LEARNING THE SEQUENCE DETERMINANTS OF MAMMALIAN TRANSCRIPTIONAL GENE REGULATION ACROSS CELL-TYPES

by

Dustin Shigaki

A dissertation submitted to Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland

July 2024

Abstract

Gene expression is controlled by *cis*-regulatory DNA elements that contain binding sites for a class of proteins known as transcription factors. The set of transcription factors that bind one of these DNA elements determine its regulatory function. Mutations to these binding sites have been associated with or directly cause complex and common human disease. Prediction of the impact of regulatory variants requires knowledge of the sequence determinants of regulatory activity. Many machine learning algorithms have been developed to predict regulatory activity directly from DNA sequence. Although these models regularly achieve high predictive performance, they are difficult to biologically interpret due to their complexity.

This dissertation is focused on the performance and interpretation of these models. First, I compare current sequence-based models on their ability to learn TFBSs and investigate their strengths and weakness. Then, to extract biological and compact features from these models, I developed a novel algorithm, gkmPWM, which learns individual Transcription factor binding site (TFBS) information by modelling gapped kmer distributions. I mathematically derive the algorithm and present a Lagrangian optimization method that extracts all the TFBS learned in these models *de novo*. Lastly, I show that gkmPWM outperforms other methods that learn TFBS sequence features.

In the second part of my dissertation, I apply gkmPWM to a wide range of experiments to derive sequence rules for different classes of regulatory elements. I characterize the cell-type independent binding of promoters and insulators with a small set of transcription factors. Additionally, I identify the combinations of TFBSs that determine the cell-type specific activity of enhancers. I also use gkmPWM to learn the sequence preferences of different functional characterization assays and show that reporter assays have unique sequence preferences.

In the last part of my dissertation, I map the TFBSs learned by gkmPWM to their specific positions in regulatory elements at nucleotide resolution. I present a dynamic programming algorithm to efficiently map combinations of binding sites using information from gkm-SVM models. I show that these TFBS predictions align with experiments that target specific TFBSs. I mapped TFBSs to distal enhancers in a wide range of cell-types, which are publicly available.

Thesis readers

Dr. Michael A. Beer (Primary Advisor)
Professor
Department of Biomedical Engineering and Genetic Medicine
Johns Hopkins University

Dr. Andy McCallion
Professor
Department of Genetic Medicine
Johns Hopkins School of Medicine

Dr. Elena MacFarlane
Associate Professor
Department of Genetic Medicine
Johns Hopkins School of Medicine

Dedication

This thesis is dedicated to my mom and dad, Roxane and Eric Shigaki.

Acknowledgements

I would like to first express my gratitude to my mom and dad who dedicated countless hours of their time and effort to my development. They always encouraged me to pursue academic topics beyond what was taught in the classroom and would obtain whatever books in mathematics and science that I asked for. The habits that I built from spending the time to learn subjects independently directly translated into investigating scientific questions at the graduate level.

I would also like to thank members of the ENCODE phase 4 consortium, in particular, the Functional Characterization Capstone Group, who helped me develop the collaborative skills to work with others who specialize in different aspects of the scientific process. The results of Chapter 6 are dependent on the efforts made by this group.

In addition, I would like to thank the members of the Beer lab, Jin Woo Oh, Wang Xi, Milad Rasavi-Mohseni, Andrew Rojnuckarin, Nicolas Eng, and Gary Yang who provided much feedback on the work in this dissertation. Their suggestions improved the quality of the code written for motif extraction in Chapter 4. I would like to specifically thank Nicolas Eng, who optimized the parameters for RCmax in Chapter 3. Also, I would like to thank Gary Yang, who invested much of his time to convert the MATLAB code of the gkmPWM repository into a C implementation.

Lastly, I would like to express much gratitude to my mentor Dr. Michael Beer. During the PhD application process, he was the only one who expressed great interest in recruiting me into his lab. I am very fortunate that we happened to share the same interests in science, which include quantitative modelling of biological phenomena, abstract mathematics, and the history of scientific development over the past centuries. I cannot imagine someone who was more suited to mentor me than Dr. Beer.

Table of Contents

List of Tables

List of Figures

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Chapter 7

Chapter 1

Introduction

1.1 Overview

Complex multicellular organisms contain many different cell-types with unique functions. However, each cell in the organism originates from a single cell, which copies its DNA to its daughter cells. Even though all cell-types contain the same DNA, they are able to express different combinations of genes that are essential to their function. An important question in biology is undercovering tissue and cell-type gene expression. This requires a mechanism that can target certain genes with high specificity. The source of this specificity lies in regulatory DNA sequence. The tissue and cell specific expression of genes is controlled by *cis*-regulatory DNA elements (CREs) whose activity depends on the occupancy of cell-type specific transcription factor binding sites (TFBS). Connections between TFBS in distal enhancers regulating TF genes specifies the interconnected wiring of nonlinear gene regulatory networks which control the dynamics of cell-fate decisions, human development, and cellular responses to environmental stimuli. Genetic predisposition to common human diseases such as autism[1], diabetes[2], and cancer[3] is largely mediated by variants which alter the activity of these *cis*-regulatory elements. As a result, much effort has been made to decipher the sequence determinants of *cis*-regulatory elements.

The difficulty of understanding the sequence determinants of gene expression is best embodied in the Shh enhancer. Multiple mutations in this enhancer have been associated with different phenotypes of limb development[4]. Mutations can potentially enhance limb development or hinder it[5]. To confound this problem further, not all variants in this enhancer cause a change in phenotype. The reason for this phenomenon is that variants must severely disrupt a TFBS or strongly increase the binding affinity. Mutations that are not within a TFBS will not affect regulation of the gene. In addition, mutations within a TFBS may not significantly affect the binding affinity of the TF. Therefore, to predict the impact of these variants, we must know 1) what TFs are binding, 2) the location of the binding sites, and 3) the change in binding affinity of the TF induced by the mutation, which requires accurate knowledge of relative binding affinity to the wild-type and mutant sequences.

Many machine learning models have been developed to predict these elements[6,7] and the impact of regulatory sequence variation on chromatin accessibility[8] and gene expression[9]. These models can be trained on DNA sequence alone, implying that learning the binding preferences of TFs is sufficient to accurately determine cis-regulatory activity. However, in order to achieve high predictive power, these models must be complex enough to learn the binding preferences (or motifs) of multiple TFs, making it difficult to interpret most predictive features. For example, support vector machines trained on gapped k-mer counts (gkm-SVM) require ~10^6 gapped k-mers to achieve high performance, and convolutional neutral networks use even more parameters in multiple complex network layers and filters. As a consequence, identifying the TF motifs learned by the models remains a significant challenge. Further investigation and interpretation of enhancer function requires reliable predictions of the locations of TFBS's within the elements. While Position Weight Matrix (PWM) models are compact and interpretable representations of TFBS, they are often experimentally derived from the strongest binding sites, and perform poorly relative to gkm-SVM and DNN models when predicting genomic TF binding[10,11]. Thus, it is necessary to develop methods to extract TFBS motifs from complex machine learning models and map their multiple occurrences in promoters and enhancers.

In this dissertation, I will assess the quality of modern sequence-based models and identify their strengths and weaknesses. I will then present the main focus of the dissertation, which is the development of a novel algorithm, which I call gkmPWM, to extract TFBS motifs/information from these models. I will demonstrate gkmPWM's superior performance over other TFBS motif learning methods. I will then apply gkmPWM in combination with gkm-SVM to learn predictive sequence features of promoters, enhancers, and insulators across many cell-types. I will also identify the sequence preferences of functional characterization assays, which were developed to quantify regulatory activity. Lastly, I map the motifs learned by gkmPWM to positions in regulatory elements at single nucleotide resolution and show that those predictions align with experiments that map TFBSs.

1.2. Thesis organization

In **Chapter 2**, I provide background information on transcription factors and *cis*-regulatory elements. I will describe their individual functions and how they interact with each other. I then review experimental techniques to map *cis*-regulatory elements in the genome. This includes TF targeting assays (ChIP-Seq) and chromatin accessibility assays (DNase/ATAC-Seq). I also detail functional characterization assays which quantitatively measure the regulatory activity of DNA. Lastly, I describe mathematical models of TF binding which includes their cooperativity and their individual sequence preferences.

In **Chapter 3**, I compare two leading sequence-based models, the support vector machine with gapped kmer features, and the convolutional neural network. I assess their ability to learn multiple binding sites simultaneously on synthetic data. I also compare their performances on real data, which include predicting regions with high chromatin accessibility and dsQTLs. I then discuss their strengths and weaknesses.

In **Chapter 4**, I derive a mathematical model to extract TFBS motifs from any set of sequences, which can be extended to sequence-based models. The foundation of this model comes from modelling the distribution of gapped kmer counts in regulatory DNA. I present two different approaches: one which uses only previously trained PWMs (gkmPWMlasso), and one that learns PWMs *de novo* (gkmPWM). The algorithm to learn PWMs de novo uses Lagrangian optimization techniques to train the frequencies within each model. At end the chapter, I compare gkmPWM to other motif learning algorithms on synthetic and real data.

In **Chapter 5**, I use gkm-SVM and gkmPWM in tandem to learn the sequence determinants of different classes of regulatory elements in the mapping data generated by ENCODE. I identify the combinations of motifs that predict promoter, enhancer, and insulator activity. I present the cell-type specific TFs detected across many cell/tissue types and show that these motifs are sufficient to predict the cell-type.

In **Chapter 6**, I once again use gkm-SVM and gkmPWM to learn the motifs that predict functional characterization data, which includes massively parallel reporter assays, self-transcribing active regulatory region sequencing (STARR-Seq), and CRISPR interference (CRISPRi). I compare the predictive sequence features of these assays and present other novel analyses.

In **Chapter 7**, I present a method to map gkmPWM motifs to regulatory elements at nucleotide resolution using information from gkm-SVM. I will show how to perform this quickly with dynamic programming. I assess the quality of the TFBS predictions by comparing them to saturation mutagenesis assays, which can detect TFBSs at nucleotide resolution, and ChIP-Seq, which can correctly identify TF binding at a region. Lastly, I present an online resource that contains predictions of TFBSs for 1888 different DNase/ATAC-Seq experiments performed on many cell-types.

Chapter 2
Background

While most of the protein-coding human genome has been deciphered, the non-coding regions dedicated to transcriptional regulation are still under intense investigation, despite much effort. Like genes, the sequence largely determines the function of regulatory DNA. The sequence regulates expression through the binding of transcription factors, proteins that are responsible for assembling the transcriptional machinery, initializing transcription, and stabilizing interactions between elements. This thesis is focused on identifying the transcription factor binding sites that determine the function of regulatory DNA. In this section, I will outline the function of transcription factors and three classes of regulatory DNA: promoters, enhancers, and insulators. I will then describe experimental approaches to identify these regions in the genome and to quantitatively measure their activity. Lastly, I will describe mathematical models of transcription factor binding. This information will serve as the basis for comparing the performance of leading sequenced-based models of regulatory activity (**Chapter 3**), extracting TFBS information from these models (**Chapter 4**), learning the predictive motifs of regulatory activity (**Chapter 5 and 6**), and finally mapping these motifs to their exact positions in regulatory DNA.

2.1. Pre-transcriptional gene regulation

There are two main components to pre-transcriptional gene regulation: *trans*-regulatory elements or transcription factors, and *cis*-regulatory elements or the regions of

DNA to which TFs bind*. There are 2000 different transcription factors coded in the genome and likely more than a million *cis*-regulatory elements[12]. *cis*-regulatory elements can be further separated into different classes. Although other classes of elements may exist[13], there are three classes that have been shown to be functionally distinct: promoters, distal enhancers, and insulators.

Transcription factors

TFs are regulatory proteins that initiate (or potentially repress) gene expression. They contain a DNA binding domain, which contains the sequence specificity. Although there are 2000 TFs, there are only a few hundred recognition patterns of sequence[14]. A specific pattern of sequence that is recognized by the DNA binding domain is called a transcription factor binding site (TFBS), or more simply: "motif." TFs also contain an activation domain, which interacts with other proteins that help recruit and stabilize the transcription machinery. These include histone modifiers and proteins involved in enhancer promoter interactions such as Mediator[15]. Transcription factors are pivotal components of cellular function. Cells process information and make decisions by activating and repressing the expression of TFs. As we will see, TFs determine the function of the regulatory element to which they bind.

Promoters

Promoters are regions of the genome located near transcription start sites that recruit RNA Polymerase II and the general transcription factors. Genes often contain more than one promoter, which can both be active simultaneously. Interestingly, PolII transcribes in both directions, but prematurely terminates its transcription when going in

* For the sake of convenience, I will refer to *trans*-regulatory elements as transcription factors or TFs, and *cis*-regulatory elements as regulatory elements.

the unproductive direction. This implies that promoters have a degree of orientation independence. In addition, even when not actively transcribing, PolII is normally bound to promoters. In order to initiate transcription, one or more active enhancers must interact with the promoter. Although the sequence content of all classes of regulatory elements are unique, promoters may have the most variation. These regions have much higher fractions of the G and C nucleotide (GC content) compared to the rest of the genome. In particular, the frequency of the CG dinucleotide (CpG) is much higher since promoters can be methylated to repress function. Promoters contain a wide variety of sequence features that bind to TFs that recruit PolII. The most recognizable one is the TATA box, even though it is only present in 24 percent of human promoters[16]. In future chapters, I will identify the core set of promoter sequence features in humans.

Enhancers

Enhancers are regulatory elements located much further from the TSS than promoters. They are required for most promoters to initiate transcription. The specific biophysical mechanism(s) of enhancer activation has not been investigated thoroughly, but enhancers are believed to stabilize promoter activity through direct interactions. One or more enhancers loop to contact the target promoter, and a variety of proteins such as Mediator assemble in a large complex. Like promoters, enhancers contain multiple binding sites for multiple types of TFs, which enable enhancer activity. Experiments have shown that enhancers are largely orientation independent[17,18]. In addition, there seems to be few restrictions on specific combinations of TFs required for enhancer function. While many have proposed a class of TFs called pioneer factors which open the enhancer to enable the other TFs to bind, other experiments have shown that any TF will bind to its target if the concentration of protein is high enough, if the binding site contains strong motifs, or if the number of binding sites in the enhancer is large enough[19,20]. There are some examples of strict ordering and positioning of TFBSs, but these seem to be exceptions. It seems that a wide range of combinations of TFs are permissible for enhancer function. Enhancers also can interact with their target promoter at a distance up to 1 Mb. However, most enhancers are closer to their promoters (~20-100kb). Interestingly, 40% of enhancers lie in introns. A difficult problem in genomics is predicting enhancer-promoter pairs that interact. It turns out that the last class of regulatory element, insulators, are the determinants of those interactions.

Insulators

Insulators are a unique class of regulatory elements. They are responsible for organizing the genome into compartments called topologically associated domains (TADs). Pairs of insulators interact to form large DNA loops that restrict the set of possible enhancer-promoter interactions. The TF that binds to insulators is CTCF, which binds to a long motif and blocks cohesin extrusion along with a partner CTCF binding site to stabilize the loop. Unlike promoters and enhancers, insulators are orientation dependent with respect to their CTCF motif, which is not reverse palindromic. Convergent orientation pairs form over 90% of loops[21,22]. Computational analysis has shown that the most predictive feature of predicting enhancer-promoter pairs is the enhancer and promoter being in the same CTCF loop[23].

Despite the fact that much of the function of promoters, enhancers, and insulators have been characterized, identifying these elements in the genome is necessary to learn their sequence features. Many assays have been developed to do so, and I will describe the most successful ones in the next section.

2.2 Experimental approaches to identify *cis*-regulatory elements and transcription factor binding sites

Before modelling the activity of regulatory elements, a set of real regulatory elements must be generated to train the models on. There are two general classes of experiments that can identify regulatory elements: mapping assays and functional characterization assays. The prior maps specific regions in the genome that may have regulatory function. The most common approaches are to target TFs while they are bound to their binding sites or to target accessible chromatin. Functional characterization experiments quantitatively measure the activity of putative regulatory elements. In this section, I give an overview of the design of these assays, their strengths, and their weaknesses.

Transcription factor targeting assays

The most straightforward approach to identifying potential regulatory elements is to map the binding sites of the active TFs genome wide. The first approach developed to accomplish this is Chromatin Immunoprecipitation (ChIP-Seq)[24,25], an assay that isolates transcription factors through immunoprecipitation while they are bound to genomic DNA. To summarize the methodology, transcription factors are cross-linked to their binding sites in the genome with formaldehyde, which covalently binds them together. To separate the binding sites from the rest of the genome, sonication is performed to shear the DNA into 300 bp fragments. Finally, the binding site fragments are isolated using immunoprecipitation by treating the TF-DNA complex with antibodies developed to target the TF. The fragments recovered in this process are then sequenced. One will typically find anywhere from a few thousand to tens of thousands of binding sites using ChIP-Seq.

An important extension of ChIP-Seq is that the targeted protein does not necessarily need to be a DNA binding protein. The secondary co-factors that are recruited by the DNA-binding TFs can also serve as reliable targets. In particular, the protein EP300, which is often recruited to enhancers, can be a reliable target to map cell-type specific enhancers[26]. Other interesting targets include RNA Polymerase II[27], histone modifiers, such as histone deacetylases (HDACs)[28], and histone modifications themselves[27,29]. Interestingly, histone modifications were discovered to be associated with classes of regulatory elements. In particular, enhancers are enriched with the H3K27ac and H3K4me1 marks, and promoters are enriched with the H3K27ac and H3K4me3 marks. These epigenetic marks of enhancers and promoters are highly conserved and are often used to identify potential regulatory elements across species[30–32].

Although using ChIP-Seq to identify regulatory elements is a reasonable idea, there are a few shortcomings, most of which are practical concerns. First, an antibody with a strong affinity to the target TF that is also highly specific to the TF must be developed. This process is highly dependent on sheer luck. Low affinity antibodies will fail to produce peaks, and non-specific antibodies will produce false positives. Second, the set of cell-type specific, enhancer-binding TFs must be known in order to correctly identify regulatory elements. This is difficult to accomplish with ChIP-Seq alone. RNA-Seq can be used to identify TFs with higher differential expression compared to other cell-types, but an expressed TF is not necessarily active. Lastly, for ChIP-Seq experiments that target histone modifications, the signals produced (peaks) are very broad and are typically not centered on the actual regulatory elements. Furthermore, there are many histone modifications whose function is still being investigated. For example, the histone modification H3K27me3 marks regulatory elements in embryonic cells to repress their function until further differentiation[33]. As a result, ChIP-Seq is not ideal for identifying most regulatory elements. The information needed for ChIP-Seq to map regulatory elements can be found in other assays that can accomplish this task, namely the chromatin accessibility assays.

Chromatin accessibility assays

The most powerful assay to identify regulatory elements are the two chromatin accessibility assays, DNase-Seq and ATAC-Seq. The idea of these assays is to make cuts in genomic regions that are "open," or in other words, not occupied by histones. The most common regions that are open are regulatory elements, which are bound by transcription factors instead of histones. Due to the highly stochastic nature of TF binding, TFs often release from their binding sites, allowing enzymes that cleave DNA (DNase-I for DNase-Seq and Tn5 transposase for ATAC-Seq) to bind and induce a strand break. Although DNase-Seq and ATAC-Seq both target accessible regions, there are some differences between their methodologies. For DNase-Seq, DNA fragments are filtered to isolate reads with short lengths, which are more likely to originate from regulatory elements. This is followed by sequencing and genome mapping. A common terminology used for genomic regions with high DNase-Seq signal is "DNase I Hypersensitivity Sites" (DHS), which I will use throughout this dissertation. ATAC-seq targets accessible regions using a prepared Tn5 transposase, which has sequencing adapters already bound to it. The Tn5 transposase cuts accessible regions and inserts the adapters immediately. ATAC-Seq is a cheaper and easier assay to perform than DNase-seq[34], and thus it is now much more commonly used than DNase-Seq. There are other minor differences as well, but the signal measured by DNase-seq and ATAC-seq are highly correlated[35]. Thus, when identifying regulatory elements, there is not a significant advantage using one assay over the other. Furthermore, 95% percent of ChIP-Seq peaks overlap with a DHS[36], implying that 1) the chromatin accessibility is driven by TF binding and 2) DHS peaks encapsulate nearly all the information of multiple ChIP-Seq experiments. Therefore, DNase-Seq and ATAC-Seq avoid the problem of requiring *a priori* knowledge of all active TFs in the cell type of interest.

An interesting phenomenon is that promoters and insulators tend to be accessible across many cell-types, while the accessibility of distal enhancers is cell-type specific. As a result, investigating the sequence rules of cell-type specific behavior will involve heavy analysis of enhancer sequence features. Furthermore, this allows for a simple rule for separating regulatory elements into promoters, enhancers, and insulators with fairly high accuracy. Regions with high accessibility across cell-types that are near TSSs are most likely promoters. Distal regions with high accessibility that are cell-type specific are most likely enhancers. The regions that remain are most likely insulators. There are certainly flaws with this rule, namely, defining cutoffs for distance to be considered a promoter and what constitutes "cell-type specificity." Nevertheless, these simple rules are more than adequate for determining sequence features of each class of element, which I will demonstrate in Chapter 5.

Functional characterization assays

The last group of assays are the functional characterization assays, which are designed to measure regulatory activity of elements in terms of mRNA transcript levels, rather than just number of mapped reads within an accessible element. These reporter assays attempt to answer the question of whether a potential regulatory element defined by DNase/ATAC-Seq actually has regulatory activity, and to what degree. There are three main classes of large-scale functional characterization assays: massively parallel reporter assays (MPRA), self-transcribing active regulatory region sequencing (STARR-Seq), and CRISPR interference (CRISPRi). MPRA and STARR-Seq are plasmid-based episomal assays and CRISPRi is an *in vivo* assay. I will give a general overview of each in this section, and a more detailed description of these in Chapter 6.

The design of the MPRA plasmid places the regulatory element of interest (the insert) upstream of a reporter gene (usually luciferase) and a barcode[37,38]. There are separate constructs used to assay enhancers and promoters. For promoters, the insert is simply placed upstream of the gene and barcode. For enhancers, a "minimal promoter" (a promoter with low activity) is placed in between the insert and gene. To measure regulatory activity, the mRNA counts, which are identified by the barcodes, are divided by the associated DNA plasmid counts. The base-2 logarithm is normally applied to this ratio, which is called the log fold expression. As implied in its name, many regulatory elements can be tested simultaneously with MPRA. The STARR-Seq assay, like MPRA, creates readout of mRNA from a plasmid. The metric of expression levels is also the log fold expression. The main difference between MPRA and STARR-Seq is that in STARR-seq the insert is placed downstream of the minimal promoter[39]. As a result, the insert transcribes itself. Thus, this assay is strictly to measure enhancer activity. MPRA and STARR-Seq assays are widely used to quantitatively characterize regulatory activity. However, there are a couple of concerns. The primary one is that they are episomal assays, which take the inserts outside of their native biological contexts. The scope of regulatory activity is limited to direct interactions of a single enhancer and promoter in a small span of distance. Histones, which block regulatory activity, may not bind to the plasmid in the same way that they bind to genomic DNA. The other concern is that the log fold expression of MPRA and STARR-Seq do not necessarily correlate[40]. Thus, it is uncertain whether we should define regulatory elements as having either a positive MPRA or STARR-Seq readout, or both having positive readouts.

The last class of functional characterization assays is CRISPRi, which targets elements *in vivo*. An engineered dCas9 protein, which is a Cas9 protein modified to no longer induce strand breaks, is fused with a KRAB domain, which recruits histone modifiers to silence the regulatory element by inducing H3K9me3 marks[41]. This assay is highly specific since it requires a guide RNA of length ~20nt to target the region. CRISPRi has been modified to target and characterize multiple regions in a single experiment[42,43]. This assay has upsides not present in MPRA or STARR-Seq. The readout is the decrease in expression of a specific gene. Therefore, CRISPRi can link enhancers to their gene targets. Additionally, it is currently the only *in vivo* assay. Phenomena that cannot be replicated in MPRA or STARR-Seq, like the disruptions of a single enhancer that us near other enhancers with the same target. However, there are some weaknesses. The scale of the assay is not as high as the plasmid-based assays since it requires enhancer-gene pairs to form its quantitative metric. Thus, the scale increases as a function of a quadratic rather than a linear function. Also, while perturbations to the promoter permanently affect gene expression, disruptions to the enhancer are often temporary[23]. This can be interpreted as both a strength and weakness since robustness of expression is a key feature of genetic networks. However, in certain contexts, such as development, the timing of regulatory activity can potentially play important roles. In order to detect these changes in expression, the experiment must be performed at the time of activation, which is often difficult to determine *a priori*. Functional characterization assays are not meant to replace mapping data, but to complement them. Identifying regulatory elements remains a challenging task, but dramatic progress has been made over the past few years.

2.3 Mathematical Modelling of *cis*-Regulatory Activity

The primary focus of this paper is the mathematical modelling of regulatory elements and their activity with respect to their DNA sequence. There are two main components to this. The first is the cooperativity between transcription factors to bind in a non-linear (or non-additive) manner to regulatory elements. The second is the individual binding sequence preferences of each TF. Both play important roles in the function of regulatory elements, as a single mutation to a single binding site can interfere with the cooperative binding of neighboring TFs and disrupt regulatory function. Conversely, a variant in a regulatory element may not have any effect. In this section, I discuss mathematical models of these two components.

Cooperative binding of transcription factors

Eukaryotic transcription factors operate in a non-linear fashion. Combinations of TFs bind together to displace the histones bound to regulatory elements. Mirny derived a model of this phenomenon through thermodynamic laws[44]. He modelled the occupancy of the regulatory element Y as

$$Y = \alpha \frac{(1+\alpha)^{n-1} + Lc(1+c\alpha)^{n-1}}{(1+\alpha)^{n} + L(1+c\alpha)^{n}}$$

Where α is the concentration of the TF, L is an equilibrium term of the nucleosome bound state and nucleosome free state in the absence of TFs, and c is free energy cost of DNA unwrapping to allow a TF to bind. In the limit of low TF concentration, Y approaches zero. In the limit of high TF concentration $\alpha \rightarrow \infty$, the occupancy behaves like

$$\lim_{\alpha \rightarrow \infty} Y = \alpha \frac{\alpha^{n-1} + Lc(c\alpha)^{n-1}}{(\alpha)^{n} + L(c\alpha)^{n}} = 1$$

A notable feature of this model is that it behaves much like a hill function $\frac{x^n}{x^n+k^n}$, where increasing n increases the “switch-like” behavior of the system. In addition, decreasing L reduces both the switch-like behavior of the system and makes it easier for lower concentrations of TF to occupy the regulatory element (the limit as $L \rightarrow 0$ is 1). This models biological mechanisms such as the application of H3K27ac modifications. The most interesting claims made in this model are that multiple low affinity sites create more stable nucleosome bound versus TF bound states and that the ideal number of binding sites per nucleosome (per 147 bp) is 4-6. These insights explain much of the odd behavior of regulatory elements, in particular, their sequence content, which I will go over in the next section.

Sequence-based modelling of transcription factor binding preferences

One of the trickiest components to model in regulatory elements is transcription factor binding. As stated earlier, TFs do not just bind to a particular sequence, but also to other sequences with similar nucleotide compositions. The collection of sequences to which a TF binds is called a motif. Because TFs can bind to a wide range of sequences, the binding is also described as "degenerate." The level of degeneracy varies among TFs. Some TFs bind strongly to a very small set of sequences, while others bind promiscuously. Mathematical models of transcription binding must be able to fully capture the span of sequences to which a TF can bind.

In 1987, von Hippel showed that the free energy of binding can be directly related to the frequency of each nucleotide in each position of the binding site[45]. He assumed independence of the positions in this derivation. This mathematical structure would become the most popular form of TF models, the position weight/frequency matrix. This is a 4 by L matrix where the rows index each nucleotide, and the columns index the positions of the binding site. Each entry contains the frequency of a nucleotide in each position (**Fig 2.1A**). The matrix of frequencies/probabilities was referred to as the position frequency matrix (PFM) and the negative logarithm of the PFM was named the position weight matrix (PWM). Over time, the term PWM became synonymous with both forms of the matrix. Another convenient form of the PWM is the motif logo, where the letters are scaled by their frequency and stacked on top of each other from largest to smallest (**Fig 2.1B**). The total height of the column is scaled by the information in the column, which is the difference in entropy of the distribution of the column and entropy of a flat distribution. In the case of PWMs, this is always

$$2 + \sum_{N \in \{A,C,G,T\}} p_N \log_2 p_N$$

A

$$\begin{array}{c} A \\ C \\ G \\ T \end{array} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0.05 & 1 \\ 0 & 0 & 0 & 0.5 & 0 & 0.95 & 0 \\ 0 & 1 & 0 & 0.5 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

B

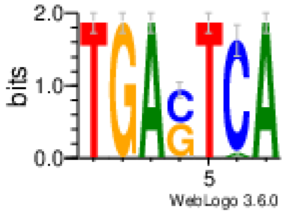


Figure 2.1 An Example of a Position Weight Matrix and Motif Logo

***A**) The position weight matrix is comprised of the frequencies of nucleotides in each position. **B**) The matrix can be converted into a logo by scaling the letter by their information (right). The example provided is an AP1 motif.*

PWMs are typically trained using expectation maximization[46]. The training set is a large number of sequences that are bound by the TF. These sequences are normally gathered using ChIP-Seq or SELEX[10,47]. A PWM is initialized to begin training. Each sequence is scored by scanning all subsequences of length L. The score is defined by taking the product of the entries associated with the correct nucleotide in each column (see 4.1.2 for more details). In each sequence, the subsequence with the highest score is selected to construct a new PWM, and the process is repeated until convergence. There are a few problems that can occur in training for PWMs, which I will describe in 4.1.7. Nevertheless, PWMs for the vast majority of human TFs have been trained and are available in multiple databases[48,49].

Although there have been criticisms that the simplicity of the PWM limits its predictive power, the main shortcoming of the PWM is that it only models a single TF. As mentioned earlier, combinations of TFs cooperate to bind to their genomic target. The problem can be conceptualized as follows: a sequence of length L appears $3 \times 10^9/_{4^L}$ times in the genome, not accounting for reverse complements. A binding site is normally 6-10 bp long, with a few exceptions. Taking $L = 8$, the average number of times a sequence appears in the genome is 46,000. This does not account for degenerate binding, which would increase that number dramatically. Therefore, using PWMs to predict binding sites in the genome typically yields a large number of false positives. To complicate matters further, TFs cooperate with each other to increase their probability of occupancy to a regulatory element. As a result, many ChIP-Seq peaks do not contain a strong motif of the targeted TF[50].

In order to properly model TF binding, the binding of other TFs that cooperate with the TF of interest must also be accounted for. Algorithms to incorporate multiple PWMs have been developed[51,52]. However, the most successful models do away with PWMs entirely, and learn sequence features discriminatively[6,7,53]. These include kmer-based methods and convolutional neural networks (CNNs). I will describe these algorithms in detail in the next chapter, but they both have the ability to learn multiple binding sites in the training process of a single model. For the vast majority of ChIP-Seq datasets, kmer-based methods and CNNs outperform single PWMs[6,54]. However, they both have a downside, which is their interpretability. It is difficult to know what binding sites are being learned by these complex multiple TF models.

Chapter 3

Comparisons of Current Sequence-based Models

TF binding models representing a single TFs rarely achieve high performance when predicting ChIP-Seq peaks. Sequence-based models need to be complex enough to learn multiple binding sites simultaneously to classify regulatory elements with high predictive performance. Currently, there are two types of machine learning approaches that have been successful: gapped kmer-based models[6] and convolutional neural networks[7,55]. In this chapter, I will compare the performance of the linear gapped kmer approach gkm-SVM, and the non-linear CNN. I will describe the mathematical properties of both methods that give them their strengths and weaknesses. I will then generate synthetic sequences that mimic real regulatory sequences, which will be used to test the sensitivity and robustness of the models' capabilities to learn TF binding motifs. Next, I will evaluate the performance of gkm-SVM and CNN models on putative regulatory elements across many cell/tissue types. Lastly, I will compare their effectiveness in predicting variant effects.

3.1 Introduction

Kmer-based models use kmer counts as features to predict regulatory elements[53]. "kmers" are strings comprised of a given alphabet. In the case of genomes, the alphabet is just comprised of the nucleotides A, C, G, and T. The "k" indicates the length of the string (e.g. ATGTC is a 5-mer). The number of possible gapped kmers is 4^k, which means k does not need to be large to generate many features. For larger k, the potential to learn more TF binding motifs increases. However, the total number of kmers in the training set increases linearly with the size of the training set and the length of the sequences. Gandhi et al.[6,56] were able to overcome this problem by using gapped kmers, which are strings with an alphabet that includes a gap "-". These features have been shown to useful in describing both functional DNA and protein sequence motifs.[10,32,57–61] The gapped indicates positions to be ignored during the counting process. Gapped kmers are parameterized by two variables, (l, k), where l is the length of the string and k is the number of ungapped terms (e.g. AG-T- is a (5,3) gapped kmer). Each l-mer contains $\binom{l}{k}$ gapped kmers and there are $4^k\binom{l}{k}$ possible gapped kmers. In addition to the sequence length and number of training sequences, the gapped kmer counts also scales with $\binom{l}{k}$. Thus, gapped kmers manage the sparsity problem more effectively than kmers[56]. By using gapped kmers as features in a linear support vector machine, which they call gkm-SVM, Ghandi et al. were able to outperform conventional kmer methods across many ChIP-Seq datasets.

The other class of models are convolutional neural networks (CNNs). The input to a CNN is a one-hot encoded sequence, which is a matrix of dimension 4 by L where L is the length of the sequence. The 4 rows index the nucleotides A, C, G, and T and the column index the position. The entries of the matrix are either 1 or 0, indicating whether a position contains a particular nucleotide. For example, the one-hot encoded matrix of ACAGT is

$$ACAGT \rightarrow \begin{array}{c} A \\ C \\ G \\ T \end{array}\begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The first few layers of a CNN are comprised of convolution filters, which are mathematically identical to PWMs. Convolution filters are 4 by l matrices that are used to create the input to the next layer by creating scores through pointwise multiplication of the input in moving windows. The last couple layers contain dense and the output node. Other layers can be incorporated into the CNN, such as max pooling or dropout. In comparison to kmer-based approaches, many more CNNs have been created to predict regulatory elements, each with their own network structure. A consensus, optimal network structure has yet to be agreed upon.

