4}$$

where j denotes a possible binding site, and $r_{binding}(j)$ is the rate of binding of active complexes at that site, which will be clarified in Section 2.2, and 3.

Once isomerization has proceeded, the mature, active complexes drift through the cell body and attempt to bind to host DNA sites immediately upstream (see Figure 1.1) PAM sites. In this thesis, the diffusion of active complexes within the cell is assumed isotropic. We note, however, that this assumption seems unlikely in eukaryotic cells, where chromatin structure may significantly affect how the interrogation process occurs [16, 13, 17, 8].

Upon successful attachment to a PAM site along the host genome, R-loop formation occurs. This process, which is discussed in more detail within Chapter 3, effectively consists of a sequence of pairwise binding events between the gRNA and complementary DNA site along the host genome (see Figure 1.1). Successful completion of this process then leads to either cleavage (in the case of dCas9), or nearly irreversible binding to the site in question (in the case of dCas9).

2.2 DNA Interrogation by the Active Complex

Next, we specify the rates of binding in equation (2.4), following the model in [8]. Intuitively, the binding rate should be (1) proportional to the number of active complexes within the cell $N_{Complex}$ (i.e., the more active complexes within the cell, the more often at least one will interact with a PAM site), (2) inversely proportional to the volume V of the cell (i.e., the bigger the cell volume, the less likely it is for the gRNA-dCa9 to interact with any of the target sites), and (3) proportional to a diffusivity constant D, representing how easily the active complex can actually drift within the cell body. Finally, binding rates should be (4) proportional to the average separation between binding sites (λ). With these observations in hand, it is natural to define the molar flow rate (r_{RW}) for the active complex as:

$$r_{RW} := \frac{\lambda \cdot D \cdot N_{Complex}}{V}. \tag{2.5}$$

A more rigorous derivation of this rate can be found in [9]. It should be noted, however, that this formulation assumes a well-mixed cellular compartment and that the rate of net molar flow is zero. These assumptions translate to all chemical components utilized in the reaction being effectively equidistant and intermingled, and no new reactants being added to the mixture. These assumptions are widely held in differential equations that model sequences of chemical reactions. For a more thorough overview of the implications of these assumptions, as well as how models like these ones described here are constructed, see [28].

To properly define the binding rate at a site j one also needs to consider super-coiling. This is the process where the DNA double helix twists itself into a coil, guarding against other cellular processes that may transcribe or corrupt its genetic material. Double-stranded DNA with extra helical twists is called positive super-coiled, whereas unwound DNA is called negative super-coiled.

To measure super-coiling at a site j, one needs to consider the length of untwisted DNA (ℓ), and initial and final super-helical density (σ_F, and σ_I). Using these quantities, we can define the

amount of energy required to uncoil a strand of DNA as:

$$\nabla G_{supercoiling} = 10 \cdot \ell \cdot (\sigma_{F,j}^2 - \sigma_{I,j}^2))k_b T \tag{2.6}$$

If we then place this value into the Boltzmann distribution, we get a quantity proportional to the probability of being in an uncoiled state. If we cancel out temperature and the Boltzmann constant, the coefficient of dissociation ($k_{d,j}$) from a site j is found to be:

$$k_{d,j} = k_d^* \cdot \exp(-\nabla G_{supercoiling}/k_b T) \tag{2.7}$$

$$k_{d,j} = k_d^* \cdot \exp(-10 \cdot \ell \cdot (\sigma_{F,j}^2 - \sigma_{I,j}^2)) \tag{2.8}$$

However, as we did not have access to site-wise super-coiling for this experiment, we utilized experimental results given in [8]:

$$k_d = k_d^* \cdot \exp(-10 \cdot \ell \cdot (\sigma_F^2 - \sigma_I^2)) \tag{2.9}$$

Where k_d^* is also a constant term used within previous sources [8]. A more thorough out explanation of this equation—based on a free energy perspective—lies in [8]. However for a shorter explanation: when relaxed B-form helical DNA of length ℓ is (un)twisted by 10σ turns, the change in free energy will be approximately equal to $10\ell\sigma k_b T$, where k_b is the Boltzmann constant, T is temperature, and *ell* is the length of the targeting strand.

Within bacterial genomes, much of the genetic information exists in a mostly untwisted DNA double helix that is easily accessible by the dCas9. As a result, in the systems considered in this thesis, this coefficient will often play a negligible role within the model. It has been suggested that some super-coiling structure does exist across certain bacterial genomes [18], and binding sites that are close to each other can cause conflicting binding effects. That is to say, if one site is bound, the

DNA within the site is negatively coiled, while the DNA around the site is positively coiled. This then makes sites that are near this one more difficult to bind to. While this was presented as one of the major deciding factors within [8], it is not the principle topic of this thesis, and as such no specific mechanism for spatial dependence of sites was included within the model.

Finally, if we define k_c as a final normalizing constant that represents the rate of irreversible binding (or in the case of Cas9 irreversible cleavage). While redundant within the model that we present, as the coefficient is absorbed in the bias of our probability term, within [8] only the ratio of k_c, and k_d.

$$r_{binding}(j) = \frac{k_c}{k_c + k_d} \cdot r_{RW} \cdot P(gRNA, site(j)) \tag{2.10}$$

where $P(gRNA, site(j))$ is the probability that dCa9 successfully binds to site j. A more thorough discussion about this probability is provided in Chapter 3. For now, we only note that this probability is a function of the type of PAM that the complex initially binds to, the gRNA sequence, and the specific site being targeted.

With binding rates now defined, we can complete the description of the kinetics model. Returning for a moment to the last molecular concentration considered, $N_{Complex}$, we can now define the change in the number of a given host target site j over time using the following equation:

$$\frac{dN_{Target,j}}{dt} = \mu \cdot (N_{Total,j} - N_{Target,j}) - r_{binding}(j). \tag{2.11}$$

Here, $N_{Target,j}$ is the percentage of unbound sites of type j in the host genome, μ is the growth rate of cells within the cell population, and $N_{Total,j}$ is the total number of sites of type j on the genome (also referred to as the copy number). In particular, the quantity $\mu(N_{Total,j} - N_{Target,j})$ is equivalent to the net production rate of available DNA sites, and distinguishes the initially available target sites of type j versus steady state promoter activity within the cell. That is to say that even while

sites are bound to by the dCas9, at steady state, there will still likely be a few bacteria that contain the gene in question. As will be displayed below, the cell growth rate μ, when increased even a bit can drastically change the percentage of bounds sites, and in turn the steady state promoter repression. The full description for a genome with n off-target site types is given in Table 2.2.

Systemwide Biophysical Model	
Biological Process	Corresponding Differential Equation
Transfection of dCas9 and gRNA	$\frac{dN_{Cas9}}{dt} = r_{Cas9} - \delta_{Cas9} \cdot N_{Cas9} - k_f \cdot N_{Cas9} \cdot N_{gRNA}$
Formation of the Intermediary Complex	$\frac{dN_{Inter}}{dt} = k_f \cdot N_{Cas9} \cdot N_{gRNA} - \delta_{Complex} \cdot N_{Inter} - k_I \cdot N_{Inter}$
Isomerization of the Intermediary Complex	$\frac{dN_{Complex}}{dt} = k_I \cdot N_{Inter} - \delta_{Complex} \cdot N_{Complex} - \sum_j r_{binding}(j)$
Principal target interrogation by the Active Complex	$\frac{dN_{target,p}}{dt} = \mu \cdot (N_{total,p} - N_{target,p}) - r_{binding}(p)$
First off-target interrogation by Active Complex	$\frac{dN_{target,1}}{dt} = \mu \cdot (N_{total,1} - N_{target,1}) - r_{binding}(1)$
Second off-target interrogation by Active Complex	$\frac{dN_{target,2}}{dt} = \mu \cdot (N_{total,2} - N_{target,2}) - r_{binding}(2)$
⋮	⋮
n-th off-target interrogation by Active Complex	$\frac{dN_{target,n}}{dt} = \mu \cdot (N_{total,n} - N_{target,n}) - r_{binding}(n)$

Table 2.2: Full system of differential equations describing the modified biophysical model in [8].

To end the discussion of the creation of this model, we would like to reiterate the key ways in which it differs from the previous formulation given by [8]. Firstly, this system of equations no longer includes an explicit notion of genome length; however, by including larger numbers of potential off-target sites (something likely in long DNA sequences), the model accounts for the off targeting effects and reduced expression described by [8, 37, 13]. Secondly, as we are no longer considering Cas9 enzymes that have the capacity to cleave their targets, we have also omitted kinetic coefficients associated with cleavage rates. Lastly, as a result of our redevelopment of probability of R-loop formation, we have also omitted some of the coefficients associated with super-coiling.

2.3 Model Behavior

In this section, we test the systemwide kinetic model summarized in Table 2.2 to ensure that it works as expected. Our main interest is in understanding the effect of parameters in binding. The values reported in Table 2.3 were derived from experiments reported in the literature [8, 34, 29].

Kinetic Model Parameters					
Rate Constants	Value	Interrogation Constants	Value	Nucleotide Pair	Free Energy
μ	1.25×10^{-4}	σ_F	5×10^{-3}	(AT,TA)	(-1.27,-0.12)
δ_{Cas9}	10^{-3}	σ_I	5×10^{-2}	(GC,CG)	(-2.70,-1.44)
δ_{gRNA}	10^{-3}	k_d^*	1.0	(TC,CT)	(-1.66,-1.29)
$\delta_{Complex}$	10^{-3}	V	1.0	(GT,TG)	(-2.04,-0.78)
r_{Cas9}	10^{-3}	D	4.0	(AC,CA)	(-2.04,-1.97)
r_{gRNA}	10^{-3}	λ	1.0	(AG,GA)	(-1.29,-1.66)
k_I	10^{-1}	k_d	6.5×10^{-1}	(GG,CC)	(-1.97,-1.97)
k_f	8×10^{-3}	k_c	2.0	(AA,TT)	(-1.04,-1.04)

Table 2.3: Parameters used within the following model experiments were chosen so as to give an idea of what one can expect from the average promoter repression experiment. The first two columns give rate parameters

As an initial test, we trained the model using the parameters given in Table 2.3. As a baseline metric, we can first look at how the system progresses through time for a partially complimentary guide RNA (AATTGACTTAAAGTCTAACCTATAGGATACTT where the underline represents where a mismatch occurs). For this simulation, I created a cut off for an off target site to be considered within the integration scheme based on the minimum probability of binding. In this way we avoided integration of roughly five thousand off target sites In Figure 2.1, we can see how

the concentration of each intermediary complex within the CRISPR-Cas system behaves in time, where the row includes the percentage of bound locations for both on and off-target PAMs. Due to the small number of off target sites we can also see that this system is stable, as the percent binding levels out and stabilizes at the a single value instead of slowly decaying.

Next we considered the relationship between binding probability and cell growth rate (μ) as well as cell volume (V). As can be seen in Figure 2.2, as μ increases, the rate of successful binding decreases. This makes sense: if sites are replicated at rates that outpace the attacks made by the active complex, no amount of interrogation will affect the hosts DNA in steady state if we hold the concentration of gRNA-dCas9 constant. In particular, rapid cell growth may almost entirely negate the effects of the dCas9 mediated promoter repression. Similarly, as expected, increasing the cell volume decreases the overall binding within the cell. This makes sense physically as the larger the cell, the more the active molecule must traverse to interact with a given site.

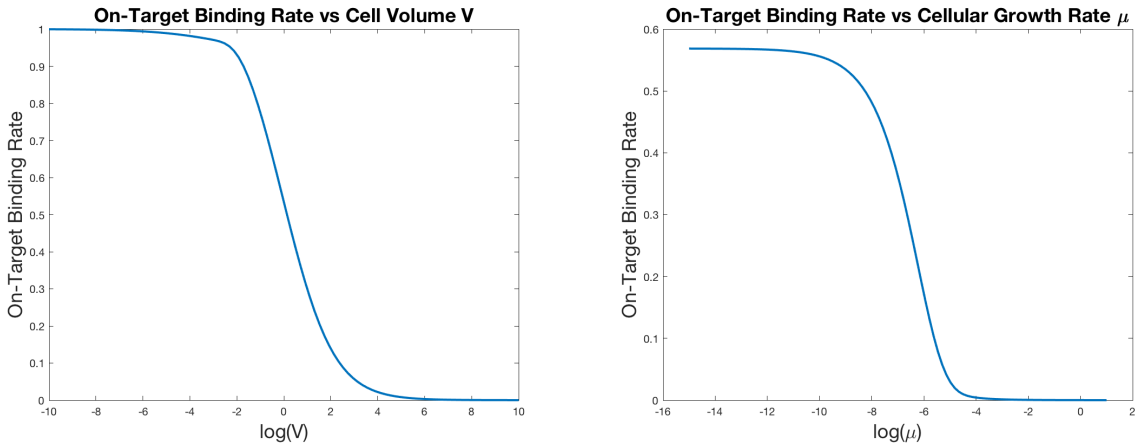

Figure 2.2: Relationship between cell growth (μ) versus binding, and cell volume (V) versus binding.

We also looked at how the rates of expression (r_{dCas9}, r_{gRNA}), deterioration (δ_{dCas9}, δ_{gRNA}, $\delta_{Complex}$), and the kinetic coefficients of formation and isomerization (k_f and k_I) affect binding. Starting with the rates of expression in Figure 2.4 and 2.5, we see some behavior that is a bit harder to interpret then previous results. This is because these parameters control the net flow of molecules into the system, and as a result have major effects on the timescale of the reaction.

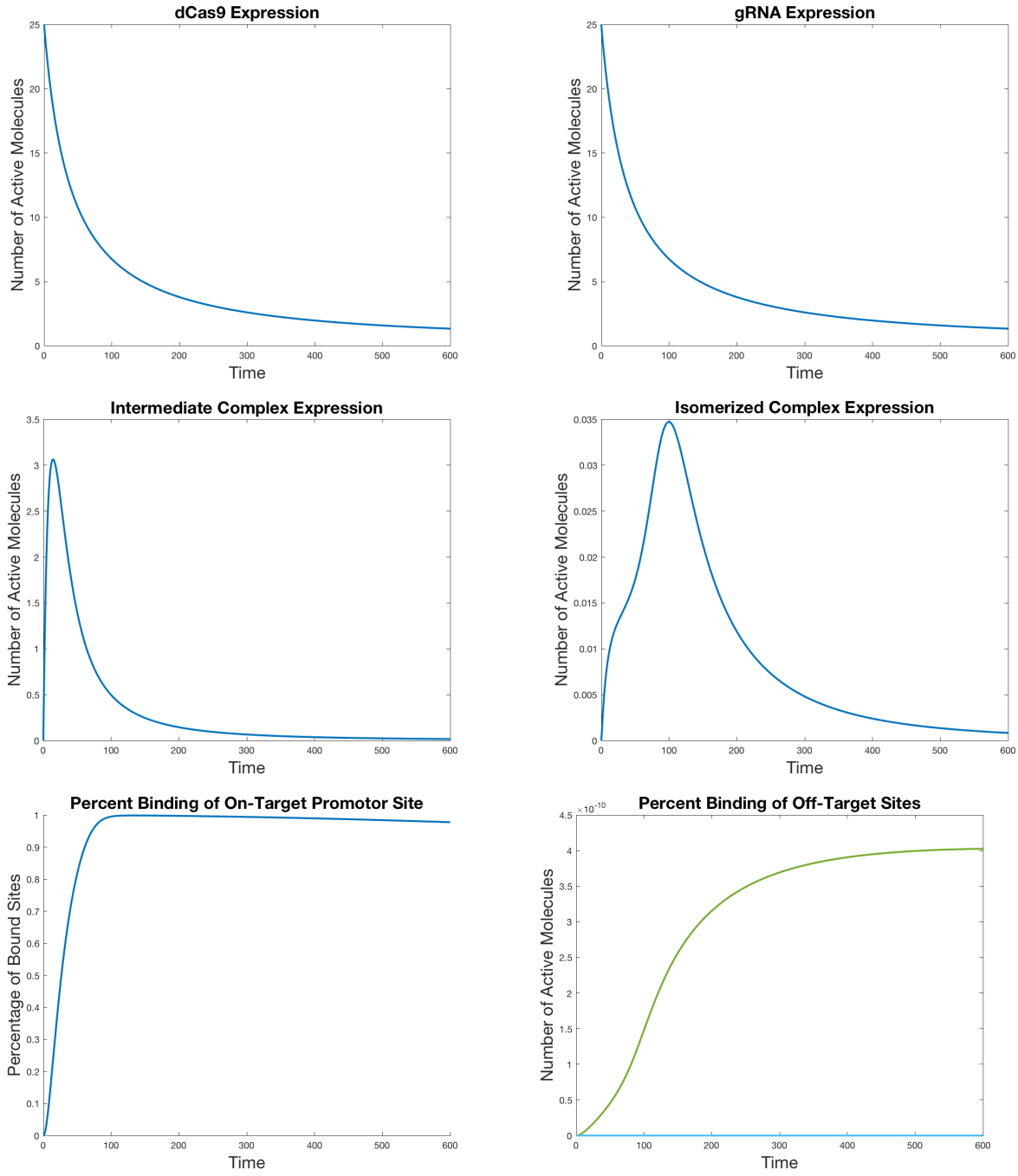

Figure 2.1: Simulation of systemwide kinetic model of dCas9 binding. Each plot displays either a molecular concentration, or the percentage of bound sites along the genome as a function of time (measured in seconds). The top column displays the change over time N_{Cas9} and N_{gRNA}. The middle row, displays the change over time $N_{Isomerized}$ and $N_{Intermediate}$. The bottom row, displays the percent binding for both on and off target sites.