3.1.1 An overview of the support vector machines

gkm-SVM uses gapped kmer counts as features in a linear support vector machine to predict regulatory elements. The support vector machine learns the hyperplane that optimally separates the positive and negative class (**Fig 3.1a**). The output of gkm-SVM is the vector $\boldsymbol{w}$ that parameterizes the hyperplane, which is a set of weights that indicate the predictive power of each gapped kmer (**Fig 3.1b**). In addition, there is an offset b. For a given data point i, $\boldsymbol{x_i}$ is the feature vector (in this case gapped kmer counts), and y_i is the class of i (either 1 or -1). To understand what the SVM is learning, we can look at the primal and dual formulations of the SVM. The primal problem is given below.

$$\text{minimize} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{subject to} \quad y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1, \forall i$$

The SVM is minimizing the size of $\boldsymbol{w}$ while maintaining a buffer between the plane and each point, which are called margins defined by $\boldsymbol{w}^T\boldsymbol{x} + b = 1$ and $\boldsymbol{w}^T\boldsymbol{x} + b = -1$ (the dotted lines in **Fig 3.1a**). This assumes that the data is linearly separable, but that is rarely the case. Thus, slack variables are regularly used to allow for misclassifications in the training process. For the sake of brevity, I will not include them here, as their inclusion does not contradict any of the arguments made here. The primal problem gives an intuition of what the SVM learns, but the dual problem below tells us what determines the parameters of the SVM.

$$\text{maximize} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \boldsymbol{x_i}^T \boldsymbol{x_j}$$

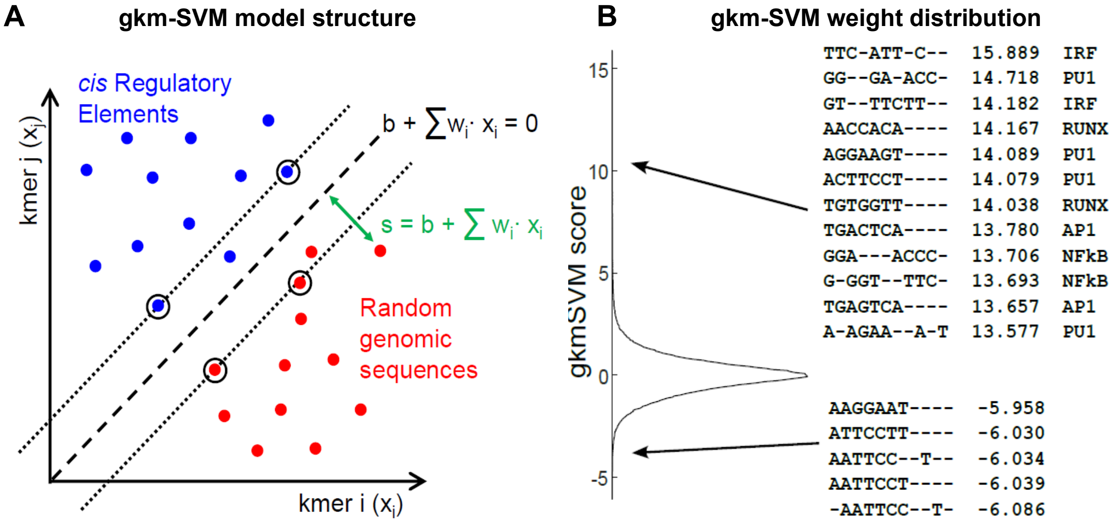
$$\text{subject to} \quad \alpha_i \geq 0 , \sum_i \alpha_i y_i = 0, \forall i$$

The dual problem is obtained by using Lagrange multipliers on the primal problem by optimizing the Lagrangian

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_i \alpha_i \big(1 - y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b)\big)$$

Where $\boldsymbol{\alpha}$ is the vector of Lagrange multipliers, and α_i is the Lagrange multiplier associated with the constraint $y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b) \geq 1$. The first constraint $\alpha_i \geq 0$ follows from the Karush-Kuhn-Tucker Conditions, which are necessary conditions for the feasibility of finding a solution for this type of problem. The second constraint arise from taking the partial derivative with respect to b and setting it equal to zero.

$$\frac{\partial}{\partial b} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \sum_i \alpha_i y_i = 0$$


A
gkm-SVM model structure
cis Regulatory Elements
kmer j (x_j)
$b + \sum w_i \cdot x_i = 0$
$s = b + \sum w_i \cdot x_i$
Random genomic sequences
kmer i (x_i)
B
gkm-SVM weight distribution
15
10
5
0
-5
gkmSVM score

TTC-ATT-C--	15.889	IRF
GG--GA-ACC-	14.718	PU1
GT--TTCTT--	14.182	IRF
AACCACA----	14.167	RUNX
AGGAAGT----	14.089	PU1
ACTTCCT----	14.079	PU1
TGTGGTT----	14.038	RUNX
TGACTCA----	13.780	AP1
GGA---ACCC-	13.706	NFkB
G-GGT--TTC-	13.693	NFkB
TGAGTCA----	13.657	AP1
A-AGAA--A-T	13.577	PU1

AAGGAAT----	-5.958
ATTCCTT----	-6.030
AATTCC--T--	-6.034
AATTCCT----	-6.039
-AATTCC--T-	-6.086

Figure 3.1 The support vector machine with gapped kmer features

A) *The support vector machine is a linear model that learns the hyperplane that best separates the positive and negative classes. It maximizes the distance between the hyperplane and the nearest points.* ***B)*** *A vector of weights parameterizes the hyperplane, which assigns the predictive value of each gapped kmer. Gapped kmers that have large positive weights are from TFBS motifs.*

When taking the partial derivative with respect to $\boldsymbol{w}$, we obtain an especially important quantity[1].

$$\frac{\partial}{\partial \boldsymbol{w}} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \boldsymbol{w} - \sum_i \alpha_i y_i \boldsymbol{x_i} = 0 \rightarrow \boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{x_i}$$

The weight vector is a linear combination of the gapped kmer counts of the data, and the Lagrange multipliers are the coefficients. Also, since $\alpha_i \geq 0$, we can determine which data points contribute to the solution. By the Karush-Kuhn-Tucker Conditions, we have that $y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b) - 1 \geq 0,\ \ \alpha_i \geq 0,\ \ \alpha_i(y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b) - 1) = 0,\ \ \forall i$. For point i, that lie outside of the margins, we have $y_i(\boldsymbol{w}^T \boldsymbol{x_i} + b) - 1 > 0$. Thus, $\alpha_i = 0$ for points outside the margin, implying that they do not contribute to $\boldsymbol{w}$. Therefore, the only points that contribute to $\boldsymbol{w}$ must lie on the margins, and hence, are called support vectors. The points that are the

[1] *Substituting this term into $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ also yields $\sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \boldsymbol{x_i}^T \boldsymbol{x_j}$.

most difficult to separate contribute to the solution. This is beneficial for classifying regulatory elements, especially enhancers, where most TFBSs differ from their consensus motifs.

3.1.2 An overview of the convolutional neural networks

Convolutional neural networks can be thought of as constrained dense nodes, which help prevent overfitting. In addition to TFBS motifs, they are capable of learning positional preferences within the sequence, something of which gapped kmers counts alone are incapable. Different kernels can be applied to the SVM, to allow for more non-linear decision boundaries, but these will not incorporate position preferences between sequence features. One can argue that CNNs can learn more than just those two features, but that has yet to be shown. Despite their capacity to learn non-linear patterns, there are some major drawbacks. The number of parameters is large, often comparable to the number of training data. This makes CNNs prone to overfitting in the context of regulatory genomics, since increasing the size of the training set is impossible due to the limited size of the genome. As a result, measures to avoid this must be taken, such as regularization, dropout, and max pooling. In addition, the error function is non-convex, and usually contains multiple local minima, which make different predictions, unlike the SVM. The algorithm used to find the optimal parameters is back-propagation, a dynamic programming method derived from gradient descent. Thus, different initializations of parameters do not necessarily converge to the global minima. Various measures have been developed to account for this, such as training multiple models and averaging the predictions or picking the best performing model. The lack of robustness in CNNs is a major drawback, since it is never certain whether the model learns the parameters that achieves the global minimum.

3.2 Methods

In the next sections, I will compare the predictive power of gkm-SVM and CNNs. Previous network structures have been created to be trained on all epigenetic marks/peaks from the ENCODE database. However, cell-type specific features tend to be lost in this type of CNN. Therefore, we developed a new CNN, that we named RCmax, that is optimized for training on a smaller number of peaks. I will compare gkm-SVM and RCmax on their ability to predict synthetic sequences, which are sequences embedded with transcription factor binding motifs[*]. I will then compare gkm-SVM, RCmax, and another CNN, ChromBPnet[62], on predicting real promoter and distal enhancer peaks. ChromBPnet is a CNN developed to predict the read profile of DNase/ATAC-Seq peaks. I will then evaluate the models' performance on predicting variant effect biological data in the form of DNase-1 sensitivity quantitative loci and saturation mutagenesis assays,

3.2.1 Optimization of parameters for RCmax

We designed a DNN architecture based on DeepSea, but modified network and regularization parameters for optimal performance when trained on a single experiment, using exactly the 300bp inputs we used to train gkm-SVM on DHS and ATAC-seq data (**Fig 3.2**). We also use as the first input layer both the peak sequence and its reverse complement and take the maximum convolutional filter score of the two sequences as input to the next convolutional filter layer. Because of this reverse complement, we call this algorithm RCmax. We also optimized the regularization parameters of RCmax to improve its performance over the Deep-Sea network design (Table 3.1).

[*] Nicholas Eng, during his time as an undergraduate, performed many of the simulations for gkm-SVM and RCmax comparisons

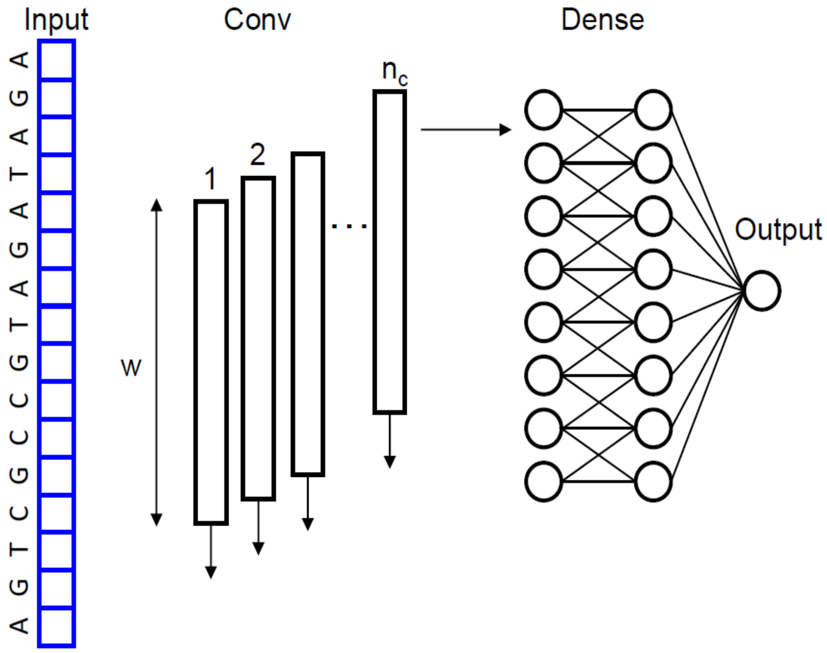


Figure 3.2 The network structure of RCmax

RCmax contains 3 convolution layers, followed by 2 dense layers. The input to the second layer is the maximum of the scores of the sequence and its reverse complement.

Table 3.1 Changes to the network structure of RCmax

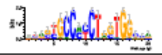
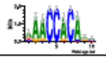
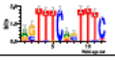
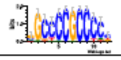
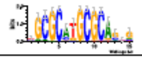
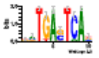
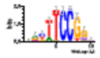
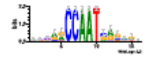
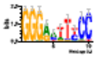
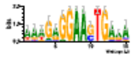
Network Configuration	DeepSea-like	RCmax
# of Convolution filters	360, 480, 960	200, 140, 9
Kernel Size	8, 8, 8	13, 10, 10
Dense Layer	925 nodes	100 nodes
Output Layer	1 node	1 node
L2 regularization factor	5^{-7}	10^{-9}
L1 regularization factor	10^{-8}	10^{-9}

3.2.3 Training gkm-SVM and RCmax on synthetic sequences and evaluating their performance

We created multiple sets of synthetic sequences and varied the overall motif frequency per sequence from 0.2 to 3 motifs per sequence. This creates a range of difficulties when learning binding sites. For this task, we were interested only in the model's ability to learn TF motifs, so no constraints on binding site position were imposed.

To simulate the motif distribution of real regulatory elements, we seeded 10 motifs with non-uniform relative frequencies (**Table 3.2**).

Table 3.2 The PWMs Seeded into Synthetic Sequences and their Relative Frequencies

CTCF: 0.15 	RUNX: 0.10 	IRF: 0.10 	SP1: 0.10 	NRF1: 0.05 
AP1: 0.15 	ETS: 0.10 	NFY: 0.10 	NFkB: 0.05 	PU.1: 0.05 

We trained gkm-SVM and RCmax models on all synthetic sequences against a 2 times larger negative set compromised of background DNA. The qualities of these models were evaluated by their ability to classify synthetic dsQTLs, which are pairs of synthetic sequences, where one contains a strong motif of a given TF, and the other contains a variant to disrupt the motif. A 50 times larger negative set was generated by mutating a random base in background DNA. gkm-SVM models were trained on each synthetic training set, and variant effect scores were generated on the synthetic dsQTLs using deltaSVM. Since RCmax does not always converge to the same set of parameters, we trained 10 different RCmax models and averaged their scores. We used the area under the precision-recall curve (AUPRC) as a performance metric.

3.2.4 Processing DNase/ATAC-Seq and ChIP-Seq datasets

Following our standard analysis pipeline, we called MACS2 peaks (with $p=10^{-9}$) after combining replicates of ENCODE 2[12], ENCODE 3[63], ENCODE 4, and Roadmap[64] human and mouse DHS, ATAC-Seq, and ChIP-Seq for hg38 downloaded from www.encodeproject.org. We separated MACS2 peaks into promoter peak sets by their distance to a transcription start site (TSS). Since promoters could be found up to 2kb from TSS, we called all peaks a promoter if they were within 2kb of a GENCODE v24 TSS.

The peaks that were greater than 2kb of a TSS are comprised of enhancers and CTCF. To remove as many CTCF peaks as possible, we removed DNase/ATAC-Seq peaks that were called in 30 percent of the datasets. To define sequence sets, we extended +/−150 bp from each MACS2 summit.

3.2.5 Training and evaluating the performance of gkm-SVM, RCmax, and ChromBPnet

We evaluated the models on their prediction accuracy of regulatory elements. We ran gkm-SVM using default parameters l=11 k=7 d=3 t=2) using the gkm-SVM R-package and ls-gkm for large training sets. To prevent overfitting[65], We used 5 similar sized chromosomal test-set splits (1,3,6; 2,8,9,16; 4,11,12,15,Y; 5,10,14,18,20,22; and 7,13,17,19,21,X). For each CV split, we trained gkm-SVM and RCmax on the top 10,000 peaks (in terms of MACS signal) against an equal sized negative set of genomic background matched to its GC content and common genomic repeat frequency. Because RCmax does not always converge to a good model, we restarted RCmax if the training set area under the receiving curve (AUROC) was less than 0.8. We trained ChromBPnet (https://github.com/kundajelab/chrombpnet) on all DNase/ATAC-Seq peaks and used predicted counts as a feature to calculate the AUROC. We evaluated gkm-SVM and ChromBPnet's performance when predicting dsQTLs using the same method in Lee et al. To summarize, dsQTLs from lymphoblastoid cell lines that were within 50 bp of a lymphoblastoid DHS peak were used as a positive set (579 total). SNPs that were not-significantly associated with dsQTLs were used as a negative set. The ratio of the positive to negative set is 1 to 50. A gkm-SVM model trained on GM12878 DHS peaks was used to score the variants. We used the absolute value of the deltaSVM score since we did not use sign of the change in accessibility in the prediction task. A ChromBPnet model trained on ATAC-Seq peaks was used to score the dsQTLs. The absolute value of the difference in counts between the reference and alternate alleles was used to score the dsQTLs.

3.2.6 Evaluating the performance of models when predicting saturation mutagenesis variant effects

This work was done for the Critical Assessment of Genome Interpretation 5[66] Regulation Saturation Challenge for which Michael A. Beer and I were asked to be evaluators. The challenge was to predict saturation mutagenesis variant effects of 15 regions (10 promoters and 5 enhancers[17]) with a limited number of training data relative to the test set (20-80 training to test set ratio). An example of the saturation mutagenesis data is given in **Fig 3.3**. Seven groups submitted machine learning models to predict the variant effects. We also decided to test gkm-SVMs performance on this task by using deltaSVM scores from models trained on the same or a similar cell-type to the cell-type on which the experiment was done. Performance was measured with the Pearson correlation of the predicted and experimental variant effects.

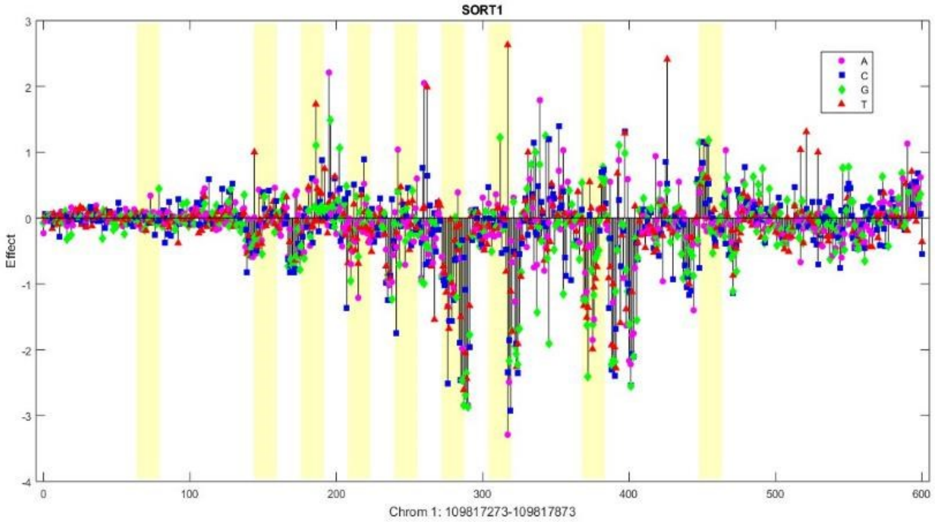


Figure 3.3 Saturation mutagenesis of the SORT1 enhancer in HepG2 cells

Saturation mutagenesis is a type of massively parallel reporter assay (MPRA) that measures the impact of variants in a genomic region. To generate all variants, a universal nucleotide is added at a low concentration relative to the other bases (usually around 0.01 of the total base concentration). The result is a set of sequences that have a few random mutations. MPRA is ran on these sequences, and the signal of each sequence is compared to the original sequence. The impact of each variant is estimated by using linear regression. This stem plot shows the linear regression weight of each variant. Areas with stretches of negative values are indicative of TF binding sites. In the CAGI 5 challenge, participants were given the regions in yellow to train models to predict the rest of the enhancer.

3.2.7 Combining models trained on different biochemical assays to improve predictions of saturation mutagenesis assays

Many groups used various features in combination to predict variant effects. The most popular model used to combine them was a random forest regressor. We trained random forest regression models and LASSO regression models using deltaSVM scores of enhancer, promoter, and gkm-SVM ChIP-Seq models. We used the 20-80 training-test set ratio given in the challenge and a 50-50 training-test set ratio that we generated ourselves. Performance was measured with the Pearson correlation of the predicted and experimental variant effects.

3.3 Results

3.3.1 Evaluating models on synthetic sequences

Our goal for this first task is to evaluate gkm-SVM and RCmax's ability to learn multiple TFBSs simultaneously. By varying the number of motifs per sequence, we can precisely measure the minimum amount of information that is needed to learn each motif. In addition, having the models predict individual TF motif disrupts allows us to determine the motif with which each model struggles. In **Fig 3.4**, it is apparent that gkm-SVM vastly outperforms RCmax for this task. gkm-SVM learns TF motif information even at the lowest motif frequency (**Fig 3.4a**). On the other hand, RCmax required at least 0.7 motifs per sequence for it to perform better than random assortment for any of the motifs (**Fig 3.4b**). When the number of seeded motifs is less than the number of sequences, neural networks are likely to find features that do not generalize because they will try to learn features from sequences with no motifs to decrease the training error. This seems to suggest that the performance of RCmax is overly sensitive to the quality of the training set. However, even when the number of motifs per sequence is high, gkm-SVM learns motifs more effectively than RCmax (**Fig 3.4C**). We believe that this results from the SVM using the sequences that are the most difficult to classify to form its decision boundary, whereas RCmax may not necessarily need as detailed information of individual motifs to reduce the training error. From this, we concluded that gkm-SVM is more proficient at learning TF binding information. However, gkm-SVM tends to perform similarly when training and testing on real data. In the next section, we will assess the differences between gkm-SVM and RCmax on biological data.

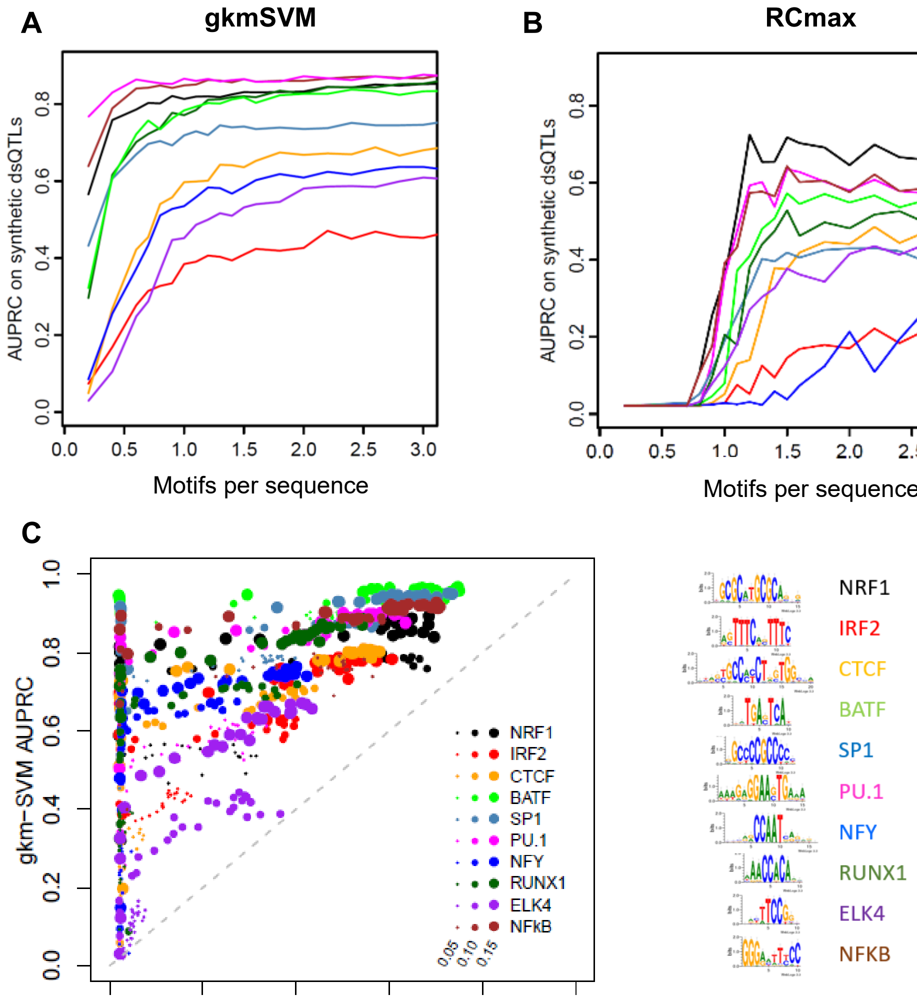


Figure 3.4 gkm-SVM and RCmax performances on predicting synthetic dsQTLs.

*gkm-SVM (**A**) more effectively and robustly predicts synthetic dsQTLs than RCmax (**B**). As the motif frequency increases, gkmSVM is quicker to obtain its max AUPRC for all TF motifs. RCmax needs a higher motif frequency to obtain an AUPRC higher than random assortment (0.02). In addition, gkm-SVM yields a higher AUPRC across all motifs and motif frequencies (**C**).*

3.3.2 Classifying real peaks from chromatin accessibility assays

To compare the SVM and CNN, we gathered a large number of datasets that encompassed as many cell-types as possible. We downloaded 1270 DNase-Seq and 374 ATAC-Seq datasets from the ENCODE portal and split them into promoter and distal enhancer datasets. Since enhancers bind cell-type specific TFs, we focused much of our comparisons to the distal enhancer datasets. During the training process for RCmax, we attempted to produce the best possible RCmax models as we could by retraining RCmax with a different initialization if the model had a training AUROC below 0.80. The results are quite different from the synthetic dataset predictions (**Fig 3.5A**). RCmax seems to slightly outperform gkm-SVM when the dataset is easy to learn (AUROC > 0.90). However, for datasets below that threshold, gkm-SVM vastly outperforms RCmax. We believe that datasets with lower AUROCs may contain more sequences without regulatory potential, and thus lack TFBSs. Similar to the synthetic sequence analysis, the CNN will attempt to find features in sequences that have no TFBSs, and thus, learn irrelevant features. On the other hand, gkm-SVM's weight vector is a linear combination of the data points that are difficult to separate from the other class. Although sequences without TFBSs will contribute to the weight vector, their influence on the decision boundary will be limited by other sequences with regulatory potential.

Because gkm-SVM and RCmax outperform each other in under different circumstances, we decided to see if we could create a superior model by combining them. We transformed RCmax scores by using the logit function (inverse of a logistic function) and added the scores together. The combined model, which we named gkm-DNN hybrid, outperforms both gkm-SVM (**Fig 3.5B**) and RCmax over all ranges of dataset quality. This suggests that RCmax and gkm-SVM are learning different features. It is difficult to determine whether RCmax is learning spatial preferences between TFBSs, since the negative set is comprised of sequences with low frequencies of TFBS motifs. It could be possible that RCmax learns features that we are not aware of in the negative set better than gkm-SVM.

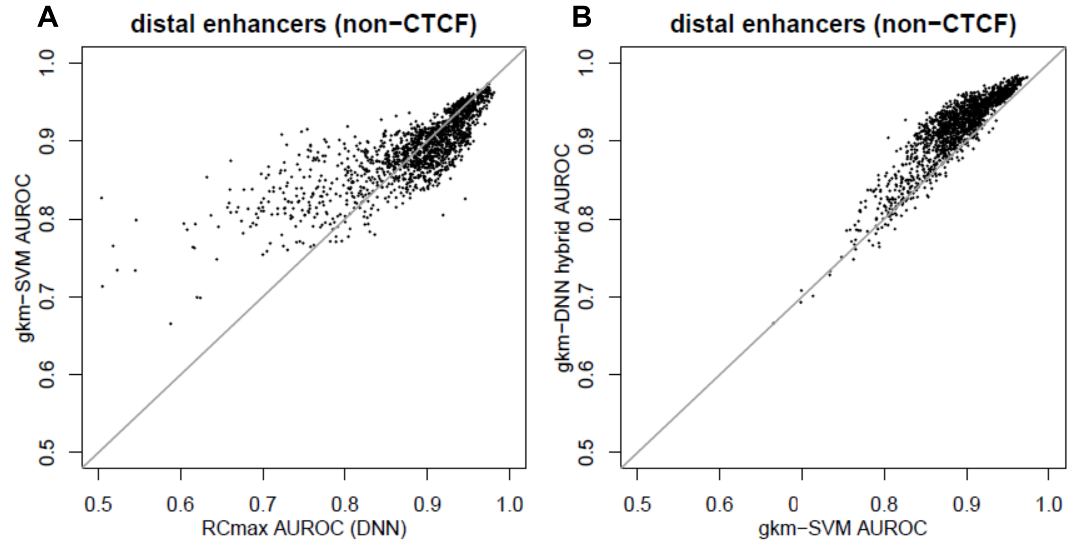


Figure 3.5 gkm-SVM vs RCmax performances for classifying distal enhancers.

***A**) For datasets that are easy to classify (AUROC > 0.90), RCmax appears to slightly outperform gkm-SV. However, for datasets below that threshold, gkm-SVM vastly outperforms RCmax. **B**) Combining the predictions scores of gkm-SVM and RCmax (y-axis) consistently produces a better model than gkm-SVM.*

Since there are many CNNs that use sequence to predict regulatory elements, we decided to compare gkm-SVM against another model, ChromBPnet. ChromBPnet is a CNN that predicts the read profile (the peak shape) of a regulatory element from only the sequence. We trained ChromBPnet with the default parameters on all 1644 promoter and distal enhancer datasets and used peak counts as a score to calculate AUROCs. ChromBPnet predicts promoters more effectively than gkm-SVM (**Fig 3.6A**). However, it struggles to learn distal enhancers (**Fig 3.7B**). We think that ChromBPnet learns promoters more effectively than distal enhancers because promoters have very large DHS/ATAC peaks compared to distal enhancers. Promoter features will be weighed more than distal enhancer features to increase the predicted signal at promoters. We would also like to add the promoter datasets are highly redundant because most promoters are accessible across all cell-types, so it is highly likely that a model performing well on one promoter dataset implies that it would perform well on all promoter datasets.

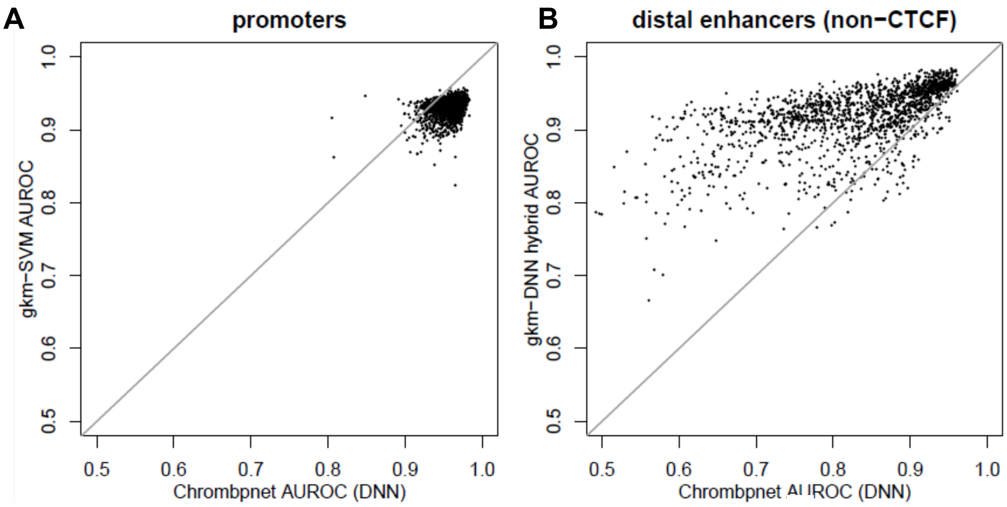


Figure 3.6 ChromBPnet performance when predicting promoter and distal enhancers.

***A**) ChromBPnet is able to predict promoters slightly better than gkm-SVM. **B**) However, it struggles to predict distal enhancers, especially when compared to the gkm-DNN hybrid.*

CC

We also compared gkm-SVM and ChromBPnet on predicting dsQTLs, which are variants that are statistically associated with large changes in chromatin accessibility. We trained a gkm-SVM model on a GM12878 DNase-Seq dataset on all peaks. We used a ChromBPnet model trained on a GM12878 ATAC-Seq dataset to compare against. dsQTLs are comprised of all types of regulatory elements, including promoters, enhancers, and insulators, so the biases to each model's strengths and weaknesses should be limited. We expected ChromBPnet to perform well on this task since it performs regression on the ATAC signal, rather than just classifying peaks versus non-peaks (**Fig 3.7** blue curve). When using the model from Lee et al., gkm-SVM performs poorly (**Fig 3.7** red curve). We noticed that many large delta-SVM scores occurred in regions with very low predicted activity or very high predicted activity. Strengthening a single binding site in a region without other binding sites, or weaking a single site in a very strong enhancer is very unlikely to change the accessibility by a large margin. Therefore, we applied a logistic transformation to gkm-SVM scores prior to running deltaSVM to flatten very large or very low gkm-SVM scores. This change doubled the AUPRC of delta-SVM (**Fig 3.7** black curve). Despite these results, there is a limitation of this dataset, in that many of these variants are not likely to be the causal factor in the change in accessibility. dsQTLs, like GWASs, suffer from linkage disequilibrium. Furthermore, the samples used for this study were from only 91 patients, in contrast to GWAS, which typically has thousands of participants.

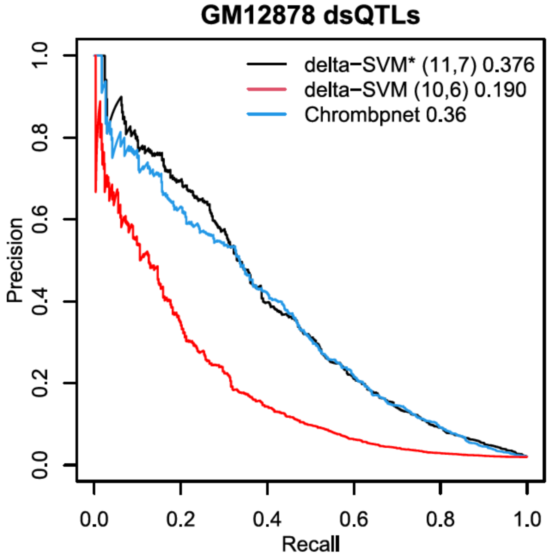


Figure 3.7 gkm-SVM and ChromBPnet's precision-recall curves for predicting lymphoblastoid dsQTL's

gkm-SVM and ChromBPnet performance when predicting significant changes in chromatin accessibility for Lymphoblastoid cell lines. The predictions from Lee et al. are in red, where deltaSVM scores without any modifications were made. In black, the deltaSVM scores are modified by applying a logistic transformation to the alternate and reference allele scores. This version of deltaSVM slightly outperforms ChromBPnet.

3.3.3 Predicting saturation mutagenesis activity with sequence-based models

In the CAGI 5 Regulation Saturation Challenge, participants developed machine learning models to predict the results of saturation mutagenesis data on 10 promoters and 5 enhancers[9] (**Fig 3.8A**). As the evaluators of the challenge, we were tasked to investigate the reasons why groups were able to have success. Most groups used non-linear machine learning methods like random forests to combine multiple biological features, to varying degrees of success[67,68]. We associated the features that each group used to the level of success. We noticed that groups who used DeepSea features had considerably better results than the other groups (**Fig 3.8B**). They integrated the variant effect scores of DeepSea models trained on different cell-types and epigenetic marks. It seemed that sequence-based models were contributing largely to the predictive power of those group's models.