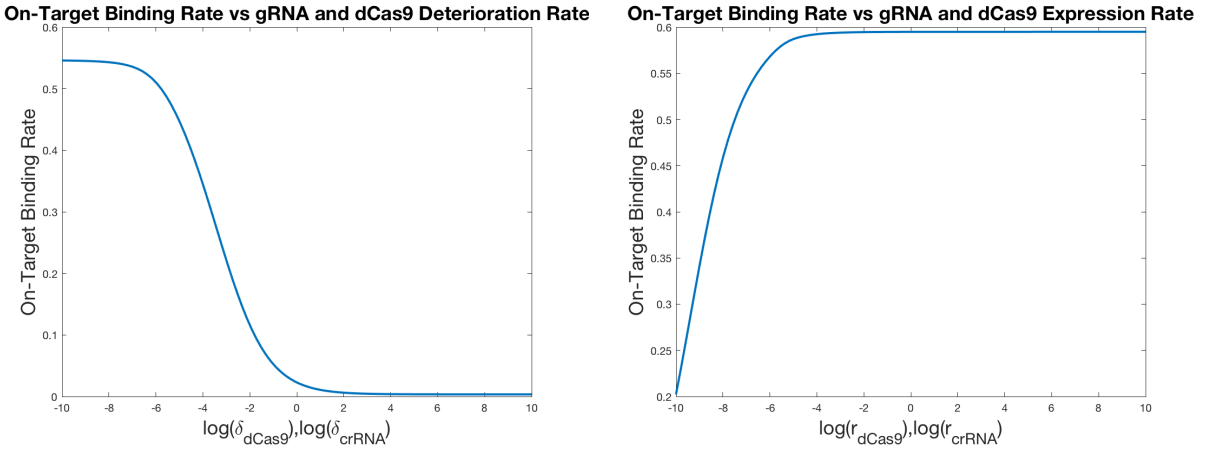

Figure 2.3: Relationship between coefficients associated with rates of expression (r_{dCas9}, r_{gRNA}) versus binding on the left, as well as the coefficients associated with rates of deterioration (δ_{dCas9}, δ_{gRNA}) on the right

Therefore, once the rate of expression is high enough relative to the other coefficients there will be no measureable deterioration, and thus the target, barring off-target sites, will eventually be completely bound. Conversly if the rate of deterioration is high enough, the initial molecules in this sequence of reactions will deteriorate before they have time to react, and then attack the target site. This type of variation has practical consequences on the numerical stability of the system, and given a large enough difference in magnitude between parameters the equations will eventually become stiff. To avoid this we tested a few of the values discussed in the table above, and compared them to experimental results found in [34, 8]. As a final note, the plots given in 2.4 represent the change over simultaneous change in both expression and deterioration parameters. This is because the two differential equations are exactly the same given equimolar initial concentrations of each molecule, and thus any changes to one would have the same effect on the system as any changes on the other. However if one of the two molecules (dCas9 or gRNA) where to have a significantly smaller starting quantity (act as a limiting reagent), then drastic changes in either of these rate parameters would cause the substance to deteriorate even more rapidly.

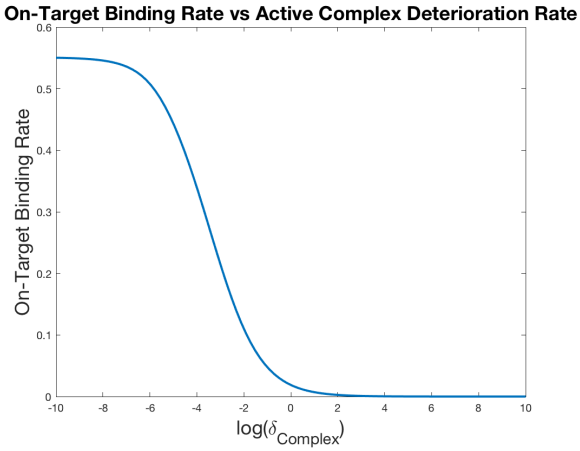

Figure 2.4: Relationship between the coefficients associated with rates of deterioration ($\delta_{Complex}$) for the active complex.

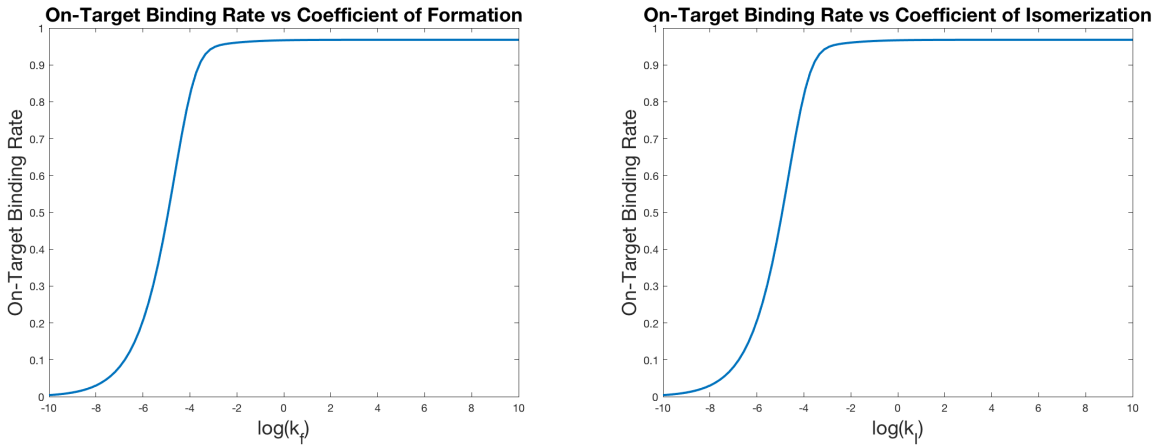

Figure 2.5: Relationship between coefficients of formation and isomerization versus binding. (k_f and k_I)

2.4 Initial Conditions Considerations

Finally, we discuss how the initial concentrations of dCas9 and gRNA may affect the systemwide kinetic model evolution. Within dCas9 systems we can generally consider two different regimes: saturating and non-saturating conditions of the gRNA and dCas9. Aside from affecting the binding strength, saturating conditions greatly increase the possibility for off-targeting effects, which in turn can lead to transcriptional repression at unintended sites. Off targeting effects can lead to unforeseen issues within the host, sometimes leading to unwanted mutations or even death of the cell body. Under the parameters given above, and over the timescale considered, we achieve steady state dynamics as they are reported within the literature [8, 34, 13]. However by varying the rate parameters this behavior can be achieved for for a variety of initial conditions, allowing this architecture to adapt to a variety of dCas-Cas9 binding systems.

We can also look at a few aspects of off-target effects, namely: off-target copy number (e.g. number of off-target site duplicates), and the total number of off-target sites included within the model. Plots for each of these quantities are not included in this thesis as the the average copy number required to create any meaningful change within the system is many magnitudes larger then what is seen in bacterial genomes. This is because, even relatively complimentary sites tend to have a relative fluorescence less than 10^{-8}. This means that in order to affect the rate of binding at the primary target site, the off target site in question would need a binding rate roughly $\propto 10^5$, and a probability of binding close to 1. Off-target copy number is however of major concern in human genomes, where the length of the genome saturates the number of possible off-target sites to almost this proportion (among sets of off target sites). Similarly, incorporation of higher on-target copy number has an equally modest effect on the system.

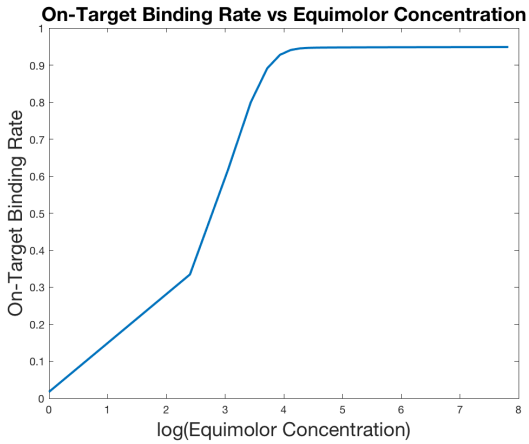

Figure 2.6: Relationship between Initial equimolar concentrations of dCas9 and gRNA molecules versus binding.

Chapter 3

A Zipper Model of R-Loop Formation

This chapter introduces a probabilistic model of R-loop formation. It begins describing the biological mechanisms behind R-loop formation. A Markov chain model of this process is then proposed, followed by a novel formulation of transition probabilities. The chapter concludes by reviewing whether features of R-loop formation reported in the literature still hold under this new formulation.

3.1 A Bio-physical Perspective

In the process of generating repression via the CRISPR-dCas9 mechanism, the isomerized CRISPR-crRNA complex must successfully form an R-loop with the corresponding host DNA site. To better understand this process, we first clarify two terms: complementarity, and DNA melting.

In the context of this thesis, the term complementarity refers to two molecules that can react, and combine; and is in many ways a generalization of the so-called "lock and key principle." This is a metaphor for the interaction that occurs between a substrate (the key), and an enzyme (the lock) that are compatible.

In our setting, DNA acts as a lock and RNA as a key, and their interaction is facilitated by base-pairing. (The DNA bases T, G, C, and A, are complementary, i.e., can base-pair with the RNA bases, A, C, G, and U, respectively.) In what follows, we extend this simple analogy that occurs in a binary fashion, to a continuous representation; that is to say, we will consider complementarity to be a measure of how well a key fits into a given lock, instead of whether it does or does not.

The next important concept, DNA melting, is simply the process by which two complementary single strand DNA sequences are uncoiled and sundered by an enzyme (in this case the dCas9). As the DNA is melted, complimentary base pairs within the DNA are pulled apart, providing space for the gRNA to sequentially bind to the complimentary bases along a single DNA strand. While in reality the sequential reaction is likely effected by the dynamics of DNA melting, this is not explicitly addressed by our model. Similarly this process is also affected by the temperature of the cell, however, again, this is not explicitly included in our model. To ensure that this information was not lost, we included a bias term within the Markov chain. In this way we implicitly account for the thermodynamic conditions within the cell culture.

Other factors regarding DNA melting and DNA super-coiling have also been reported to in some cases to drastically affect the binding at specific sites along the genome. For instance, DNA melting was recently reported to be affected by sites near the target who were already bound by another dCas9 complex [8, 35]. This is thought to be a result of the negative super-coiling at the site where the DNA is bound. However, further research is required to thoroughly verify this result, as well as under what experimental conditions it is most prominent.

Upon completion of the R-loop, the Cas9 can cleave the targeted site from the host genome. In the case of dCas9, the R-loop remains bound to the site, and thereby inhibits transcription. In a laboratory setting, the effectiveness of the guide sequence and endonuclease can be tested through the measurement (in lumens) of yellow fluorescent protein (YFP). Our model, assumes that upon successful completion of the R-loop, the complex remains irreversibly bound to the host site. This assumption is held in the majority of the literature and presents a significantly simplified view of the biochemical process [8, 37, 34].

3.2 A Probabilistic Perspective

Motivated by the previous description, we now define a Markov chain that mirrors the biophysical process. To fix ideas, we start with a simple example (see Figure 3.1). In this example we have a six nucleotides long single DNA strand, and a six nucleotide long RNA strand, that are

attempting to bind. Each pair of nucleotides is associated with a unique state in our Markov chain (see Figure 3.3). While non-complimentary base pairs cannot bind, these are associated with a state that will deter the R-loop formation. We note that earlier formulations of our model disregarded non-complementary base pairs as states. Such formulations, however, weighted disproportionately poorly complimentary strands due to the occurrence of shorter chains for sequences with an abundant number of mismatches.

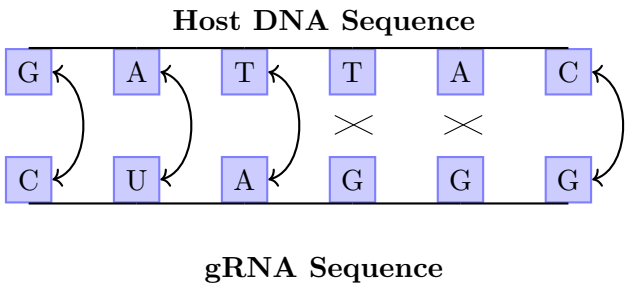

Figure 3.1: Short sequence of base pairs, whose sequential reaction can be represented as a Markov chain. Double-pointed arrows denote complementary bases. Crosses indicate non-complementary bases.

Two other states are added to the chain. One associated with the PAM site, which is always the initial state of the chain. The other, which we call the Anchor, represents the probability that given the R-loop is completely formed, the site will become irreversibly bound. This state was added based upon the assumption that during the attacks made by the active complex, irreversible binding occurs. While for the majority of interactions between host site and active complex, the anchor state accounts for the small percentage of instances where the complex is kicked off the site after completion of the R loop.

Additionally the PAM state acts as the primary buffer for the successful formation of the R-Loop. This is due to experimental results reported by [8, 13] which indicate that the majority of the free energy that is used within the R-loop formation process comes from the PAM site itself.

Furthermore, they also act as a buffer for off-target sites by severely downplaying binding rates due in non-cannonical PAMs. Meaning that the type of PAM is an important feature when attempting to designate the probability of successful binding between large sets of potential binding sites.

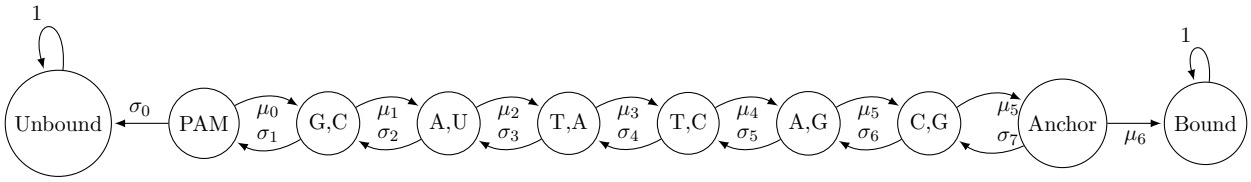

Figure 3.2: Markov chain representing the example in Figure 3.1.

Finally, our model includes two absorbing states. The Unbound state, which represents the complex being kicked out, and the Bound state, which represents a successful and irreversible binding. See Figure 3.1. The probability of successful R-loop formation is therefore given by the probability of absorption at the Bound state when the chain starts at the PAM state.

We note that a full analytic solution to this absorption probability is included in the following section. This problem can be generalized and reformulated as the solution to a system of linear equations. This approach is discussed in Appendix A, and extension to other chains outside of the Gambler's Ruin Markov model described above will be briefly considered in Chapter 5.

A general schematic of the chain when the DNA and the RNA each consists of n nucleotides is given in Figure 3.3. (For instance, for the experiments using fluorescence data, $n = 32$.) The total number of states of the associated Markov chain is therefore $(n + 4)$.

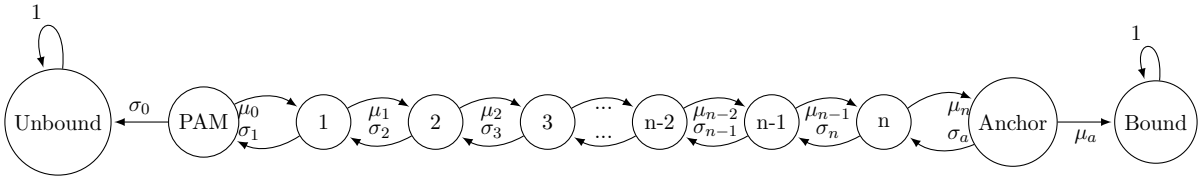

Figure 3.3: Generalized Markov chain model for R-loop formation in CRISPR-Cas9 when the gRNA consists of n nucleotide pairs, and therefore the DNA also consists of n nucleotides.

3.3 Formulation of Transition Probabilities

The next important question now that the Markov chain model has been formulated is: how can we define transition probabilities? From a statistical mechanics perspective, each transition probability implicitly describes how much more stable an adjacent state is relative to the current state. Therefore we can define this "local stability" via some linear combination of the "complimentarities" between each adjacent base pair.

Next, we address the transition probabilities of the Markov chain model. From a statistical mechanics perspective, each transition probability implicitly describes a "local stability" i.e. how much more stable an adjacent state is relative to the current state. Our model defines the forward transition probabilities (i.e., towards the binding state) at a given state in terms of the bases ahead in the DNA and RNA strand. Similarly, the reverse transition probabilities (i.e., towards the unbinding state) in terms of all bases behind. These transition probabilities are defined in term of free energy between pairs of bases (complementary or not), with values derived from earlier sources [29].

However, because our goal is to capture local stability, the free energy associated with states far away from a state contributes only marginally to the forward and backward transition probabilities. Free parameters determine the precise way in which this happens. Moreover, the functional form was selected so that small perturbations to the free parameter resulted in small prediction errors of the model. At state i these are defined as:

$$\hat{\mu}_i := \exp\left\{\text{bias}_F + \sum_{d=i}^{Anchor} f_F(d-i)\cdot\left(2\cdot[\![state\ d = match]\!]-1\right)\cdot\hat{\phi}(d)\right\} \tag{3.1}$$

$$\hat{\sigma}_i := \exp\left\{\text{bias}_B + \sum_{d=PAM}^{i} f_B(i-d)\cdot\left(2\cdot[\![state\ d = mismatch]\!]-1\right)\cdot\hat{\phi}(d)\right\} \tag{3.2}$$

In order to understand the coefficients within the expression above, we can first consider the forward and backward transitions as rates, later normalizing to convert rates into transition probabilities. First, let $\hat{\phi}(d)$ represent the complementarity measure between the target nucleotide and the gRNA nucleotide. Additionally, using Iverson's brackets ($[\![\ ]\!]$) to denote indicator functions, we can ensure that matches (i.e., complementary bases) favor the forward reaction but inhibit the reverse reaction. Similarly, mismatches (i.e., non-complementary bases) promote the reverse reaction and repress the forward reaction.

Next, we can define how the affects of the interaction at state d ($\hat{\phi}(d)$) affect the binding event at the next state given that we are currently in state i by the function f($\|d-i\|$). If we do not assume radial symmetry of this function, then it can be split into two separate functions, each denoting behavior in front of (f_F), and behind (f_B) the nucleotide pair of the current state.

As a final note, we must also define the quantities $\hat{\phi}(0)$:= complementarity measure at PAM state, and $\hat{\phi}(n+1)$:= complementarity measure at Anchor state; where both are considered to be "match" states. Additionally, a bias term is added to each transition probability in order to encompass auxiliary factors that affect R-loop formation such as temperature, and local super-coiling. Lastly, these rates are exponentiated so as to ensure that they are non-negative, and can therefore be converted into probabilities. Lastly, we can then convert these rates into probabilities via the following normalization:

$$\mu_i := \frac{\hat{\mu}_i}{\hat{\mu}_i + \hat{\sigma}_i} \tag{3.3}$$

$$\sigma_i := \frac{\hat{\sigma}_i}{\hat{\mu}_i + \hat{\sigma}_i} \tag{3.4}$$

In particular, $(\mu_i + \sigma_i) = 1$ and $\mu_i, \sigma_i \geq 0$.

3.4 Analytic Expression for Probability of R-loop Formation

Having defined the probability transitions for the Gambler's Ruin chain in Figure 3.3, we now deduce the probability the chain gets absorbed at the Bound state when starting at the PAM state. The argument follows closely that of Example 1.43 in Chapter 1 of [6].

Define a_k as the probability that the chain gets absorbed at the Bound state given it started at state k. We aim to determine a_{PAM} explicitly. Conditioning on the first step, and as long as k is neither of the two absorbing states, it follows that

$$a_k = \sigma_k \cdot a_{k-1} + \mu_k \cdot a_{k+1}. \tag{3.5}$$

Next we can note that $a_k = (\mu_k + \sigma_k) \cdot a_k$, and with some simplification:

$$(a_{k+1} - a_k) = \frac{\mu_k}{\sigma_k} \cdot (a_k - a_{k-1}).$$

Since $a_{\text{Unbound}} = 0$, we can iterate the above expression recursively to obtain that

$$(a_{k+1} - a_k) = a_{\text{PAM}} \cdot \prod_{j=\text{PAM}}^{k} \frac{\mu_j}{\sigma_j}.$$

Next we can utilize a telescoping series to deduce that:

$$(a_{k+1} - a_{\text{PAM}}) = a_{\text{PAM}} \cdot \sum_{i=\text{PAM}}^{k} \prod_{j=\text{PAM}}^{i} \frac{\mu_j}{\sigma_j},$$

i.e.

$$a_{k+1} = a_{\text{PAM}} \cdot \left(1 + \sum_{i=\text{PAM}}^{k} \prod_{j=\text{PAM}}^{i} \frac{\mu_j}{\sigma_j}\right).$$

Finally, we can chose k above so that $(k+1) = \text{Bound}$. In particular, since $a_{\text{Bound}} = 1$, we can solve for a_{PAM} to finally obtain that

$$a_{\text{PAM}} = \left(1 + \sum_{i=\text{PAM}}^{\text{Anchor}} \prod_{j=\text{PAM}}^{i} \frac{\mu_j}{\sigma_j}\right)^{-1}.$$

In terms of the model parameters given in equations (3.1)-(3.2) and (3.3)-(3.4), we can rewrite the above absorption probability as

$$\mathrm{a_{PAM}} = \left(1 + \sum_{i=\mathrm{PAM}}^{\mathrm{Anchor}} \frac{\exp\left\{\mathrm{bias}_F + \sum_{d=i}^{Anchor} f_F(d-i)\cdot\left(2[\![state\ d{=}match]\!]-1\right)\cdot\hat{\phi}(d)\right\}}{\exp\left\{\mathrm{bias}_B + \sum_{d=PAM}^{i} f_B(i-d)\cdot\left(2[\![state\ d{=}mismatch]\!]-1\right)\cdot\hat{\phi}(d)\right\}}\right)^{-1}$$

This can then be simplified even further with a little bit of algebra, and a few additional definitions. Indeed, we find that the probability of successful R-loop formation in CRISPR-Cas9 Systems is given (in terms of free parameters) by

$$a_{\mathrm{PAM}} = \frac{1}{1 + \sum\limits_{i=\mathrm{PAM}}^{n} \exp\left\{\mathrm{bias} + \sum\limits_{d=\mathrm{PAM}}^{n} f(i-d)\cdot\phi(d)\right\}} \tag{3.6}$$

where

$$\mathrm{bias} := \mathrm{bias}_F - \mathrm{bias}_B; \tag{3.7}$$

$$f(i-d) := \begin{cases} f_F(i-d) & ,\text{for } i > d; \\ f_F(i-d) - f_B(d-i) & ,\text{for } i = d; \\ f_B(d-i) & ,\text{for } i < d; \end{cases} \tag{3.8}$$

$$\phi(d) := ([\![\text{state d} = \text{match}]\!] - [\![\text{state d} = \text{mismatch}]\!])\hat{\phi}(d) \tag{3.9}$$

3.5 Model Properties

Here, we check that our Markov model behaves as expected, namely that (i) PAM proximal mismatches should be the most impactful, (ii) consecutive mismatches should have a compounding effect, and (iii) specific pairs of bases should be associated with higher local stability. In order to briefly review whether these characteristics hold for our chain we considered the following sequences listed in Figure 3.1.

ID	Probability of R-Loop Formation	Percent Repression	Model Error
1	0.1173	0.9770	5.3094e-4
2	0.0031	0.5094	6.4000e-3
3	0.0193	0.9007	9.3075e-4
4	0.0015	0.3332	1.5000e-3
5	0.0018	0.3700	1.7857e-3

Table 3.2: Binding information for the gRNA sequences in Table 3.1.

ID	gRNA	Characteristic
1	TATTGACTTAAAGTCTAACCTATAGGATACTT	Complementary to target
2	AATTGACCTAAAGTCTAACCTATAGGATACTT	Non-consecutive mismatches
3	TATTGACTTAAAGTCTAACCTATAGGTTACTT	PAM distal mismatch
4	TATTGACTTAAAGTCTAACCTATAGGAACTGC	Consecutive mismatches
5	AATTGACTTAAAGTCTAACCTATAGGATACTT	PAM proximal mismatch

Table 3.1: Table of four gRNA sequences of length 32, with varying complementarity to the target sequence, as defined implicitly by the first sequence (ID 1). Mismatched positions have been underlined, and the cannonical PAM for these sequences is CTC.

Each of the gRNA sequences given within Table 3.1 is associated with the same target site within the bacterial genome, and each was released into the cell independently from the others. In this way each of the individual gRNA sequences may be compared to the others without any conflicting interactions. Techniques that utilize multiple guide sequences (such as paired nickases) do exist, but are not investigated here. As can be seen in Table 3.1 and Figure 3.2, PAM proximal mismatches weight more heavily the probability of R-loop formation, and thus, the probability of binding , while mismatches that do not have a smaller impact. Furthermore, PAM distal mismatches—even when occurring consecutively—have a substantial collaborative effect on the probability of binding. Additionally, we see that the chance of binding is usually much lower than that of repression. This is consistent with the high rate of attack and rejection of Cas9 (described

in Chapter 2): even though the Cas9 has a relatively small chance of binding within any given encounter, the probability of successful repression is heavily impacted by how often the complex and the target site interact.

There is one interesting thing that should be noted however that justifies the true difficulty of modeling this problem, and that is the relationship between gRNA 2, and gRNA 5. As can be seen, gRNA 5 has the same initial mismatch as gRNA 2, however gRNA 2 has an additional mismatch at position 8. What is most interesting, is that while gRNA 2 has more mismatches then gRNA 5, and the sequences are almost identical, gRNA 5 has a lower rate of binding experimentally. This is what makes this problem difficult; while heuristically the more mismatches a gRNA has, the less repression will be experienced, sometimes based upon some intermolecular stability, a sequence will act in an unexpected way. Additionally it displays why this process cannot be approximated using a linear model, as in many cases, the sequences in question will not follow the simple heuristics that describe average behavior.

Next we can test the effects of increasing the free energy coefficient associated with both canonical, and non-canonical PAM site within the model. Within CRISPR-dCas9 systems, the PAM acts a significant energy barrier for many of the off target sites, and as a result ensures that in most cases minimal off-targeting effects will occur. That is not to say that non-canonical PAM sites cannot bind, however in practice the type of PAM proves to be a major distinguishing factor for the percent repression of a given host site gRNA pair. The effects of the canonical PAM within the model is given below, in Figure 3.4 and Figure 3.5. The sequence used is the gRNA ID 2 from 4.1, and said sequence is held constant while the canonical PAM coefficient varies.

As expected, as the cannonical PAM coefficient decreases, so does the binding; however, in all cases, the PAM only defines roughly a third of the total percent repression. This affirms that our models incorporation of the PAM free energy coefficient follows experimental results given in [8, 37, 13] supporting that sites with heavy energetic constraints on the PAM, can in fact still bind with the active complex. It should be noted, however, that the PAM free energy values used in previous sources [8] might be a poor representation within our system as the results from [8]

suggest that PAMs play a much larger role in binding affinity than was observed here.

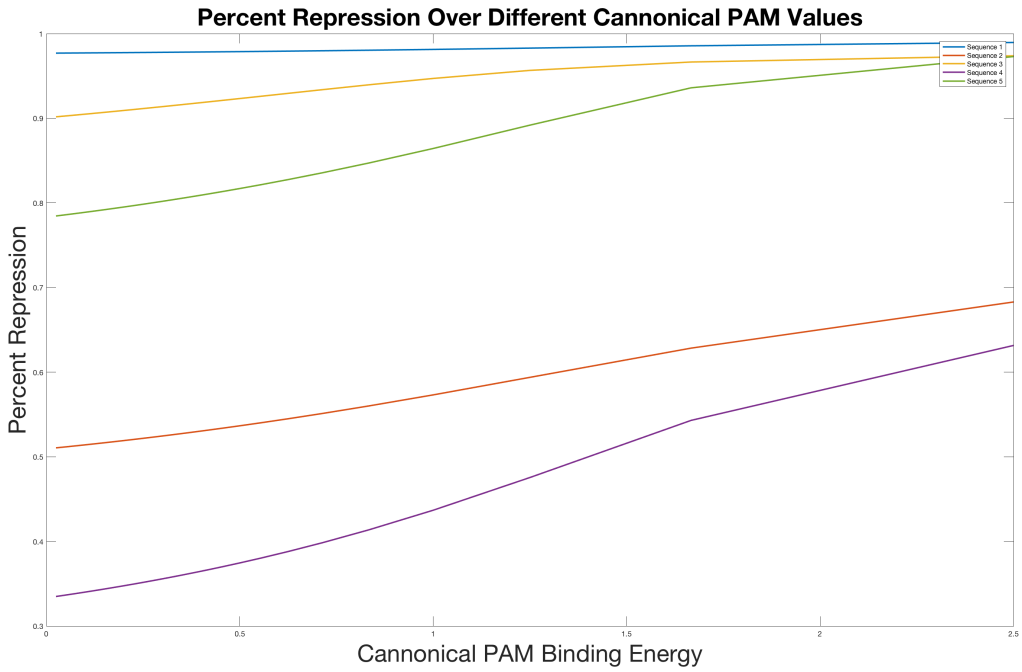

Figure 3.4: Effects of canonical PAM complementarity on the percent repression within the dCas9 binding system for each of the example sequences displayed within Figure 3.1

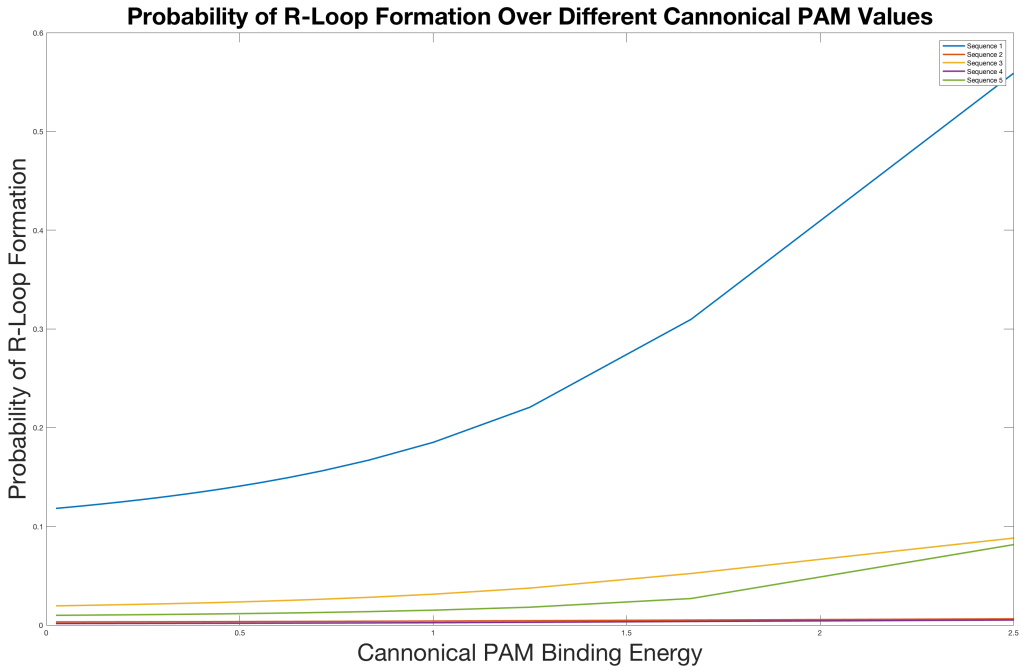

Figure 3.5: Effects of canonical PAM complementarity on the probability of successful R-loop formation within the dCas9 binding system for each of the example sequences displayed within Figure 3.1.

Chapter 4

Results and Model Analysis

This chapter addresses the fit of our Markov model to a small dataset of green fluorescent protein (GFP) data provided by the Gill Lab and performs a sensitivity analysis of the model parameters using bootstrapped data. The end of the chapter is devoted to two algorithms developed as part of this thesis to generate gRNA libraries for fitting and testing CRISPR models under a rich variety of match-mismatch configurations and off-targets.

4.1 Green Fluorescent Protein Data Set

Using the Markov chain model described in Chapter 3, coupled with the system of differential equations described in Chapter 2, we implemented a predictive model for normalized relative fluorescence measurements provided by Katia Tarasava of the Gill Lab in the Department of Chemical and Biological Engineering at the University of Colorado, Boulder. These measurements are taken from sets of cell cultures that have been inoculated with a mixed concentration of dCas9 enzyme and varying guide RNAs of varying sequences. Within each of these bacteria is a gene that corresponds to a fluorescent protein who's luminosity is easily quantifiable, and who's associated gene is known (i.e. we know the location and composition of the DNA that encodes this specific protein). Using gRNA that are complimentary to this site along the genome, we can measure how "complementary" each of the sequences are by how much of this florescent protein is suppressed. This is measured by how much the total luminosity of a given cell culture decreases with respect to an un-inoculated cell culture (i.e. the control).

The data-set provided by the Gill Lab includes three measurements of thirty experimental cell cultures, each injected with a different targeting RNA (see Table 4.1) that acted as a guide for transcriptional repression within the bacteria using dCas9. gRNA were chosen based upon experimental and time constraints such that given the small data set provided, each of the hypothesis of interest was at the very least addressed. This means that the data-set was in created so as to include a one instance of a mismatch at each position, a few instances of consecutive mismatches, one fully complimentary gRNA, and a non-complimentary gRNA (zero repression). Considerations were also take to ensure that minimal off-targeting would occur, as well as various other factors regarding the position and type of promoter that was used within the experiment. Each of these measurements is then normalized based upon controls florescence, and then normalized to be between zero and one. This implies that a measurement taken as 1 describes a sequence that has a high probability of R-loop formation, and as a result a high rate of irreversible binding at the target site. Similarly, a measurement taken as 0 describes a sequence that has a low probability of R-loop formation, and as a result a low rate of irreversible binding at the target site.

Parameters for the Markov chain were then estimated via minimization of the L_2 norm of the model output and the experimental results. For more on the algorithms used, see Appendix B, which describes the minimization problem, as well as the algorithms applied to approximate the global minimum. Finally, 10,000 iterations was chosen for the standard metric, as the model on average converged to the near optimum in practice (i.e. $\leq .05$ sum of squared errors). There were instances where the algorithm was able to find a better approximation r the standard metric, however achieving this minimum required substantially longer training time, and did not seem worth the minor improvement.