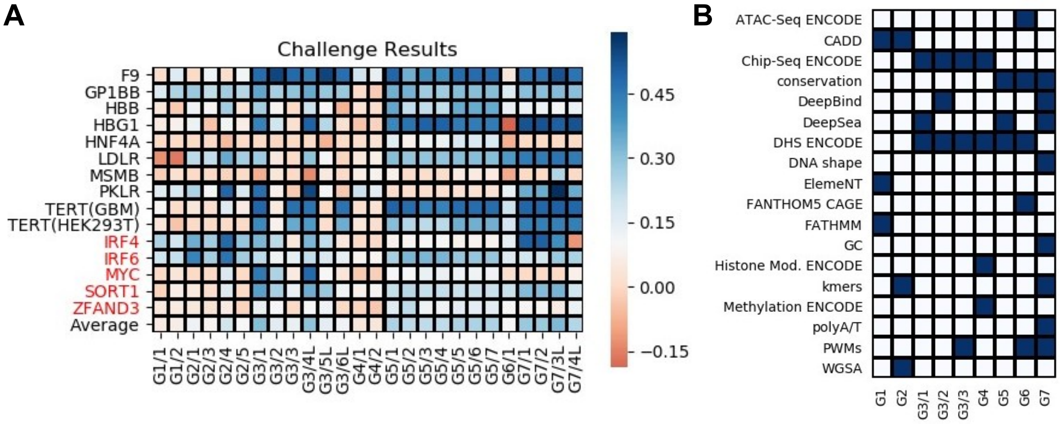


Figure 3.8 CAGI 5 Regulation Saturation results

A) The correlations of the predicted and actual variant effects for each group. The rows indicate the promoter (black) or enhancer (red). The columns indicate the group number (left) and the submission (right). B) The features that were used in each group's models. The three groups who had the most success (3, 5, 7) all used DeepSea features in their models.

We decided to use delta-SVM scores to predict the variant effects of each region. We had three primary approaches. 1) We would use one gkm-SVM model trained on a single DNase-Seq dataset the same or closely matching cell-type as was done in the experiment. 2) We would combine multiple gkm-SVM models from DNase-Seq and ChIP-Seq[2*] by using a random forest regressor by training on the same variants given to the

[*] At the time of the challenge, the datasets from ENCODE phase 4 were not available. We used 676 promoter, 654 enhancer, and 1044 ChIP-Seq datasets.

participants. 3) We could combine multiple gkm-SVM models with a LASSO regression model, which would allow us to see which models are contributing to the predictive power.

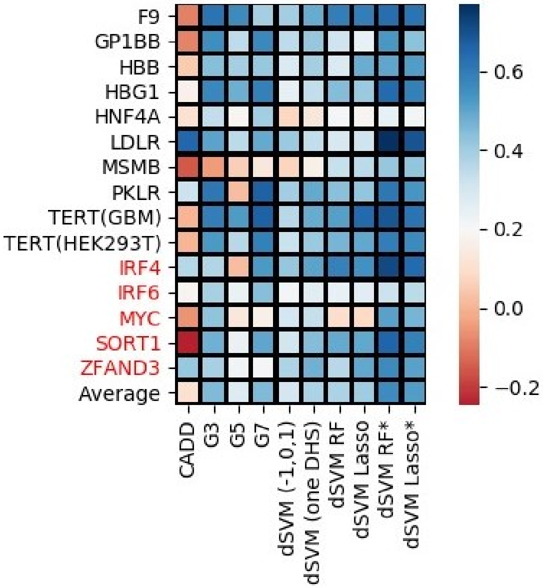


Figure 3.9 Combining datasets increases the performance of delta-SVM when predicting variants

Predictive performance of saturation mutagenesis variant effects increases with training random forest and LASSO regression models that combine multiple delta-SVM scores derived from various models. CADD, which had been the best model to predict variants, is vastly outperformed by the other models. Columns 2-8 show the performance of models trained on 20 percent of the region. Columns 9-10 are delta-SVM regression models trained on 50 percent of the region.

For all regions, there was a noticeable improvement in the performance of delta-SVM when combining multiple models (**Fig 3.9**). Interestingly, the random forest only slightly outperformed the LASSO regression model. This was promising since LASSO regression is a very interpretable model. The most predictive models for each region are given in **Table 3.3**. For enhancers, the most predictive models were trained on datasets from the same or similar cell-type. On the other hand, promoter was well predicted by datasets from a wide range of cell-types. Interestingly, ChIP-Seq datasets that targeted GABPA and NFY were commonly given high regression weights across promoters.

Table 3.3 gkm-SVM models with the largest LASSO regression weights for predicting saturation mutagenesis variant effects

F9 (HepG2)	**LDLR** (HepG2)	**IRF4** (SK-MEL-28)
HepG2: ETV4 ChIP-seq HepG2: 3xFLAG-KAT8 ChIP-seq K562: FOXK2 ChIP-seq	MCF-7: SREBF1 ChIP-seq HepG2: 3xFLAG-SP5 ChIP-seq HEK293: eGFP-SP3 ChIP-seq	SK-MEL-5: DHS enhancers foreskin melanocyte: DHS enhancers (Roadmap) GM12878: RAD51 ChIP-seq
GP1BB (HEL 92.1.7)	**MSMB** (HEK293T)	**IRF6** (HaCaT)
K562: GABPB1 ChIP-seq CMK: DHS enhancers K562: GATA2 ChIP-seq	HEK293: eGFP-PRDM6 ChIP-seq K562: ATF2 ChIP-seq adrenal gland: DHS enhancers	foreskin keratinocyte: DHS enhancers (Roadmap) bronchial epithelial cell: DHS enhancers keratinocyte: DHS enhancers
HBB (HEL 92.1.7)	**PKLR** (K562)	**MYC** (HEK293T)
HEK293: eGFP-SP2 ChIP-seq L1-S8R: DHS enhancers K562: NFYB ChIP-seq	K562: DHS enhancers MCF-7: eGFP-KLF9 ChIP-seq liver embryo: DHS enhancers	HeLa-S3: CTCF ChIP-seq HepG2: RAD21 ChIP-seq CWRU1: DHS enhancers
HBG1 (HEL 92.1.7)	**TERT** (GBM)	**SORT1** (HepG2)
GM12878: NFYB ChIP-seq HEK293: eGFP-KLF1 ChIP-seq K562: NFYB ChIP-seq	HepG2: GABPA ChIP-seq CMK: DHS enhancers HepG2: 3xFLAG-SP5 ChIP-seq	HepG2: CEBPB ChIP-seq HepG2: FOXA1 ChIP-seq K562: eGFP-ZNF148 ChIP-seq
HNF4A (HEK293T)	**TERT** (HEK293T)	**ZFAND3** (MIN6)
K562: NFYB ChIP-seq K562: EGR1 ChIP-seq K562: eGFP-ZFX ChIP-seq	K562: GABPB1 ChIP-seq HepG2: GABPA ChIP-seq HepG2: 3xFLAG-GABPA ChIP-seq	adrenal gland: DHS enhancers A549: USF1 ChIP-seq brain embryo: DHS enhancers

Compared to Groups 3, 5, and 7, deltaSVM performance was slightly weaker when predicting promoters, and stronger when predicting enhancers, like the results in section 3.3.2. Interestingly, for many of the promoter regions, the vast majority gkm-SVM models trained on DHS promoters, could predict the variant effects with fairly high accuracy. We assessed each individual gkm-SVM DHS enhancer and promoter models on every region. For enhancer regions, the best performing model was from an enhancer model. Similarly, the best performing model for promoters was from a promoter model. However, when we compare the average performances of all models, most promoter models can predict promoter regions nearly as well as the best performing one. This is not true for the enhancer regions.

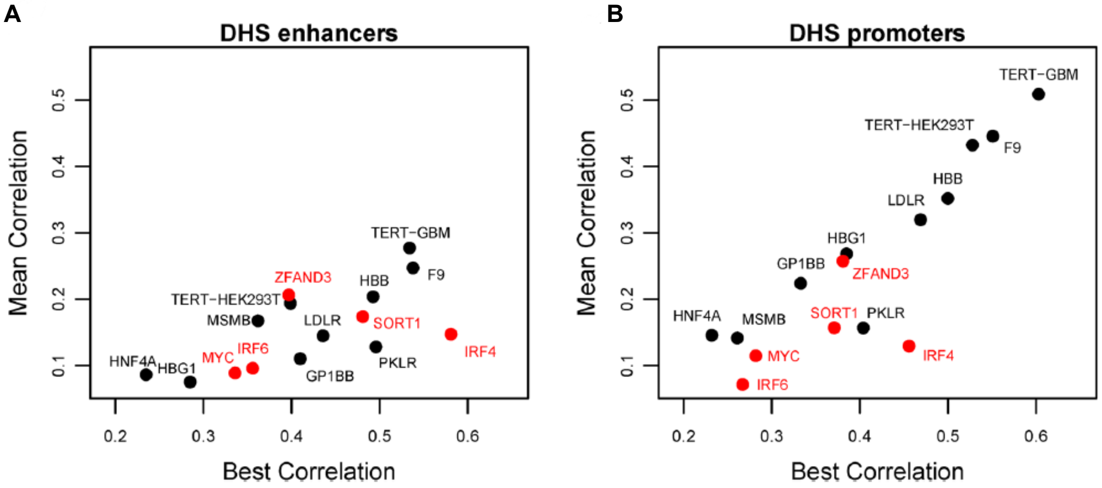


Figure 3.10 Comparing the best performing model to the average performance of all models

The models that best predict promoter and enhancer regions are trained on peaks from promoters and enhancers respectively. ***A****) However, most DHS enhancer models' performance does not match the performance of the best performing one.* ***B****) Most promoter models can predict promoter regions with similar accuracy to the best performing model.*

3.4 Discussion

In this section, I gave an overview of the support vector machine and the convolutional neural network. I discussed their respective strengths and weaknesses and how those may manifest themselves in predicting regulatory activity. I then tested both methods in a controlled manner by generating synthetic datasets of varying difficulty to find the limits of their predictive power. Lastly, I tested their performances on predicting real biological data, and investigated what types of data on which each method performed well.

It seems clear that gkm-SVM is better capable of learning TF binding motifs, especially when the data quality is poor. We can see this in both the synthetic and real datasets, where datasets with low motif frequency or AUROC were better predicted by gkm-SVM. However, even though CNNs do not learn motifs as well as gkm-SVM, they are clearly learning features that are able to compensate for this deficiency for high quality datasets. Interestingly, CNNs learn promoters more accurately than gkm-SVM, whereas the gkm-SVM learns enhancers more accurately. It is difficult to determine the precise reason for this. The motifs in promoters may not be as degenerate as they are in enhancers, and thus may be easier to learn. There could indeed be spatial preferences between binding sites that are learnable. Nevertheless, there are still improvements that can be made to sequence-based models.

Chapter 4

Extracting Transcription Factor Binding Site Motifs from Sequence-Based Models

In the previous chapter, I focused the analysis of sequence-based models on their predictive power. One very important fact that I have not addressed is the interpretation of their parameters. It is very difficult to integrate the parameters in a neural network into compact, understandable features. Many different algorithms have been developed to address this problem, but there is no consensus on a single algorithm or methodology to extract features from a neural network. In contrast to neural networks, the linear SVM is highly interpretable because features with large positive or negative weights are predictive of one of the classes. However, in the case of gkm-SVM, the parameters (in particular l and k) required to learn sequence features with high accuracy result in an extremely large number of gapped kmers $4^k\binom{l}{k}$. For $(l,k) = (10,6)$ and include reverse complement equivalence (e.g. $AAA{-}{-}{-} = {-}{-}{-}TTT$), the total number of gapped kmers is approximately 500,000. Complicating things further, many gapped kmers with high weights are from the same TF motif (**Fig 3.1B**). In order to find all the motifs that were learned by the model, one must search through a very large number of gapped kmers (**Fig 4.1**) and associate each gapped kmer to one out of hundreds of TF motifs. To definitively say that motif is learned by the model, there must be a large number of gapped kmers with high enough weight to statistically support that claim.

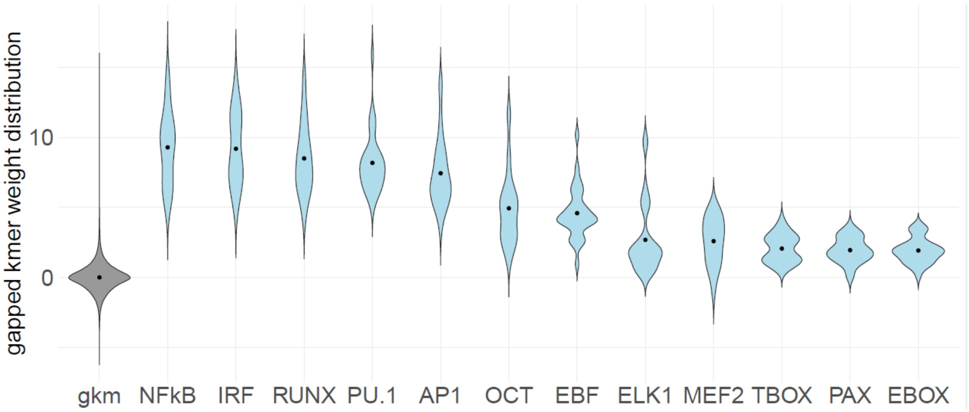


Figure 4.1 The weights of the top gapped kmers for TFs in GM12878 distal enhancers

In gray is the gapped kmer weight distribution (in units of standard deviation) of a GM12878 distal enhancer model. In blue are the weights of the top 30 gapped kmers associated with the motif on the x-axis. Although some TFs are easy to find at the top of the weight distribution (Fig 3.1B), the weights of gapped kmers for a given TF are often located closer to the mean.

Without knowledge of the binding sites that are learned in the model, we are stuck with a black box. At best, we can make predictions of what they were trained to do, which is predict the regulatory potential of a sequence. Extracting TF binding information from these models can extend their utility to other mathematical questions, such as constructing gene regulatory networks.

In this chapter, I mathematically derive a method to extract TF binding motifs from sequence-based models in the form of position weight matrices. I show that the gapped kmer weights of gkm-SVM are a linear combination of the gapped kmer distribution of the PWMs embedded in the training set sequences. I developed two methods of motifs extraction: 1) Identifying the PWMs that best explain the gkm-SVM weight vector from a set of pre-trained PWMs (called gkmPWMlasso) and 2) Learning the PWMs *de novo* (called gkmPWM). At the end of the chapter, I will demonstrate that gkmPWM learns motifs with higher sensitivity than other methods on both synthetic and real datasets.

4.1 Mathematical derivation of gkmPWM

Suppose that we had a large number of regulatory sequences. We are interested in observing the frequency of gapped kmer counts of this set of sequences relative to what we would observe in the background. As a reminder, a gapped kmer is a string that is comprise of elements from the alphabet $\{A, C, G, T, -\}$, where $-$ are ``gaps" in the string.

We define a (l, k) gapped kmer to have l total elements from the alphabet (i.e. the length of the string), and $l - k$ gaps. For example, $ACT{-}AA{-}G$ is a (l, k) gapped kmer. For a given sequence, we can count all possible gapped kmers and store the counts in a large vector. Define S as a set of sequences with regulatory activity and bounded length and let $\boldsymbol{G}(S)$ be the function that maps S to its gapped kmer count vector, which contains the sum of gapped kmer counts of all sequences in S. When normalizing over the length of the average sequence in S, we obtain $E[\boldsymbol{G}(S)]$, the expected gapped kmers counts of S. We are interested in the gapped kmers that are significantly higher in frequency in $E[\boldsymbol{G}(S)]$ compared what we expect the in entire genome. These gapped kmers are indicative of transcription factor binding sites. We could infer what TFs bind to them by quantitatively associating each gapped kmer to a TFBS. However, as the length l of the gapped kmer becomes larger, the number of gapped kmers scales as $4^k \binom{l}{k}$. Thus, our goal is to computationally reduce the complexity of a gapped k-mer distribution with larger pairings of (l, k) into more compact representations of TFBSs. We call this algorithm **gkmPWM**, which learns the frequency of multiple TFBSs (in the form of position weight matrices) from the gapped kmer counts of a set of sequences. It can either utilize a collection of pretrained PWMs or learn the PWMs *de novo*.

In this section, we will show:

1. A derivation of $E[\boldsymbol{G}(S)]$ as a probabilistic model of a linear combination of the gapped k-mer counts of individual TFBS models, which we denote as θ.

2. How to map each θ to gapped k-mer count vector. Our prime example will be a position weight matrix, but this model generalizes to any TFBS model that generates kmers.

3. An algorithm to learn the TFBSs that are in a set of regulatory elements *de novo*.

4. A method to extract TFBS features sequence-based models.

As I go through each section, I will reference the specific function that implements the mathematics from the gkmPWM repository (https://github.com/shigakiD/gkmPWM). There is a MATLAB and a C version of each function.

4.1.1 Counting gapped kmers

In this section, we will show that the gapped kmer composition of set S contains information about the frequency of TFBSs. If we sum the gapped kmer counts of all sequences in S and normalize by the number sequences S, we obtain the expected gapped kmer counts $E[\boldsymbol{G}(S)]$. We define $\{s_i\}$ to be the set of unique sequences in S, $\boldsymbol{G}(s_i)$ to be the gapped kmer count vector of s_i,and $P(s_i)$ to be the fraction of sequences equal to s_i in S. We can express $E[\boldsymbol{G}(S)]$ as a linear combination of the $\boldsymbol{G}(s_i)$.

$$E[\boldsymbol{G}(S)] = \sum_{i=1}^{m} P(s_i)\boldsymbol{G}(s_i)$$

where $m = |s_i|$. $E[\boldsymbol{G}(S)]$ can be used as an estimate of the gapped k-mer distribution of regulatory elements $P(\boldsymbol{g}|RE)$ by scaling by the mean the number of gapped kmers of each s_i, $\tilde{n} = \sum_{i=1}^{m} P(s_i)n_i$, where n_i is the number of gapped kmers in s_i.

$$P(\boldsymbol{g}|RE) = \frac{1}{\tilde{n}} E[\boldsymbol{G}(S)]$$

To learn individual features from $E[\boldsymbol{G}(S)]$, we will model the gapped kmer counts of $P(\boldsymbol{g}|RE)$. We assume that regulatory elements are comprised of DNA from transcription factor binding sites or background DNA. If we randomly select a gapped kmer from a regulatory element, it can only be picked from either from one of the individual binding site models θ_j or the background BG. The probability of picking a gapped kmer from a TFBS θ_j is donated as $P(\theta_j)$ and the probability of picking a gapped kmer from the background is denoted as $P(BG)$. It follows that

$$P(BG) + \sum_j P(\theta_j) = 1$$

(Eq. 1)

We can express $P(\boldsymbol{g}|RE)$ as

$$P(\boldsymbol{g}|RE) = P(\boldsymbol{g}, BG|RE) + \sum_j P(\boldsymbol{g}, \theta_j|RE)$$

We will assume that the gapped kmer distribution of background DNA and all θ_j are independent of whether they are in a regulatory element or not. Therefore, we can remove the condition on RE.

$$P(\boldsymbol{g}|RE) = P(\boldsymbol{g}, BG) + \sum_j P(\boldsymbol{g}, \theta_j)$$

By Bayes rule,

$$P(\boldsymbol{g}|RE) = P(BG)P(\boldsymbol{g}|BG) + \sum_j P(\theta_j)P(\boldsymbol{g}|\theta_j)$$

(Eq. 2)

Substituting (Eq.1) into (Eq. 2) yields

$$P(\boldsymbol{g}|RE) = P(BG)P(\boldsymbol{g}|BG) + \sum_j P(\theta_j)P(\boldsymbol{g}|\theta_j)$$

$$= \left(1 - \sum_j P(\theta_j)\right)P(\boldsymbol{g}|BG) + \sum_j P(\theta_j)P(\boldsymbol{g}|\theta_j)$$

$$= P(\boldsymbol{g}|BG) + \sum_j P(\theta_j)\left(P(\boldsymbol{g}|\theta_j) - P(\boldsymbol{g}|BG)\right)$$

$$P(\boldsymbol{g}|RE) - P(\boldsymbol{g}|BG) = \sum_j P(\theta_j)\left(P(\boldsymbol{g}|\theta_j) - P(\boldsymbol{g}|BG)\right)$$

The difference of gapped kmers distributions between regulatory elements and the background is a linear combination of the differences of the gapped kmer distributions of the individual binding site models θ_j and the background. We can bring this to a more workable form. Multiplying both sides by $\tilde{n}$,

$$E[\boldsymbol{G}(S)] - P(\boldsymbol{g}|BG) = \tilde{n}\sum_j P(\theta_j)\left(P(\boldsymbol{g}|\theta_j) - P(\boldsymbol{g}|BG)\right)$$

We can use the same arguments for $E[\boldsymbol{G}(S)]$ to get an estimate of the gapped kmer distribution of background DNA. We call B the set of sequences sampled from the background and b_i to be an element of B. We then have $E[\boldsymbol{G}(B)] = \sum_{\boldsymbol{i}} P(b_i)\boldsymbol{G}(b_i) = \tilde{n}P(\boldsymbol{g}|BG)$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] = \tilde{n}\sum_j P(\theta_j)\left(P(\boldsymbol{g}|\theta_j) - P(\boldsymbol{g}|BG)\right)$$

(**Eq. 3**)

We will use the left hand to estimate the gapped kmer counts directly from data. The right-hand side will be used to computationally reduce the complexity of the left-hand side to individual TFBS models. Note that the form of (Eq. 3) is a regression problem.

$$y - \tilde{y} = \sum_j w_i(x_i - \tilde{x}_i)$$

Each $P(\boldsymbol{g}|\theta_j)$ is a feature (independent variables), $E[\boldsymbol{G}(S)]$ is the dependent variable, and $E[\boldsymbol{G}(B)]$ and $E[\boldsymbol{G}(B)]$ behave like``means". The $P(\theta_j)$ are the regression weights. Thus, if we model $P(\boldsymbol{g}|\theta_j)$, we can use linear regression techniques to learn the $P(\theta_j)$ to give us the frequency of each TFBS in the set of regulatory elements. However, we must still choose a sequence generating TFBS model to obtain $P(\boldsymbol{g}|\theta_j)$. Luckily, the simplest model, the position weight matrix (PWM), has very convenient properties that allow for easy and fast calculation despite the number of gapped kmers with which we must work.

4.1.2 Mapping PWMs to gapped kmers

In this section, we will describe how to obtain gapped kmer distributions from position weight matrices (implemented in PWM2kmers). The position weight matrix is a TFBS model that contains the frequencies of each nucleotide at every position. The rows are indexed by $\{A, C, G, T\}$ and the columns are indexed by the position. The entries in each column sum to one. A key property of the PWM is that it assumes that the columns are conditional independent, i.e. a 0th order Markov model. Thus, calculating the probability of a kmer being generated by the PWM is merely multiply the correct entry in each position. For example, consider the following PWM.

$$\theta = \begin{array}{c} A \\ C \\ G \\ T \end{array} \begin{bmatrix} 0.2 & 0.1 & 0.1 & 1 & 0.1 \\ 0.2 & 0.7 & 0.8 & 0 & 0 \\ 0.3 & 0.1 & 0 & 0 & 0 \\ 0.3 & 0.1 & 0.1 & 0 & 0.9 \end{bmatrix}$$

The probability of generating the kmer $ACCAA$ is

$$P(ACCAA|\theta) = (0.2)(0.7)(0.8)(1)(0.1) \ = \ 0.0112$$

To calculate the probability of generating a gapped kmer, we sum over all the probabilities of generating the full kmers that contain that gapped kmer and normalize by the number of gapped kmers contain a full kmer $\binom{l}{k}$. Because of the distributive property, summing over the probabilities can be done by ignoring the gapped positions because we can factor out the ungapped positions, and what remains sums to one. For example,

$$P(AC\text{–}AA|\theta) = \frac{P(ACAAA|\theta) + P(ACCAA|\theta) + P(ACGAA|\theta) + P(ACTAA|\theta)}{\binom{5}{4}}$$

$$= \frac{(0.2)(0.7)(0.1)(1)(0.1) + (0.2)(0.7)(0.8)(1)(0.1) + (0.2)(0.7)(0)(1)(0.1) + (0.2)(0.7)(0.1)(1)(0.1)}{5}$$

$$= \frac{(0.2)(0.7)(0.1 + 0.8 + 0 + 0.1 + 0.1)(1)(0.1)}{5}$$

$$= \frac{(0.2)(0.7)(1)(1)(0.1)}{5} = 0.0028$$

This holds even if there are multiple gapped positions.

Our calculations so far assume that the length of the gapped kmer, l, is equal to the number of columns for a given PWM, which we denote as L. This is often not the case. We do need to consider cases where $l > L$ because we extend the PWM on the right and left by $l - 1$ to match genomic background. This maximizes the amount of information we can obtain by the PWM. Contributions to the gapped kmer counts by the flanking $l - 1$ parts are eliminated by $P(\boldsymbol{g}|BG)$ in (Eq. 3). Thus, we will always run into the case where $l < L$. Full length kmers contain smaller full length kmers (e.g. $ACCAA$ contains $ACCA$ and $CCAA$). Therefore, to calculate $P(\boldsymbol{g}|BG)$ for $l < L$, we need to break up θ into $L - l + 1$ subPWMs ϕ_j of length l. Each ϕ_j is a moving window of θ in steps of one. We then calculate $P(\boldsymbol{g}|\phi_j)$ over all j, sum, and normalize.

$$P(\boldsymbol{g}|\theta) = \frac{1}{L - l + 1} \sum_{j=1}^{L-l+1} P(\boldsymbol{g}|\phi_j)$$

For example, to calculate $P(AC\text{–}A|\theta)$

$$P(AC\text{–}A|\theta) = \frac{1}{5 - 4 + 1} \sum_{j=1}^{5-4+1} P(AC\text{–}A|\phi_j)$$

$$\phi_1 = \begin{array}{c} A \\ C \\ G \\ T \end{array} \begin{bmatrix} 0.2 & 0.1 & 0.1 & 1 \\ 0.2 & 0.7 & 0.8 & 0 \\ 0.3 & 0.1 & 0 & 0 \\ 0.3 & 0.1 & 0.1 & 0 \end{bmatrix} \qquad \phi_2 = \begin{array}{c} A \\ C \\ G \\ T \end{array} \begin{bmatrix} 0.1 & 0.1 & 1 & 0.1 \\ 0.7 & 0.8 & 0 & 0 \\ 0.1 & 0 & 0 & 0 \\ 0.1 & 0.1 & 0 & 0.9 \end{bmatrix}$$

$$P(AC\text{–}A|\phi_1) = \frac{(0.2)(0.7)(1)}{\binom{4}{3}} = 0.035$$

$$P(AC\text{–}A|\phi_2) = \frac{(0.1)(0.8)(0.1)}{\binom{4}{3}} = 0.002$$

$$P(AC\text{–}A|\theta) = \frac{0.035 + 0.002}{2} = 0.0185$$

We will now go over how to calculate $P(\boldsymbol{g}|BG)$ computationally (implemented in BGkmers). We estimate the dinucleotide frequency from genomic background by hard counting. We use the dinucleotide distribution because of the very low frequency of the CG dinucleotide. This yields a matrix of dinucelotide frequencies where the rows index the first position, and the columns index the second. When calculating $P(\boldsymbol{g}|BG)$, we are interested in the probability of generating the second nucleotide given the first, so we normalize the rows to sum to one.

$$M = \begin{array}{c|cccc} & A & C & G & T \\ \hline A & 0.30 & 0.20 & 0.28 & 0.22 \\ C & 0.35 & 0.27 & 0.05 & 0.33 \\ G & 0.27 & 0.22 & 0.27 & 0.24 \\ T & 0.18 & 0.23 & 0.29 & 0.30 \end{array}$$

We model this as a first order Markov process where the probability of the next nucleotide only depends on the previous nucleotide. We denote the specific positions of each nucleotide as N_i where N_1 is the first position, N_2 is the second, etc. Thus, $P(ACAA|BG)$ is

$$\begin{aligned} P(ACAA|BG) \; = \; & P(N_1 = A|BG)P(N_2 = C|N_1 = A, BG)P(N_3 = A|N_2 = C, BG)P(N_4 = A|N_3 \\ & = A, BG) \end{aligned}$$

The first term $P(N_1 = A|BG)$ can be obtained by the *GC* content of background DNA. For enhancers, it is around 0.46, which yields an A content of 0.27.

$$\begin{aligned} P(ACAA|BG) \; = \; & P(N_1 = A|BG)P(N_2 = C|N_1 = A, BG)P(N_3 = A|N_2 = C, BG)P(N_4 = A|N_3 \\ & = A, BG) \\ & = (0.27)(0.20)(0.35)(0.3) = 0.00567 \end{aligned}$$

To incorporate gaps, we must use the matrix M^{j+1} for the number of gaps, j, between two ungapped positions. This sums over all possible paths from N_i to N_{i+j+1}. Like for $P(\boldsymbol{g}|\theta)$, we also must normalize by $\binom{l}{k}$. For example,

$$M^2 = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{array}{c} \begin{array}{cccc} \quad A & \quad C & \quad G & \quad T \end{array} \\ \begin{bmatrix} 0.27 & 0.23 & 0.24 & 0.26 \\ 0.27 & 0.23 & 0.22 & 0.28 \\ 0.27 & 0.23 & 0.23 & 0.27 \\ 0.26 & 0.23 & 0.23 & 0.28 \end{bmatrix} \end{array}$$

$$P(AC\text{–}A|\theta) = \frac{P(N_1 = A|BG)P(N_2 = C|N_1 = A, BG)P(N_4 = A|N_2 = C, BG)}{\binom{4}{3}}$$

$$= \frac{(0.27)(0.20)(0.27)}{4} = 0.003645$$

With $P(\boldsymbol{g}|\theta_j)$ and $P(\boldsymbol{g}|BG)$, we now can calculate every term on the right-hand side of (Eq. 3) except for $P(\theta_j)$. We will do some algebraic manipulation to obtain a more convenient form. When summing over the entire sequence, the expectation of the background is

$$E[\boldsymbol{g}|BG] = \binom{l}{k}(\tilde{L} - l + 1)P(\boldsymbol{g}|BG)$$

where $\tilde{L}$ is the average length the sequences in S. The expectation of a PWM is summed over the number of length l kmers, and NOT over the sequence.

$$E[\boldsymbol{g}|\theta_j] = \binom{l}{k}\left(L_{\theta_j} - l + 1\right)P(\boldsymbol{g}|\theta_j)$$

where L_{θ_j} is the length of PWM θ_j.

Since the number of gapped kmers is proportional to the number of full length kmers in the sequence, $\tilde{n} = \binom{l}{k}(\tilde{L} - l + 1)$. Thus, (Eq. 3) becomes

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] = \tilde{n}\sum_{j} P(\theta_j)\left(P(\boldsymbol{g}|\theta_j) - P(\boldsymbol{g}|BG)\right)$$

$$= \binom{l}{k}\left(\tilde{L} - l + 1\right)\sum_j P(\theta_j)\left(\frac{E[\boldsymbol{g}|\theta_j]}{\binom{l}{k}\left(L_{\theta_j} - l + 1\right)} - \frac{E[\boldsymbol{g}|BG]}{\binom{l}{k}\left(\tilde{L} - l + 1\right)}\right)$$

$$= \sum_j \frac{\tilde{L} - l + 1}{L_{\theta_j} - l + 1} P(\theta_j)\left(E[\boldsymbol{g}|\theta_j] - \frac{L_{\theta_j} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG]\right)$$

We define two more terms for convenience.

$$F(\boldsymbol{g}|\theta_j, BG) = E[\boldsymbol{g}|\theta_j] - \frac{L_{\theta_j} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG]$$

$$f(\theta_j) = \frac{\tilde{L} - l + 1}{L_{\theta_j} - l + 1} P(\theta_j)$$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] = \sum_j f(\theta_j) F(\boldsymbol{g}|\theta_j, BG)$$

(**Eq. 4**)

$F(\boldsymbol{g}|\theta_j, BG)$ is the difference of gapped kmers counts that you would expect in a particular TFBS compared to the background. $f(\theta_j)$ is the average number of times a TFBS appears per sequence. (Recall that $P(\theta_j)$is the probability of picking any gapped kmer from TFBS θ_j). Calculating $F(\boldsymbol{g}|\theta_j, BG)$ is easier since we avoid much unnecessary normalization when calculating $P(\boldsymbol{g}|\theta_j)$ and $P(\boldsymbol{g}|BG)$.

To calculate the gapped kmer distribution of a PWM, you must find $P(\boldsymbol{g}|\theta_j)$ for all possible gapped kmers. This seems like a computationally difficult task, but with efficient memory utilization, vectorization, and proper indexing, the gapped kmer distribution of a PWM can be calculated in approximately a tenth of a second.

4.1.3 Regression to learn the frequency of TFBSs

As stated earlier, equations (3) and (4) are in the form of a regression problem. The left-hand side is determined by directly counting the gapped kmers from two set of sequences S and B. $F(\boldsymbol{g}|\theta_j, BG)$ serves as our features, and $f(\theta_j)$ are the regression weights, which must be learned. If we want to learn the frequencies of J total PWMs, the basic form of a regression model, $\boldsymbol{Y} = \boldsymbol{Xw}$, where we minimize the squared error $||\boldsymbol{Y} - \boldsymbol{Xw}||_2^2$, where $||v||_2$ is the $l2$-norm. $\boldsymbol{Y}$ is a $4^k \binom{l}{k}$ by 1 vector of gapped kmer counts. $\boldsymbol{X}$ is a $4^k \binom{l}{k}$ by J matrix, whose columns contain $F(\boldsymbol{g}|\theta_j, BG)$ for all j. $\boldsymbol{w}$ is a J by 1 vector of the $f(\theta_j)$. We have two type of problems that we can solve.

1. $\boldsymbol{Y}$ and $\boldsymbol{X}$ are known quantities, and $\boldsymbol{w}$ must be learned.
2. Only $\boldsymbol{Y}$ is known, and we must learn both $\boldsymbol{X}$ and $\boldsymbol{w}$.

We will discuss case 2 in the next section, where we show how to create *de novo* PWMs with convex optimization techniques. In this section, we will go over case 1, the much easier of the two.

Let's suppose that we have a collection of PWMs that are already trained, and therefore can calculate $F(\boldsymbol{g}|\theta_j, BG)$. Under most circumstances, we will be using values of (l, k) where $J \ll 4^k \binom{l}{k}$. Typically, we will have hundreds of thousands of gapped kmers, compared to 2000 different TFs and only a few hundred significantly different binding motifs. Combinations of l and k that work the best result in $4^k \binom{l}{k} > 10^5$, since it requires many unique gapped kmers to learn multiple binding sites simultaneously. If we were to use the simplest regression algorithm, ordinary least squares, the model will be prone to overfitting, especially when J is large. Thus, when working with larger numbers of J, we must use regularization.

We chose to use LASSO regression, which minimizes the quantity $||\boldsymbol{Y} - \boldsymbol{Xw}||_2^2 + \lambda||\boldsymbol{w}||_1$, where $||\boldsymbol{v}||_1$ is the $l1$-norm. λ is the regularization parameter, which controls how much the regression weighs the squared errors against the regularization. For large enough λ, LASSO regression forces many of the regression weights to zero, which is ideal for our problem. We expect only a handful of TFs to be active in a given cell/tissue type (typically around, 5-20). There are a few ways to pick the optimal λ. We start with a large λ, forcing most of the weights to zero, and decrease its value until the MSE begins to plateau (find the “knee" in the error vs lambda curve). We can also find a λ that identifies a set number of PWMs with non-zero weight, say 30 or so, and remove each feature and calculate the increase in error. PWMs, when removed, that result in a large increase in error, strongly contribute to fitting $\boldsymbol{Y}$. Our last method requires utilizing sequence-based models which will be explained in 2.1.6. We call this method gkmPWMlasso, which is implemented in gkmPWMlasso.m.

There is one more detail that must be discussed. There are many different PWM databases available, all of which contain slightly different motifs for the same TFs. Furthermore, many TFs contain the same binding domain, (e.g. GATA1 and GATA2). Thus, we must cluster the PWMs to prevent linear dependence and either combine PWMs in the same cluster into the same feature or select one to represent the entire cluster. We opt to do the latter, but from our experience both methods work well.

4.1.4 Convex optimization with Lagrange multipliers to learn *de novo* PWMs

In this section, we will show how to create PWMs *de novo* with convex optimization techniques (code found in gkmPWM.m). To preview, we will train the PWMs themselves by iteratively optimizing each column of the PWMs, and occasionally updating the PWM frequencies. We will show that the solution space of the optimal column is convex and derive the exact solution using Lagrange Multipliers.

We begin by observing a mathematical property of PWMs. This is most easily conveyed by using an example. We take the PWM in the third section and replace the third column with unknowns.

$$\theta = \begin{array}{c} A\\ C\\ G\\ T \end{array}\begin{bmatrix} 0.2 & 0.1 & p_A & 1 & 0.1\\ 0.2 & 0.7 & p_C & 0 & 0\\ 0.3 & 0.1 & p_G & 0 & 0\\ 0.3 & 0.1 & p_T & 0 & 0.9 \end{bmatrix}$$

We are interested in the contributions of the p_N to the gapped kmer distribution of θ. We will use $(l, k) = (3{,}2)$ to reduce the total number of gapped kmers to a more manageable number to fit in a document. There are 3 subPWMs of length 3.

$$\phi_1 = \begin{array}{c} A\\ C\\ G\\ T \end{array}\begin{bmatrix} 0.2 & 0.1 & p_A\\ 0.2 & 0.7 & p_C\\ 0.3 & 0.1 & p_G\\ 0.3 & 0.1 & p_T \end{bmatrix}, \phi_2 = \begin{array}{c} A\\ C\\ G\\ T \end{array}\begin{bmatrix} 0.1 & p_A & 1\\ 0.7 & p_C & 0\\ 0.1 & p_G & 0\\ 0.1 & p_T & 0 \end{bmatrix}, \phi_3 = \begin{array}{c} A\\ C\\ G\\ T \end{array}\begin{bmatrix} p_A & 1 & 0.1\\ p_C & 0 & 0\\ p_G & 0 & 0\\ p_T & 0 & 0.9 \end{bmatrix}$$

For convenience, we calculate $P(\boldsymbol{g}|\theta)$ only for gapped kmers of the form $-NN$ since there are 48 total $(3{,}2)$ gapped kmers, which is still a bit unmanageable.

$$P(\boldsymbol{g}|\theta) = \frac{P(\boldsymbol{g}|\phi_1) + P(\boldsymbol{g}|\phi_2) + P(\boldsymbol{g}|\phi_3)}{\binom{3}{2}}$$

$$3P(\boldsymbol{g}|\theta) = \begin{array}{c} \_AA\\ \_AC\\ \_AG\\ \_AT\\ \_CA\\ \_CC\\ \_CG\\ \_CT\\ \_GA\\ \_GC\\ \_GG\\ \_GT\\ \_TA\\ \_TC\\ \_TG\\ \_TT \end{array}\begin{bmatrix} 0.1p_A\\ 0.1p_C\\ 0.1p_G\\ 0.1p_T\\ 0.7p_A\\ 0.7p_C\\ 0.7p_G\\ 0.7p_T\\ 0.1p_A\\ 0.1p_C\\ 0.1p_G\\ 0.1p_T\\ 0.1p_A\\ 0.1p_C\\ 0.1p_G\\ 0.1p_T \end{bmatrix} + \begin{bmatrix} p_A\\ 0\\ 0\\ 0\\ p_C\\ 0\\ 0\\ 0\\ p_G\\ 0\\ 0\\ 0\\ p_T\\ 0\\ 0\\ 0 \end{bmatrix}, + \begin{bmatrix} 0.1\\ 0\\ 0\\ 0.9\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0 \end{bmatrix} = \begin{array}{c} \_AA\\ \_AC\\ \_AG\\ \_AT\\ \_CA\\ \_CC\\ \_CG\\ \_CT\\ \_GA\\ \_GC\\ \_GG\\ \_GT\\ \_TA\\ \_TC\\ \_TG\\ \_TT \end{array}\begin{bmatrix} 0.1p_A + p_A\\ 0.1p_C\\ 0.1p_G\\ 0.1p_T\\ 0.7p_A + p_C\\ 0.7p_C\\ 0.7p_C\\ 0.7p_C\\ 0.1p_A + p_G\\ 0.1p_C\\ 0.1p_G\\ 0.1p_G\\ 0.1p_A + p_T\\ 0.1p_C\\ 0.1p_G\\ 0.1p_G \end{bmatrix} + \begin{bmatrix} 0.1\\ 0\\ 0\\ 0.9\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0 \end{bmatrix}$$

We can separate this vector into the form $\boldsymbol{Ap} + \boldsymbol{b}$ where $\boldsymbol{p} = [p_A \quad p_C \quad p_G \quad p_T]^T$. $\boldsymbol{A}$ is a $4^k\binom{l}{k}$ by 4 matrix that stores the contribution of each p_N to $P(\boldsymbol{g}|\theta)$, $\boldsymbol{b}$ is a $4^k\binom{l}{k}$ length vector of constants.

$$3P(\boldsymbol{g}|\theta) = \begin{array}{r} \_AA \\ \_AC \\ \_AG \\ \_AT \\ \_CA \\ \_CC \\ \_CG \\ \_CT \\ \_GA \\ \_GC \\ \_GG \\ \_GT \\ \_TA \\ \_TC \\ \_TG \\ \_TT \end{array}\!\begin{bmatrix} 0.1p_A + p_A \\ 0.1p_C \\ 0.1p_G \\ 0.1p_T \\ 0.7p_A + p_C \\ 0.7p_C \\ 0.7p_C \\ 0.7p_C \\ 0.1p_A + p_G \\ 0.1p_C \\ 0.1p_G \\ 0.1p_G \\ 0.1p_A + p_T \\ 0.1p_C \\ 0.1p_G \\ 0.1p_G \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 0.9 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \\ 0.7 & 1 & 0 & 0 \\ 0 & 0.7 & 0 & 0 \\ 0 & 0 & 0.7 & 0 \\ 0 & 0 & 0 & 0.7 \\ 0.1 & 0 & 1 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \\ 0.1 & 0 & 0 & 1 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \begin{bmatrix} p_A \\ p_C \\ p_G \\ p_T \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0 \\ 0 \\ 0.9 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \boldsymbol{Ap} + \boldsymbol{b}$$

$\boldsymbol{p}$ contributes linearly to $P(\boldsymbol{g}|\theta)$ and therefore $E[\boldsymbol{g}|\theta]$. Since $E[\boldsymbol{g}|\theta_i] = \binom{l}{k}\left(L_{\theta_i} - l + 1\right)P(\boldsymbol{g}|\theta_i)$, we will just absorb $\binom{l}{k}\left(L_{\theta_i} - l + 1\right)$ into $\boldsymbol{A}$ and $\boldsymbol{b}$. This generalizes to all combinations of (l, k). We can write Eq. 4 as a linear function of a column of $\boldsymbol{p}$ for a given θ_i.

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] = \sum_j f(\theta_j) F(\boldsymbol{g}|\theta_j, BG)$$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{j \neq i} f(\theta_j) F(\boldsymbol{g}|\theta_j, BG) = f(\theta_i) F(\boldsymbol{g}|\theta_i, BG)$$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{j \neq i} f(\theta_j) F(\boldsymbol{g}|\theta_j, BG) = f(\theta_i)\left(E[\boldsymbol{g}|\theta_i] - \frac{L_{\theta_i} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG]\right)$$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{j \neq i} f(\theta_j) F(\boldsymbol{g}|\theta_j, BG) + \frac{L_{\theta_i} - l + 1}{\tilde{L} - l + 1} f(\theta_i) E[\boldsymbol{g}|BG] = f(\theta_i) E[\boldsymbol{g}|\theta_i]$$

$$E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{j \neq i} f(\theta_j) F(\boldsymbol{g}|\theta_j, BG) + \frac{L_{\theta_i} - l + 1}{\tilde{L} - l + 1} f(\theta_i) E[\boldsymbol{g}|BG] = f(\theta_i)(\boldsymbol{Ap} + \boldsymbol{b})$$

$$\frac{1}{f(\theta_i)}\left(E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{j \neq i} f(\theta_j) F(\boldsymbol{g}|\theta_j, BG)\right) + \frac{L_{\theta_i} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG] - \boldsymbol{b} = \boldsymbol{Ap}$$

If we hold everything constant except for $\boldsymbol{p}$, the left-hand side becomes a $4^k\binom{l}{k}$ length vector of constants, which we will denote as $\boldsymbol{r}$.

$$\boldsymbol{r} = \boldsymbol{Ap}$$

Once again, we have a regression problem, where we must solve for a single column of a single PWM. However, since $\boldsymbol{p}$ is a probability distribution, we must impose two constraints. We now have the following optimization problem.

minimize $\quad \|\boldsymbol{Ap} - \boldsymbol{r}\|_2^2$

subject to $\quad \mathbb{1}^{\boldsymbol{T}}\boldsymbol{p} = 1, \quad \boldsymbol{p} \succcurlyeq 0$

The first constraint ensures the p_N sum to one ($\mathbb{1}$ is the vector of all ones). The second constraint ensures p_N is strictly positive. We do not need to impose $p_N \leq 1$ because these two constraints in combination guarantee that all p_N will be no greater than 1.

This optimization problem is convex. This allows us to worry about the inequality constraint only if our optimal solution does not lie within the boundaries. For now, we only need to impose the equality constraint. We proceed with Lagrange multipliers to find $\boldsymbol{p}^*$, the optimal $\boldsymbol{p}$.

$$L = \frac{1}{2}\|\boldsymbol{Ap}^* - \boldsymbol{r}\|_2^2 - \lambda\mathbb{1}^{\boldsymbol{T}}\boldsymbol{p}^*$$

$$\nabla L = \boldsymbol{A}^{\boldsymbol{T}}(\boldsymbol{Ap}^* - \boldsymbol{r}) - \lambda\mathbb{1} = 0$$

$$\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{Ap}^* = \boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{r} + \lambda\mathbb{1}$$

$$\boldsymbol{p}^* = \left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{r} + \lambda\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\mathbb{1}$$

$$\mathbb{1}^{\boldsymbol{T}}\boldsymbol{p}^* = \mathbb{1}^{\boldsymbol{T}}\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{r} + \lambda\mathbb{1}^{\boldsymbol{T}}\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\mathbb{1}$$

$$1 = \mathbb{1}^{\boldsymbol{T}}\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{r} + \lambda\mathbb{1}^{\boldsymbol{T}}\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\mathbb{1}$$

$$\lambda = \frac{1 - \mathbb{1}^{\boldsymbol{T}}\left(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{r}}{\mathbb{1}^{\boldsymbol{T}}(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A})^{-1}\mathbb{1}}$$

$$\boldsymbol{p}^* = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{r} + \left(\frac{1 - \mathbb{1}^T\left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{r}}{\mathbb{1}^T(\boldsymbol{A}^T\boldsymbol{A})^{-1}\mathbb{1}}\right)\left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\mathbb{1}$$

(Eq. 5)

If the solution to Eq. 5 lies outside of our inequality constraints, (i.e. one or more of the $p_N < 0$), then we must check the solutions at the boundaries, which are all combinations of $p_N = 0$. Normally, we would have an exponential increase in the number of combinations, but since there are only 4 nucleotides, we only need to check at most $2^4 - 1 = 15$ cases. Furthermore, we can just use Eq 6. repeatedly for each combination since we are forcing certain p_N to be zero. The solution to Eq. 5 is guaranteed to decrease the mean squared error more than any other allowable combination of p_N because our error function is still convex. Therefore, we can create PWMs *de novo* by iteratively optimizing through each column of every PWM. We call this algorithm gkmPWM. Below is the pseudocode.

```
procedure gkmPWM($S, B$)
    For $j = 1, \dots, J$   do
        Initialize PWM $\theta_j$
    $E[\boldsymbol{G}(S)] \leftarrow \sum_{i=1}^{m} P(s_i)\boldsymbol{G}(s_i)$
    $E[\boldsymbol{G}(B)] \leftarrow \sum_{i=1}^{m} P(b_i)\boldsymbol{G}(b_i)$
    $\boldsymbol{Y} \leftarrow E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$
    $E[\boldsymbol{g}|BG] \leftarrow \binom{l}{k}(\tilde{L} - l + 1)P(\boldsymbol{g}|BG)$
    $f(\vec{\theta}), F(\boldsymbol{g}|\vec{\theta}, BG) \leftarrow$ ComputeWeights($\vec{\theta}, E[\boldsymbol{g}|BG], \mathbf{Y}$)
    $\epsilon \leftarrow$ MeanSquaredError($\boldsymbol{Y}, \sum_j f(\theta_j)F(\boldsymbol{g}|\theta_j, BG)$)
    while $\epsilon$ not converge   do
        For $j = 1, \dots, J$   do
            For $h = 1, \dots, L_{\theta_j}$   do
                $\theta_{jh} \leftarrow \theta_j$ with $h$th column being unknown
                $\boldsymbol{A}, \boldsymbol{b} \leftarrow$ Map PWM to gapped kmers($\theta_{jh}$)
                $\boldsymbol{p}^* \leftarrow$ LagrangeAlgo($f(\vec{\theta}), F(\boldsymbol{g}|\vec{\theta}, BG), E[\boldsymbol{g}|BG], \boldsymbol{A}, \boldsymbol{b}$)
                $\theta_j \leftarrow \theta_j'$s $h$th column being updated with $\boldsymbol{p}^*$
            $f(\vec{\theta}), F(\boldsymbol{g}|\vec{\theta}, BG) \leftarrow$ ComputeWeights($\vec{\theta}, E[\boldsymbol{g}|BG], \mathbf{Y}$)
        $\epsilon \leftarrow$ MeanSquaredError($\boldsymbol{Y}, \sum_j f(\theta_j)F(\boldsymbol{g}|\theta_j, BG)$)
    return PWM $\theta_1, \dots, \theta_J$
```

```
function ComputeWeights($\vec{\theta}, E[\boldsymbol{g}|BG], \mathbf{Y}$)
```

```
    For $j = 1, \ldots, J$ **do**
        $E[\boldsymbol{g}|\theta_j] \leftarrow \binom{l}{k}\left(L_{\theta_j} - l + 1\right) P(\boldsymbol{g}|\theta_j)$
        $F(\boldsymbol{g}|\theta_j, BG) \leftarrow \left(E[\boldsymbol{g}|\theta_j] - \frac{L_{\theta_j} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG]\right)$
    $f(\vec{\theta}) \leftarrow \text{OrdinaryLeastSquares}(\mathbf{X} = F(\boldsymbol{g}|\vec{\theta}, BG), \mathbf{Y})$
    return $F(\boldsymbol{g}|\vec{\theta}, BG), f(\vec{\theta})$

**function** LagrangeAlgo($f(\vec{\theta}), F(\boldsymbol{g}|\vec{\theta}, BG), E[\boldsymbol{g}|BG], \boldsymbol{A}, \boldsymbol{b}$)
    $\boldsymbol{r} \leftarrow \frac{1}{f(\theta_j)}\left(E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)] - \sum_{q \neq j} f(\theta_j) F(\boldsymbol{g}|\theta_q, BG)\right) + \frac{L_{\theta_j} - l + 1}{\tilde{L} - l + 1} E[\boldsymbol{g}|BG] - \boldsymbol{b}$
    $\boldsymbol{p}^* \leftarrow (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{r} + \left(\frac{1 - \mathbb{1}^T(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{r}}{\mathbb{1}^T(\boldsymbol{A}^T\boldsymbol{A})^{-1}\mathbb{1}}\right)(\boldsymbol{A}^T\boldsymbol{A})^{-1}\mathbb{1}$
    if $\boldsymbol{p}^* \ngeqslant 0$ **then**
        **for** $\forall I \mid I \subset \{A, C, G, T\} \wedge I \neq \emptyset$ **do**
            **for** $N = 1, \ldots, |I|$ **do**
                $\boldsymbol{p}^*_{I(N)} = 0$
                Remove $N$th column of $\boldsymbol{A}$
            $\boldsymbol{p}^*_I \leftarrow (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{r} + \left(\frac{1 - \mathbb{1}^T(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{r}}{\mathbb{1}^T(\boldsymbol{A}^T\boldsymbol{A})^{-1}\mathbb{1}}\right)(\boldsymbol{A}^T\boldsymbol{A})^{-1}\mathbb{1}$
        $\boldsymbol{p}^* \leftarrow \min_{\boldsymbol{p}^*_I \geqslant 0} \|\boldsymbol{A}\boldsymbol{p}^*_I - \boldsymbol{r}\|_2^2$
    return $\boldsymbol{p}^*$
```

The primary advantage of gkmPWM over gkmPWMlasso is that it requires no prior knowledge of motifs. As a result, we also do not need to perform regularization. It also needs much less memory to run, since it does not need to calculate $F(\boldsymbol{g}|\vec{\theta}, BG)$ for a large number of PWMs. It does have one weakness, which is that it can get stuck in a local minimum. However, when seeding enough motifs, this problem goes away.

4.1.5 Integrating sequence-based models with gkmPWM

Many sequence-based models have been developed to predict regulatory elements. Because multiple TFs bind cooperatively to the same regulatory elements, these models must be capable of learning multiple binding sites simultaneously to achieve high predictive performance. As a result, the complexity of the models must be high, making them difficult to biologically interpret. In this section, we will demonstrate how to use gkmPWM and gkmPWMlasso to extract PWM features from support vector machines (gkm-SVM) and other machine learning methods.

We will use our estimate of the expected gapped kmer counts of sequences for regulatory elements S and background DNA B.

$$\mathrm{E}\left[\boldsymbol{G}(\mathrm{S})\right] = \sum_{i=1}^{m} P(s_i)\boldsymbol{G}(S_i)$$

$$E[\boldsymbol{G}(B)] = \sum_{i=1}^{m} P(b_i)\boldsymbol{G}(b_i)$$

To extract features from machine learning models, we will use the models to define $P(s_i)$ and $P(b_i)$ for a given S and B and create an estimate of $E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$. We can then run gkmPWM or gkmPWMlasso to learn PWMs.

We will first show how to extract features from gkm-SVM. Recall that the decision function of the SVM is of the form $h(\boldsymbol{x}) = sign(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b)$, where $\boldsymbol{x}$ is the gapped kmer count vector of the input sequence and $\boldsymbol{w}$ is the vector of weights assigned to each gapped kmer, which the SVM learns. From section 3.1, we showed that the dual problem of the SVM is a convex optimization problem is

$$\text{maximize} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \boldsymbol{x_i}^T \boldsymbol{x_j}$$

$$\text{subject to} \quad \alpha_i \geq 0\,, \sum_i \alpha_i y_i = 0, \forall i$$

where y_i is class of sequence i (either -1 or 1), and α_i is the Lagrange multiplier of sequence i. Additionally, we showed that the SVM weight vector is $\boldsymbol{w} = \sum_i \alpha_i \, y_i \, \boldsymbol{x_i}$. We will show that we can directly interpret $\boldsymbol{w}$ with Eq. 4. We can split $\boldsymbol{w}$ into positive class ($y_i = 1$) and negative class ($y_i = -1$) parts. Define $\mathbf{1}_{\boldsymbol{A}}(y)$ as the indicator function, which outputs 1 if $y \in \boldsymbol{A}$ is true and 0 if false.

$$\boldsymbol{w} = \sum_i \alpha_i y_i \boldsymbol{x_i} = \sum_i \mathbf{1}_1(y_i)\alpha_i \boldsymbol{x_i} - \sum_i \mathbf{1}_{-\mathbf{1}}(y_i)\alpha_i \boldsymbol{x_i}$$

Note that α_i are all non-negative, which forces positive class sequences to contribute positive or zero values to the weight vector, and negative or zero values for negative class sequences. We will similarly split the second constraint $\sum_i \alpha_i y_i = 0$ as $\sum_i \mathbf{1}_1(y_i)\alpha_i = \sum_i \mathbf{1}_{-\mathbf{1}}(y_i)\alpha_i$. We define $\rho = \sum_i \mathbf{1}_1(y_i)\alpha_i = \sum_i \mathbf{1}_{-\mathbf{1}}(y_i)\alpha_i$ and can make estimates for $E[\boldsymbol{G}(S)]$ and $E[\boldsymbol{G}(B)]$.

$$E[\boldsymbol{G}(S)] = \sum_i^m P(s_i)\boldsymbol{G}(s_i) = \sum_{i|y_i=1}^m \frac{\alpha_i}{\rho}\boldsymbol{x_i}$$

$$E[\boldsymbol{G}(B)] = \sum_i^m P(b_i)\boldsymbol{G}(b_i) = \sum_{i|y_i=-1}^m \frac{\alpha_i}{\rho}\boldsymbol{x_i}$$

Recall that $\boldsymbol{G}(s_i) = \boldsymbol{x_i}$. We now define $P(s_i) = {}^{\alpha_i}/_{\rho}$. ρ is a normalization term that allows us to treat the α_i as probabilities. Therefore, $\boldsymbol{w} = \rho(E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)])$, which allows us to use Eq.4 to extract motifs.

We can also extract features from other machine learning models as long as they give a real valued score to each test sequence. Simple estimates of $P(s_i)$ and $P(b_i)$ can be obtained by normalizing the scores appropriately depending on whether they are positive or negative. However, a more preferred method is to derive estimates of the average contribution of each gapped kmer to a positive or negative score. First, if the output of the machine learning method is a probability, it should be mapped to an unbounded score by using the inverse of a logistic function $f(p) = \ln\left(\frac{p}{1-p}\right)$. (The constant term k of $f(x) = \frac{1}{1+e^{-kx}}$ can be ignored since we will eventually normalize). We now use support vector regression with gapped kmers (gkmSVR) to learn the contributions of each gapped kmer. We will see that this works just as nicely as gkmSVM. gkmSVR uses the same features as gkmSVM, except it learns continuous values y_i. Its decision function is also linear $\boldsymbol{w^T x} + b$ which tolerates deviations in error up to some error, $\left|y_i - (\boldsymbol{w^T x} + b)\right| \leq \epsilon$. It has a very similar dual problem to gkmSVM, except that there are two

Lagrange multipliers per sequence α_i and α_i^*.

$$\text{maximize} \quad \epsilon \sum_i (\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i^* - \alpha_i) + \frac{1}{2}\sum_i \sum_j (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\boldsymbol{x_i}^T \boldsymbol{x_j}$$

$$\text{subject to} \quad \alpha_i \geq 0\,, \alpha_i^* \geq 0, \sum_i (\alpha_i - \alpha_i^*) = 0, \forall i$$

The solution to the weight vector is given by $\boldsymbol{w} = \sum_i (\alpha_i - \alpha_i^*)\boldsymbol{x_i}$. We can use the same trick from earlier by splitting the sequences into two sets $\{s_i|(\ \alpha_i\ - \alpha_i^*) > 0\}$ and $\{s_i|(\ \alpha_i\ - \alpha_i^*) < 0\}$. Using the last constraint,

$$\sum_i \mathbf{1}_{x>0}(\ \alpha_i\ - \alpha_i^*) = \sum_i \mathbf{1}_{x<0}(\ \alpha_i\ - \alpha_i^*)$$

We can define $\rho = \sum_i \mathbf{1}_{x>0}(\ \alpha_i\ - \alpha_i^*) = \sum_i \mathbf{1}_{x<0}(\ \alpha_i\ - \alpha_i^*)$ and make estimates for $E[\boldsymbol{G}(S)]$ and $E[\boldsymbol{G}(B)]$ as

$$E[\boldsymbol{G}(S)] = \sum_i^m P(s_i)\boldsymbol{G}(s_i) = \sum_{i|(\ \alpha_i - \alpha_i^*)>0}^m \frac{\alpha_i\ - \alpha_i^*}{\rho}\boldsymbol{x_i}$$

$$E[\boldsymbol{G}(B)] = \sum_i^m P(b_i)\boldsymbol{G}(b_i) = \sum_{i|(\ \alpha_i - \alpha_i^*)<0}^m \frac{\alpha_i\ - \alpha_i^*}{\rho}\boldsymbol{x_i}$$

We once again have $\boldsymbol{G}(s_i) = \boldsymbol{x_i}$ and define $P(s_i) = {\alpha_i\ - \alpha_i^*}/{\rho}$. As with SVMs for classification, $\boldsymbol{w} = \rho(E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)])$. Thus, we can still interpret the weight vector of gkmSVR with Eq 4.

4.1.6 Fast algorithms to calculate gapped kmer distributions

In this section, we will show efficient ways to calculate $E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$ and $F(\boldsymbol{g}|\theta_j, BG)$. $E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$ are determined by the gapped kmer counts of sequences, and $F(\boldsymbol{g}|\theta_j, BG)$ are derived from TFBS and background models.

Counting gapped kmers in a set of sequences (implemented in getgkmcounts.m)

We begin with a sequence s of length L. s contains $L - l + 1$ full kmers (no gaps) of length l. Each full kmer contains $\binom{l}{k}$ gapped kmers. First, we create an index list for the set of gapped kmers (create a bijection between $\{1, \dots, \binom{l}{k}\}$ and the $4^k\binom{l}{k}$ gapped kmers). The index of a gapped kmer is determined 1) by the nucleotides in the ungapped positions and 2) the position of the gaps.

(1) We order the gapped kmers by the positions of their gaps. The 4^k gapped kmers with the same gaps are contained in a "block" (e.g. $AAA-, AAC-, AAG-, \dots, TTT-$). The order of the blocks will be discussed later, but for calculating $E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$, the order of the blocks with respect to each other is not important. We then assign each unique block an integer i from $\{1, \dots, \binom{l}{k}\}$.

(2) Within each block, we order the gapped kmers using base 4 indexing. Since the gaps are the same within the same block, we can ignore the gaps. For the ungapped positions, we define a map $M: \{A, C, G, T\} \rightarrow \mathbb{Z}_4$, where $M(A) = 0, M(C) = 1, M(G) = 2, M(T) = 3$.

We then apply M to the ungapped positions to obtain a vector $\boldsymbol{g_I} \in Z_4^k$, where the first entry corresponds to the first ungapped position, the second entry corresponds to the 2nd, etc. We take the inner product of $\boldsymbol{g_I}$ and $[4^0 \ 4^1 \ \dots \ 4^{k-2} \ 4^{k-1}]$ and add one. In order words, $\boldsymbol{g_I}$ is the coefficients of the base 4 representation of a unique integer in $\{0,1, \dots, 4^k - 1\}$. For consistency with the rest of the document, we add one to maintain one-indexing to obtain a value in $\{1, \dots, 4^k\}$. The index of a given gapped kmer is the sum of the result of 2 and $4\hat{}i$ from 1. This completely spans the integers from 1 to $4^k\binom{l}{k}$.

For each full kmer, we create a 4^l by $\binom{l}{k}$ matrix E that stores the indices of the $\binom{l}{k}$ gapped kmers that are contained in the full kmer. The rows index the full kmers using the base 4 transformation (2). Once this matrix is created, we can apply M from (2) to the entire sequence and map each full kmer to its index. We can then use E to add one to the proper indicies in the gapped kmer count vector. We use this format of indexing for two reasons. First, it is faster than a hash table. Second, it conveniently works with our next algorithm to get $F(\boldsymbol{g}|\theta_j, BG)$.

<u>Computing the gapped kmer distribution of a PWM (implemented in PWM2kmers.m)</u>

For a given PWM, we take advantage of the fact that the columns are conditionally independent. To generate all expected gapped kmer counts, we apply a series of $k-1$ tensor multiplications to the columns that correspond to ungapped positions (recall that we can ignore gapped positions for PWMs). We must apply this to all combinations of gapped positions. The expected gapped kmer count vector will be indexed in the same manner as the previous section. Let p_{ij} be an entry in θ where i is the nucleotide and j is the *j*th ungapped position. We define $\boldsymbol{t^j}$ recursively, where $\boldsymbol{t^1} = [p_{A1}\ \ p_{C1}\ \ p_{G1}\ \ p_{T1}]^T$ and $\boldsymbol{t^j} = \boldsymbol{t^{j-1}} \otimes [p_{A1}\ \ p_{C1}\ \ p_{G1}\ \ p_{T1}]^T$. We arrange $\boldsymbol{t^j}$ as

$$\boldsymbol{t^j} = \begin{bmatrix} p_{Aj}\boldsymbol{t^{j-1}} \\ p_{Cj}\boldsymbol{t^{j-1}} \\ p_{Gj}\boldsymbol{t^{j-1}} \\ p_{Tj}\boldsymbol{t^{j-1}} \end{bmatrix}$$

This preserves the indexing from (2) in the previous section (base 4 indexing).

We apply this recursion for all combinations of gapped positions and concatenate the resulting vectors in the order as in 1. However, unlike in the previous section, we can use the order of the blocks to speed up computation.

Consider two arrangements of ungapped positions: $[1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6]$ and $[1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 7]$. The $\boldsymbol{t^j}$ are the same for both arrangements until the sixth position. Thus, if we save $\boldsymbol{t^5}$ from the computation of $\boldsymbol{t^6}$ of the arrangement $[1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6]$, we only need to apply one tensor multiplication to get $\boldsymbol{t^6}$ of the arrangement $[1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 7]$. We can index the blocks to minimize the number of multiplications by placing blocks with similar consecutive ungapped positions. For example:

$$[1\quad 2\quad 3\quad 4\quad 5\quad 6]$$

$$[1\quad 2\quad 3\quad 4\quad 5\quad 7]$$

$$[1\quad 2\quad 3\quad 4\quad 6\quad 7]$$

$$[1\quad 2\quad 4\quad 5\quad 6\quad 7]$$

$$[2\quad 3\quad 4\quad 5\quad 6\quad 7]$$

Thus, we use this method to order the blocks for both the calculation of $E[\boldsymbol{G}(S)] - E[\boldsymbol{G}(B)]$ and $F\left(\boldsymbol{g}\middle|\theta_j, BG\right)$ (implemented in genIndex.m).

To obtain the expected gapped kmer counts of the background, we brute force the computation over the length of the gapped kmer, l. This needs to be computed only once, so despite its lack of efficiency, it does not take much time at all.

4.1.7 Advantages of gkmPWM over current PWM training methods

Current methods to construct PWMs from cis regulatory elements are confounded by three problems. The first two originate from the assumption that there is zero or one TFBS per sequence (ZOOPS). Since it is very unlikely for a particular TF motif to be present in every sequence in the training set, algorithms which assume ZOOPS train PWMs with an arbitrary cutoff to determine whether a sequence contains a motif or not[69]. Second, ZOOPS algorithms only pick one TFBS per sequence when building PWMs. The same TF often has multiple binding sites within the same sequence. This likely biases the PWM training toward stronger TFBS matches to the PWM and to under-sample weaker affinity binding site. gkmPWM alleviates these two problems by counting all of the gapped k-mers in each sequence, maximizing the representation of all binding sites in the TFBS model. The third problem many current methods face is that they have a difficult time differentiating repetitive sequences from the background model and as a result, report many repetitive and low complexity sequences.

In the next sections, I will demonstrate that gkmPWM outperforms two motif extraction methods, HOMER[69], and tfmodisco-lite[70]. HOMER is a very popular algorithm that uses the methods described above. tfmodisco-lite, like gkmPWM, can extract motifs from sequence-based models. I will show that gkmPWM learn motifs more sensitively than both methods by evaluating their performances on extracting motifs from synthetic sequences and real regulatory elements.

4.2 Methods

4.2.1 Training gkm-SVM and gkmPWM on GM12878 distal enhancers

We used a GM12878 DNase-Seq distal enhancer set (defined the same as from 3.2.4) of peaks to train a gkm-SVM model. We took the 10,000 peaks with the highest MACS2 signal as a positive set. We generated a negative set by sampling sequences from genomic background matched to the positive set's GC content and common genomic repeat frequency, following our standard analysis procedures.[71] We can train gkm-SVM using either the gkmSVM-R package[72] or lsgkm[73], with the default parameters (-l 11 -k 7 -t 2 from lsgkm). We ran gkmPWMlasso to identify the top 24 motifs using a set of 1968 motifs available at https://beerlab.org/gkmpwm/. We ran gkmPWM by initializing 24 motifs using the 11-mer weights generated by running gkmpredict from lsgkm on all possible 11-mers.

4.2.2 Generating synthetic sequences for HOMER and tfmodisco-lite comparisons

We created two different sets of synthetic sequences.

1) For HOMER and gkmPWM comparisons, we created synthetic positive sets for training by seeding motifs. We varied the number of training sequences (2500,

5000, 7500, 10000), the number of unique motifs (2, 4, 6, 8, 10), and the motifs per sequence (1, 2, 3, 4, 5) motif frequencies into random genomic background for a total of 100 different synthetic datasets. We used the same set of motifs from Chapter 3. We created equal sized negative sets by sampling sequences from genomic background. gkm-SVM models for gkmPWM were trained with default parameters.

2) For HOMER, gkmPWM, and tfmodisco-lite comparisons. We created 19 synthetic datasets, each with 10,000 sequences, by seeding 10 PWMs using the same relative frequencies from Table 3.2. We varied the overall motifs per sequence from 0.2 to 2 in increments of 0.1. We created equal sized negative sets by sampling sequences from genomic background.