Our model predictions are displayed in Figure 4.3, where 95% confidence intervals assume normality of experimental errors. As should be expected, after training the model over the entire data-set, it works well (i.e. produces very little error), but displays one possible issue in the solution found. This issue can be seen by noticing that the model regularly under-fits the true experimental florescence. Why this occurs is not currently known, however it is possible that the bias term

ID	gRNA	Percent Repression
1	ACCACAACGGTTTCCCTCTACAAATAATTTTG	0.0000
2	TATTGACTTAAAGTCTAACCTATAGGATACTT	1.0000
3	AATTGACTTAAAGTCTAACCTATAGGATACTT	0.3718
4	TAATGACTTAAAGTCTAACCTATAGGATACTT	0.7496
5	TATGTACTTAAAGTCTAACCTATAGGATACTT	0.5588
6	TATTCACTTAAAGTCTAACCTATAGGATACTT	0.6854
7	TATTGATTTAAAGTCTAACCTATAGGATACTT	0.9758
8	AATTGACCTAAAGTCTAACCTATAGGATACTT	0.3465
9	TATTGACTCAAAGTCTAACCTATAGGATACTT	0.9482
10	TATTGACTTCAAGTCTAACCTATAGGATACTT	0.9608
11	TATTGACTTATAGTCTAACCTATAGGATACTT	0.8233
12	TATTGACTTAAGGTCTAACCTATAGGATACTT	0.9475
13	TATTGACTTAAAGCCTAACCTATAGGATACTT	0.9775
14	TATTGACTTAAAGTGTAACCTATAGGATACTT	0.9541
15	TATTGACTTAAAGTCGAACCTATAGGATACTT	0.9617
16	TATTGACTTAAAGTCTTACCTATAGGATACTT	0.9784
17	TATTGACTTAAAGTCTAACCTCTAGGATACTT	0.9456
18	TATTGACTTAAAGTCTAAACTATAGGATACTT	0.9569
19	TATTGACTTAAAGTCTAACCCATAGGATACTT	0.9684
20	TATTGACTTAAAGTCTAACCTCTAGGATACTT	0.9658
21	TATTGACTTAAAGTCTAACCTACAGGATACTT	0.8744
22	TATTGACTTAAAGTCTAACCTATTGGATACTT	0.9792
23	TATTGACTTAAAGTCTAACCTATATGATACTT	0.7343
24	TATTGACTTAAAGTCTAACCTATAAAATACTT	0.4716
25	TATTGACTTAAAGTCTAACCTATAGGTTACTT	0.7563
26	TATTGACTTAAAGTCTAACCTATAGGACACTT	0.9291
27	TATTGACTTAAAGTCTAACCTATAGGATCCTT	0.9630
28	TATTGACTTAAAGTCTAACCTATAGGATAGTT	0.9896
29	TATTGACTTAAAGTCTAACCTATAGGATACCT	0.9428
30	TATTGACTTAAAGTCTAACCTATAGGAACTGC	0.4361

Table 4.1: gRNA sequences tested by the Gill Lab with their associated Percent Repression derived from the Normalized fluorescence intensity.

over-corrects the average florescence during the optimization process as it is the most influential parameter, and then while the model is fine tuning the distance parameters we are left in a state where florescence is slightly under-predicted. This behavior does display why the bias parameter is so crucial to correct prediction within the model, and how small perturbations in it can drastically change model error. In the future, upon completion of the global optimization process, it might be best to optimize over subsets of the parameters. This proved to be a useful approach in earlier

iterations of the model, and if this was done for the case given in 4.3, we could likely avoid the "under-prediction" observed.

We can also look at squared relative errors (in this context, squared error divided by the sample mean, over the entire data set, of the normalized relative fluorescence) that are displayed in Figure 4.4. From this plot, we can examine the two largest errors. These these are associated with the gRNAs with ID 11 and 21, namely

TATTGACTTCAAGTCTAACCTATAGGATACTT

TATTGACTTAAAGTCTAACCTCTAGGATACTT

where again the underlined nucleotide pairs represent mismatches along the sequence. The fact that both of these mismatches include a C might be indicative of the program having difficulty predicting sequences of this mismatch type. However it is more likely from consideration of the other sequences (which also include a C-mismatch but do not have a significant error) that this inaccuracy is simply due to a lack of training iterations. That is, given a higher maximum iterations cap, it is likely that these approximations would settle into the true experimental values.

Additionally, we then performed ten fold cross validation on the data, holding out 3 out of the thirty data points within the full set of averages, and then running the global optimization algorithm (explained in greater detail in Figure B) for 10,000 iterations. All parameters were initialized uniformly at random between -1 and 1. The average error on the held out data points was ≈ 0.1295, plots displaying the error for the individual folds is displayed in Figure 4.1, which displays box plots for the errors, as well as bar plots for the average error at each data point.

Given that the high number of parameters within the model relative to number of data-points used in training, this result does prove that the model is robust to variations in the data. Whats more, some of the folds within the cross validation excluded instances at positions along the genome. That is to say, the training data included no longer even included a mismatch at every possible position. Next we can look at how the parameters vary to see how stable the solution found for each of these problems is under variations within the data.

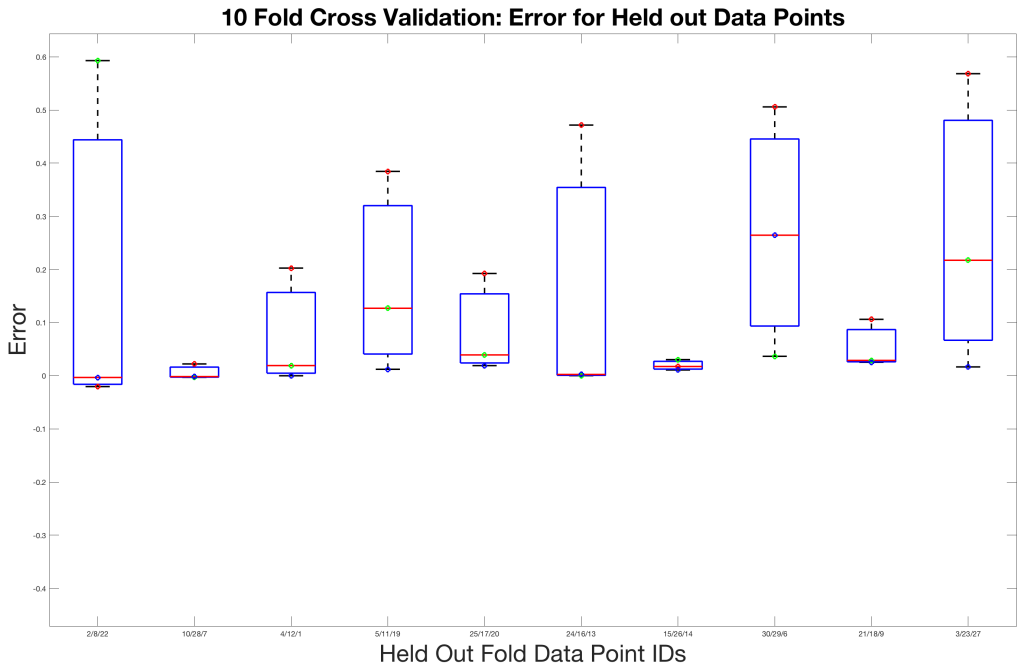

Figure 4.1: Box Plot for error in held out data points. Data points within folds are indicated by "data point 1/ data point 2 / data point 3"; where the red dot indicates the first data point, the green indicates the second, and the blue indicates the third.

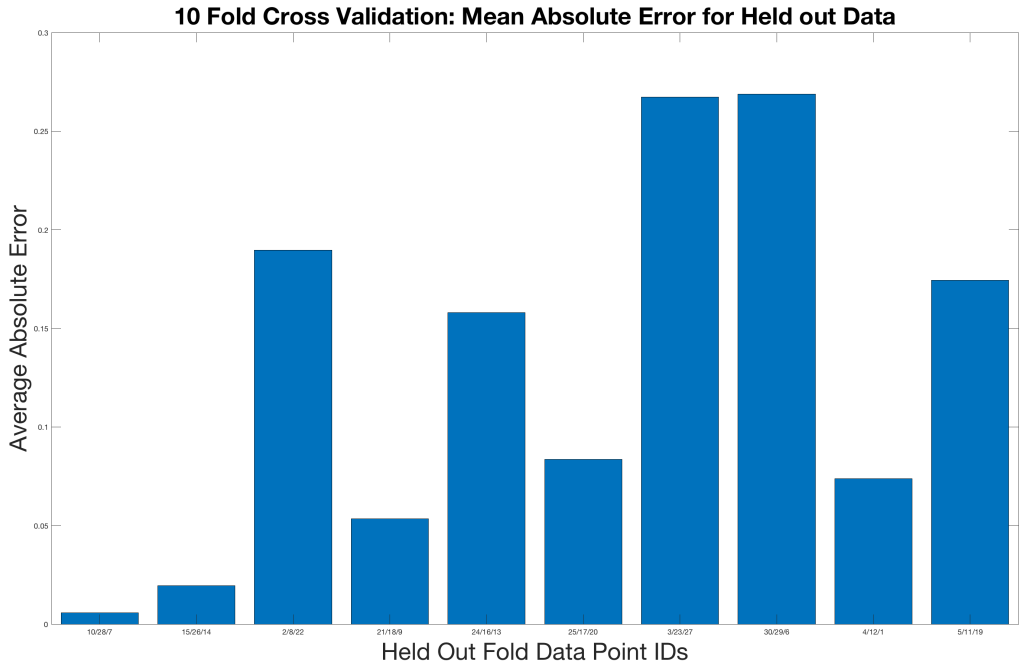

Figure 4.2: Bar graph of average absolute error for each fold generated during ten fold cross validation. Data points within fold are indicated by "data point 1/ data point 2 / data point 3".

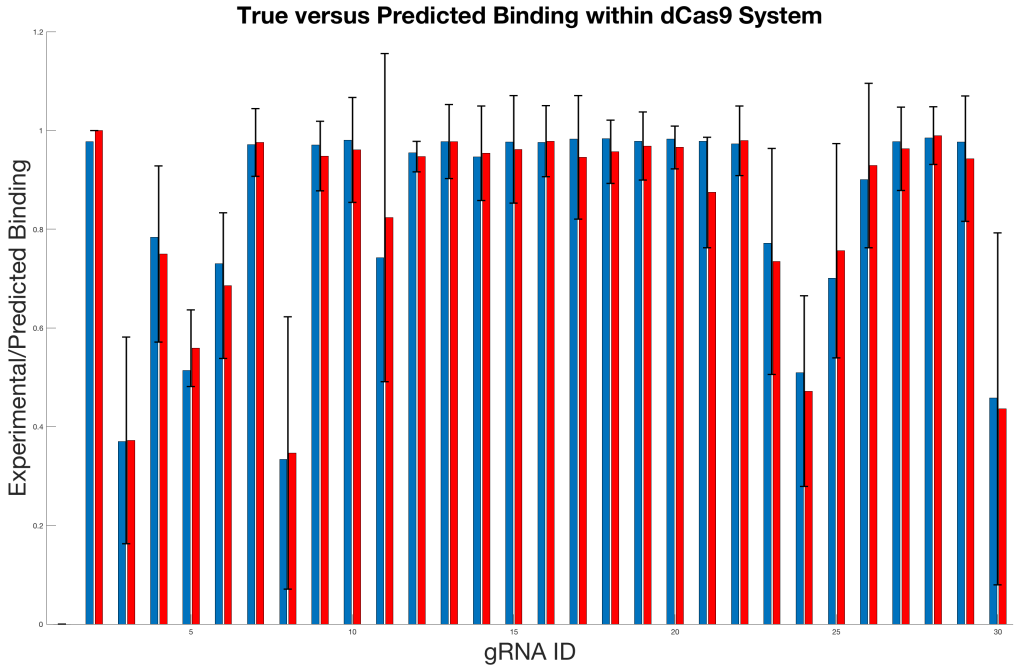

Figure 4.3: Model predictions after $10,000$ iterations of training on the full data set The red bars indicate the experimental values, blue bars the model prediction, and black segments represent 95% confidence intervals.

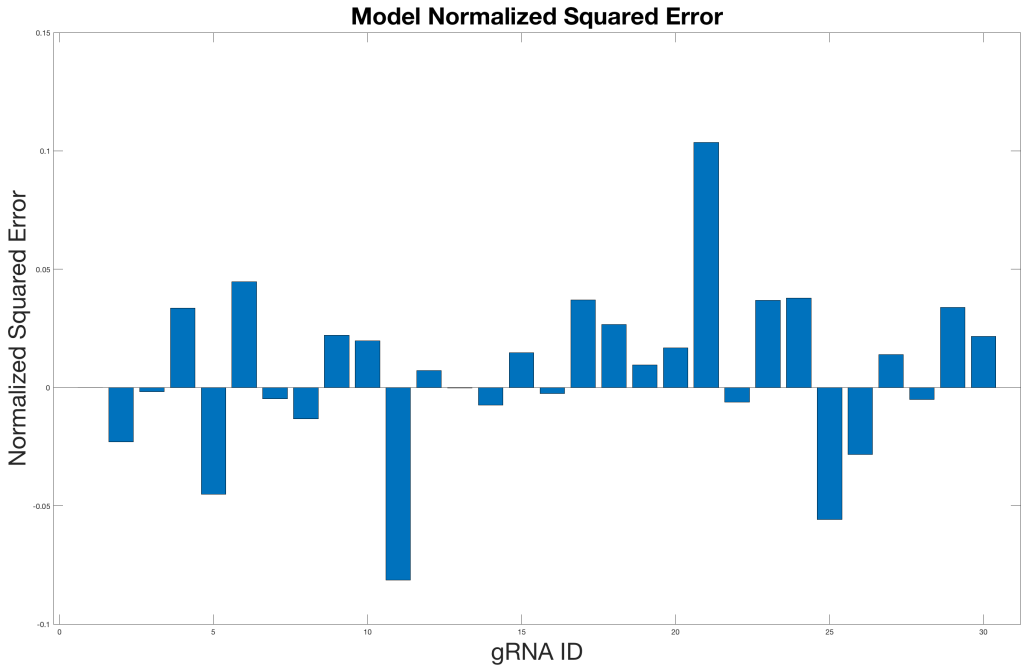

Figure 4.4: Normalized squared error (i.e. squared error divided by the mean of the training data set) for each gRNA in Table 4.1 after $10,000$ training iterations.

4.2 Parameter Sensitivity Analysis

We next performed sensitivity analysis on our parameters. In order to do this, we added normally distributed error to each data point based upon the mean and standard deviation at that data point. In order to estimate the variance of the noise at each data point, we only had access to a minimum, median, and maximum value (as the data set provided by the Gill lab had only three realizations). We can estimate the variance of a sample by noting that its range should be approximately approximately four times the standard deviation. This metric was used because the true range for the sample is well known, and further, because constraints exist on the distribution (i.e. all values fall between zero and one), by using the sample variance we substantially overestimate the possible variance for each of the data realizations, and in doing so throw the parameters to values that would unlikely exist within true data. Ideally, we would want to come up with a more realistic distribution for each of these data, however as we are really only attempting to investigate how the parameters vary when the data is perturbed it was not investigated in detail here.

Using this approximated value, Figure 4.5 displays 100 different realizations of the data, while Figure 4.6 give trained distance parameters under this new data. These plots suggest that our approximation of distance parameters (f(i-d)) are adequately robust (i.e. all realizations seem to follow the same basic trend with minor variation around the mean value). What is unexpected, is that all of the forward distance parameters seem to be significantly less stable then the reverse. This could be a result of the optimization algorithm arbitrarily favoring these parameters over reverse, as all that is actually required to calculate the probability of absorption is the ratio of these values. However it is also likely that each is varying at the same magnitude relative to the scale of the parameter itself. This is reinforced by the fact the many of the backwards distance parameters are themselves extremely small compared forward parameters.

Indeed, upon looking at the coefficient of variation we see that the parameters seem to vary somewhat uniformly. This information along with the results from cross validation suggest that the local distance model chosen is also robust to small variations within the data, and may be extended

to laboratory situations that are carried out within similar environments to the data provided by the Gill Lab. Unfortunately, simulations at a larger scale (1000 + data realizations) are somewhat inhibited by the models lengthy training time. It should also be noted that for numerical stability, transition probabilities at the PAM state and the anchor state were enforced due to numerical instability at these locations. The parameter sensitivity analysis is omitted here however, as it follows the same trends discussed regarding the distance parameters given above.