4.2.3 Training HOMER on synthetic and real datasets

We downloaded HOMER from http://homer.ucsd.edu/homer/. We ran HOMER using the command “homer2 denovo” on all 100 datasets from 4.2.2 (1) and all 19 datasets from 4.2.2 (2) with the default parameters except for length, which we set at 16. We used the negative set generated for gkm-SVM training as the background input (-b). We also ran HOMER on the GM12878 distal enhancer set in 4.2.1 with the same parameters.

4.2.4 Training tfmodisco-lite on synthetic and real datasets

We downloaded tfmodisco-lite from https://github.com/jmschrei/tfmodisco-lite. Tfmodisco-lite requires scoring sequences from a testing set using gkmExplain (https://github.com/kundajelab/gkmexplain). We created 5 80-20 cross validation splits for the 19 datasets from 4.2.2 (2) and trained gkm-SVM models and ran gkmExplain on all test sets. We then ran tfmodisco-lite with the default parameters on the gkmExplain importance scores from all 10,000 simultaneously. We ran tfmodisco-lite on GM12878 DNase-Seq distal enhancers with the same procedure.

4.2.5 Training gkmPWM on synthetic datasets

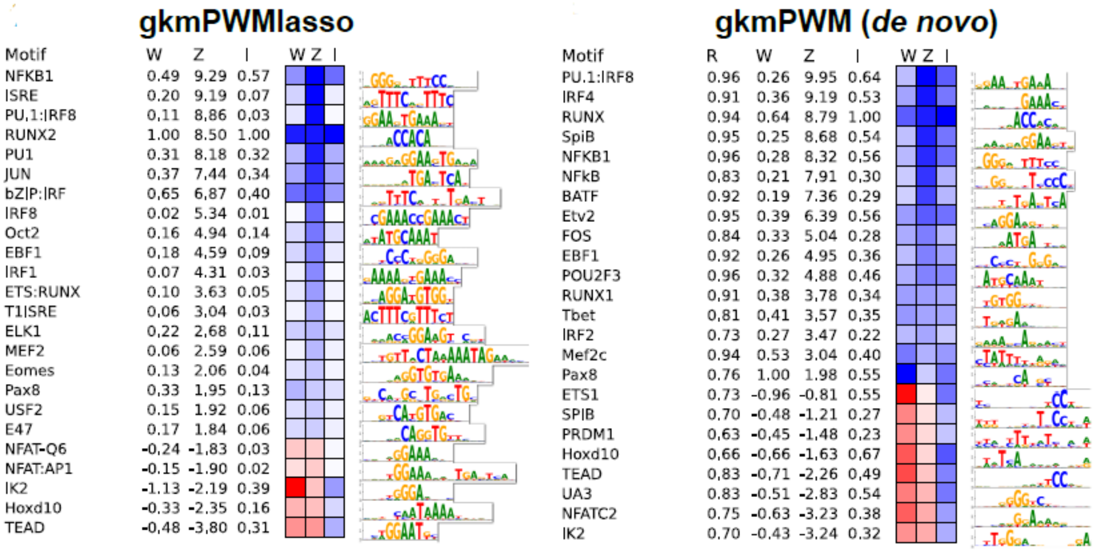
For 4.2.2 (1) and (2), we trained gkmPWM models on all synthetic datasets using the default parameters, except for "PNratio", which we set to 2. For each dataset, we set gkmPWM to learn 1.5 times the number of unique motifs seeded to allow for reasonable sensitivity.

4.2.6 Evaluating the performance of gkmPWM, HOMER, and tfmodisco-lite at learning motifs from synthetic sequences.

Evaluation of performance was measured by determining if the seeds motifs were successfully learned. For each seeded motif, we took the maximum Pearson correlation of all PWMs learned from one of the algorithms. If the correlation is high, then the motif is successfully learned.

4.3 Results

First, I will discuss the similarities and differences between gkmPWMlasso, which uses a list of pre-trained PWMs, and gkmPWM, which learns PWMs *de novo*. I ran gkmPWMlasso and gkmPWM on a gkm-SVM model trained on GM12878 distal enhancers (**Fig 4.2**). Both methods learn $f(\theta)$ (from Eq. 4, which is referred to as W in **Fig 4.2**) using linear regression. Motifs with positive W, are predictive of distal enhancers, and the motifs with negative W are predictive of genomic background. Although the specific values of W differ between similar motifs, the overall list of motifs learned by both methods is very similar, even for the motifs with negative W. Both methods are capable of learning not just the strong motifs (AP1, NF-kB, RUNX1, PU1, and IRF), but also the weaker motifs TF's (MEF2, EBF, Pax5, Oct1/2).



Motif	W	Z	I
NFKB1	0.49	9.29	0.57
ISRE	0.20	9.19	0.07
PU.1:IRF8	0.11	8.86	0.03
RUNX2	1.00	8.50	1.00
PU1	0.31	8.18	0.32
JUN	0.37	7.44	0.34
bZIP:IRF	0.65	6.87	0.40
IRF8	0.02	5.34	0.01
Oct2	0.16	4.94	0.14
EBF1	0.18	4.59	0.09
IRF1	0.07	4.31	0.03
ETS:RUNX	0.10	3.63	0.05
T1ISRE	0.06	3.04	0.03
ELK1	0.22	2.68	0.11
MEF2	0.06	2.59	0.06
Eomes	0.13	2.06	0.04
Pax8	0.33	1.95	0.13
USF2	0.15	1.92	0.06
E47	0.17	1.84	0.06
NFAT-Q6	-0.24	-1.83	0.03
NFAT:AP1	-0.15	-1.90	0.02
IK2	-1.13	-2.19	0.39
Hoxd10	-0.33	-2.35	0.16
TEAD	-0.48	-3.80	0.31

Motif	R	W	Z	I
PU.1:IRF8	0.96	0.26	9.95	0.64
IRF4	0.91	0.36	9.19	0.53
RUNX	0.94	0.64	8.79	1.00
SpiB	0.95	0.25	8.68	0.54
NFKB1	0.96	0.28	8.32	0.56
NFkB	0.83	0.21	7.91	0.30
BATF	0.92	0.19	7.36	0.29
Etv2	0.95	0.39	6.39	0.56
FOS	0.84	0.33	5.04	0.28
EBF1	0.92	0.26	4.95	0.36
POU2F3	0.96	0.32	4.88	0.46
RUNX1	0.91	0.38	3.78	0.34
Tbet	0.81	0.41	3.57	0.35
IRF2	0.73	0.27	3.47	0.22
Mef2c	0.94	0.53	3.04	0.40
Pax8	0.76	1.00	1.98	0.55
ETS1	0.73	-0.96	-0.81	0.55
SPIB	0.70	-0.48	-1.21	0.27
PRDM1	0.63	-0.45	-1.48	0.23
Hoxd10	0.66	-0.66	-1.63	0.67
TEAD	0.83	-0.71	-2.26	0.49
UA3	0.83	-0.51	-2.83	0.54
NFATC2	0.75	-0.63	-3.23	0.38
IK2	0.70	-0.43	-3.24	0.32

Figure 4.2 Predictive motifs in GM12878 distal enhancers

The output of gkmPWMlasso and gkmPWM when extracting motifs from gkmPWMlasso and gkmPWM for 24 total motifs. Motifs with positive regression weights (W) are predictive of the positive class and motifs with negative W are predictive of the negative class. Interestingly, both positive and negative motifs are consistent between gkmPWMlasso and gkmPWM. Besides W, other quantities Z and I are calculated for convenience. Z is average of gapped kmer score of the top gapped kmers (n=30) for each PWM, in units of standard deviations. I is the relative increase in error with removed that motif from the list. For gkmPWM only, I calculate R, the correlation of a de novo motif and the best matching motif in a set of pre-trained PWMs.

There are advantages to using either method. gkmPWMlasso is faster, taking only a few minutes to run, whereas gkmPWM takes 30-45 minutes. However, gkmPWMlasso requires more memory since it must map more PWMs to their expected gapped kmer weights than gkmPWM. Additionally, gkmPWM more reliably models the gkm-SVM weight vector than gkmPWMlasso not just for GM12878 distal enhancers (**Fig 4.3AB**), but also for all ENCODE DHS and ATAC datasets (**Fig 4.3C**). Thus, *de novo* motifs more accurately represent the learned gkm-SVM score function. The observation that *de novo* motifs more accurately describe the gkm-SVM weight vector is likely due to a bias in known PWM models towards the strongest binding sites and suggests that weak binding sites are contributing to the performance of gkm-SVM models and their ability to describe weaker TFBS motif disruptions.

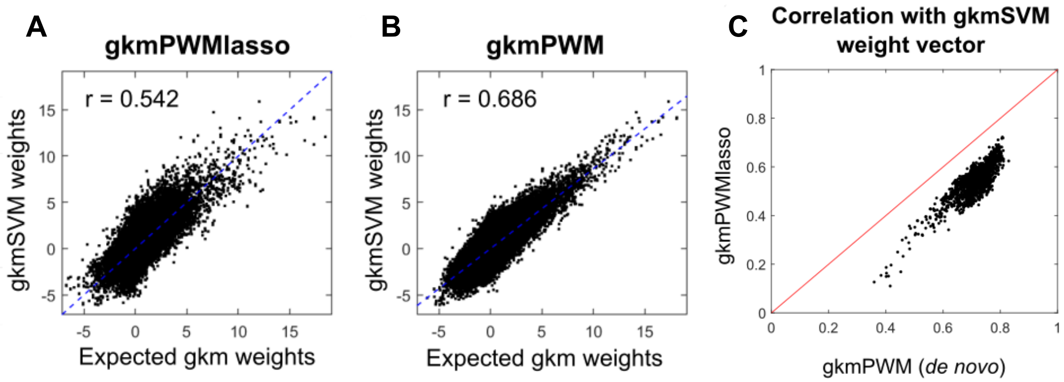


Figure 4.3 Fitting the gkm-SVM vector with PWMs

A) The 24 pre-trained PWMs identified by gkmPWMlasso from Fig 4.2 can fit the gkm-SVM weight vector with moderate accuracy. B) However, learning the motifs de novo increases the correlation with the gkm-SVM weight vector. C) gkmPWM fits the gkm-SVM weight better than gkmPWMlasso across all 1644 human distal enhancer datasets.

Systematic comparison of gkmPWM with other motif extraction methods

We first performed a systematic comparison of gkmPWM to HOMER, a popular motif finding tool which uses hypergeometric enrichment of TFBS. We evaluate their performance by creating synthetic motif learning tasks of varying difficulty. We ran HOMER and gkm-SVM/gkmPWM on each set of synthetic sequences and evaluated their performance by calculating the fraction of the seeded motifs that were successfully learned. For all assessments, a motif was considered to be learned if it had a correlation of at least 0.9 with one of the PWMs detected the method. Both algorithms have a parameter controlling the number of motifs to report, which we set to 25 for HOMER and 1.5 times the number of unique seeded motifs for gkmPWM. gkmPWM consistently learned a higher fraction of the motifs, whereas HOMER struggled to learn motifs, especially when the total number of the number of unique motifs was high and the motifs per sequence was low (**Fig 4.4AB**). To simulate more realistic data, we generated sequences with a diverse set of motifs seeded at variable frequencies, and we used frequencies of motifs learned from lymphoblastoid cell line (GM12878) DHS gkm-SVM models. We created 19 more synthetic datasets with ten total unique motifs where we varied the relative motif frequencies of each PWM. The AUROCs of the gkm-SVM models over this range of motifs per sequence increased steadily from 0.5 to 0.85. **Fig 4.4CD** shows the similarity (correlation) between the seeded and learned motifs, for both gkmPWM and HOMER. Over much of this range, HOMER struggled to learn shorter motifs such as RUNX1, ETS, NFY.

We next compared gkmPWM to tfmodisco-lite, an algorithm that can extract features from sequence-based models such as gkm-SVM. We used the 19 synthetic datasets from 4.2.2 (2). Since tfmodisco-lite requires a separate test set to extract importance scores using gkmexplain, we cross validated using a 80-20 split. **Fig 4.4EF** shows the similarity (correlation) between the seeded and learned motifs, for both gkmPWM and tfmodisco-lite. Just like HOMER, tfmodisco-lite had a difficult time consistently learning RUNX1, NFY, ETS, and NFKB, all of which are shorter motifs.

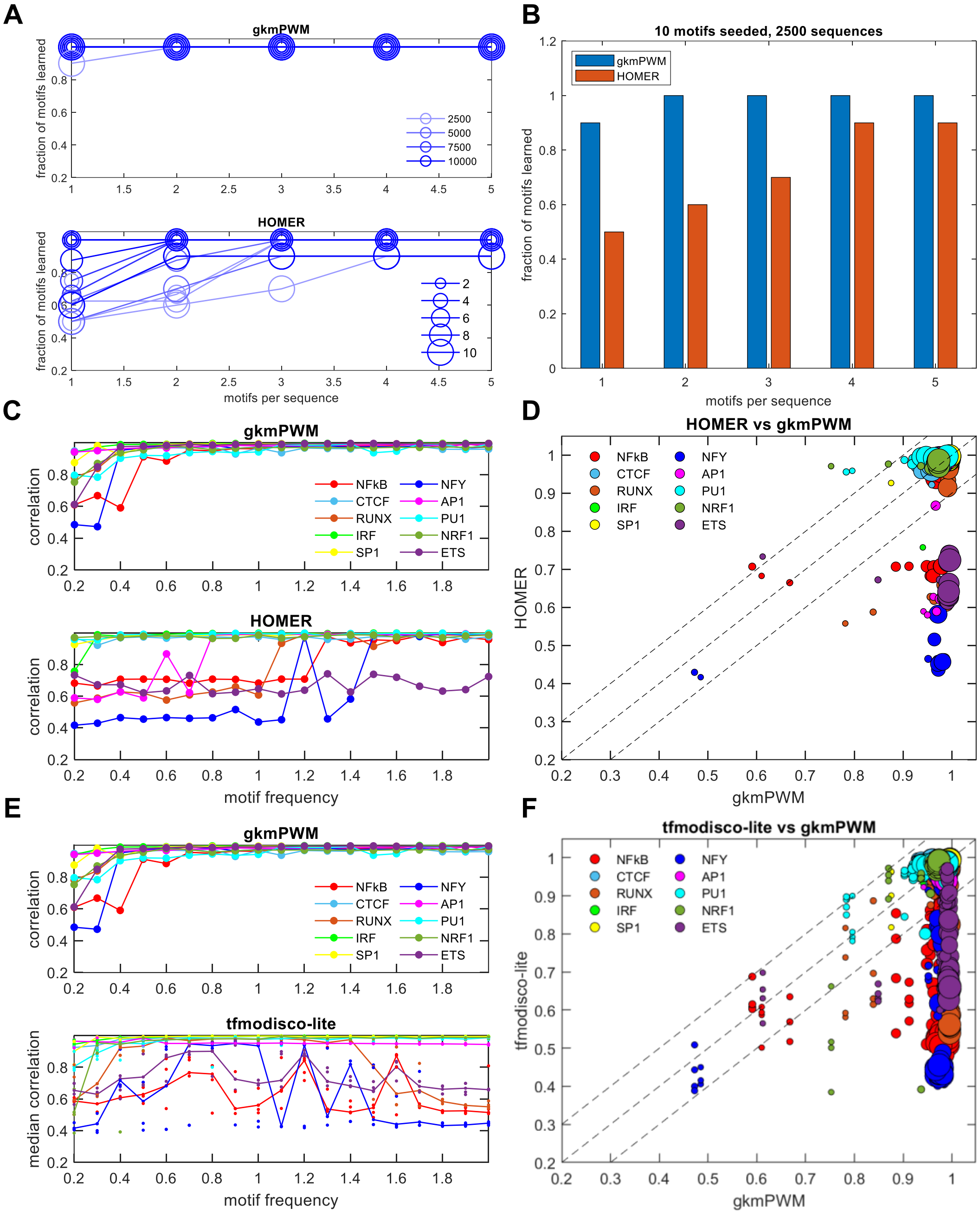
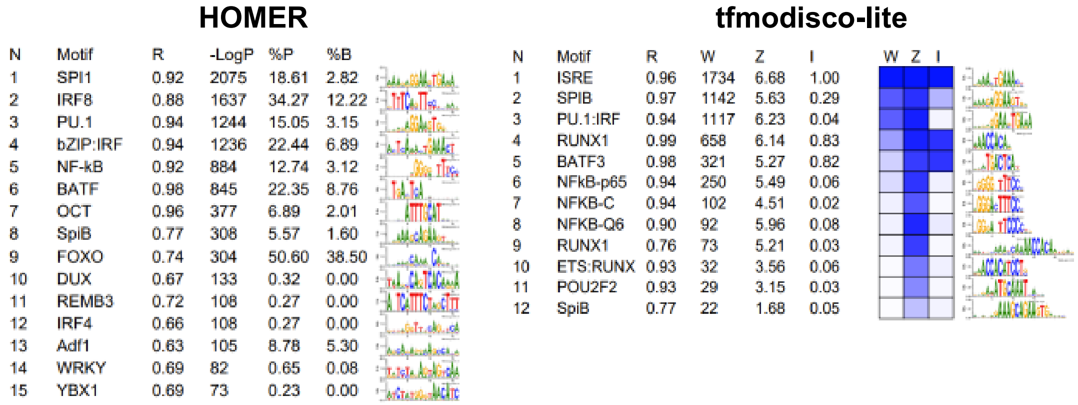


Figure 4.4 gkmPWM outperforms HOMER and tfmodisco-lite on predicting synthetic regulatory elements

A) gkmPWM outperforms HOMER on the 100 synthetic datasets that varied the size of the training set (shade of the marker), number of unique motifs seeded (size of the marker), and the number of motifs per sequence (x-axis). B) A bar plot of the performances of HOMER and gkmPWM for 2500 training sequences and 10 unique motifs seeded. C, D) The performance of HOMER and gkmPWM when the 19 synthetic sequences with different relative motif frequencies. The size of the marker in D is proportional to the AUROC of the model, indicating the difficulty of extracting motifs. E, F) tfmodisco-lite and gkmPWM comparisons of the same synthetic sequences in C and D.

We next applied all three methods to real DHS distal enhancers from the lymphoblastoid cell line GM12878. All models learn a subset of similar motifs, but HOMER (**Fig 4.5A**) and tfmodisco-lite (**Fig 4.5B**) do not detect the full set of motifs learned by gkmPWM, all of which have literature and ChIP-seq support for being relevant in B-lymphoblasts. HOMER does not identify RUNX1, one of the top motifs detected by the other methods, with a known role in B-cells. HOMER and tfmodisco-lite miss both EBF1 (Early B-cell Factor) and MEF2. All models detect some redundant motifs, which gkm-PWM identifies with a low I score.



HOMER

N	Motif	R	-LogP	%P	%B
1	SPI1	0.92	2075	18.61	2.82
2	IRF8	0.88	1637	34.27	12.22
3	PU.1	0.94	1244	15.05	3.15
4	bZIP:IRF	0.94	1236	22.44	6.89
5	NF-kB	0.92	884	12.74	3.12
6	BATF	0.98	845	22.35	8.76
7	OCT	0.96	377	6.89	2.01
8	SpiB	0.77	308	5.57	1.60
9	FOXO	0.74	304	50.60	38.50
10	DUX	0.67	133	0.32	0.00
11	REMB3	0.72	108	0.27	0.00
12	IRF4	0.66	108	0.27	0.00
13	Adf1	0.63	105	8.78	5.30
14	WRKY	0.69	82	0.65	0.08
15	YBX1	0.69	73	0.23	0.00

tfmodisco-lite

N	Motif	R	W	Z	I
1	ISRE	0.96	1734	6.68	1.00
2	SPIB	0.97	1142	5.63	0.29
3	PU.1:IRF	0.94	1117	6.23	0.04
4	RUNX1	0.99	658	6.14	0.83
5	BATF3	0.98	321	5.27	0.82
6	NFkB-p65	0.94	250	5.49	0.06
7	NFKB-C	0.94	102	4.51	0.02
8	NFKB-Q6	0.90	92	5.96	0.08
9	RUNX1	0.76	73	5.21	0.03
10	ETS:RUNX	0.93	32	3.56	0.06
11	POU2F2	0.93	29	3.15	0.03
12	SpiB	0.77	22	1.68	0.05

Figure 4.5 GM12878 distal enhancer motifs from HOMER and tfmodisco-lite

The output of homer2 denovo contains 4 quantities. R is the correlation with a pre-trained motif. -LogP is the negative log p-value of enrichment. %P and $B are the percent of sequences in the positive set and background (negative set) that contain the motif. The output of tfmodisco-lite outputs just the number of sequences (W) used to generate the motif. Since it extracts motifs from a gkm-SVM model, I added Z and I from Fig 4.2.

4.4 Discussion

In this chapter, I showed how to extract TFBS motifs from a set of sequences using gapped kmer counts. I also demonstrated how to extract motifs from sequence-based models such as gkm-SVM and RCmax. I then compared the performance of gkmPWM to a widely used motif learning algorithm, HOMER, and another feature extraction method tfmodisco-lite. In 4.1.7, I outlined many of the shortcomings of current motif extraction methods. They are apparent when training on both the synthetic and real datasets.

HOMER had a difficult time learning shorter motifs. The probability of short motifs occurring by chance throughout the genome is exponentially lower than longer motifs. Therefore, their enrichment with respect to the genomic background is likely to be lower. gkmPWM, when paired with gkm-SVM, overcomes this problem by weighing gapped kmers that are predictive of the regulatory elements. Additionally, HOMER learns motifs that have low fractions of occurrence in the positive set that are still enriched because they occur at an even lower rate in the background (motifs 10-15 in **Fig 4.5A**). This can be avoided by using a sequence-based model. gkmPWM's stronger performance cannot just be attributed to its compatibility with gkm-SVM. It outperforms tfmodisco-lite, which also learned motifs from sequence-based models, when learning motifs from both synthetic and real datasets.

The improved performance of gkmPWM over other methods in detecting moderately perturbed TF activity has been shown to be effective in detecting and interpreting mechanisms of dysregulated chromatin state in cancer,[74–79] in response to drug treatment,[80] and regulatory responses to CRISPRi perturbation[23,81], genetic perturbations[82–84], and other environmental stimuli. While the results reported here are robust, it is also important to compare and validate algorithm performance in blind computational assessments on completely held out biological test data, and we look forward to assessing gkmPWM through mechanisms similar to CAGI.[9,60,71,85]

Chapter 5

The Sequence Composition of *cis*-Regulatory Elements

In this chapter, I will use gkmPWM and gkmPWMlasso to learn the TFBSs that characterize the three common classes of regulatory elements: distal enhancers, promoters, and insulators. gkmPWM's high sensitivity, which I demonstrated in the previous chapter, should allow me to generate as complete of a motif list as possible for each dataset. I will apply gkmPWM and gkmPWMlasso on gkm-SVM models trained on 1644 enhancer and promoter human DNase/ATAC-Seq datasets and 2879 ChIP-Seq datasets. I will show that that the TFs that bind distal enhancers are cell-type specific, whereas the TFs that bind promoters are cell-type independent. In addition, I will show that the motifs extracted from distal enhancer models are highly predictive of the cell-type, and that motifs extracted from promoters have little predictive power. I will cluster all ChIP-Seq datasets using gkmPWMlasso motifs and explain the patterns of clustering. Lastly, I will define sequence rules for distal enhancers, promoters, and insulators.

5.1 Introduction

There are multiple definitions of regulatory elements. There is a purely descriptive definition, which characterizes enhancers and promoters by their function. Promoters are sequences that recruit RNA Polymerase II, and enhancers are distal elements that help stabilize the interaction of the promoter with RNA Polymerase II. The only quantitative component that can be inferred from this definition is the distance from the gene. There are also definitions that characterize regulatory elements by their epigenetic marks. Promoters and enhancers both have high chromatin accessibility and H3K27ac signal.

Promoters also have high H3K4me3 signal and enhancers have high H3K4me1 signal. Although these definitions are more quantitative than the previous one, histone marks tend to be very broad, making it difficult to find the exact location of the enhancer or promoter. The definition that I have used are a combination of these two, where I define promoters as DNase/ATAC-Seq peaks within 2kb of a TSS, and enhancers as cell-type specific DNase/ATAC-Seq peaks (not in 30% of other DNase/ATAC-Seq datasets) and greater than 2kb from a TSS. Like the previous two, these definitions are not perfect. There is no reason an enhancer cannot be within 2kb of a TSS, and the cutoff 30% is arbitrary. Regulatory elements contain sequences to which TFs bind. Upon binding, TFs recruit other proteins that initiate the transcription process. Thus, a more concise and rigorous way to define regulatory elements is by the TFs that bind them. In order to learn which TFs bind the different classes of regulatory elements, I will run gkmPWM and gkmPWMlasso on gkm-SVM models trained on a large number of experiments across many cell-types. Additionally, I will demonstrate the predictive power of these motifs by using them to predict the cell-type of the experiments.

5.2 Methods

5.2.1 Training gkm-SVM models on human DNase-Seq, ATAC-Seq, and ChIP-Seq datasets and evaluating their performance

I used the same DNase-Seq and ATAC-Seq datasets that were processed for gkm-SVM and RCmax comparisons (3.2.5). This created 1644 promoter and distal enhancer datasets. Additionally, I called MACS2 peaks ($p=10^{-9}$) on 2879 ChIP-Seq datasets. I did not split them into promoter and enhancer datasets. I ran gkm-SVM using default parameters l=11 k=7 d=3 t=2). To prevent overfitting, I used five similar sized chromosomal test-set splits (1, 3, 6; 2, 8, 9, 16; 4, 11, 12, 15, Y; 5, 10, 14, 18, 20, 22; 7, 13, 17, 19, 21, X). For each CV split, I trained gkm-SVM on the top 10,000 peaks against an equal sized negative set of genomic background matched to its GC content and common genomic repeat frequency. The models from each CV split were tested on the remaining chromosomes. The performance of the model was measured by the mean AUROC. The models are available on the ENCODE portal (https://www.encodeproject.org/search/?type=Annotation&status=released&annotation_type=gkm-SVM-model).

5.2.2 Clustering a large set of motifs to reduce redundancy

I used a list of known PWMs from three motif databases TRANSFAC, JASPAR, and UniPROBE, and removed replicants for a list of 1968 motifs (https://www.beerlab.org/gkmpwm/combined_db_v4.meme). However, many of the PWMs that remained were very similar to other motifs in the list. To filter this list further, I mapped all PWMs to the (11,7) gapped kmer space, and calculated the covariance matrix. I created an adjacency matrix from the covariance matrix using the Heaviside step function $u(x - 0.80)$, which maps all correlations greater than 0.80 to one, and the rest to zero. I defined clusters by using connected components. In other words, if we construct a graph from the adjacency matrix, then two datasets are in a cluster if there exists a path between them. This created 448 total clusters. For each type of dataset (enhancers, promoters, and ChIP-Seq), I picked a motif to represent each cluster by the number of times it was selected by running gkmPWMlasso on the entire 1968 motif list with default parameters.

5.2.3 Training gkmPWMlasso and gkmPWM

For each dataset, I combined the five models trained on different CV splits by averaging their weight vectors. Since the SVM is a linear model, averaging the weight vectors should retain all information learned in each model. For gkmPWMlasso, I extracted 30 PWMs using the full 1968 motif list and the reduced list of 448 motifs with default parameters (l = 11, k = 7, fraction of total gapped kmers = 0.19). For gkmPWM, I set the number of positive PWMs to initialize to 25 and the number of negative PWMs to 5 to maximize sensitivity. PWMs were seeded using kmer weights (available on the ENCODE portal under gkmsvm-model annotations).

5.2.4 Clustering DNase/ATAC-Seq datasets into 53 cell/tissue types

We clustered all the human ENCODE chromatin accessibility data by the similarity of their accessibility profiles. We generated a consensus set of 972,649 peaks using 1D clustering to merge peaks, and mapped DHS/ATAC signal from 1644 experiments into these 300bp genomic bins. We then generated 53 clusters of experiments with similar accessibility profiles using k-means. Experiment labels within each cluster are very consistent, and we assigned labels to each cluster according to its most prevalent cell-type. Cluster assignments are available in the second column of https://www.beerlab.org/gkmpwm/expt_table.html.

5.2.5 Running t-SNE on DNase/ATAC-Seq and ChIP-Seq datasets using gkmPWMlasso motifs as features

Using the appropriate reduced motif list, we set gkmPWMlasso to learn 30 motifs from the combined models. I ran t-SNE using 448 Boolean motif features, where a feature was given a 1 if the motif was learned with a positive regression weight, and a 0 otherwise. I avoided using the exact weights to prevent biasing towards the strongest motifs. I used MATLAB’s t-NSE with the Jaccard distance to cluster the motifs.

5.2.6 Training and testing Naïve Bayes models on DNase/ATAC-Seq distal enhancer and promoter datasets using gkmPWM *de novo* motifs as features

I used a Bernoulli Naïve Bayes classifier using gkmPWM (*de novo*) motifs_as features to predict the assigned cell/tissue type (cluster) of each dataset. I used the same 448 motifs in the t-NSE clustering as features. A feature was given a 1 if had a correlation greater than 0.8 with one of the gkmPWM motifs. The likelihood function of generating $\boldsymbol{x}$ given cluster C_k over n features is

$$p(\boldsymbol{x}|C_k) = \prod_{i=1}^{n} p(x_i = 1|C_k)^{x_i}\, p(x_i = 0|C_k)^{1-x_i}$$

x_i are the Boolean features of whether the motif i was learned in that dataset with positive weight. $p(x_i = 1|C_k)$ is the fraction of datasets that learned motif i in cluster C_k. Since we are working with Boolean variables, $p(x_i = 0|C_k) = 1 - p(x_i = 1|C_k)$. We used uniform priors ($p(C_k) = p(C_{k'})$ for all k, k'). I performed 5-fold cross validation and used fraction misclassified as the performance metric. Since k-means often creates multiple clusters if there are many data points with similar features (i.e. the T-cell and brain datasets), I considered a dataset's prediction to be correct if that dataset had an average correlation with the DNase/ATAC-Seq distal signal profile of 0.4 or greater with all other datasets in the predicted cluster. I removed poorly clustered datasets (average correlation with the DNase/ATAC-Seq distal signal of all other datasets assigned to the same cluster is less than 0.4). These mostly included models with poor performance or cancer cell-lines.

5.3 Results

We next apply gkmPWM to detect sequence features predictive of chromatin accessibility, TF binding, and enhancer activity in a large range of datasets generated by ENCODE. These datasets include the 1644 DNase/ATAC-Seq promoter and enhancer peaks from Chapter 3 and 2879 ChIP-Seq datasets, primarily from nine cell-types (GM12878, K562, HepG2, A549, HeLa-S3, HEK293, MCF-7, SK-N-SH, and H1). Overall performance of the models was high (**Fig 5.1** median promoters = 0.933, distal = 0.897, ChIP-seq = 0.907). I will demonstrate that the motifs extracted with gkmPWM can predict the tissue or cell-type of the dataset that was used to train each model.

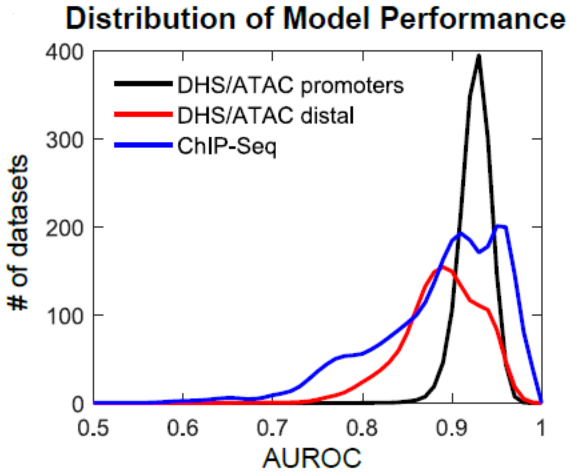


Figure 5.1 The performances of gkm-SVM models trained on promoters, enhancers, and ChIP-Seq

The distributions of the AUROCs of gkm-SVM models trained on a 1644 promoter (black), 1644 enhancer (red), and 2879 ChIP-Seq (blue) datasets. AUROCs were calculated using 5-fold cross-validation by splitting the training and test sets by chromosome. Most models achieve high performance (>0.9).

Although most of the models learn motifs bound by cell-specific TFs that one would expect from known biology (e.g. NFkB is learned in lymphocytes), the overall predictive power of the learned motifs is not obvious because of two confounding factors. First, many different TFs contain the same binding domain and sequence preferences. The number of TFs is vastly greater than the number of unique motifs. While it is possible to define a map TF to motifs, the inverse map is not well-defined. As a result, a single motif usually gives little information about the cell-type. Second, TFs can cooperate with other TFs and cofactors to bind at distinct genomic locations in distinct cell types, and it is not obvious which cofactors are required for cell specificity. Although these sequence models are trained on individual datasets and achieve high test set CV performance, this by itself does not ensure the cell specificity of the motifs learned. If the models are learning the important cell-type specific TFBSs, we should be able to determine the cell-type based on the sequence features we learn. To test this hypothesis, we clustered all the human ENCODE chromatin accessibility data by the similarity of their accessibility profiles. We generated a consensus set of 972,649 peaks using 1D clustering to merge peaks, and mapped DHS/ATAC signal from 1644 experiments into these 300bp genomic bins. We then clustered the experiments by their accessibility profiles using k-means (k=53). The cell-types of the individual datasets within each cluster are very consistent, and thus we assigned labels to each cluster according to its most prevalent cell-type. I next showed that the features learned by gkmPWM can reconstruct these clusters, by generating a t-SNE plot, using binary motif features and a Jaccard similarity metric on the positive weight motifs of distal enhancer models. The accessibility cluster labels are almost completely preserved in the t-SNE clusters (**Fig 5.2**). However, when using the motifs extracted from promoter models, we were not able to preserve the clustering as well (**Fig 5.3**).

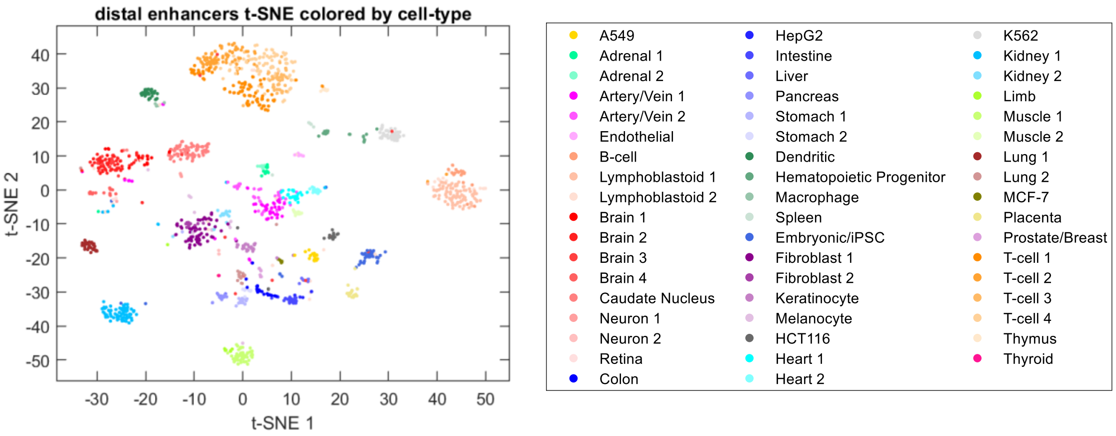


Figure 5.2 t-SNE plot of distal enhancer models using gkmPWMlasso motifs as features

t-SNE plot of 1644 distal enhancer gkm-SVM models using gkmPWMlasso motifs. t-SNE was ran with 448 Boolean motif features, indicating if a motif was extracted with positive weight, with the Jaccard distance. Each dataset is colored by their cell-type cluster (n=53), which was determined by running k-means on the DHS/ATAC signal on a combined list of 972,649 accessible peaks. Clusters with similar cell-type compositions (e.g. T-cell 1 and T-cell 2) were given different shades of the same color. Overall, the gkmPWMlasso motifs from distal enhancers were able to produce a very similar clustering to the DHS/ATAC peaks.

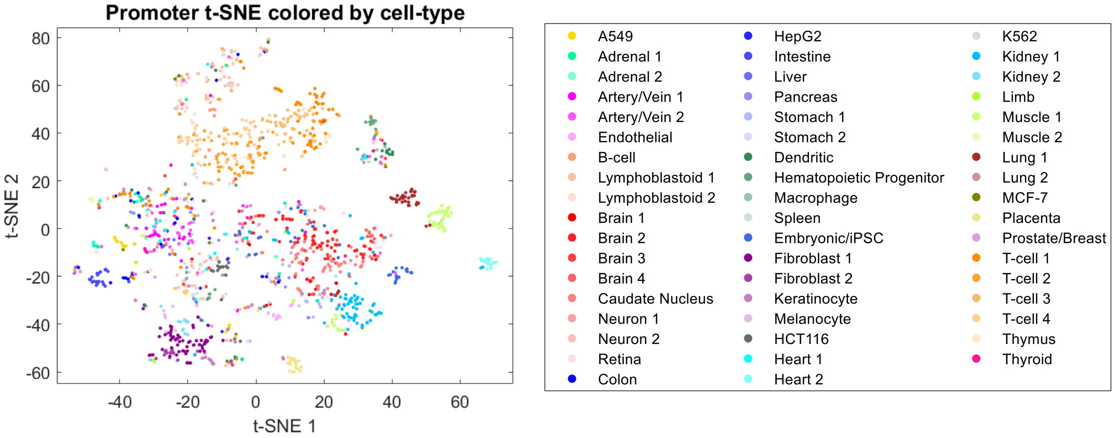


Figure 5.3 t-SNE plot of promoter models using gkmPWMlasso motifs as features

t-SNE plot of 1644 promoter gkm-SVM models using gkmPWMlasso motifs. t-SNE was ran with 448 Boolean motif features, indicating if a motif was extracted with positive weight, with the Jaccard distance. Each dataset is colored by their cell-type cluster (n=53), which was determined by running k-means on the DHS/ATAC signal on a combined list of 972,649 accessible peaks. Clusters with similar cell-type compositions (e.g. T-cell 1 and T-cell 2) were given different shades of the same color. In contrast to the distal enhancer motifs, promoter motifs were not able to maintain the clusters from DHS/ATAC peaks as cleanly.

Although the clusters in **Fig 5.2** are well-defined and pure, t-SNE clustering does not rigorously demonstrate the predictive power of the extracted motifs. To quantitative measure this, I trained a Naïve Bayes classifier to predict the assigned cluster of each distal enhancer dataset using only the gkm-PWM derived motifs as features (**Fig 5.4**). Some of the clusters are very similar (e.g. T-cell 1 vs T-cell 2), so cluster centers with C>0.4 were not considered errors. We removed 65 datasets that poorly matched all cluster (C<0.4 to any cluster center). The Naïve Bayes classifier was able to correctly classify greater than 96% of these 1579 datasets correctly using only the gkmPWM motifs. Some of the gkmPWM motifs learned in different datasets are slight variants of the same TFBS, so to find a more compact set of motifs, we iteratively ran gkmPWM by removing the bottom 10% of the least commonly learned motifs. The Naïve Bayes classification accuracy is shown in (**Fig 5.5A**). Surprisingly, performance remains quite high as the size of the total motif set is reduced, and a significant drop in performance does not occur until the total motif set is reduced to approximately 70 motifs. We show the frequency of motif usage across all clusters for *n*=68 motifs in **Fig 5.4**). Since there is significant redundancy in the immune, brain, and cancer/EBV transformed cell lines, the number of motifs that are shared amongst these clusters is large. However, although most motifs are detected in many clusters, datasets can be predicted with high accuracy with combinations of motifs. This is possible because of the overall sparsity of the motif frequency matrix. It might seem surprising that only $n \approx 70$ motifs can predict 53 clusters when there are $n \approx 10$ motifs per cluster, but given the number of unique motifs n, and the number of motifs m per cluster, the number of unique combinations of motifs is $\binom{n}{m}$. n does not need to be exceptionally large since small increases in m can drastically increase $\binom{n}{m}$. As we reduce the total motif set, we can also calculate the average number of motifs detected per cluster

(**Fig 5.5B**). Total predictive performance is high until ~ 8 motifs per cluster, which suggests that most human cell types have between 8 and 10 active core lineage determining TFs. This compact set of distal enhancer TFs are sufficient to establish the chromatin accessibility landscape of the cell/tissue and shows that most human distal enhancer peaks are specified by a relatively small number of TFs.

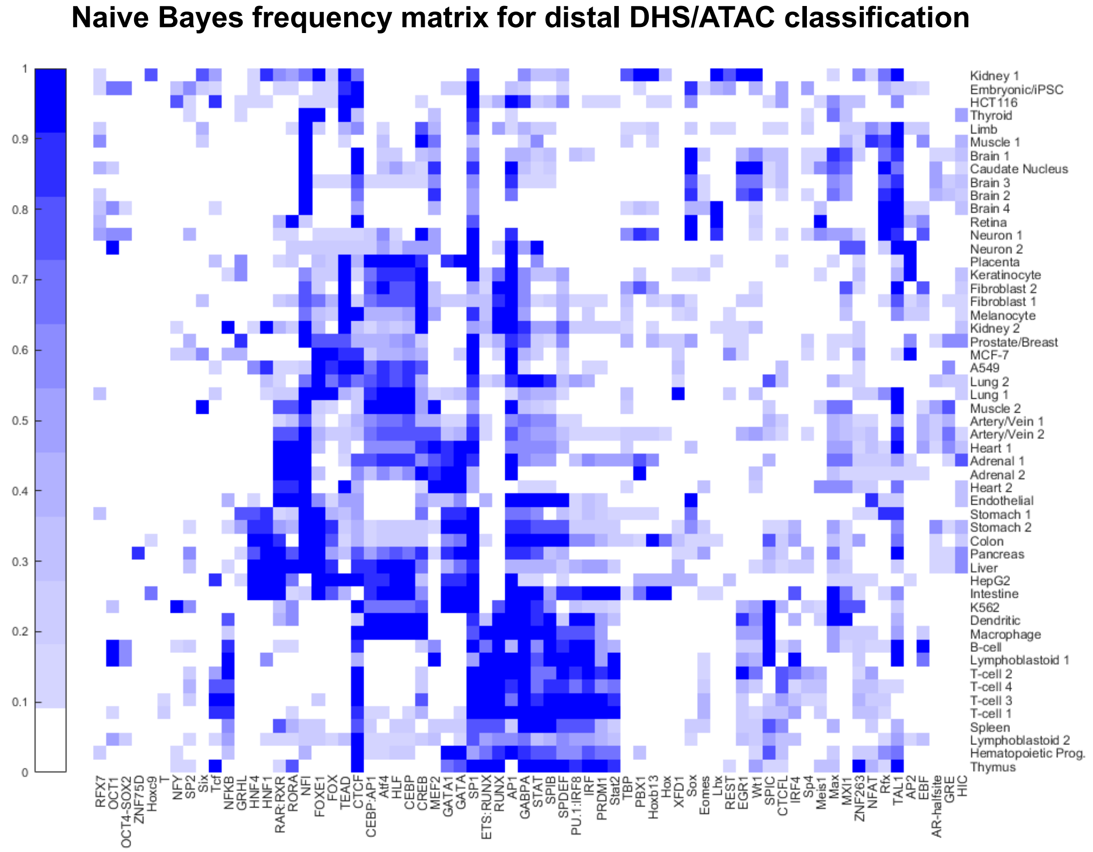


Figure 5.4 Naive Bayes frequency matrix of predictive motifs across cell-types in distal enhancers

The probability of a feature given the class $p(x_i = 1|C_k)$ *for a Bernoulli Naïve Bayes classifier can be stored in a m by n matrix where m is the number of classes (m=53), and n is the number of features/motifs (n=68). The prediction accuracy for this number of motifs is 92%. Each row contains the motifs that are present in distal enhancers for a cell-type. The rows and columns are clustered hierarchically (dendrogram not shown). Cell-types when common lineages are arranged close together.*

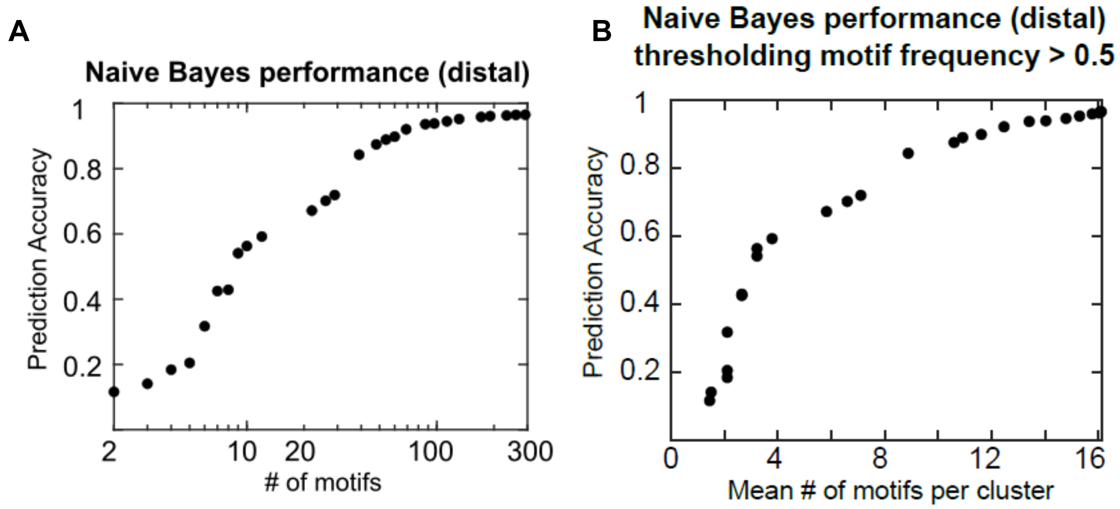


Figure 5.5 Naive Bayes classification of distal enhancers when reducing the number of motif features

A) *The prediction accuracy (fraction of datasets classified correctly) as 10% motif features were recursively removed.* ***B)*** *The prediction accuracy against the mean number of motifs per cluster that were learned in greater than 0.5 of the datasets in a cluster* $(p(x_i = 1|C_k) > 0.5)$.

We next perform the same analysis on the promoter DNase/ATAC-Seq datasets, but as expected, since promoter activity and promoter regulatory vocabulary varies little across cell types, the predictive performance is not as strong as it is for distal enhancers (**Fig 5.5A**). The definition of promoters used for creating training sets (>2kb from a TSS) is not perfect and likely contains enhancers. When including all motifs, the fraction correctly classified is around 90%. However, when the least predictive motifs are removed, the drop in performance occurs sooner (**Fig 5.5B**). In fact, when the least predictive motifs are removed, the average number of common motifs per cluster does not change as quickly. This implies that cell-type specific motifs, likely ones learned from enhancers that leaked into the promoter set, are removed first. When smaller number of motifs remain, promoters have nearly no predictive power. This is a result of a small number of common and highly predictive TF motifs that are extracted from the vast majority of DNase/ATAC-Seq promoter sets (**Table 5.1**). These motifs, despite their high frequency of occurrence, are not predictive of any cluster (**Fig 5.7)**. When taking the correlation across all pairs of gkm-SVM weight vectors, we found that DNase/ATAC-Seq promoter models were quite correlated, especially in comparison to pairs of distal enhancer models. This result is consistent with our previous work, where we showed that different DHS promoter datasets were equally predictive of saturation mutagenesis assays that were done on promoters, regardless of the cell-type. This analysis shows that human promoters are broadly of one class and are all active across all cell-types.

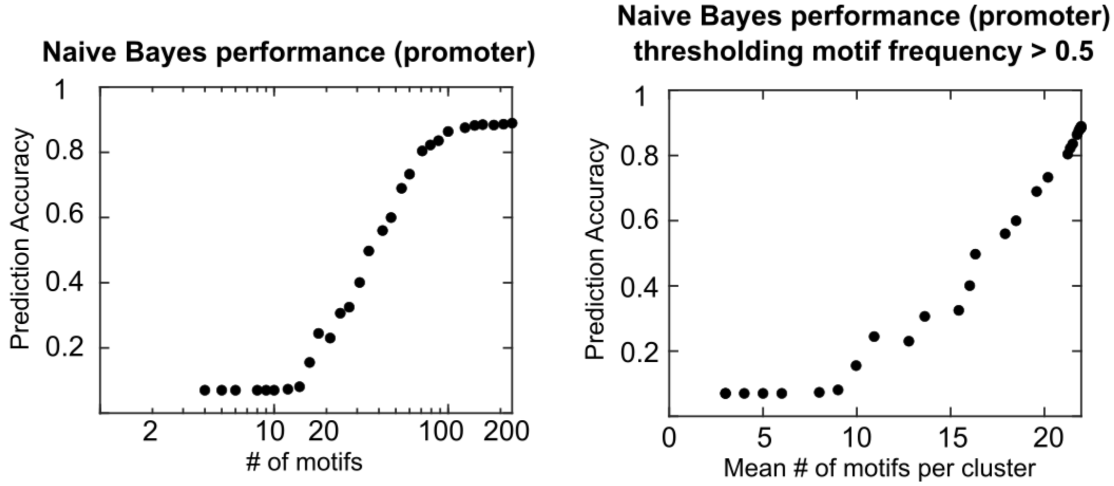
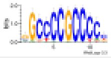
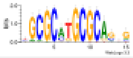
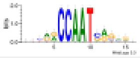
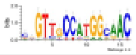
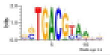
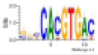
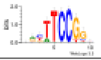
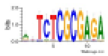
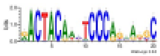
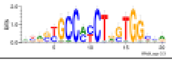
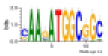
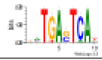


Figure 5.6 Naive Bayes classification of promoters when reducing the number of motif features

A) *The prediction accuracy (fraction of datasets classified correctly) as 10% motif features were recursively removed.* ***B)*** *The prediction accuracy against the mean number of motifs per cluster that were learned in greater than 0.5 of the datasets in a cluster. In contrast to distal enhancers, the prediction accuracy rapidly falls when the number of motifs is removed.*

Table 5.1 Predictive promoter motifs in greater than 0.50 of datasets

SP1 	NRF1 	NFY 	RFX 	CREB 	Max 
GABPA 	GFX 	ZNF143 	CTCF 	YY1 	AP1 

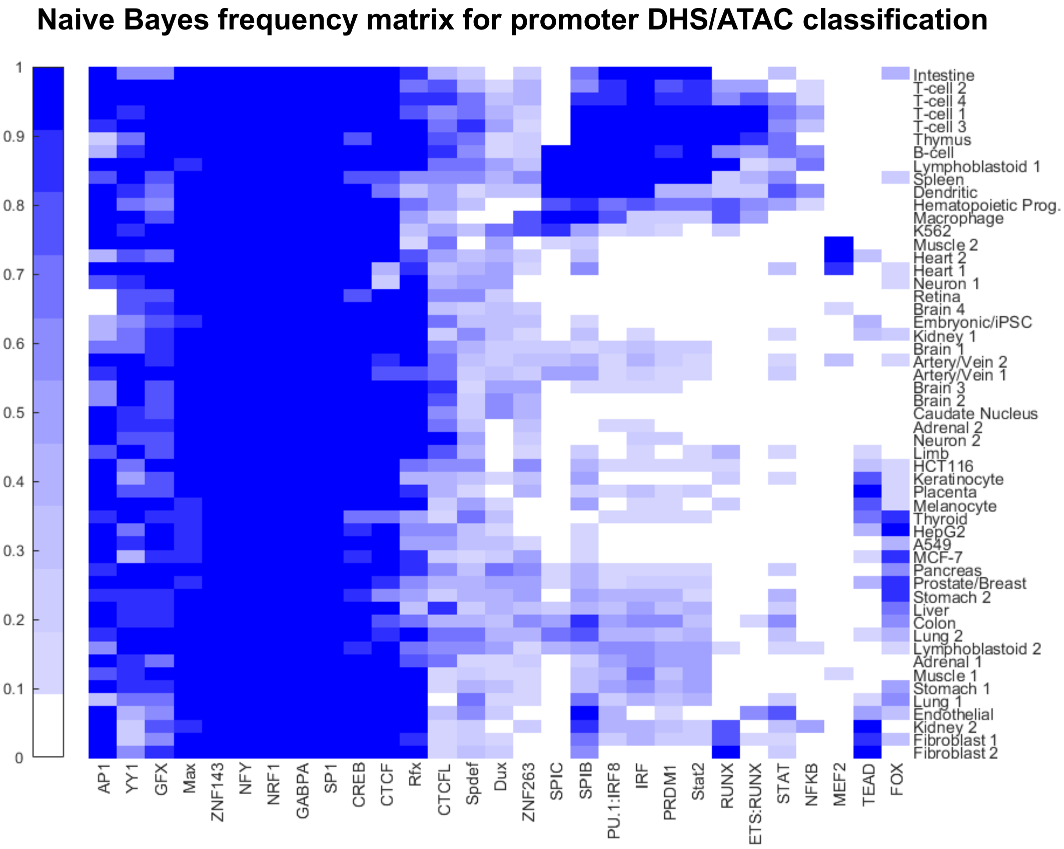


Figure 5.7 Naive Bayes frequency matrix of predictive motifs across cell-types in promoters

The Bernoulli Naïve Bayes classifier for promoters with n = 60 features/motifs (only 29 shown after removing redundant motifs). The prediction accuracy for this number of motifs is 72%. For this number of motifs, there are still cell-type specific motifs that are found in Fig 5.4. However, there are motifs that are cell-type independent located on the left of the figure that have no predictive power.

We next performed t-SNE clustering on the motifs learned from ENCODE TF ChIP-seq datasets using the same approach. Most of the TF datasets clustered by their cell-type (**Fig 5.8A**) (clusters of similar color), but there were two noticeable exceptions. The highly variable region near the cell-type specific clusters had a high promoter content (**Fig 5.8AB**, dashed circle).

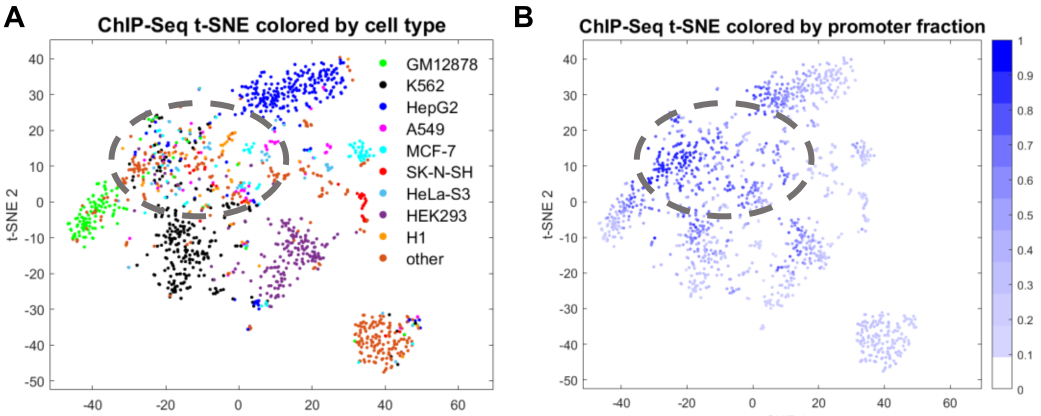


Figure 5.8 ChIP-Seq t-SNE with gkmPWMlasso motif features primarily clusters peaks by cell-type and promoter content

A) *Most ChIP-Seq datasets from the same cell-type have similar sequence features.* ***B)*** *However, there are many ChIP-Seq datasets from different cell-type that cluster together (gray dashed circle). These TFs tend to bind to promoters. The most prevalent protein targets in this region are Pol II and TFs from Table 5.1.*

The other variable region contains ChIP-Seq experiments that targeted CTCF or cohesin subunits (RAD21, SMC1, SMC3) (**Fig 5.9A**). The only motif that is consistently extracted in CTCF ChIP-Seq models is CTCF (**Fig 5.9B**), and its PWM is very predictive of CTCF binding. Interestingly, CTCF seems to be the only motif that is highly predictive of CTCF binding.

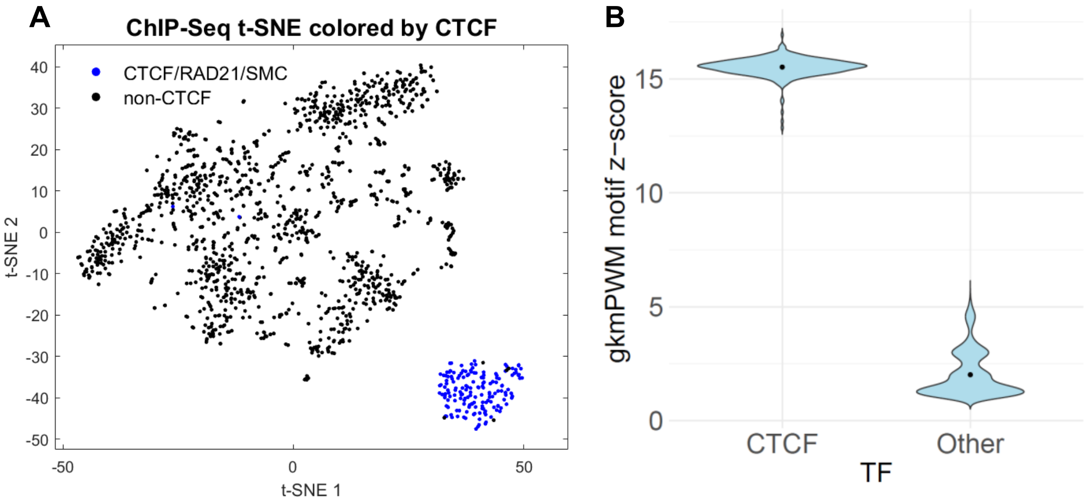


Figure 5.9 The largest cell-type independent cluster in ChIP-Seq t-SNE is CTCF

***A**) The most noticeable cell-type independent cluster are ChIP-Seq datasets that target CTCF or cohesion components. This cluster is very different from other ChIP-Seq datasets because the only predictive motif of CTCF ChIP-Seq seems to be the CTCF motif. **B**) The distribution of the z-score (Fig 4.1) of the CTCF motif relative to other the z-scores of all other motifs that were called by gkmPWMlasso. The CTCF motif is weighed very heavily by gkm-SVM.*

Prior to generating the distal enhancer and promoter models, we merged and removed all DHS and ATAC-seq peaks that were active in over 30% of the experiments (ubiquitous peaks). The largest single class of element in this set are CTCF binding sites, and 63% of the ubiquitous peaks overlap with one of the CTCF ChIP-seq peaks (**Fig 5.10**). Thus, CTCF is the third broad class of genomic element and is mostly accessible and bound in all cell/tissue types. Previous analysis of cell-type specific CTCF binding has identified variable occupancy driven by methylation. We find some variation in CTCF binding depending on cell type that depends strongly on the CpG content of the CTCF binding site and the flanking sequence context, but these are at weaker CTCF binding sites.

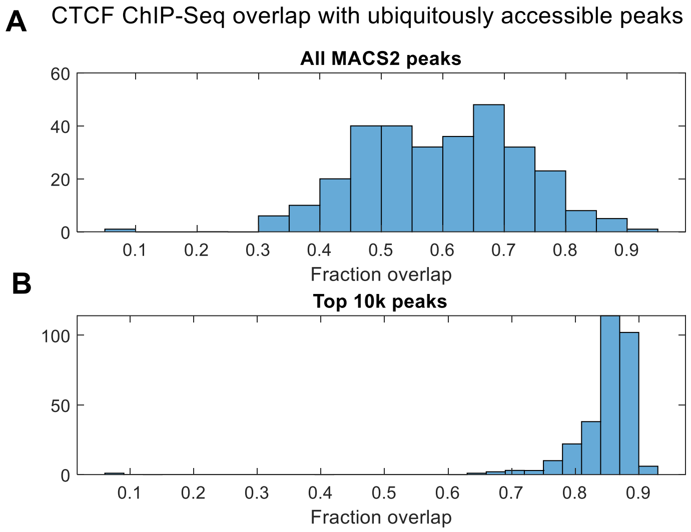


Figure 5.10 CTCF binding sites are prevalent in common accessible peaks

A) *Overlapping CTCF ChIP-Seq peaks with DHS/ATAC peaks that are called in more than 30 percent of datasets, most datasets have a fairly large fraction of their peaks in these regions.* ***B)*** *When restricting to the 10,000 strongest peaks, an even larger fraction of peaks overlaps with ubiquitously accessible peaks.*

5.4 Discussion

To summarize, we can combine this analysis of 1644 ENCODE chromatin accessibility experiments and 2879 ENCODE ChIP-seq experiments to construct simple rules of human CREs based on their motif content and predictive modeling:

1. Distal cell-specific enhancers contain cell-type specific TF motifs. Each cell type has a distinct set of 8-10 enhancer binding TFs whose activity is sufficient to determine the cell-type, but many of these TFs are active across multiple cell types.

2. Promoters are active across multiple cell-types, and all contain a remarkably similar set of promoter TF motifs.

3. The third class of non-cell specific peaks contain CTCF motifs and are mostly bound by CTCF in a cell-type independent manner.

While there are some exceptions to these rules, they explain the vast majority of the strongest peaks. Our analysis is limited to the top 10,000 strongest peaks in each dataset, but weaker peaks largely follow these rules.

Understanding the sequence features driving regulatory element activity is a crucial step toward eventually using ENCODE functional epigenomic datasets to build dynamic gene regulatory network models to predict cellular behavior. While we have made some progress in well-defined cellular transition model systems,[23] more complete models will need to incorporate how regulatory element activity is coupled to target gene transcriptional activation through DNA looping interactions constrained by cohesin extrusion and CTCF. As we have shown, CTCF binding is largely cell-type independent, and therefore TAD structure and enhancer-promoter interactions can now be predicted with reasonable accuracy.[22,23,86,87]

Chapter 6

Predictive Sequence Features of Functional Characterization Assays

In the previous chapter, I extracted sequence features from regulatory elements defined by DNase-Seq, ATAC-Seq, and ChIP-Seq, all of which are considered "mapping data." They give us information on the location of DNA with certain biomarkers, which is valuable in itself. However, they tell us little about the degree to which those regulatory elements contribute to gene expression. For DNase/ATAC-Seq, the signal in promoters is not correlated with gene expression since promoters tend to stay accessible across cell-types. On the other hand, the signal in distal enhancers, which are cell-type specific, is more predictive of gene expression. However, multiple enhancers often have the same target gene, which makes it difficult to parse the individual contributions of each enhancer. Lastly, ChIP-Seq only targets a single TF, and therefore does not encapsulate the information from all TFs like DNase/ATAC-Seq. A new class of experiments called functional characterization (FC) assays (or reporter assays) were developed to address these issues. These assays were developed to measure regulatory activity at the RNA level. As mentioned in Chapter 2, these include massively parallel reporter assays (MPRA), self-transcribing active regulatory region sequencing (STARR-seq), and CRISPR interference (CRISPRi). In ENCODE phase 4, labs that developed extensions of these assays to overcome shortcomings of the original assays attempted to characterize a large number of elements in K562, a cancer cell-line derived from an erythroblast[*]. In this

[*] Much of work done to process the experiments was done by Junke Zhang, Keith Siklenka, Alex Barrera, Kuei-Yueh Ko, Revathy Venukuttan, and Galip Gurkan Yardimci from the ENCODE 4 functional characterization centers.

chapter, I will describe these assays, and evaluate and compare their readouts from a sequence analysis perspective. I will test their ability and robustness to detect real regulatory signal and identify their sequence preferences (bias). In addition, I will present novel sequence information about regulation that FC assays can detect, but mapping assays fail to detect. Lastly, I will show how to manage the large GC bias in functional characterization assays.

6.1 Methods

There are two types of MPRA assays that will be examined: episomal and lentiviral. A description of episomal MPRA is provided in Chapter 2. For the analysis in this chapter, only the plasmid construct for enhancers was used. Episomal MPRA was applied to characterize entire gene loci, defined as a genomic region up to 1 Mbp in length that contains the gene of interest and the potential regulatory elements around it. The goal was to completely dissect each locus of all regulatory activity. MPRA was performed using 50 bp sliding windows across the entire locus, and higher resolution 25 bp sliding windows were used near or within DHS/ATAC peaks. Thus, this method will be referred to as “tiling MPRA” for the rest of this chapter. Genes were selected based on their activity in erythroblasts: and I will analyze data generated at the GATA1, MYC, HBE1, LMO2, and RBM38 loci. Although the size the dataset is quite large (> 100,000 regions tested), most of the tested regions do not have regulatory activity *in vivo*. Thus, training a sequence-based model can be difficult due to the small number of regions with true positive signal.

The second MPRA experiment is lentiviral MRPA (lentiMPRA). The construct of the plasmid is similar to episomal MPRA, but the plasmid is integrated into the genome with the aid of a lentivirus[88]. The purpose of this addition is to place the plasmid into the proper biological context. However, the degree to which lentiviral MPRA is different from episomal MPRA is uncertain. Inoue et al. demonstrate that when testing the same inserts, the correlation between episomal and lentiviral MPRA is lower than the correlation between replicates of the same assay. However, the cause of these differences remains unclear. The sequences selected as plasmid inserts were 200 bp regions centered on all chromatin accessible peaks in K562. As a result, the fraction of regions with regulatory effect is extremely high. 30 negative controls were also tested to compare against the elements derived from chromatin accessible peaks.

Two variants of STARR-Seq were tested: ATAC-STARR-Seq[89] (ATAC-STARR) and Whole Genome-STARR-Seq[90] (WG-STARR). In contrast to the two MPRA assays, whose tested regions were selected with specific criteria, the STARR-Seq assays tested randomly selected regions from the entire genome. For ATAC-STARR, a library was developed using Tn5 to extract sequences from the K562 genome. These sequences were then used as inserts for STARR-Seq. For WG-STARR, a library was developed by randomly shearing the genome, and using the fragments are inserts for STARR-Seq. Both of these assays yield high coverage, with WG-STARR covering a majority of the genome. However, like episomal MPRA, they both remove the tested sequences from their native genomic context.

The last assay under investigation is the CRISPRi proliferation screen (CRISPR growth screen). This assay used CRISPRi in parallel by targeting chromatin accessible peaks in K562. Each peak was targeted by multiple guide RNAs, which were designed to tile across the entire peak. The gRNAs recruit dCas9-KRAB, which deacetylates the nearby histones to reduce enhancer or promoter activity. These guides are randomly transfected into many cells. Cells with different combinations of targeted enhancer and promoter targeting gRNAs compete for proliferation. After 20 days, the guides were sequenced to determine which targeted enhancers or promoters had a positive effect on proliferation. This assay will only identify regulatory elements that affect proliferation or cell death and is therefore limited. However, it is the only *in vivo* FC assay of the five, which may allow detection of the effects of other factors impinging on transcription, such as 3-D architectural changes which can impact enhancer promoter interactions.

Although all these methods are designed to measure the regulatory effects, differences between the assays have not been thoroughly investigated. Previous work has shown that there can be differences between MPRA and STARR-Seq based on the design of the plasmid, but little analysis on the predictive power of the sequence features has been done[40]. Just like sequence-based models, functional characterization assays are used to predict regulatory activity. Previous work from CAGI has shown that sequence-based models can predict MPRA variant effects with fairly high accuracy[9,68]. Thus, I hypothesized that the sequence features of the inserts would be highly predictive of the assays' signals. Furthermore, I hypothesized that these sequence features are TF binding motifs. Thus, I created two types of evaluations for the FC assays: 1) Distinguishing the tested regions with high assay signal from genomic background (gkm-SVM). 2) Predicting the signal with regression (gkm-SVR). For both cases, I will extract motifs using gkmPWMlasso and show the similarities and differences of predictive sequence features between the assays.

6.1.1 Training and testing gkm-SVM models on high signal regions

All functional characterization data used is available on the ENCODE portal. Specific ENCODE ascension codes for datasets that were used are:

Tiling MPRA: ENCSR917SFD, ENCSR363XER

Lentiviral: ENCSR460LZI

ATAC-STARR-Seq: ENCSR312UQM, ENCSR926NDZ

Whole Genome STARR-Seq: ENCSR661FOW

CRISPR proliferation screen: ENCSR162HMH

For the plasmid-based assays, which include tiling (classical) MPRA, lentiviral MPRA (lentiMPRA), ATAC-STARR, and Whole Genome-STARR (WG-STARR), the metric of regulatory activity is the logarithm of RNA counts that were generated by the plasmid over the DNA counts of the plasmid. For the CRISPR growth screen, the output is the number of guide RNA counts at the end of the experiment over the guide RNA counts at the beginning of the experiment. Low signal regions with a DNA count lower than 5 were removed. For plasmid-based assays, a high log fold change implies high regulatory activity. For the CRISPR growth screen, a low log fold change implies that the targeted region is important for survival and/or proliferation, which would confirm that the targeted region has regulatory effect. I split the regions tested into promoter and distal regions by distance from a TSS (< 2kb → promoter).