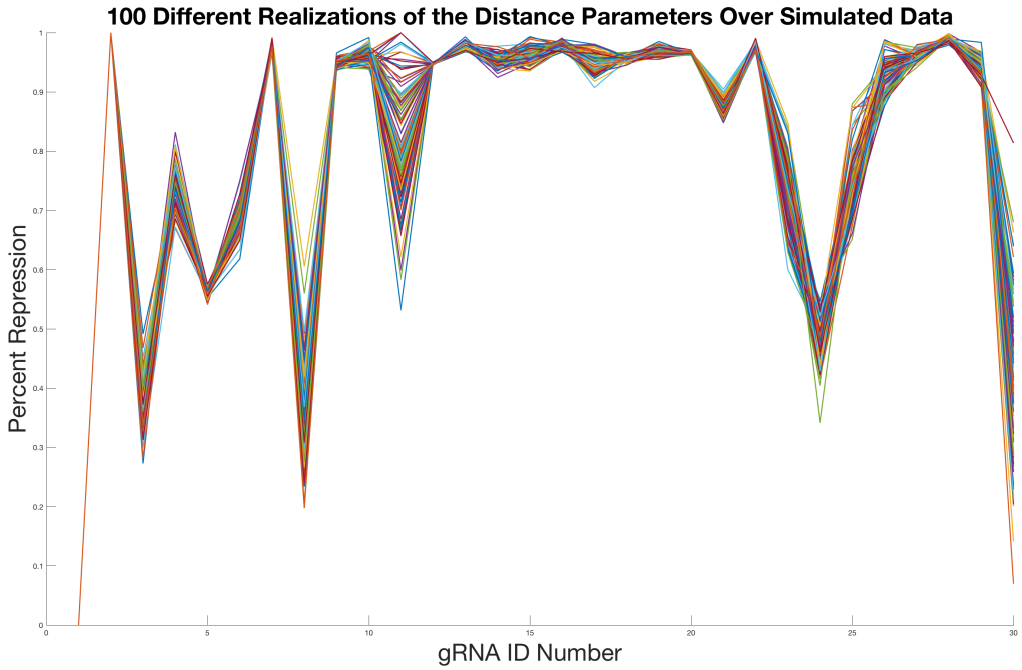

Figure 4.5: Simulated fluorescent data using Gaussian noise.

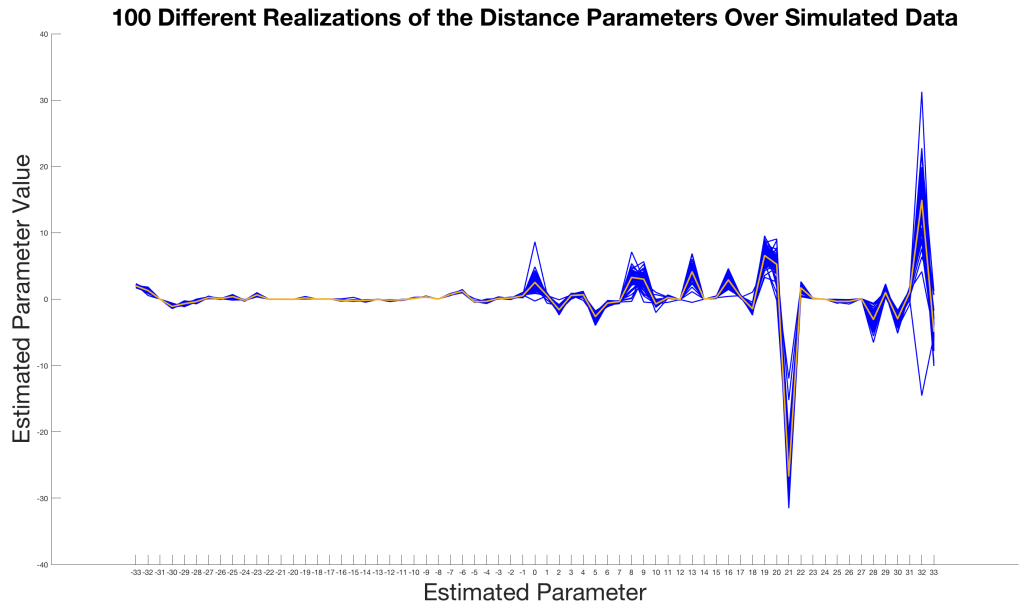

Figure 4.6: Trained Distance Parameters from Simulated Data

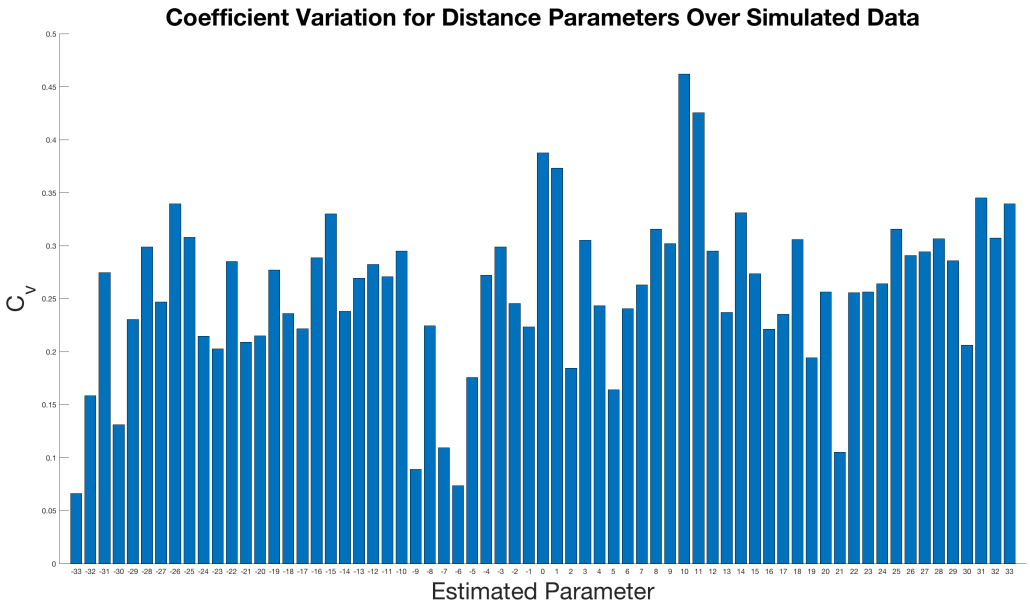

Figure 4.7: Coefficient of Variation for distance parameters

4.3 Sequence Generation

The principal reason that this model was created was to produce a reliable method of sequence generation within future experiments, to improve the economy of CRISPR-Cas9 experimentation further. Therefore in order to begin applying the method described above we began collaboration with the Gill Lab at CU boulder and undertook the generation of sequences that would allow us to investigate many of the important aspects of CRISPR-dCas9 described in earlier sections.

Based upon various hypothesis that we were interesting in investigating in order to better understand sequence dependence (including position dependence, nucleotide composition, the collaborative effect of multiple mismatches, and finally the effects of different Canonical PAMs), we set up a sequence generation algorithm to create a experimental sequences to test these hypothesis. Additionally we added in programs that probabilistically sampled from larger sets of sequences in order to get subsets of sequences with specific characteristics, or distributions of characteristics. For instance if we wanted the data set to consist of a set of sequences where 50% had PAM proximal mismatches, and 50% did not, we set up a program which would sample from the random sequences generated, and achieve a new subset with this characteristic.

The core distribution that we sample from was based off of the desired distribution of mismatch number within the data set. In this case, to ensure that we could produce the desired percentage of sequences with specific numbers of mismatches, we first generated random permutations from binary sequences of a length equal to the number of desired mismatches plus one. Where the reason that an extra binary value was added, was to ensure that the sequence could always have the possibility of being flanked matches on either side. This also led the sequences generate to slightly favor clustering for sets of mismatches with longer spans (the distance between the first and last mismatch), allowing us to garner more information about the collaborative effect of multiple mismatches.

Using this method, we could select precisely the percentage of the sequences that would have a specific number of mismatches, while including a range of different match-mismatch configurations.

Primary Targeting Sequence Generation Algorithm

Inputs:	Bacterial Genome, number of gRNA sequences, number of sequences per number of mismatches
Step 1:	Generate random binary sequences encoding positions of spans of mismatches for every number of mismatches specified by the input
Step 2:	Iterate through binary sequences and generate uniform random partitions for each
Step 3:	Assign mismatches at the corresponding positions uniformly at random (selecting between A,C,T,G)
Step 4:	Apply trained model, or score function, to each sequence based upon the PAM and generated targeting sequence
Step 5:	Sample sequences based upon desired complimentary distributions, and other factors such as the number of PAM proximal mismatches, PAM motif, position of first mismatch, etc

Table 4.2: Algorithm describing procedural generation of targeting gRNA sequences for hypothesis testing in CRISPR experimentation.

This procedure was then applied to sequences with up to 10 mismatches (the threshold after which repression would no longer be possible for the majority of match-mismatch configurations, as can be seen in 4.8).

The process of generating sequences with a comparatively high chance of off-targeting was much more computationally intensive then the algorithm described above (the full algorithm for off target generation is described concisely within 4.3). To find sequences with the possibility of high off targeting, we iterated through the proposal sequences already generated via 4.2, for each committing to a genome wide screen of possible sites that had a high complimentary to the gRNA sequence considered.

This was done by brute force using sliding windows, and in general considered each of the roughly 65,000 sequences with an accepted PAM site (i.e. a site where the specific type of dCas9 being used was known to bind with). Due to how small the bacterial genome is, this approach was still tractable and required a much more straightforward implementation than other viable methods. Furthermore, it was easily reformulated as a series of parallel processes, which drastically sped up the genome-wide search.

High off-targeting Sequence Generation Algorithm

Inputs:	Host Genome, list of current proposal sequences, number of desired high off-targeting sequences
Step 1:	Partition sequences by score (defined in previous algorithm)
Step 2:	(In Parallel) Iterate through sequences within each partition
Step 2a:	For each sequence, search genome for n closest matches (sequence composition and score function)
Step 2b:	Sort chosen off-targets and pick k most complimentary
Step 3:	Sample off-targets based upon desired complimentarity distribution
Step 4:	Based upon desired percentage of high off-targeting sequences, integrate new sequences into the data set

Table 4.3: Algorithm describing procedural generation of targeting gRNA sequences with high off-targeting effects for within the data set generated for the Gill Lab at the University of Colorado, Boulder.

As displayed in 4.3 Determination of off-target sites was done by ordering a set of 5,000 sequences per gRNA based upon a naive sigmoidal scoring function or the probability calculation given by the Markov Chain. After one full sweep was returned, we would then sample from this set of sequences, replace randomly distributed values within the the sampled off target sequence to make it more complimentary to the original target site, and store the new gRNA as a high off-targeting sequence.

The reason that the sequences that were found had to be augmented was the result of the extremely low average complimentary between the gRNA and other sites along the genome. As can be seen by a Figure 4.8, in general, proposal sequences did not have any places on the genome where other sites had less than 12-14 mismatches. Furthermore, the threshold (i.e. the sequence with zero percent repression) for a host to have the possibility of binding to the gRNA that was exhibited by the fluorescence measurements provided by the Gill Lab indicated that greater complimentarity at the off target site would be needed to affect the rate of repression on the primary site.

Using these sequences we can then supplement the original data set produced with the algorithm in 4.2, in order to ensure that on top of the other hypothesis we could also include the affects of off targeting. Ideally this information will allow the Gill lab to see what the effects of off

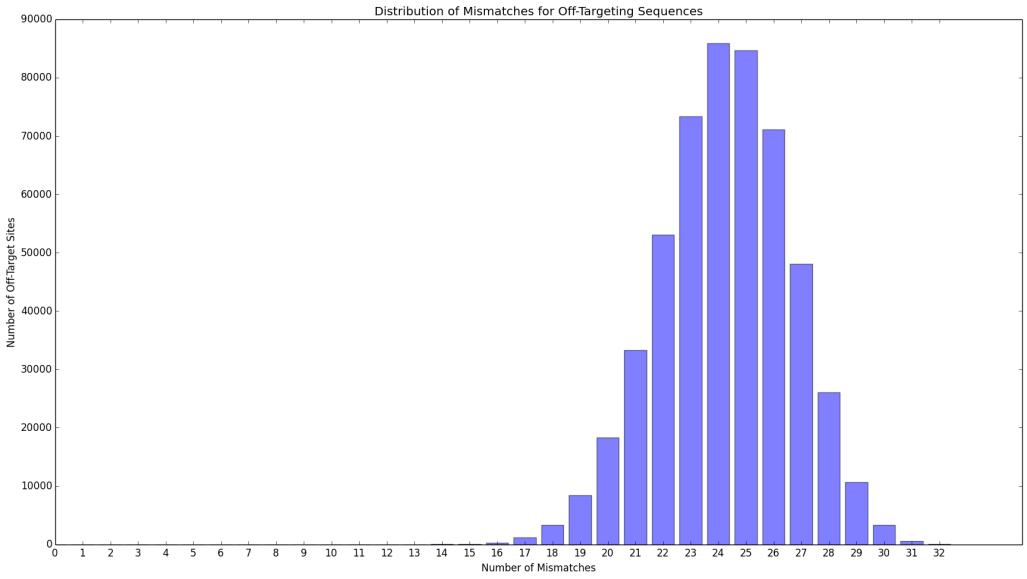

Figure 4.8: One realization of the distribution of mismatch number within a dCas9 system provided a given gRNA (in this case TATTGACTTAAAGTCTAACCTATAGGATACTT) with a strain of Escherichia coli bacteria. The smallest number of mismatches for an off-target site is 13 and occurs along its genome twice.

targeting look like, so they will know what to look like in the future. In the end we partitioned ten percent of our final data set to off target testing, and ninety percent to the general sequences created by the sampling approach described 4.2.

The final step of the whole process was picking which sequence characteristics were to be considered, and what the desired distributions of these characteristics were. For this stage we worked closely with Katia Tarasava of the Gill lab who gave us a list of the properties that should be included, and what types of sequences we wanted based upon these characteristics. Some of the characteristics that were considered were: sum of pairwise ΔG energies, the distance between the first and last mismatch (span), whether or not the sequence had PAM proximal mismatches, the position of the first mismatch, the total number of mismatches, whether more mismatches where PAM proximal or PAM distal, position along the genome, percent GC content, PAM motif, complementary score, and copy number. Some of these distributions are displayed in Figure 4.9.

Finally, we also made sure to include all possible combinations of one and two mismatches within our data set, to better understand local interactions between short runs of mismatches which should hopefully allow the algorithm to better approximate the individual effects of adding a mismatch, or two to a given sequence. Unfortunately by 3 mismatches, the number of all possible combinations greatly exceeded the number of sequences that were being generated, and so these, as well as all other types of on-target sequences were generated via 4.9.

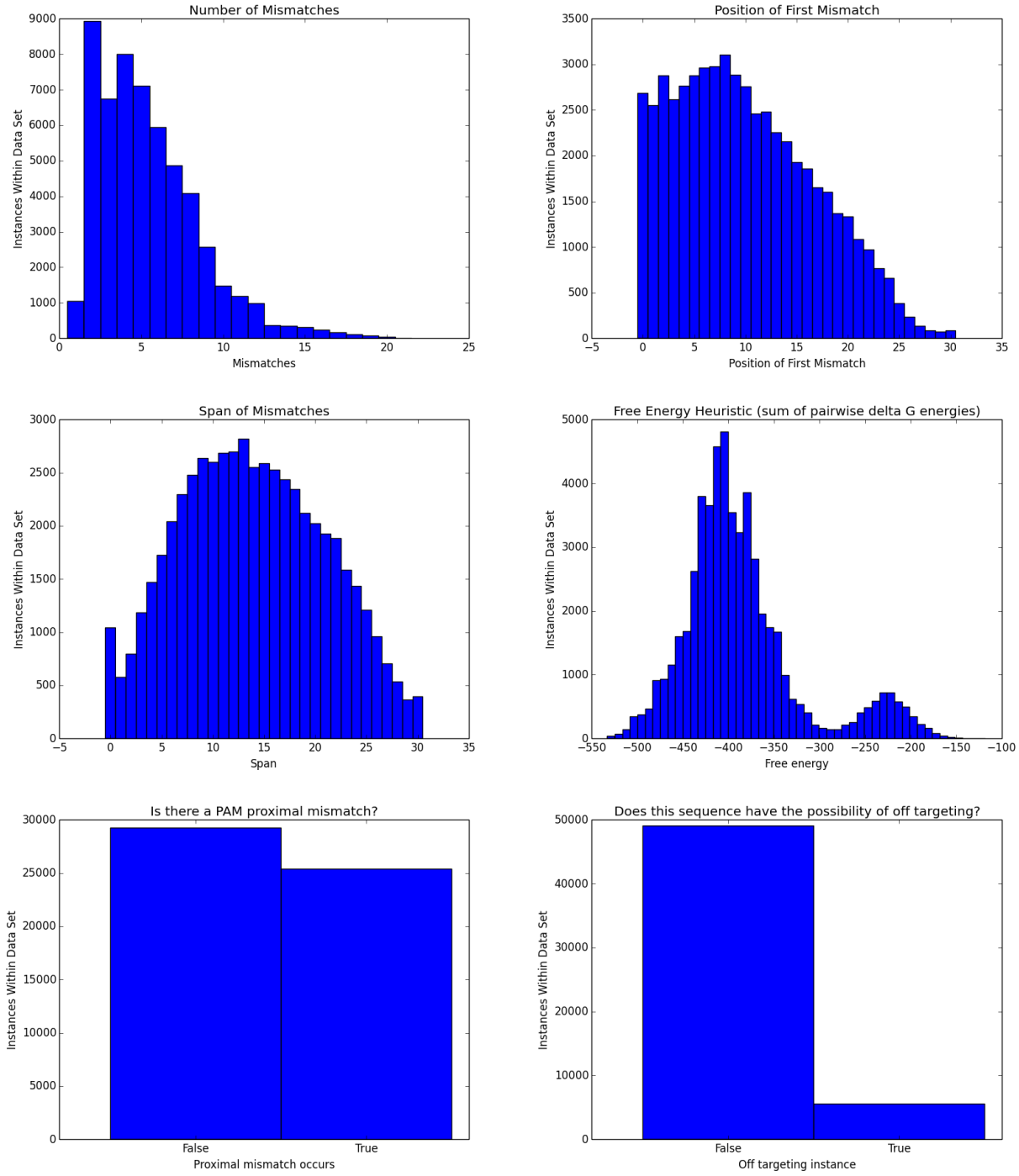

Figure 4.9: Examples of chosen sampled distributions for experimental data.