There was an abundance of tested regions for plasmid-based assays (>10,000 tested regions). Using the chromosomal CV splits from Chapters 3 and 5, I took the 1000 regions with the highest log fold change in each CV split for each plasmid-based assay, for a total of 5000 regions. For tiling MPRA, which tested gene loci, I used individual gene loci as CV sets (CV 1 = GATA1, CV 2 = MYC, CV 3 = HBE1, CV 4 = LMO2, CV 5 = RBM38). Log fold changes from CRISPR growth screens tend to have a very small number of regions that yield significantly lower log fold changes compared to the plasmid-based assays. Therefore, I took 200 regions per CV split instead of 1000. I created equal sized negative sets that were GC and repeat matched. I trained gkm-SVM models using default parameters on four of the CV splits and testing on the other. I measured the performance using the AUROC. I trained promoter and distal models separately for each assay.

6.1.2 Training and testing gkm-SVR on a set of regions with a large range of signal

Since the functional characterization assays output a continuous value, I trained gkm-SVR (support vector regression) models to predict the log fold changes of tiling MPRA, lentiMPRA, ATAC-STARR, and WG-STARR. Tiling MPRA and lentiMPRA had 154276 and 115345 regions tested respectively, which gkm-SVR train on in a reasonable amount of time. ATAC-STARR and WG-STARR had far more regions tested. Many of the regions overlap with each other. I merged these overlapping regions into one, selected the middle 300 bp, and averaged their log fold changes. This resulted in 57436 regions for ATAC-STARR and 54330 sequences for WG-STARR. I split each set of sequences into five CV sets by chromosome. I trained gkm-SVR models on four of the CV sets and tested on their other. The predictive performance is measured by the correlation of the predicted log fold change with the experimentally measured log fold change.

6.1.3 Extracting motifs using gkmPWMlasso

For both gkm-SVM and gkm-SVR models, I averaged the 5 models from different CV splits. I ran gkmPWMlasso on the average models to extract 20 motifs. In **Fig 6.2** and **Fig 6.4**, motifs that were either learned in multiple assays, or were top motifs in at least one of the assays were included on the list.

6.1.4 Controlling for GC bias in high signal vs low signal regions in ATAC-STARR

I split ATAC-STARR regions by their GC content in bins of 5%. Most of the sequences had GC contents within 45-70%, so bins between 0-45% and 70-100% were merged. If I used the non-overlapping regions from 6.1.2, I would have too few sequences. Thus, I downsampled the sequences in each bin by sampling regions based on their log fold change. Most tested regions have log fold regions near zero, and thus, random sampling would yield very few regions with strong regulatory effect. To prevent this, I created 150,000 bins of uniform length, ranging from the lowest to the highest log fold change. Many of these bins had zero sequences. Therefore, sampling a sequence from each bin yielded 35,000 to 61,000 thousand sequences for each GC bin. I then trained and evaluated gkm-SVR models using the same method as in 6.1.2.

6.2 Results

6.2.1 The sequence features of high signal regions varies among the assays

The first evaluation of the functional characterization assay concerns their ability to consistently generate signal to distinguish sequences from genomic background. To do this, I trained gkm-SVM models similarly to the DNase/ATAC-Seq models in Chapter 5. However, I did not remove CTCF peaks like I did for DNase/ATAC distal enhancer models. In **Fig 6.1**, most assays produce high signals for sequences that are very distinguishable from the background. However, both tiling MPRA models and the CRISPR growth screen enhancer model yielded poor AUROCs. This is not surprising for tiling MPRA since most of the inserts selected are from genomic background. However, it is interesting that performance of the distal enhancer model of the CRISPR growth screen (AUC = 0.757) was significantly lower than its promoter model (AUC = 0.906). One would expect that disrupting enhancers that targeted the promoters that affected fitness should also affect fitness to the same degree. However, it has been shown that in CRISPRi experiments, targeting promoters has a long-lasting repressive effect on the expression of the associated gene, while targeting single enhancers only produces a transient effect, where a noticeable reduction in gene expression is only temporary[23], likely due to the activation of other enhancers acting on the same gene, which were not targeted by CRISPRi. This may be the reason few enhancers affect the fitness of cells, especially when compared to the number of promoters.

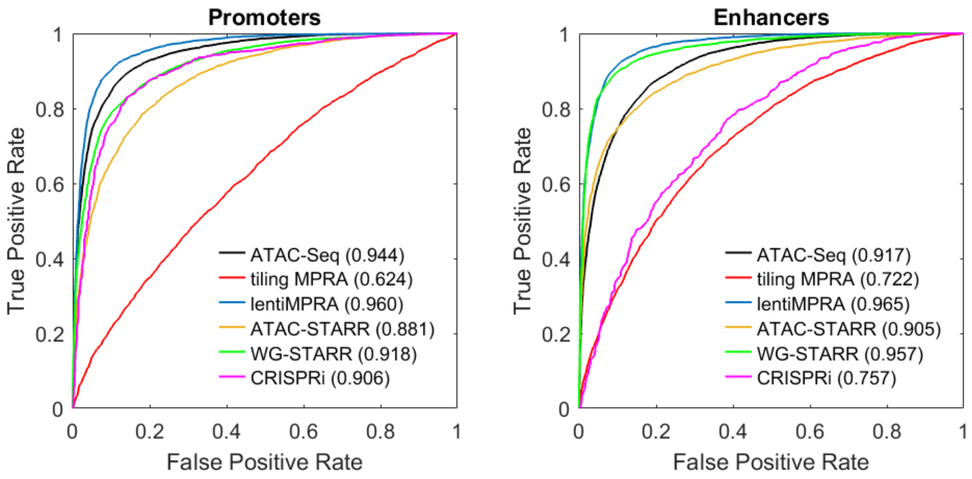


Figure 6.1 The performance of classifying high signal regions versus genomic background

The predictive power of sequence features to classify regions with high signal. To benchmark the assays, the performance of ATAC-Seq peaks was included. Most models have comparable performance to ATAC-Seq, with the exception of tiling MPRA (for promoters and enhancers), and CRISPR growth screen (enhancers).

Since most models produced high AUROCs, the models are learning consistent sequence features in their training sets. I ran gkmPWMlasso to determine what TF motifs are predictive in each assay. While there was some commonality of the predictive TF motifs of the assays, there were quite a few differences. In **Fig 6.2**, the z-scores (normalized to max(Z) = 1) of the predictive motifs of all five assays and ATAC-Seq are displayed. For most assays, the list of predictive promoter motifs is not as comprehensive as ATAC-Seq's list. Although lentiMPRA has a similar list to ATAC-Seq's, the inserts selected for testing were regions centered at DHS peaks. WG-STARR promoter motifs also had considerably overlap with ATAC-Seq motifs, but there were a few other motifs that were not associated with accessible promoters (**Table 5.1**). For distal enhancers, the motif list for all assays is considerably different from ATAC-Seq. Interestingly, GATA, which is the strongest motif by far in distal enhancer ATAC-peaks, is not the most predictive of any of the FC assays' enhancer models.

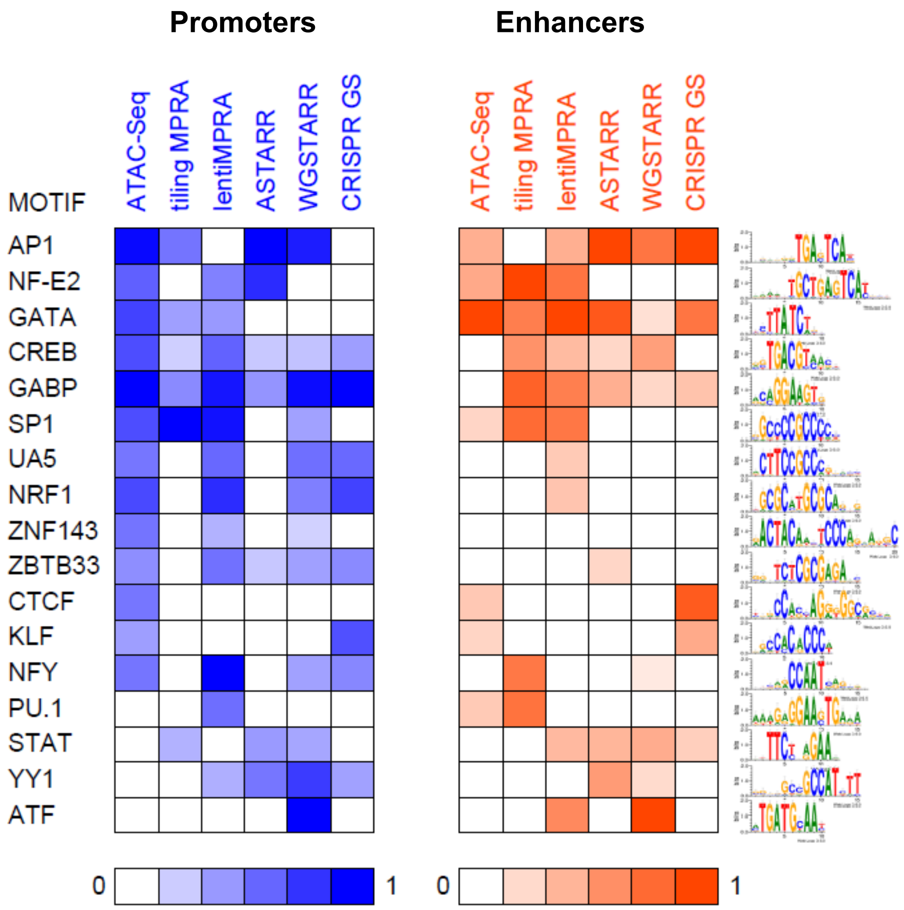


Figure 6.2 Predictive motifs of high signal in functional characterization assays

A condensed list of motifs extracted from the models in Fig 6.1 using gkmPWMlasso. Only motifs with positive W and Z that were highly predictive of at least one assays or motifs that were predictive of multiple assays were included.

Since the ATAC-Seq motifs seem to be quite different from the FC assays' motifs, I took the positive sets from the FC models trained in **Fig 6.1** and trained them against ATAC-Seq's positive set. I limited this analysis to lentiMPRA, ATAC-STARR, and WG-STARR because I was more confident that they had enough true positive signal compared to tiling MPRA and CRISPR growth screen. The evaluation of these models will be the same as in **Fig 6.1**, except the goal is to distinguish the high signal regions of each assay from ATAC-Seq peaks. A high AUROC implies that the models learn vastly different sequence features, and an AUROC near 0.5 means the models are not significantly different. For promoters (**Fig 6.3A**), lentiMPRA and WG-STARR have similar sequence features to ATAC-Seq. On the other hand, ATAC-STARR promoters are quite different. The reason for this will be discussed later in an analysis focused on ATAC-STARR. For enhancers (**Fig 6.3 B**), all assays have different sequence features compared to ATAC-Seq. From **Fig 6.2**, it is evident that all FC assays have noticeably different sequence features in enhancers than ATAC-Seq. In particular, the motifs in lentiMPRA enhancers are common motifs found in promoters. This is true for tiling MPRA as well. Differences in WG-STARR and ATAC-Seq enhancers seems to be a result of a strong ATF motif.

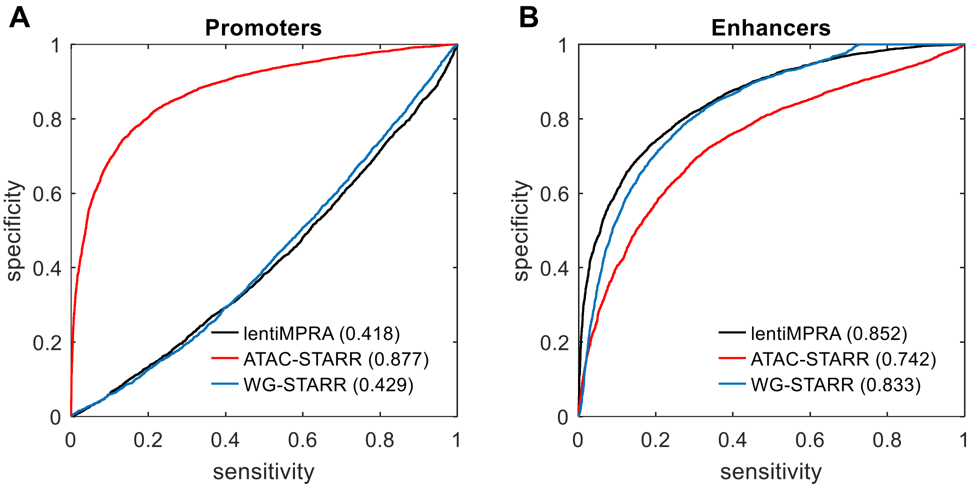


Figure 6.3 Comparing the sequence features of ATAC-Seq peaks to high signal regions

The performance of gkm-SVM models trained on the top 5,000 ATAC peaks and the top 5000 regions from lentiMPRA, ATAC-STARR-Seq, and Whole Genome-STARR-Seq in promoters and enhancers. The same sequences and CV splits from Fig 1 were used. Quantities in parentheses are the AUROC of the assay.

The most interesting result comes from the CRISPR growth screen, where a strong predictive motif in distal enhancers is CTCF. Since I did not remove CTCF sites like I did for DNase/ATAC-Seq models, these are most likely insulators rather than enhancers. Although the AUROC of this model is modest, it is significantly higher than 0.5, which gives confidence that the strongest motifs in the model are predictive. None of the other FC assays had the CTCF motif as a predictive sequence feature. Since their plasmid designs enable direct proximal enhancer-promoter interaction, and do not require looping, it is not surprising that CTCF does not generate a high signal for MPRA and STARR-Seq. However, since CRISPR growth screens are an *in vivo* assay, the effects of 3-D conformation changes can potentially be measured. Disruptions in a CTCF site can affect expression in two ways. First, it can prevent the formation of a loop that increases the probability of interaction of an enhancer and promoter pair, and thus decreasing gene expression. Conversely, the disruption of a loop can allow for the interaction of an enhancer and promoter that was previously blocked by the loop, increasing expression. This feature of CRISPRi is not limited to the growth screens but is applicable to all CRISPRi experiments.

It is also necessary to quantify the differences between sequence features between assays. Like in **Fig 6.3**, the highest signals of the FC assays were trained against each other. For both promoters and enhancers (**Fig 6.4 A and B** respectively), the differences in sequence features seem to be on the larger side, especially for enhancers. It may be that case that the plasmid design of MPRA and STARR-Seq bias these assays toward detection of specific TFs are more likely to generate a large RNA readout. In addition, the selection process of the regions may have biased the data towards sequences with certain features.

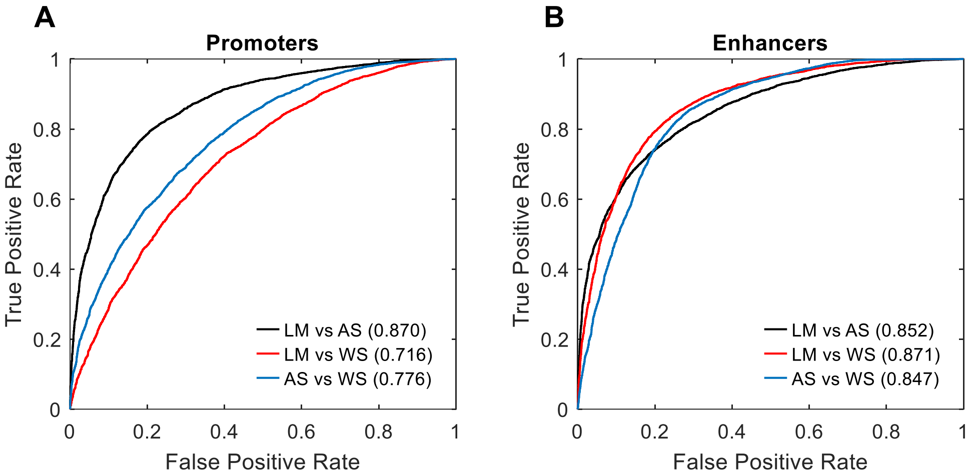


Figure 6.4 Distinguishing high signal regions between assays

The performance of gkm-SVM models when training high signal regions against each other. The same sequences and cross-validation splits from Fig 6.1 were used. LM = lentiMPRA, AS = ATAC-STARR-Seq, WG = Whole Genome-STARR-Seq. Quantities in parentheses are the AUROC of the assay.

6.2.2 Predicting the readouts of the assays with regression

The signals generated by the assays distinguish sequences with regulatory activity from genomic background, even though their predictive features vary. However, when extracting TF motifs, some of them may not be truly generating the signal, but instead, coexist within sequences flanking the real causal motifs. This is likely since many of the regions tested are in open chromatin, which contain a wide range of motifs. Thus, to determine the motifs that are generating the signal, I trained support vector regression models to predict the log fold changes of the MPRA and STARR-Seq assays. The CRISPR growth screen did not generate enough data with high or low signal and was thus excluded from this analysis.

I used gkm-SVR to train regression models on tiling MPRA, lentiMPRA, ATAC-STARR, and WG-STARR. The training sets included sequences that spanned the entire range of log fold changes. There was no separation of sequences into promoter and enhancer datasets. As shown in Chapter 4, gkm-SVR is similar to gkm-SVM. It uses the same feature set of gapped kmers and outputs a vector of gapped kmer weights. All gkm-SVR models were able to predict the log-fold changes with fairly strong accuracy (r = 0.6 - 0.7 **Fig 6.5**). This includes tiling MPRA, which performed poorly in the previous task. Thus, the poor performance of tiling MPRA is likely due to the lack of regulatory elements in the positive set. The correlation of replicates for MPRA and STARR-Seq is very high (>0.90)[40,88,91]. Thus, there is potential to improve the predictive power of sequence models. I used the linear kernel for gkm-SVR, which does not learn positional preferences of TFBSs nor multiplicative effects of combinations of motifs that may improve the performance of sequence-based models on this task[92,93]. Nevertheless, additively incorporating TF binding motifs explains 40-50% of the variance.

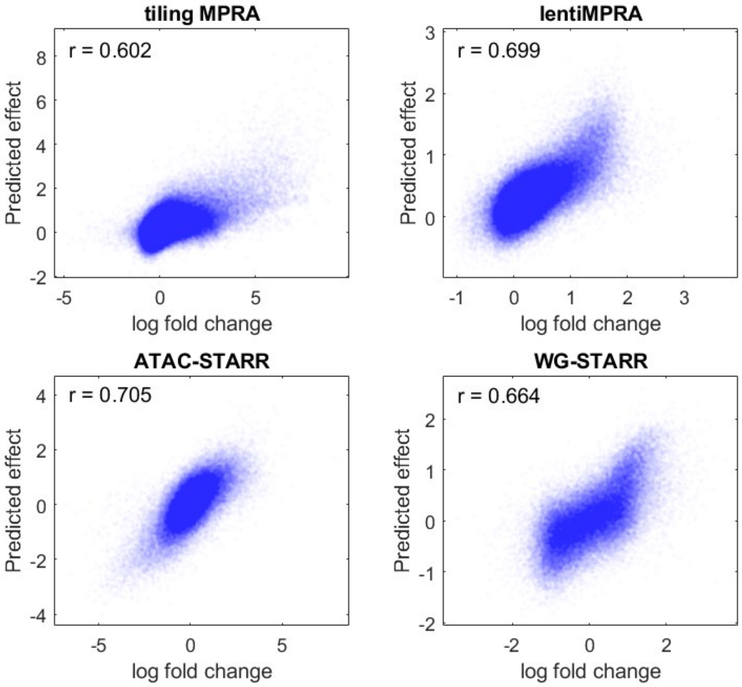


Figure 6.5 Performance of gkm-SVR models when predicting log fold changes

Support vector regression with gapped k-mer features to predict the signal of functional characterization assays. The performance metric is the Pearson correlation of the predicted effect from the model with the measured log fold change from the experiment. All predictions from the five cross-validation splits are shown.

To determine the TF binding motifs that are predictive of log fold changes, I ran gkmPWMlasso on the models to learn the motifs that were predictive of both high and low signal (**Fig 6.6**). Unsurprisingly, most of the motifs in **Fig 6.2** are also learned in this task. There are some notable exceptions. The most glaring one is that GATA, which is by far the most predictive motif of K562 ATAC-Seq, is no longer predictive for STARR-Seq assays. Additionally, there are a couple other weak motifs from **Fig 6.2** that are now missing (ZNF143, ZBTB33). For the MPRA assays, the motifs associated with promoters are more predictive of activity than the distal enhancer motifs. This is another perplexing result since the plasmid design that was used for MPRA was the enhancer construct. It is difficult to interpret the lack of agreement of motifs predictive of positive log fold change between assays. More systematic analysis should be done to compare their readouts.

A very surprising result is the detection of known repressive TFs as motifs predictive of negative log fold change. These include HIC[94], SNAIL[95], NFI[96], and GFI[97]. There is moderate agreement between repressive TF detected across the assays. gkmPWMlasso identified NFY, a strong promoter motif, as having positive regulatory activity in tiling MPRA and lentiMPRA, but repressive activity in ATAC-STARR. NFY has been shown to have both activating and repressive activity[98], so it may be the case that the design of the plasmid influences the function of a TF.

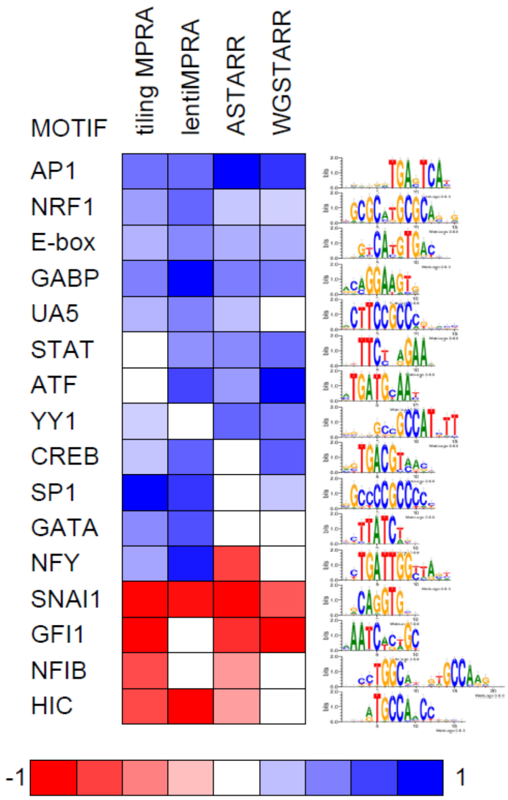


Figure 6.6 Extracted motifs from gkm-SVR models of functional characterization assays

Motifs that were predictive of high signal (blue) and low signal (red) in functional characterization assays. The heatmap contains z-scores that were normalized to 1 or -1. In contrast to models trained against genomic background, the motifs with negative weight can be meaningful features.

6.3.3 Controlling for GC in ATAC-STARR

When analyzing ATAC-STARR models, I noticed that there was a preference for AT rich sequences in the positive set. This is concerning since gkm-SVR does not use a GC and repeat matched negative to maximize the influence of TFBSs on the predictive power of the models. To ensure that the models in **Fig 6.5** are not only learning GC content, I took the top 5000 and 5000 bottom regions in terms of log fold change and used GC content as a predictor (**Fig 6.7**). For tiling MPRA, lentiMPRA, and WG-STARR, the predictive power of GC content is low (AUC < 0.7). However, GC content was a great predictor for both ATAC-STARR promoters and enhancers (AUC = 0.903, 0.877 respectively). For promoters, many of the sequences with low log fold changes have GC contents much higher than what is typically found in promoters (average GC in accessible promoters is approximately 0.65). Therefore, it is highly likely that much of the predictive power of gkm-SVR models comes from learned GC content for ATAC-STARR.

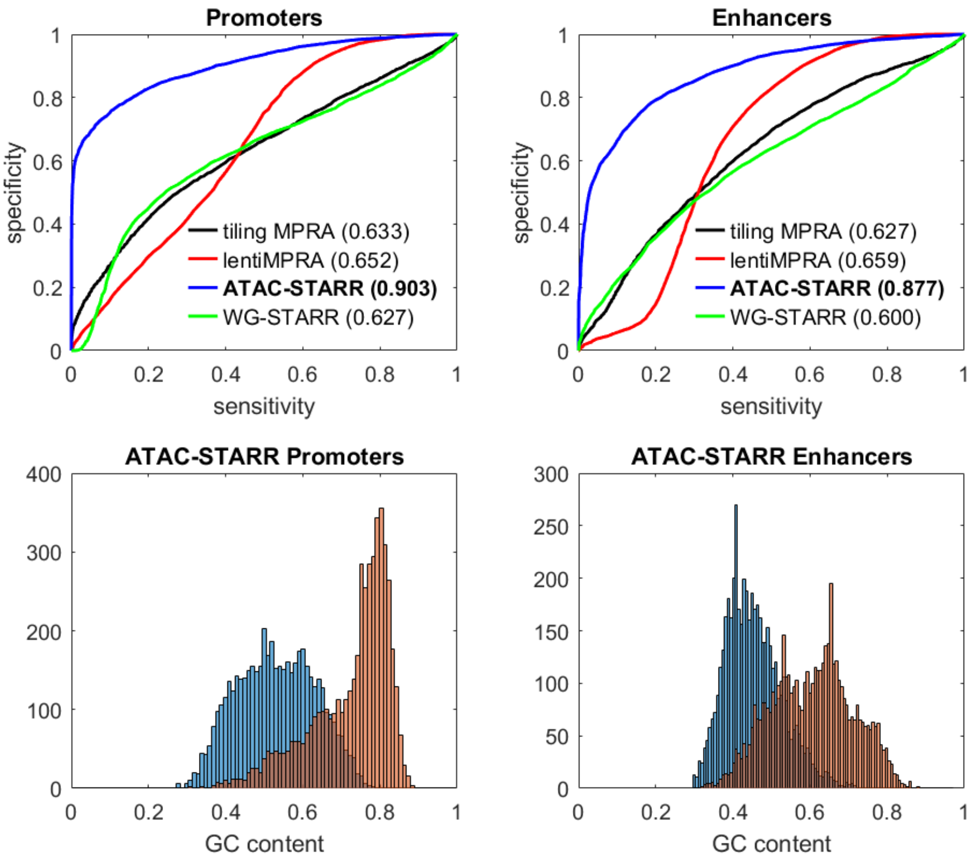


Figure 6.7 The predictive power of GC content for functional characterization assays

Top row: The AUROCs of using GC content to predict the 5000 regions with the highest signals against the 5000 regions with the lowest signal. For most assays, GC content was a weak predictor of signal. However, for ATAC-STARR, there is a clear GC bias in high and low signal. Bottom row: Histograms of the distributions of GC content for high signal regions (blue) versus low signal regions (orange).

The analysis for ATAC-STAR needed to be redone since the TFBSs that were identified as predictive in **Fig 6.5** may covary with GC. For example, NFY, which was identified as a repressor, covaries with promoters, which have high GC. To control for the influence of GC, I binned ATAC-STARR sequences by their GC content in steps of 0.05. Since there were few sequences with GC below 0.45 and above 0.70, I combined the bins outside this range into 0-0.45 and 0.70-1.0 bins. gkm-SVR models were trained on these bins, and the analysis in **Fig 6.6** and **Fig 6.7** were repeated (**Fig 6.8)**. The correlations of the predicted and actual log fold changes remained high (r = 0.6, **Fig 6.8A** blue line) across all GC bins but did not quite reach the correlation when training on the full range of GC (r = 0.7). However, the predictive power of GC content was low for most bins, except for 0.70-1.0, where the performances of gkm-SVR and GC content are approximately equal. Thus, for motif extraction outside of 0.70-1.0 can be considered reliable.

The predictive motifs across GC bins are shown in **Fig 6.8BC**. There were a few motifs that were predictive of positive signal across all bins (AP1, YY1, and GABP). The other motifs that were predictive of positive signal depended on the GC content. Motifs that were associated with enhancers (orange) were more predictive in lower GC contents, and motifs associated with promoters (blue) were predictive in higher GC contents. Interestingly, the repressive motifs (red) seem to exhibit this behavior as well. SNAIL and NFY were predictive of low log fold change across all bins. GFI is only predictive in models trained on lower GC contents. Although GFI has been shown to bind to the BCL-x and GFI-1B promoters in K562[97,99], it also binds to enhancers in hemopoietic cell lines[100]. A new repressor, REST, was learned in this analysis in higher GC content. This repressor is active in many cell-types[101,102] and binds to both enhancers and promoters. Its prevalence in high GC models is likely due to its GC rich motif, which is more likely to appear by chance in high GC sequences.

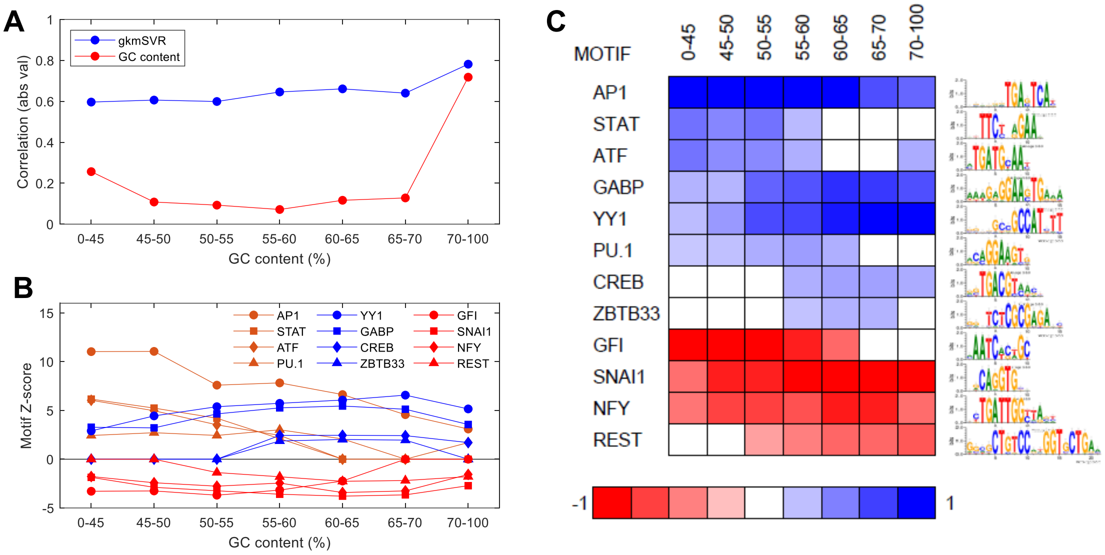


Figure 6.8 Controlling for GC content still yields quality models

A) The performances of gkm-SVR models trained on ATAC-STARR regions by separating sequences based on their GC content is comparable to the correlation in Fig 6.5. GC content is only a reliable predictor of assay signal at very high GC contents. B) Extracted gkmPWMlasso motifs of the gkm-SVR models. C) A heatmap version of B, with normalized z-scores to -1 or 1.

By controlling for GC, we can be confident that ATAC-STARR signal is generated by TF binding motifs. The specific reason the ATAC-STARR signal is low for extremely high GC rich sequences is still in question. This could be a result of the experimental protocol. It is unlikely that the Tn5 or the STARR-Seq components of the assay are biasing the signal away from high GC content because that bias would be present in ATAC-Seq or WG-STARR.

6.3 Discussion

Although I was able to show that sequence features are highly predictive of the signal readouts from functional characterization assays, it is difficult to interpret these results. The strongest sequence features of MPRA are motifs associated with promoters, even though the plasmid was constructed to measure enhancer activity. Even when using only distal genomic elements as inserts, sequences that contain promoter motifs by chance will generate the largest signals for MPRA. A potential mechanism for this phenomenon is that the insert is playing the role of both an enhancer and a secondary upstream promoter. Since the plasmid contains a minimal (weak) promoter, a strong upstream promoter can greatly increase the RNA readout. It is not obvious how to control for this confounding effect in the assay. Saturation mutagenesis is able to reduce the effect of promoter bias by measuring the activity of variant with respect to the reference sequence. However, how to develop a reference for sequences sampled across the genome is not clear.

ATAC-STARR and WG-STARR both bias towards AP1 and YY1 motifs. Identifying a potential mechanism is difficult because many TFs and combinations of TFs bind to the AP1 motif[103,104]. AP1-binding TFs are widely expressed and active across cell-types (**Fig 5.4**), so AP1 may be a commonly used site for enhancer activity. On the other hand, YY1 has been proposed to facilitate enhancer-promoter loop formations[105,106]. It is possible that the STARR-Seq plasmid design allows YY1 to act more efficiently than in MPRA plasmid designs. However, the YY1 motif is associated with promoters, not enhancers (**Table 5.1**), so this proposed mechanism is still in question.

CRISPR growth screens yielded a low number of sequences with high or low signal. Most sequences that influenced proliferation upon disrupt were promoters, and not enhancers. This is likely due to the fact that the change in phenotype upon perturbation of an enhancer can be transient[23,107]. However, one of the predictive distal motifs is the TAD associated TF, CTCF, which was not predictive of any of the other experiments. In contrast to YY1, CTCF requires a pair of CTCFs to form a loop, which may explain why it was not predictive in STARR-Seq. CRISPRi experiments in general have the potential to measure the effects of loop disruption on gene regulation.

A broad conclusion of this analysis would caution against using individual functional characterization assays to define regulatory elements. The differences between their sequence features have not been investigated thoroughly enough. Plasmid designs may influence what biological mechanisms are reliably detected. Even for CRISPR growth screen experiments, sequences that yield little changes in proliferation may be false positives due to the transient effects of enhancer perturbation. A more rigorous comparison of functional characterization assays is called for. The inserts selected for each assay were not held constant, which made the comparisons done in this chapter difficult. A proper comparison would include using the same inserts/targets across assays. In addition, a larger set of negative controls should be used. The signal of the assays could be then used as a score, and evaluations used for sequence-based models could be applied in a more robust and rigorous analysis of the various assays.

Chapter 7

Mapping gkmPWM motifs to *cis*-regulatory elements at nucleotide resolution

In previous chapters, I discussed how to extract motifs from sequence-based models and performed analysis of mapping data and functional characterization assays using the motifs themselves. While it is important to characterize classes of regulatory elements by the TFs that bind them, the analyses presented are limited to identifying the general properties of a set of sequences. Ultimately, we care about specific biological mechanisms and pathologies that can occur when regulatory elements are disrupted. In regulatory genomics, we are primarily concerned with non-coding variants that increase or decrease the expression of genes and perturb or prevent normal function of a cellular network. To model these systems properly, we must create gene regulatory networks (GRNs), which model both the regulatory elements and the concentrations of proteins whose genes are involved in the cellular system of interest. The process of constructing GRNs involves identifying the TFs involved in the system, which we can do using the mathematical tools developed in Chapter 4. However, different combinations of these TFs bind to each individual enhancer. Thus, the next major step in constructing these networks is to map the motifs learned by gkmPWM to regulatory elements at nucleotide resolution.