Chapter 5

Incorporation into a Standard Machine Learning Model

The final topic of this thesis is a short discussion considering how the Markov chain model might be extended to the more complex human-Cas9 system. First, we describe the principal differences between this system and that of the bacteria-dCas9. Then we look at a few off the shelf machine learning approaches that can be utilized to model the new system, which include: multiple regression, ridge regression, lasso regression, and random forests. Finally, we make a few minor modifications to our Markov chain formulation and see how well it performs when compared to these more classical approaches. Based upon these results, we comment on the drawbacks, as well as potential improvements. The dataset used in this chapter comes from reference [1].

5.1 Overview of a Different Biological System

In this chapter, we consider a human-Cas9 system (Recall that Cas9 cleaves irreversibly the site it binds to). This is very different from the bacterial-dCas9 system we have been addressing this far in three crucial ways: the first being a far larger genome, the second being a shorter target sequence (20 basepairs instead of 32), and finally the presence of chromatin structure. For reference, chromatin is a DNA superstructure that occurs within many eukaryotic cells, which condenses and protects genetic information in DNA.

Unfortunately, each of these seemingly innocuous modifications has enormous effects on how one can model this new system. For instance: a longer genome and a shorter targeting sequence means that off-targeting will be significantly more prominent. In terms of our "lock and key"

metaphor, in the human-Cas9 system we have a much less specific key and a great deal more locks. Further, due to the presence of chromatin, even if the targeting sequence is perfectly complementary, there is no guarantee that binding will occur as the chromatin "packaging" of DNA could negate all access to the site in question. All of these additions combined makes modeling this process a significantly harder problem. In fact, predicting chromatin structure is itself currently an active area of research. To approach this problem, a less biophysical and instead more qualitative view [1, 37, 35] may therefore be needed. (For an instance of the biophysical model applied to the human-Cas9 system, see [8].)

In what follows, we explore a statistical learning approach, as these modeling techniques perform at a much higher prediction fidelity than biophysical approaches when taking into account more information in the form of covariates. As an example data set, we consider data provided in reference [1] which contains 73 covariates (with a mixture of numeric and categorical variables). While we do not beat the algorithm presented in [1], we look at how off the shelf algorithms can be supplemented with information from the Markov chain.

5.2 A Few Classic Machine Learning Approaches

The conventional first step in most statistical modeling approaches is to apply multiple linear regression. In order to do this, we first must reformat our covariates into a design matrix that is well defined. In the case of the data set presented in [1], we have a mixture of numeric, and categorical variables. Therefore, to perform multiple regression, we first renormalized each numerical covariates by subtracting its mean and then dividing by the largest absolute value of the centered covariate (within a given column of the design matrix). By doing this we ensure that each of the columns within the design matrix used within our regression is on the same order of magnitude (i.e. between -1 and 1), and as a result ensure that no numerical instability ensues due to poorly scaled variables.

Next, because the dataset also includes categorical information, we create dummy variables in order to encode them numerically into the design matrix. This means that for every unique instance of a categorical variable, we create a an additional column the determines whether or not

the data point contains this unique instance of the categorical variable (encoded as binary zero-one values). This procedure is called one-hot-encoding. There are certainly other available methods, but for more information as to why this method is popular, see [3].

After the normalization process, we can apply multiple regression. However, since after creation of the dummy variables we have roughly 200 covariates within our design matrix, it is common practice to regularize the regression coefficients. The two most popular regularization methods are Ridge and Lasso regression. Each of these relies on the addition of a normalization term, consisting of a hyper-parameter , multiplied by either the one-norm (in the case of Ridge), or the two-norm (in the case of Lasso). Where the magnitude of the hyper-parameter affects how the regression coefficients are penalized. In this way the model is biased towards including less, or at least smaller magnitude coefficients. This can be used in feature selection, or to improve model robustness and accuracy via internal cross validation of the training data.

This leads to another question regarding how the hyper-parameter should be picked. In the case of the plots in Figures 5.1-5.4, we chose the hyper-parameter for Ridge regression based upon maximization of the AICC (the small-sample-size corrected version of Akaike information criterion), and the hyper-parameter for Lasso regression based upon minimization of the MSE in 10 fold cross-validation. For further documentation and explanation associated with these approaches, see [25, 24, 26].

We also implemented a random forest approach using Matlab ®'s built-in "Bootstrap-aggregated (bagged) decision trees" algorithm. Unlike the somewhat simpler methods described above, the random forest model works on the principle that a single model, especially a decision tree, will tend to overfit. Therefore, to get better predictions, one can aggregate a broad set of models and average their predictions. Hyper-parameters for this model were chosen base upon out-of-bag (OOB) error. For a more involved discussion on this type of random forest approach, as well as an in-depth discussion on the background theory and implementation, see [15].

Each of the methods described was applied to four different data-sets provided within [1]. Figures 5.1-5.4 display the leave-one-out performances of the methods, including their corresponding

R-squared and Spearman's correlation coefficient. While each of the different methods does well with regards to the established methods given in [1], it should be readily apparent that achieving further accuracy will require a much more refined approach. In either case, the off the shelf methods stack up quite well to the method described in [1].

For comparison, the R-squared for the bagged decision tree applied to the BLESS dataset was roughly 0.85 versus the CRISTA algorithm in [1], which gives an R-squared of about 0.9.

5.3 Markov Chain Based Regression

In order for the Markov chain to utilize the information provided by the data set in [1], we made the following changes to the original formulation of the Markov chain model. First and foremost, we no longer include the full system of differential equations given in 2.2 as the new data set no longer explicitly provides information regarding off-target sites. Additionally, as the human genome is significantly longer, and contains structures that would make determining whether an off-target site could be complimentary any significant drive to include the information is not investigated within this thesis. Additionally, because of the significantly larger data set, there is a necessity for the objective function evaluations to be more efficient for cross-validation. Secondly, we allowed both the complimentary, distance and bias parameters to vary (i.e. treat them as trainable parameters instead of known free-energy values). This drastically improves the models ability to adapt to larger classes of sequences and more meaningfully encode the information therein. And lastly, as the new data set gives access to multiple measures of super-coiling, we combined our bias term with these parameters to better account for the effects of local super-coiling on the active complex during R-loop formation.

Unfortunately, the Markov chain prediction, when compared to the classical methods, by itself does not produce any earth-shattering results. In fact, for some of the data sets it has terrible predictive accuracy. For the data sets where it does perform genuinely poorly (such as the guideseq data set), there are likely issues with over-fitting, or simply a lack of information that is occurring. Indeed, upon looking at the Spearman coefficient of the training data, we have a value that is on

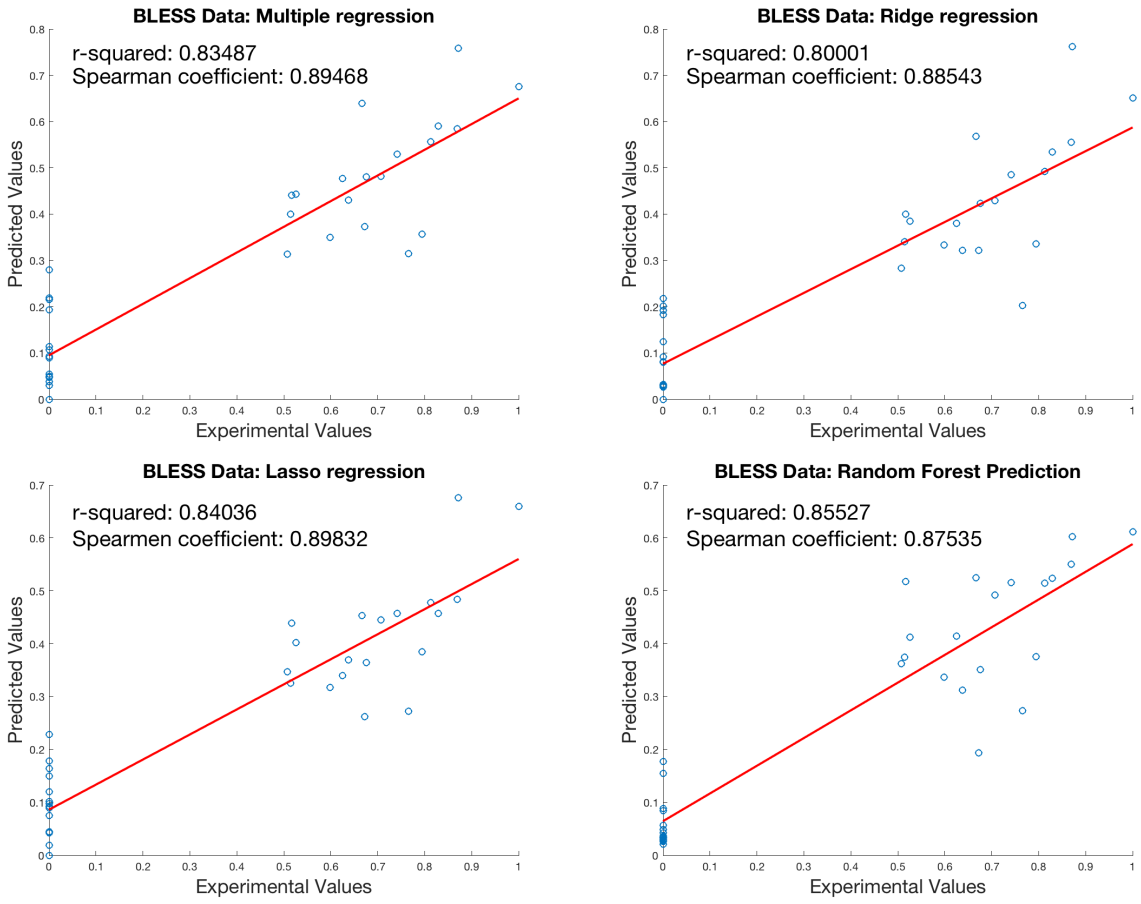

Figure 5.1: Results of leave one study out cross-validation for the BLESS from data-set provided by [1] using Multiple, Ridge, and Lasso regression, and Bagged Decision Trees.

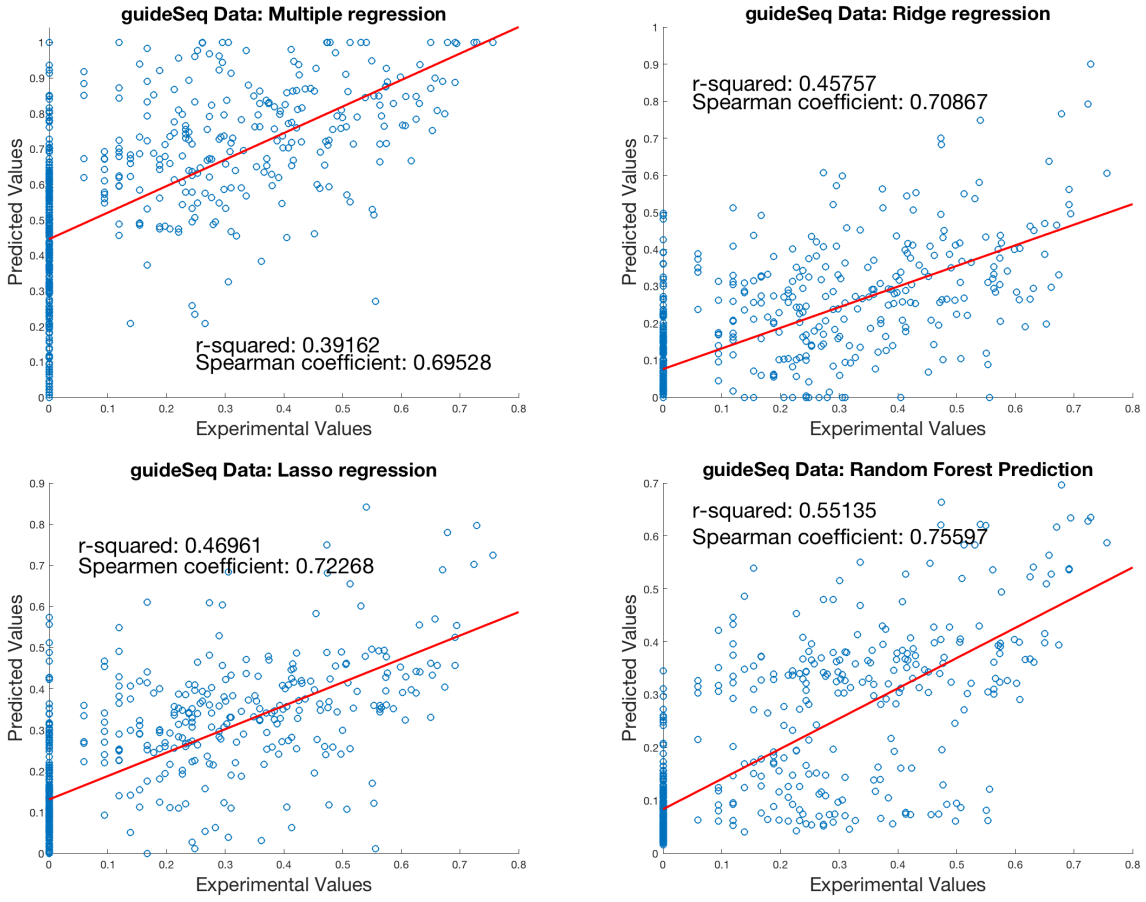

Figure 5.2: Results of leave one study out cross-validation for the guideSeq from data-set provided by [1] using Multiple, Ridge, and Lasso regression, and Bagged Decision Trees.

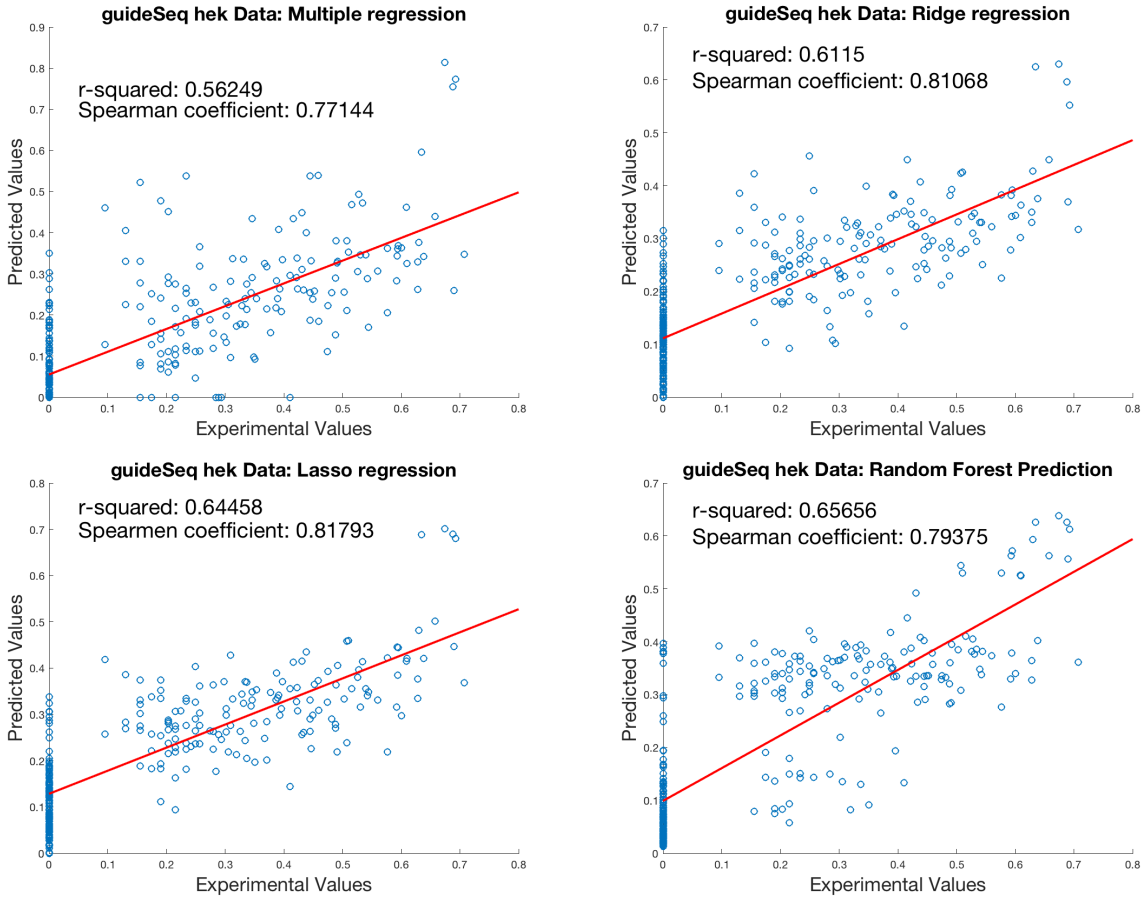

Figure 5.3: Results of leave one study out cross-validation for the guideSeq hek from data-set provided by [1] using Multiple, Ridge, and Lasso regression, and Bagged Decision Trees.

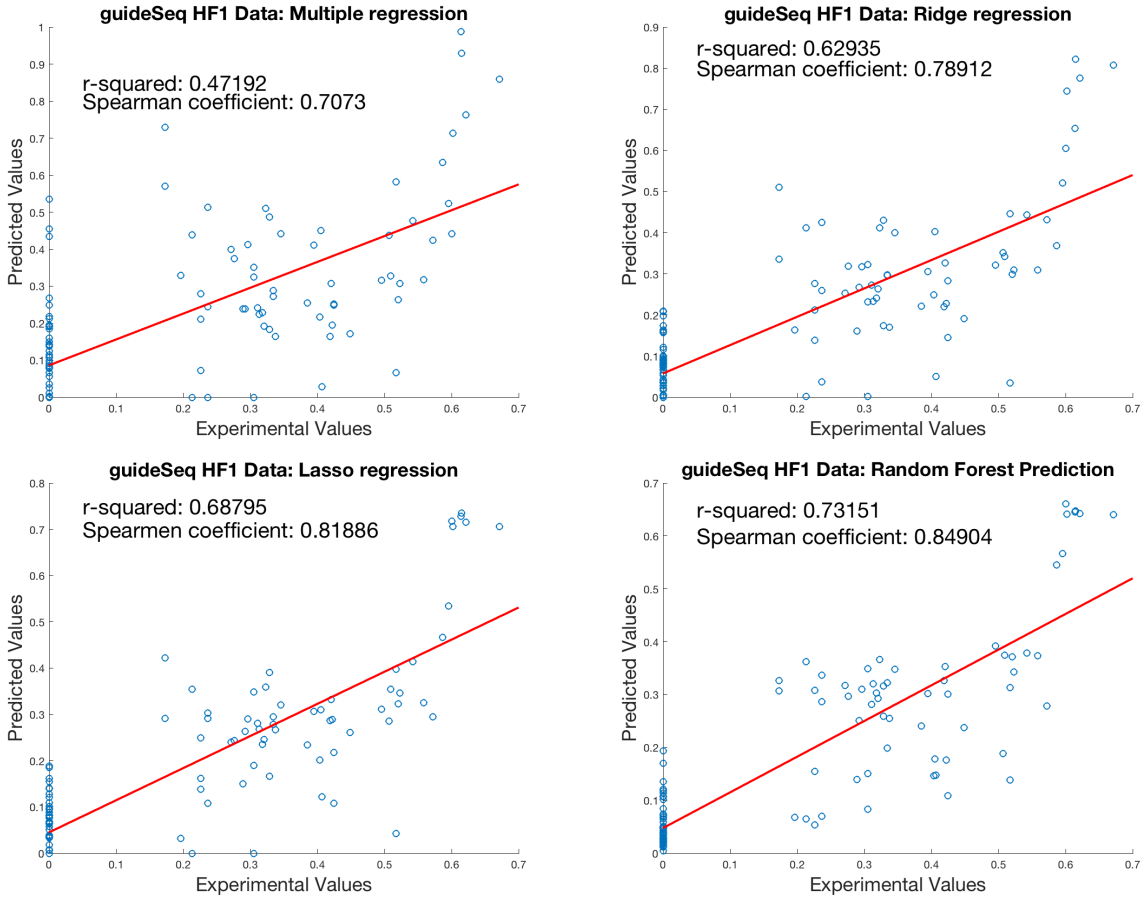

Figure 5.4: Results of leave one study out cross-validation for the guideSeq HF1 from data-set provided by [1] using Multiple, Ridge, and Lasso regression, and Bagged Decision Trees.

par with the other models at a value of about .76, however upon being applied to the test data, this drops to roughly .32. This indicates that the model needs more data then the training set that was provided for it to accurately extrapolate to new sequences of information.

More importantly however, we are no longer in the realm of the bacterial genome, and so there are a plethora of other offending constraints that could also be diminishing the Markov chains predictive power. Indeed, while the Markov chain model does encode the R-loop formation event, there is too much complexity in the human-Cas9 system expect this single aspect to encode the full process. Ideally this can hopefully be utilized within a more extensive hierarchical model in the future that describes the individual aspects of the underlying dynamics; for now, however, we can naively combine the Markov chain model with the the regularized approaches mentioned earlier.

5.4 Incorporation into a Machine Learning Model

To include information condensed from our Markov chain model into a larger machine learning ensemble, we need a model architecture that is robust to over-fitting. Above we showed a few methods above that display these attributes, and we will investigate in general how these methods can be improved by the condensed sequential information provided by the Markov chain. Full implementation of this approach is not included in this thesis, however, the code as for this project can be found at [19].

Initially, we attempted to add the Markov chain outputs as a covariate in a larger regression or random forest model, however, this proved futile as the chain tended to predict the training information well enough to skew the results of the decision tree. Instead, upon training the Markov chain, we included trained model parameters such as the PAM, and mismatch parameters (at each position). Doing this in general supplements the original off the shelf algorithm with relevant covariates that seemingly improve model performance on average, even getting pretty close to the results given in [1].

The output for the Markov chain itself was not chosen because, by including a covariate that encodes an unfair portion of the variable we are trying to predict, we may subject the output of

the decision tree to a higher variance. (Here, the variance is taken in the machine learning context and implies over-fitting.) This is because ensemble and boosting algorithms tend to work much better when the models that are being averaged are weakly correlated. In the context of other model averaging algorithms such as Adaptive Boosting (AdaBoost), this leads to an interesting phenomena known as "strength in weak learnability" (for an in-depth explanation see [31]).

In both the regression models, as well as the bagged decision trees we want each of the individual models in the ensemble to be very weakly correlated with each other. However, if one of the covariates has "seen" all the data already, we might implicitly create correlations between our sub-models. Alternatively, if one of the covariates can predict experimental values more easily then others, it will be picked more often as a branch point within the decision trees, again creating correlation between the sub models. The bagged decision approach does mitigate this by picking randomized subsets of our covariates when making the sub-models, but the possibility of sub-optimal behavior remains, and in fact persists in the human-Cas9 system.

Including the actual outputs of the chain within an ensemble method similar to what was described above might be possible. However, such an approach would require the retraining of the Markov chain each time data was bootstrapped and used to create a submodel. This approach would, however, lead to issues regarding the optimization of the Markov chain. This is because, as was discussed in chapter 3, the minimization problem defined by the analytic solution to the Markov chain is not known to be concave nor convex without an additional restriction (and in practice has exhibited multiple local extrema). This means that at each step in the bagging algorithm we would need to approximate the solution of a global minimization problem. Because of the rapidity of achieving a local minimum, this approach would be tractable, and even parallelizable; however, instead of outputting a new model prediction within a few minutes, the algorithm would likely require a few hours to run.

A simple, yet less naive approach is to apply bagging to the Markov chains themselves. Similar to the bootstrapped aggregation used in the decision tree approach, we can sample data with replacement, and then train the Markov chains on these bootstrapped samples. After the

Markov chains have been trained, we can take the mean prediction of each of the chains as its own prediction. In practice, this regularizes the Markov chain predictions in such a way as to not well predict the response variable, but generates a new covariate that well approximates complimentary of a given sequence. While the implementation here is a bit ad-hoc, it does show the possibility for this model structure to sequester additional information that was not previously available. It should be noted that in general, addition of these "Markov chain covariates" only improves accuracy within models with a proper form of regularization. An example of these covariates being added to the original design matrix within Lasso regression is given in Figure 5.5, however it also displayed significant improvement within the random forest model, PCA regression and IRLS regression. There are various pitfalls associated with this methodology, but in general (over the methods considered), it has the potential to be a useful tool in the improvement of a statistical model for the accurate prediction of cleavage in CRISPR-Cas. For a more grounded explanation of this approach, as well as a full implementation of all the methods described above, please see the GitHub repository associated with this project [19].

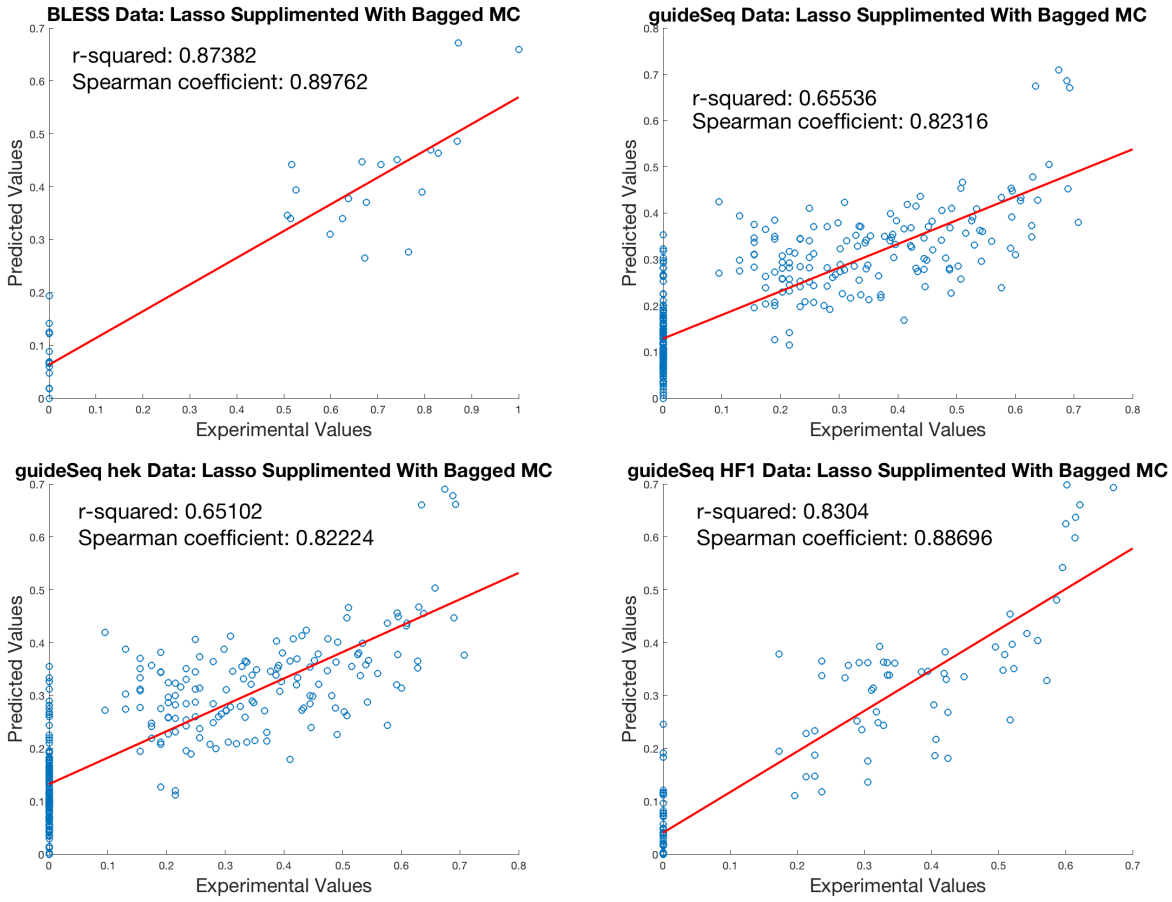

Figure 5.5: Results of leave one study out cross-validation using Lasso regression where the design matrix is supplemented by the Markov chain covariates.

Chapter 6

Conclusions and Future Work

Throughout this investigation, we have constructed a model architecture that is both physically interpretable and easy to implement. Based upon a straightforward derivation of absorption probabilities in Gambler's Ruin chains, we were able to model the highly complex process of R-loop formation in bacterial-dCas9 systems. We then incorporated this probabilistic formulation into a cell wide model adapted from [8]. Additionally, we have proved via global optimization that approximating the global minimum of the objective function defined within Chapter 3 is tractable and seems to lead to robust predictions of the data set provided by Katia Tarasava and the Gill Lab. Using this model metric, we then applied it to generate gRNA sequences for experiments being carried out within the Gill Lab at the University of Colorado, Boulder.

We have also shown that our Markov chain model may be adapted to describe CRISPR-Cas9 systems. Within these systems, the original approach does need to be modified. However, these modifications do not degrade model interpretability substantially. We did see that the heavy reliance of our model on sequence composition can skew results unfavorably, however, by incorporating our model into larger machine learning frameworks, it is possible to improve model performance.

Most importantly, we described one possible approach to leveraging Markov chain models without the presence of state transition information, creating a sort of "Markov chain regression" model. While there were drawbacks to the methodology, we did show that it was, in fact, possible to achieve adequate solutions when certain assumptions about the functional form of the transition probabilities were made (i.e. the formulation of the forward and backward transition probabilities

being proportional to some linear combination of the products of a distance metric f and a complimentary coefficient ϕ). To achieve these solutions, we did require the aid of global optimization algorithms to ensure that the solution was near optimal. In practice however these algorithms were able to converge rapidly when posed correctly, and lack of convexity did not inhibit parameter estimation.

Finally, there are many aspects of R-loop formation that were not addressed in detail, and there is still a great body of work yet to come within the subject of CRISPR modeling. One of the most relevant next steps would be a more efficient, and biologically sound representation of the interaction between the host genome and active Cas9/dCas9 molecules. While the system of differential equations in [8] does move in the correct direction, the assumption regarding three-dimensional isotropic diffusion does not likely hold well within human genomes, whose DNA is organized and safeguarded by chromatin structures.

Additionally, because the targeting sequence within dCas9 is so long, relative to sequences within other CRISPR-Cas systems, a promising direction would be to incorporate a probability mass function describing the initial state of the Gambler's ruin chain. This would avoid the numerical issues associated with numerical instability near the tail of the gamblers ruin chain. Furthermore, it would allow for an even more adaptive model and incorporate structural quirks that can occur within the R-loop such as bulges. The latter have been cited as a key factor in the determination of Cas9 mediated cleavage [1].

In regards to the Markov chain models integration into a machine learning architecture as described in Chapter 5, there are still a variety of methodologies that have yet to be attempted, and that have the potential to improve the predictions via the information provided by Markov chain covariates, and the Markov chain prediction itself. One of the most promising so far seems to be bagged Markov chain models, trained on bootstrapped data (similar to the bagged decision trees approach discussed in chapter 5).

This approach is much more time consuming then simply utilizing the output of a single chain, however it would avoid the over fitting issues, similar to a random forest. The largest

barrier that would exist for this method however, would be ensuring that each of the outputs sampled from bootstrapped data would be sufficiently uncorrelated. This is especially true, as in the bootstrapped decision trees, the program not only removes random data points, but also covariates which is no longer possible if we are considering just these values. The next major step could then be bootstrapping over generalized absorbing Markov chains. This would allow the random sub sampling of non-zero transition probabilities, further stratifying the sub-models, and ensure that they have minimal correlation (similar to drop out in neural networks).

Bibliography

[1] Shiran Abadi, Winston X. Yan, David Amar, and Itay Mayrose. A machine learning approach for predicting CRISPR-Cas9 cleavage efficiencies and patterns underlying its mechanism of action. PLoS Computational Biology, 13(10):e1005807, 2017.

[2] Stephen Boyd and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

[3] Jason Brownlee. Why One-Hot Encode Data in Machine Learning? `https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/`, Jun 2017.

[4] Richard L Burden and J Douglas Faires. Numerical analysis. 2001. Brooks/Cole, USA, 2001.

[5] Chemistry LibreTexts. Reaction Rates. `https://chem.libretexts.org/Core/Physical_and_Theoretical_Chemistry/Kinetics/Reaction_Rates`, 2018. Online; accessed 1 April 2018.

[6] Richard Durrett. Essentials of Stochastic Processes. Springer, 2016.

[7] Sergi Elizalde. Numerical Solutions to Gamblers Ruin Markov Chains. `https://math.dartmouth.edu//archive/m20x06/public_html/Lecture14.pdf`, Aug 2006.

[8] Iman Farasat and Howard M. Salis. A biophysical model of CRISPR/Cas9 activity for rational design of genome editing and gene regulation. PLoS Computational Biology, 12(1):e1004724, 2016.

[9] Aaron Fluitt, Elsje Pienaar, and Hendrik Viljoen. Ribosome kinetics and aa-tRNA competition determine rate and fidelity of peptide synthesis. Computational Biology and Chemistry, 31(5-6):335–346, 2007.

[10] D.M. Hamby. A review of techniques for parameter sensitivity analysis of environmental models. Environmental Monitoring and Assessment, 32(2):135–154, 1994.

[11] Philippe Horvath and Rodolphe Barrangou. CRISPR/Cas, the immune system of bacteria and archaea. Science, 327(5962):167–170, 2010.

[12] Stela Pudar Hozo, Benjamin Djulbegovic, and Iztok Hozo. Estimating the mean and variance from the median, range, and the size of a sample. BMC Medical Research Methodology, 5(1):13, 2005.

[13] Patrick D. Hsu, David A. Scott, Joshua A. Weinstein, F. Ann Ran, Silvana Konermann, Vineeta Agarwala, Yinqing Li, Eli J. Fine, Xuebing Wu, Ophir Shalem, et al. DNA targeting specificity of RNA-guided Cas9 nucleases. Nature Biotechnology, 31(9):827–832, 2013.

[14] Blake Hunter, A.C. Krinik, Chau Nguyen, Jennifer M. Switkes, and Hubertus F. Von Bremen. Gambler's ruin with catastrophes and windfalls. Journal of Statistical Theory and Practice, 2(2):199–219, 2008.

[15] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. An Introduction to Statistical Learning, volume 112. Springer, 2013.

[16] Spencer C. Knight, Liangqi Xie, Wulan Deng, Benjamin Guglielmi, Lea B. Witkowsky, Lana Bosanac, Elisa T. Zhang, Mohamed El Beheiry, Jean-Baptiste Masson, Maxime Dahan, et al. Dynamics of CRISPR-Cas9 genome interrogation in living cells. Science, 350(6262):823–826, 2015.

[17] Silvana Konermann, Mark D. Brigham, Alexandro E. Trevino, Julia Joung, Omar O. Abudayyeh, Clea Barcena, Patrick D. Hsu, Naomi Habib, Jonathan S. Gootenberg, Hiroshi Nishimasu, et al. Genome-scale transcriptional activation by an engineered CRISPR-Cas9 complex. Nature, 517(7536):583–588, 2015.

[18] Avantika Lal, Amlanjyoti Dhar, Andrei Trostel, Fedor Kouzine, Aswin S.N. Seshasayee, and Sankar Adhya. Genome scale patterns of supercoiling in a bacterial chromosome. Nature Communications, 7:11055, 2016.

[19] Jonathan Wilder Lavington. A probabilistic approach to modeling CRISPR-Cas9. `https://github.com/WilderLavington/research/tree/master/A_Probabalistic_Modeling_Approach_to_CRISPR_Cas9`, 2018.

[20] Marius Walter. CRISPR/Cas9. `https://commons.wikimedia.org/wiki/File:GRNA-Cas9.png`, 2018. Online; accessed 1 February 2018 from from Wikimedia Commons.

[21] Mathworks. Choose an ODE Solver - MATLAB Simulink. `https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html`, 2018. Online; accessed 1 April 2018.

[22] Mathworks. Find minimum of unconstrained multivariable function - MATLAB fminunc. `https://www.mathworks.com/help/optim/ug/fminunc.html`, 2018. Online; accessed 1 April 2018.

[23] Mathworks. Find minimum of unconstrained multivariable function using derivative-free method - MATLAB fminsearch. `https://www.mathworks.com/help/matlab/ref/fminsearch.html`, 2018. Online; accessed 1 April 2018.

[24] Mathworks. Lasso or elastic net regularization for linear models. `https://www.mathworks.com/help/stats/lasso.html`, 2018. Online; accessed 2 April 2018.

[25] Mathworks. Ridge regression. `https://www.mathworks.com/help/stats/ridge.html`, 2018. Online; accessed 2 April 2018.

[26] Mathworks. Search for optimal regularisation parameter using minimum message length and corrected AIC. `https://www.mathworks.com/help/stats/treebagger.html`, 2018. Online; accessed 2 April 2018.

[27] Christopher E. Nelson, Chady H. Hakim, David G. Ousterout, Pratiksha I. Thakore, Eirik A. Moreb, Ruth M. Castellanos Rivera, Sarina Madhavan, Xiufang Pan, F. Ann Ran, Winston X. Yan, et al. In vivo genome editing improves muscle function in a mouse model of Duchenne muscular dystrophy. Science, 351(6271):403–407, 2016.

[28] Susana R. Neves. Developing models in virtual cell. Science Signaling, 4(192):tr12–tr12, 2011.

[29] Ekaterina Protozanova, Peter Yakovchuk, and Maxim D. Frank-Kamenetskii. Stacked–unstacked equilibrium at the nick site of DNA. Journal of Molecular Biology, 342(3):775–785, 2004.

[30] SA UBCSA UBC. Transcriptional regulation via effector domain. `https://commons.wikimedia.org/wiki/File\%3ACRISPR_effectors.pdf`, 2018. Online; accessed 1 February 2018 from from Wikimedia Commons.

[31] Robert E Schapire. The strength of weak learnability. Machine Learning, 5(2):197–227, 1990.

[32] Statovic. Create bag of decision trees. `https://www.mathworks.com/matlabcentral/fileexchange/54143-fastridge-fast-ridge-regression`, 2018. Online; accessed 2 April 2018.

[33] Samuel H. Sternberg and Jennifer A. Doudna. Expanding the biologists toolkit with CRISPR-Cas9. Molecular Cell, 58(4):568–574, 2015.

[34] Samuel H Sternberg, Sy Redding, Martin Jinek, Eric C. Greene, and Jennifer A. Doudna. DNA interrogation by the CRISPR RNA-guided endonuclease Cas9. Nature, 507(7490):62, 2014.

[35] Antoine Vigouroux, Enno Oldewurtel, Lun Cui, Sven van Teeffelen, and David Bikard. Engineered CRISPR-Cas9 system enables noiseless, fine-tuned and multiplexed repression of bacterial genes. bioRxiv, page 164384, 2017.

[36] Thomas Weise. Global optimization algorithms-theory and application. Self-published, 2, 2009.

[37] Xiaojun Xu, Dongsheng Duan, and Shi-Jie Chen. CRISPR-Cas9 cleavage efficiency correlates strongly with target-sgRNA folding stability: from physical mechanism to off-target assessment. Scientific Reports, 7(1):143, 2017.

Appendix A

Numerical Solutions to the Gambler's Ruin Chain

While this thesis primarily discussed the simplest case regarding a Gambler's Ruin Chain, in reality, a great deal of time went to calculate—efficiently—the probability of absorption at either end for more general chains.

In the context of a Markov chain, a state is called absorbing if it is impossible to leave it once visited, i.e., $p(i,i) = 1$. A chain is said absorbing if it has at least one absorbing state and if from every non-absorbing state it is possible to visit an absorbing state (not necessarily in one step). Non-absorbing states in an absorbing chain are called transient.

Absorption probabilities are always described by a linear system. For example, for the Gambler's Ruin Chain, they are described by the system of linear equations in (3.5). In general, such probabilities are described by a linear system of the form $Px = b$, where x_i is the probability of absorption when the chain starts at state i. Here, P is the probability transition matrix of the chain i.e. $P(i,j)$ is the probability the chain transitions directly into state j when located at state i. To solve this linear system, it is convenient to rearrange the matrix P in the following way:

$$\underline{P} = \begin{bmatrix} T & A \\ 0 & I \end{bmatrix}, \tag{A.1}$$

where the columns in T and A are associated with the transient and absorbing states, respectively, and I is the identity matrix. In particular, if the chain contains t transient states and a absorbing states then T has dimensions $(t \times t)$, A has dimensions $(t \times a)$, and I has dimensions $(a \times a)$.

Recall that the n-th step probability transition matrix is P^n. Since for any absorbing Markov

chain the probability it gets eventually absorbed is one, in terms of the rearranged matrix $\underline{P}$ we find that

$$\lim_{n\to\infty} \underline{P}^n = \begin{bmatrix} 0 & B \\ 0 & I \end{bmatrix},$$

where $B(i,j)$ is the probability the chain gets absorbed at (the absorbing) state j given that it started at (the transient) state i. Note the matrix B has dimensions $(t \times a)$.

Next, we state two elucidating results, which can be found in [7].

Theorem: For an absorbing Markov chain the matrix $(I - T)$ has an inverse N, called the fundamental matrix, and $N = \sum_{k=0}^{\infty} T^k$. The entry $N(i,j)$ gives the expected number of times the chain is in the transient state j given that the chain starts in transient state i.

Theorem: Let t_i be the expected number of steps before the chain is absorbed, given that it starts in the transient state i, and let t be the column vector whose i-th entry is t_i. Then $t = N\mathbf{1}$, where $\mathbf{1}$ is a column vector all of whose entries are ones.

Continuing with the above notation, it can be shown that

$$B = N \cdot A = (I - T)^{-1} \cdot A.$$

The above identity allows one to compute relatively efficiently absorption probabilities. The general algorithm for this is stated in Table A.1. Moreover, in the specific case of the Gambler's Ruin Chain defined in Chapter 2, inverting the matrix $(I - T)$ has low complexity due to the banded structure of the matrix T.

For more information on the theorems discussed above, as well as other possible avenues for implementation of absorbing Markov chains, see [14, 6, 7].

Probability of Absorption in General Markov Chains

Input:	Transition matrix P of dimensions $(n \times n)$ with $a \geq 1$ absorbing states
Step 1:	Rearrange the matrix P into a matrix $\underline{P}$ so that states $1, \ldots, (n-a)$ are transient and states $(n-a+1), \ldots, n$ are absorbing
Step 2:	$T = \underline{P}(1 : n-a, 1 : n-a)$
Step 3:	$A = \underline{P}(1 : n-a, n-a+1 : n)$
Step 4:	$N = (I - T)$
Step 5:	For each column c in A, solve $Nx = c$
Step 6:	Return c vectors

Table A.1: Algorithm to find the probability of absorption from all transient states in an absorbing Markov chain.

Appendix B

Algorithms and Approaches to Global Optimization

As was discussed in earlier Chapter 3, the optimization problem defined by our model is unfortunately not convex (in practice). To combat this, we implemented various algorithms and utilized different programs based on whether we were using matlab These algorithms include decent based algorithms as well as gradient free approaches such as the Neldar-Mead algorithm. While we initially implemented gradient decent, stochastic gradient decent (see Table B.1), and eventually conjugate gradient decent, after migrating the project to MATLAB ®, we utilized fminsearch, and fminunc, as they seemed to find local solutions more effectively. These two methods use variations of the Neldar-Mead simplex algorithm (fminsearch), as well as Quasi-Newton methods such as BFGS (fminunc,fminsearch). For full documentation, see [22, 23]. As the problem did prove to be non-convex in practice, we did have to implement various global optimization algorithms including a few variations of simulated annealing (see Table B.2) (?), as well as multi-start methods (see Table B.3). While computationally expensive, these algorithms did lead to significantly better solutions, and allowed us to ensure that the algorithm found a better approximation for the global minimum of our objective function.

Stochastic Gradient Decent (Global Optimization)

Step 1:	Choose an initial vector of parameters w and learning rate η
Step 2:	Repeat until an approximate minimum is obtained:
Step 2a:	Randomly shuffle examples in the training set
Step 2b:	For $i = 1, 2, ..., n$, do: • $w := w - \eta \nabla Q_i(w)$

Table B.1: $\nabla Q_i(w)$ is the approximated gradient at one sample for a given set of weights and an objective function. η is given as the learning rate [2].

Simulated Annealing (Global Optimization)

Step 1:	Choose an initial vector of parameters w, annealing function $a(t)$, and an optimization protocol $O(w)$
Step 2:	Repeat until an approximate minimum is obtained:
Step 2a:	For $t = 1, 2, ..., n$, do: • with probability $a(t)$: move in the direction given by $r \cdot O(w)$, where r is a random vector of the same element-wise magnitude as $O(w)$ • with probability $1 - a(t)$: move in the direction given by $O(w)$

Table B.2: Simulated Annealing Optimization Algorithm pseudo-code. For more information see [36]

Multi-start (Global Optimization)

Step 1:	Choose a depth function $d(t)$, and an optimization protocol O(w_i)
Step 2a:	Generate a random set of 2^n starting points (w_i)
Step 2b:	For $t = 1, 2, ..., n$, do:
Step 2c:	• If $t = 1$: run optimization protocol at each point for $d(1)$ iterations, and store the top 50% of the parameter realizations found, as well as the objective function at these weights
Step 2d:	• If $t \neq 1$: run optimization protocol at each point of current top 50% of parameter realizations w_i for $d(t)$ iterations, and store the top 50% of the parameter realizations found, as well as the objective function at these weights
Step 3:	Return optimal parameters, as well as the objective function evaluation

Table B.3: Multi-start Global Optimization Algorithm pseudo-code. For more information see [36]

Appendix C

Numerical Integration and Analysis

The initial integration scheme used was the classic fourth order Runge-Kutta method displayed below for easy access. However, once it became apparent that early system behavior under certain parameters regimes was stiff, we switched to backward differentiation formulae (sometimes called Gear's methods). during the initial stages of this research, much of the research was done within python, and thus we implemented these algorithms by hand using numpy. However as many of these methods are readily available in MATLAB ®, we transitioned to this language and utilized both ODE45, and ODE15s, for non-stiff and stiff integration respectively.

Other more efficient or accurate methods exist for solving systems of stiff differential equations, however, the methods described in C.1 and C.2 were much more readily implemented and easy to work with. The most regularly used of these methods being Rosenbrock methods (used in ODE23s), which while more efficient, did not have the numerical precision required. It should also be noted that MATLAB ®'s implementation of these algorithms is (obviously) much more sophisticated. For example in the case of ODE45, MATLAB uses a variable step size method that switches between fourth and fifth order Runge-Kutta based upon integration tolerances. In the case of ODE15s, due to the inefficiency of Gear's methods, MATLAB switches between these, and numerical differentiation formulas (NDFs) of orders 1 to 5. For a full explanation and example implementation of these programs see [21].

Fourth Order Runge-Kutta (RK4)

Inputs:	$y_{0,1} = f_1(x, y_1, y_2, ..., y_n)$ subject to $y_1(x_0) = y_{1,0}$ $\vdots$ $y_{0,n} = f_n(x, y_1, y_2, ..., y_n)$ subject to $y_n(x_0) = y_{n,0}$
Step 1:	$k_{1,1} = hf_1(x_j, y_{1,j}, y_{2,j}, ..., y_{n,j})$, where $y_{1,j} = y_1(x_j)$ $\vdots$ $k_{1,n} = hf_n(x_j, y_{1,j}, y_{2,j}, ..., y_{n,j})$, where $y_{n,j} = y_n(x_j)$
Step 2:	$k_{2,1} = hf_1(x_j + \frac{h}{2}, y_{1,j} + \frac{k_{1,1}}{2}, y_{2,j} + \frac{k_{1,2}}{2}, ..., y_{n,j} + \frac{k_{1,n}}{2})$ where $y_{1,j} = y_1(x_j)$ $\vdots$ $k_{2,n} = hf_n(x_j + \frac{h}{2}, y_{1,j} + \frac{k_{1,1}}{2}, y_{2,j} + \frac{k_{1,2}}{2}, ..., y_{n,j} + \frac{k_{1,n}}{2})$, where $y_{n,j} = y_n(x_j)$
Step 3:	$k_{3,1} = hf_1(x_j + \frac{h}{2}, y_{1,j} + \frac{k_{2,1}}{2}, y_{2,j} + \frac{k_{2,2}}{2}, ..., y_{n,j} + \frac{k_{2,n}}{2})$, where $y_{1,j} = y_1(x_j)$ $\vdots$ $k_{3,n} = hf_n(x_j + \frac{h}{2}, y_{1,j} + \frac{k_{2,1}}{2}, y_{2,j} + \frac{k_{2,2}}{2}, ..., y_{n,j} + \frac{k_{2,n}}{2})$, where $y_{n,j} = y_n(x_j)$
Step 4:	$k_{4,1} = hf_1(x_j + h, y_{1,j} + \frac{k_{3,1}}{2}, y_{2,j} + \frac{k_{3,2}}{2}, ..., y_{n,j} + \frac{k_{3,n}}{2})$, where $y_{1,j} = y_1(x_j)$ $\vdots$ $k_{4,n} = hf_n(x_j + h, y_{1,j} + \frac{k_{3,1}}{2}, y_{2,j} + \frac{k_{3,2}}{2}, ..., y_{n,j} + \frac{k_{3,n}}{2})$, where $y_{n,j} = y_n(x_j)$
Step 5:	$k_{4,1} = hy_{1,j} + \frac{1}{6}(k_{1,1} + 2k_{2,1} + 2k_{3,1} + k_{4,1})$ $\vdots$ $k_{4,n} = hy_{n,j} + \frac{1}{6}(k_{1,n} + 2k_{2,n} + 2k_{3,n} + k_{4,n})$

Table C.1: Algorithm for Numerical integration via fourth order Runge-Kutta. This algorithm is simple to implement, and accurate for non-stiff parameter regimes. For more information, see [4]

Gear's Method (Backward Differentiation Formulas)

First Order:	$y_{n+1} - y_n = hf(t_{n+1}, y_{n+1})$
Second Order:	$y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = \frac{2}{3}hf(t_{n+2}, y_{n+2})$
Third Order:	$y_{n+3} - \frac{18}{11}y_{n+2} + \frac{9}{11}y_{n+1} - \frac{2}{11}y_n = \frac{6}{11}hf(t_{n+3}, y_{n+3})$
Fourth Order:	$y_{n+4} - \frac{48}{25}y_{n+3} + \frac{36}{25}y_{n+2} - \frac{16}{25}y_{n+1} + \frac{3}{25}y_n = \frac{12}{25}hf(t_{n+4}, y_{n+4})$
Fifth Order:	$y_{n+5} - \frac{300}{137}y_{n+4} + \frac{300}{137}y_{n+3} - \frac{200}{137}y_{n+2} + \frac{75}{137}y_{n+1} - \frac{12}{137}y_n = \frac{60}{137}hf(t_{n+5}, y_{n+5})$
Sixth Order:	$y_{n+6} - \frac{360}{147}y_{n+5} + \frac{450}{147}y_{n+4} - \frac{400}{147}y_{n+3} + \frac{225}{147}y_{n+2} - \frac{72}{147}y_{n+1} + \frac{10}{147}y_n = \frac{60}{147}hf(t_{n+6}, y_{n+6})$

Table C.2: Recursions for various backward differentiation formulas. To start, one can either rely on Taylor series approximations, or use lower order BDF formulas. These schemes are simple to implement, and robust in stiff systems of differential equations. For more information, see [4]