In this chapter, I will describe a method to map these motifs using dynamic programming. This model will incorporate both gkmPWM motifs and gkm-SVM kmer weights. gkmPWM motifs contain information about the individual TF binding sites, and gkm-SVM provides information about the predictive power of the kmers. I will show that the predicted locations of TFBSs from this map align with saturation mutagenesis data. In addition, I will show that for a given TF, regulatory elements with a predicted binding site have higher ChIP-Seq signals compared to those that lack that specific motif. Lastly, I will present the list of TFBS predictions over all promoters and distal enhancers for a wide range of cell-types.

7.1 An overview of assigning motifs to predictive kmers

First, I will discuss the current approach to mapping TBBSs to sequences. The most customary practice to scan regulatory elements for binding sites is to take a set of PWMs and score all kmers in the sequences. The probability of generating the sequence (see 4.1.2) or some variation of this quantity is typically used as a metric[48,69,108]. To make discrete predictions of binding sites, a score cutoff is arbitrarily chosen. Additionally, a large set of PWMs is used to scan the sequences, including those PWMs whose TFs are not active in the cell type. We can avoid the latter problem by using gkm-SVM and gkmPWM in tandem. Extracting motifs from a gkm-SVM model trained on a set of putative regulatory elements can reduce the number of PWMs to those with predictive of regulatory activity. To avoid the motif score cutoff problem, I associated gkm-SVM kmer weights with gkmPWM motifs. When plotting the gkm-SVM weight track along a sequence, there are stretches of kmers that have high weight. By comparing these kmers to saturation mutagenesis on the SORT1 enhancer, we can infer that these are likely TFBSs (**Fig 7.1**). However, there is a considerable amount of noise in these data, so associating every kmer with positive weight with a motif will yield many false positives. To overcome this problem, I designed an algorithm (mapTF.m in https://github.com/shigakiD/gkmPWM) to incorporate consecutive kmers into a TFBS prediction.

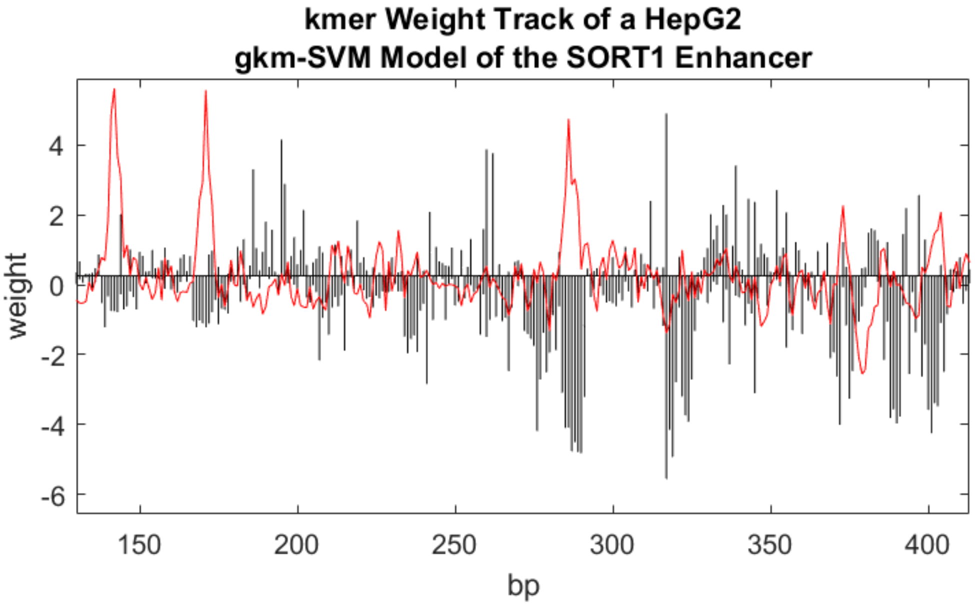


Figure 7.1 gkm-SVM kmer weights align with saturation mutagenesis signals of TFBSs

11-mers along the SORT1 enhancer (chr1:109274913-109275196) were scored by a gkm-SVM model trained on HepG2 distal enhancers (red). The variant effects of saturation mutagenesis are in black. Stretches of nucleotides with negative variant effects are likely to TFBSs. For many of these stretches, the kmer weights are consistently high.

7.1.1 Structure of the mapTF

First, I will give an overview of the algorithm of mapTF. For a sequence of length L, there are $L-l+1$ total kmers. We will denote the kmers as $\{x_1, x_2, \dots, x_{L-l}, x_{L-l+1}\}$, where x_1 is the leftmost kmer, x_2 is the next kmer to the right, and so on. Each kmer x_i has a weight associated with it $w(x_i)$, which is the kmer weight waveform (ex. **Fig 7.1**). We will propagate over the x_i and utilize $w(x_i)$ to map PWMs. We will calculate the probability of kmers being in a general TF state and a background DNA state. The general TF state is comprised of many other states, which can be assigned into M PWM/TFBS states. We can break each PWM θ_m in $L_{\theta_m} - l + 1$ subPWMs $\phi_{m,n}$. Each of these subPWMs is also a state. Thus, the total number of states is

$$1 + \sum_{i=1}^{M} \left(L_{\theta_m} - l + 1 \right)$$

Within the general TF state, $\phi_{m,n}$ emits a probability $f_{m,n}(x_i)$ that gives a probability that kmer x_i fits that subPWM (7.1.3).

Permissible transitions (i.e. the probability of entering a state is greater than zero) are defined as follows. The background state can enter any of the first subPWM $\phi_{m,1}$ of any PWM or remain in the background. The last subPWM, $\phi_{m,L_{\theta_m}-l+1}$, of any PWM can also enter the first subPWM or any PWM or the background. For all other subPWMs, $\phi_{m,n}$ can only enter $\phi_{m,n+1}$, i.e. $P(\phi_{m,n} \rightarrow \phi_{m,n+1}) = 1$ and zero otherwise.

7.1.2 Converting gkm-SVM kmer weights to probabilities

The first step is to make a gkm-SVM probability profile using the model. The distribution of the gapped kmer weights is normal (**Fig 3.1B**). Likewise, the kmer weight distribution (the sum of the weights of the gapped kmers that are contained in the full kmer) is normal. Each kmer weight was transformed into a probability by fitting a gaussian model to the full kmer distribution to the kmer weights and taking the cumulative distribution function as the probability of a kmer contributing to regulatory activity. More concretely,

$$p(x_i) = \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{x} e^{-(w(x_i)-\mu)^2/2\sigma^2} dx$$

The mean is typically close to zero. $p(x_i)$ is the probability of being in the general TF state, indicating regulatory activity, and $1 - p(x_i)$ is the probability of being in the genomic background state.

7.1.3 Assigning kmer probabilities with PWMs

To model the emission probabilities of each subPWM, we created a simple function to determine how well a kmer matched a particular subPWM Define $t_{m,n}(x_i)$ as the sum of the gapped kmer probabilities that are contained in kmer x_i for subPWM $\phi_{m,n}$.

$$t_{m,n}(x_i) = \sum_{g \subset x_i} E\left[g|\phi_{m,n}\right]$$

Here, $E\left[g|\phi_{m,n}\right]$ is the expected gapped kmer count of g using $\phi_{m,n}$ (4.1.2).

We define the probability that kmer x_i fits ϕ_{mn} as

$$f_{m,n}(x_i) = \frac{t_{m,n}(x_i) - \inf t_{m,n}(x_j)}{\sup t_{m,n}(x_j) - \inf t_{m,n}(x_j)}$$

Where $\sup t_{m,n}(x_j)$ is the maximum value of $t_{m,n}(x_j)$ over all possible kmers and $\inf t_{m,n}(x_j)$ is the minimum value of $t_{m,n}(x_j)$ over all possible kmers. This approximates the cumulative distribution probability, since calculating the sum of gapped kmer probabilities for every kmer over all sub-PWMs would be too computationally burdensome.

7.1.4 Using dynamic programming to map TFBSs

Starting with the first kmer x_i, we use dynamic programming to find the path between the states that maximize the probability of generating the sequence. The transition probability into a background state or remaining in a background state is always $1 - p(x_i)$. The transition probability of entering the general TF state is $p(x_i)$. The first instance of entering any TF state must be one of the $\phi_{m,1}$. This state emits the probability of fit $f_{m,1}(x_i)$. The process now must continue to enter successive sub-PWM states belong to the same PWM θ_m. Until the process reaches $\phi_{m,L_{\theta_m}-l+1}$, we must multiply by probability of being in a TF state $p(x_i)$, and the probability of fit $f_{m,n}(x_i)$. Thus, the probability of being in θ_m is

$$\prod_{j=1}^{L_{\theta_m}-l+1} p(x_{i+j-1}) f_{m,j}(x_{i+j-1})$$

There are many different possible paths that propagate through the background and TF states in different orders and combinations. The probability of the path will be a product of a combination of $1 - p(x_i)$ and $p(x_i)f_{m,n}(x_i)$ over $i \in \{1,2,\ldots,L-l+1\}$. For example, for the sake of simplicity, suppose all PWMs had 5 sub-PWMs. The path that goes through the background for 5 kmers, then into PWM θ_1, then into the background for 3 kmers and finally PWM θ_2 is

$$\left(\prod_{i=1}^{5}\left(1-p(x_i)\right)\right)\left(\prod_{j=1}^{5} p(x_{6+j-1}) f_{1,j}(x_{6+j-1})\right)\left(\prod_{i=11}^{13}\left(1-p(x_i)\right)\right)\left(\prod_{j=1}^{5} p(x_{14+j-1}) f_{2,j}(x_{14+j-1})\right)$$

We pick the optimal path as the path that has the highest probability. To calculate the maximum probability, we can use dynamic programming in the same manner as Markov Chains. To summarize, one calculates maximal probability of ending up in all states up to kmer x_i by using the information from calculating the maximal probability of ending up in all states up to kmer x_{i-1}. For each state, the previous state in kmer x_{i-1} that maximizes the probability up to kmer x_i is also stored. Once the end is reached, the state with the maximal probability is selected, and the optimal path is determined by backpropagating through the previous states that generated the largest probability of the next state.

7.1.5 Estimating variant effects from PWMs

I made a further addition to this algorithm to ensure that the PWMs selected are good matches. For all nucleotides in a PWM state, I calculate variant effects using the PWM.

If they poorly match deltaSVM scores, I remove the TFBS call. For a given nucleotide N_i in a PWM state, there are l full kmers that contain it. These kmers are $\{x_{i-l+1}, x_{i-l+1}, \dots, x_{i-1}, x_i\}$. These kmers align with l sub-PWMs $\{\phi_{m,n-l+1}, \phi_{m,n-l+2}, \dots, \phi_{m,n-1}, \phi_{m,n}\}$. We define the "reference" score as

$$\sum_{j=-l+1}^{0} \sum_{g \subset x_{i+j}} E\left[g|\phi_{m,n+j}\right]$$

If we induce a variant $N_i \rightarrow V_i$, the kmers that contain V_i change, which we denote as $\{y_{i-l+1}, y_{i-l+1}, \dots, y_{i-1}, y_i\}$. The sub-PWMs remain the same. The "alternate" score is

$$\sum_{j=-l+1}^{0} \sum_{g \subset y_{i+j}} E\left[g|\phi_{m,n+j}\right]$$

The variant score is therefore

$$\sum_{j=-l+1}^{0} \sum_{g \subset y_{i+j}} E\left[g|\phi_{m,n+j}\right] - \sum_{j=-l+1}^{0} \sum_{g \subset x_{i+j}} E\left[g|\phi_{m,n+j}\right]$$

I calculate this value for all variants over all nucleotides in the PWM ($3L_{\theta_m}$ total variants). I then take the cosine metric with the deltaSVM score with these variants. If that value is not at least 0.60, I remove the call.

7.2 Methods

To evaluate the quality of the TFBS calls, I compared the results of mapTF to saturation mutagenesis assays and ChIP-Seq. There is not much data that maps the specific binding sites of TFs at nucleotide resolution. Although DNase/ATAC footprints are available, statistically significant detection of footprints requires very deep sequencing. Their reliability is questionable (**Fig 7.2** and **7.3**), especially in enhancers, which have lower DNase/ATAC signal compared to promoters. Since they do not agree very well with saturation mutagenesis, I decided not to compare motifs to them.

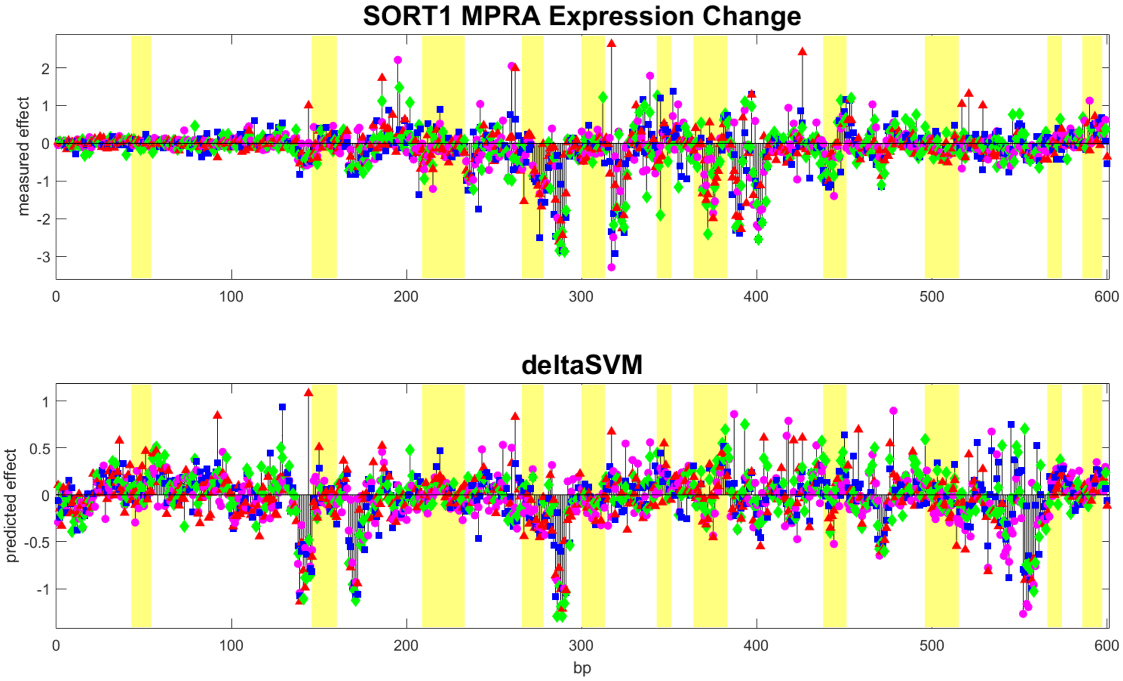


Figure 7.2 DNase-Seq footprints in the SORT1 enhancer

Footprints from HepG2 DNase-Seq (yellow) do not align with saturation mutagenesis MPRA nor delta-SVM. The correlation between MPRA and delta-SVM is 0.49. The footprints often miss regions with high measured or predicted variant effects.

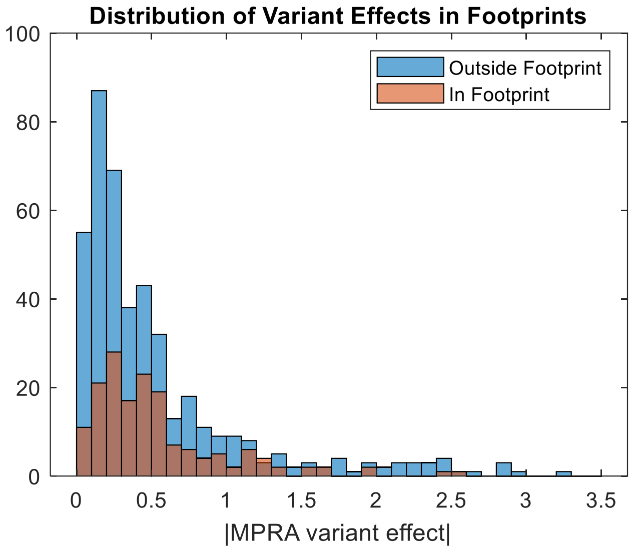


Figure 7.3 The distribution of variant effects in and out of footprints in the SORT1 enhancer

A histogram of the variants in the top row of Fig 7.2. Although a higher fraction of variants within a footprint has high effect sizes, more variants outside footprints have high effect sizes.

7.2.1 Mapping gkmPWM motifs to the CAGI5 Regulation Saturation regions

Comparisons to saturation mutagenesis are mostly visual, and not very quantitative. gkmPWM motifs are displayed above MPRA and delta-SVM tracks. The gkm-SVM models were selected by matching the cell-type to the cell line that the MPRA experiment used. If an exact match was not found, a similar cell-type was used. A variety of gkm-SVM models trained on various experiments and cell-types were used to extract and map motifs (**Table 7.1**).

Table 7.1 Experiments and cell-types of saturation mutagenesis and gkm-SVM

Saturation Mutagenesis		**gkm-SVM model**	
Region	**Cell-type**	**Experiment(s)**	**Cell Type(s)**
SORT1	HepG2	DNase-Seq	HepG2
ZFAND3	MIN6	ATAC-Seq	MIN6
IRF4	SK-MEL-28	DNase-Seq	SK-MEL-5
GP1BB	K562	Lentiviral MPRA	K562
HBG1	K562	Lentiviral MPRA	K562
LDLR	HepG2	ChIP-Seq	ZNF263 HepG2 SREBF1 MCF-7 ZNF652 HepG2

7.2.2 Predicting ChIP-Seq signals in GM12878 distal enhancer DHS peaks

I ran gkm-SVM on GM12878 distal enhancer DHS peaks and extracted motifs using gkmPWM. I mapped those motifs using mapTF to the top 10,000 GM12878 distal enhancer DHS peaks. For six of the TFs (RUNX, EBF, OCT1, AP1, NFKB, and GABPA), I separated the 10k peaks into two sets: peaks with the motif (motif+) and peaks without the motif (motif-). I also gathered ChIP-Seq datasets for each of those TFs.

I performed two sets of analysis on these sets. First, I used the motif of a particular TF to classify the 10k DHS peaks as ChIP-Seq positive or negative for the given TF. I used the number of motifs in the peak as a feature. Due to the class imbalance, I used the AUPRC as a performance metric. The motif counts of the other TFs were used as negative controls. Second, for each TF, I averaged the ChIP-Seq reads over the respective motif+ and motif- sets and calculated the enrichment scores. Again, the motif+ and motif- sets of the other TFs were used as negative controls.

7.3 Results

Mapping cell-type specific motifs to enhancers

I used mapTF to identify TFBSs in the SORT1 (**Fig 7.4**), ZFAND3 (**Fig 7.5**), and IRF4 (**Fig 7.6**) enhancers. I was able to match the cell-type of the dataset with the cell-type of the MPRA experiment for SORT1 and ZFAND3, and closely match the cell-type with IRF4 (SK-MEL-5 instead of SK-MEL-28). The quality of these predictions is mostly assessed by eye. Motifs closely align with the specific sequence (row 2), and in general, areas with large MPRA variant effects also have a motif call. In cases where there is not the delta-SVM scores are low, indicating that either gkm-SVM did learn the motif, or the motif is very weak. In the latter case, replacing a nucleotide with a variant with a large positive delta-SVM score can help mapTF identify the correct motif (**Fig 7.7**). The

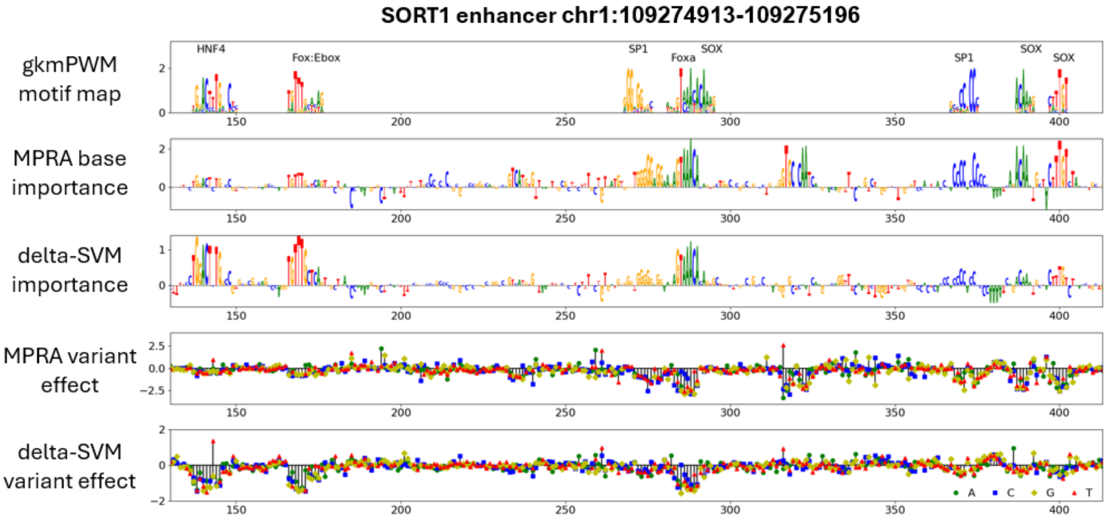


Figure 7.4 mapTF TFBS predictions of the SORT1 enhancer

Predicted TFBSs using mapTF and gkmPWM motifs from a HepG2 DNase-Seq distal enhancer gkm-SVM model. The 1st row contains the TFBS predictions from mapTF. The 2nd and 3rd rows are the negation of the average of the variant effects from the 4th and 5th rows respectively. Pearson correlation of the 4th and 5th row is 0.49. The flanks are truncated increase resolution of motifs and individual nucleotides.

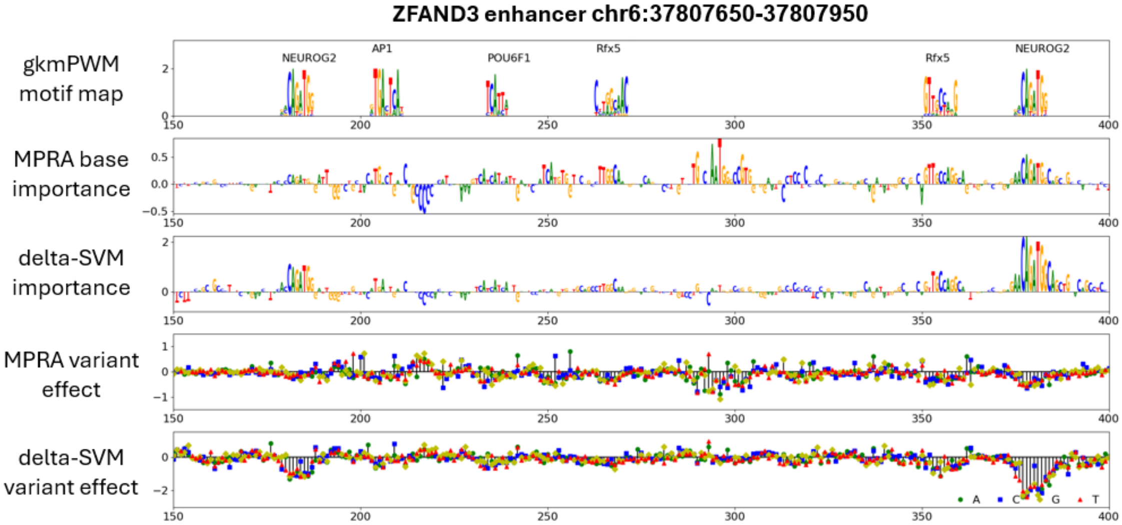


Figure 7.5 mapTF TFBS predictions of the ZFAND3 enhancer

Predicted TFBSs using mapTF and gkmPWM motifs from a MIN6 ATAC-Seq distal enhancer gkm-SVM model. The 1st row contains the TFBS predictions from mapTF. The 2nd and 3rd rows are the negation of the average of the variant effects from the 4th and 5th rows respectively. Pearson correlation of the 4th and 5th row is 0.43. The flanks are truncated increase resolution of motifs and individual nucleotides.

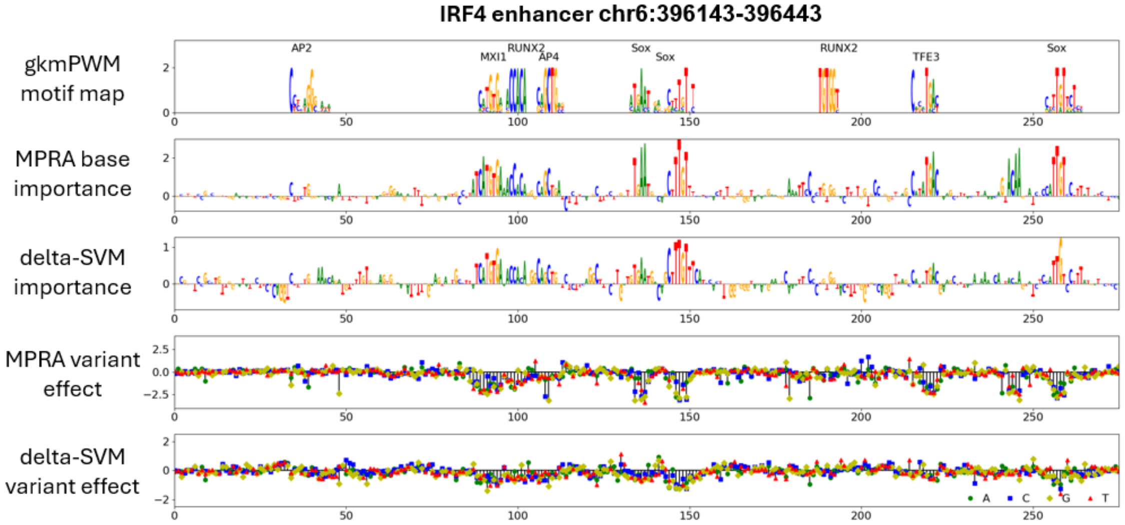


Figure 7.6 mapTF TFBS predictions of the IRF4 enhancer

Predicted TFBSs using mapTF and gkmPWM motifs from a MIN6 ATAC-Seq distal enhancer gkm-SVM model. The 1st row contains the TFBS predictions from mapTF. The second and third rows are the negation of the average of the variant effects from the 4th and 5th rows respectively. Pearson correlation of the 4th and 5th row is 0.57. The right flank is truncated increase resolution of motifs and individual nucleotides.

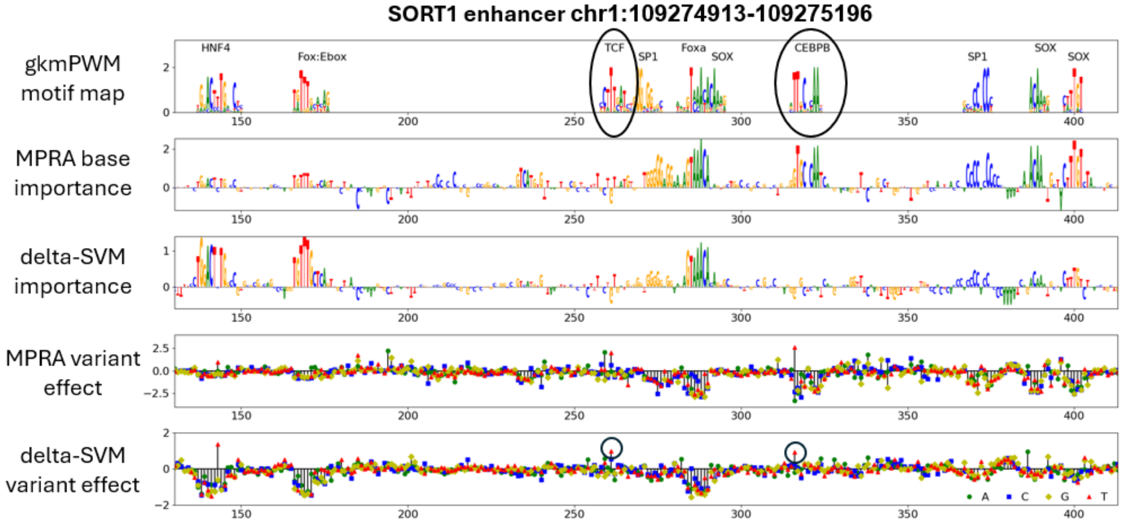


Figure 7.7 Inserting variants with high delta-SVM helps identify weak binding sites

mapTF misses the two circled motifs in the first row using the SORT1 enhancer sequence from the reference genome. However, inducing a variant (circled in the 5th row) can aid mapTF in predictions.

Mapping motifs predictive of functional characterization assays

mapTF is not limited to gkm-SVM models but can be used with gkm-SVR (regression) models. The experiments of the promoters of GP1BB and HBG1 enhancers were done in the K562 cell line. In Chapter 6, I trained lentiviral MPRA gkm-SVR models on that cell-type. Since, gkmPWM and deltaSVM can also work on gkm-SVR models, I assessed mapTF's ability to map K562 MPRA features to these two promoters. Typically, I would use a K562 DNase/ATAC-Seq promoter dataset for this task. When using a gkm-SVR model trained on MPRA instead of DHS peaks, there is an improvement in correlation between the experimental variant effects and the delta-SVM scores. It was small for GP1BB (0.36 from 0.32), but huge for HBG1 (0.52 from 0.24). Mapping the motifs to GP1BB (**Fig 7.8**) and HBG1 (**Fig 7.9**) shows many common promoter TFs (SP1, ETS, NFY) and GATA, the strongest motif in K562. Like for the distal enhancers, there is large agreement between the predicted motifs and the actual sequence.

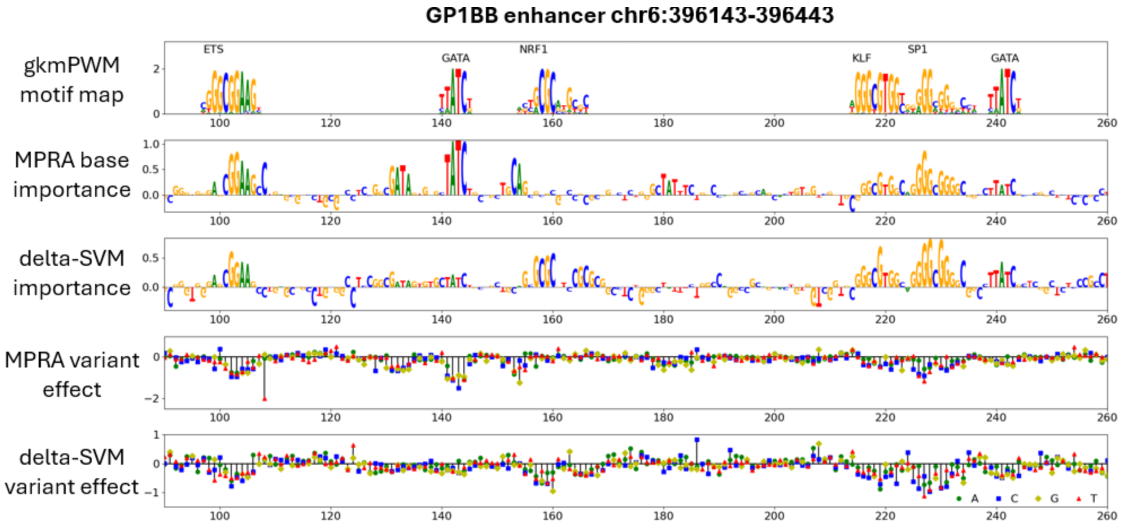


Figure 7.8 mapTF predictions of the GP1BB promoter

Predicted TFBSs using mapTF and gkmPWM motifs from a K562 lentiviral MPRA gkm-SVR model. The 1st row contains the TFBS predictions from mapTF. The 2nd and 3rd rows are the negation of the average of the variant effects from the 4th and 5th rows respectively. Pearson correlation of the 4th and 5th row is 0.36. The flanks are truncated increase resolution of motifs and individual nucleotides.

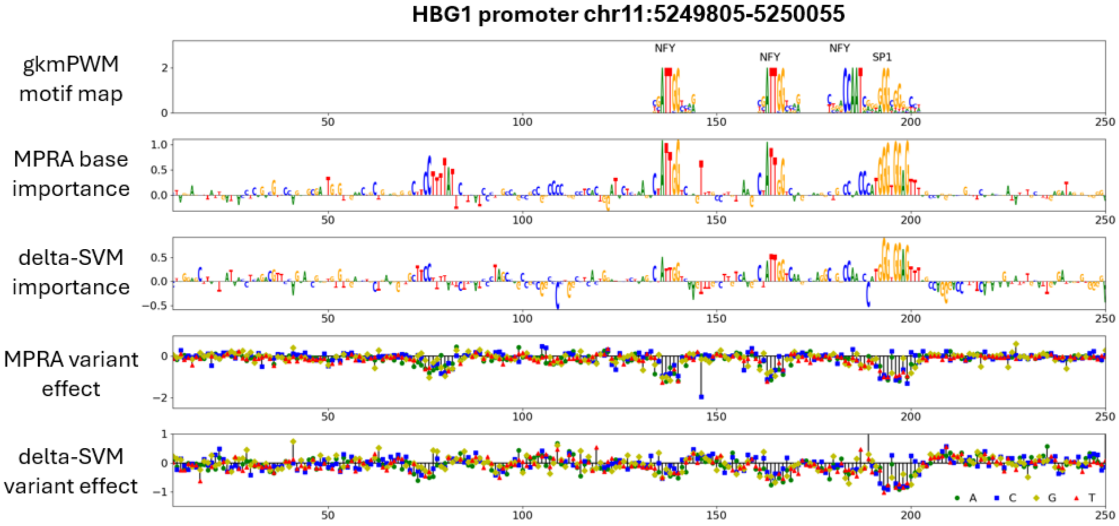


Figure 7.9 mapTF predictions for the HBG1 promoter

Predicted TFBSs using mapTF and gkmPWM motifs from a K562 lentiviral MPRA gkm-SVR model. The 1st row contains the TFBS predictions from mapTF. The 2nd and 3rd rows are the negation of the average of the variant effects from the 4th and 5th rows respectively. Pearson correlation of the 4th and 5th row is 0.52.

Combining datasets to identify all motifs

The last task I will show is how combining datasets from 3.3.3 improves the prediction of saturation mutagenesis variant effects. When using a HepG2 promoter DNase-Seq gkm-SVM model, the correlation between the MPRA and delta-SVM scores is 0.39. When using LASSO regression to combine datasets, the three top and unique datasets that had positive regression weights were ChIP-Seq assays targeting ZNF263, SREBF1, and ZNF652. When taking the least squares estimate of these three datasets with the MPRA variant effects, the correlation jumps to 0.65. There are two hypotheses for this improvement. The first is that combining datasets improves the quality of TFBS predictions that are shared between datasets. The second is that they learn different motifs, and their combination fills each other's gaps. In **Fig 7.10**, the second hypothesis definitely hold, while the first may potentially apply at the ZNF263/SP1 sites.

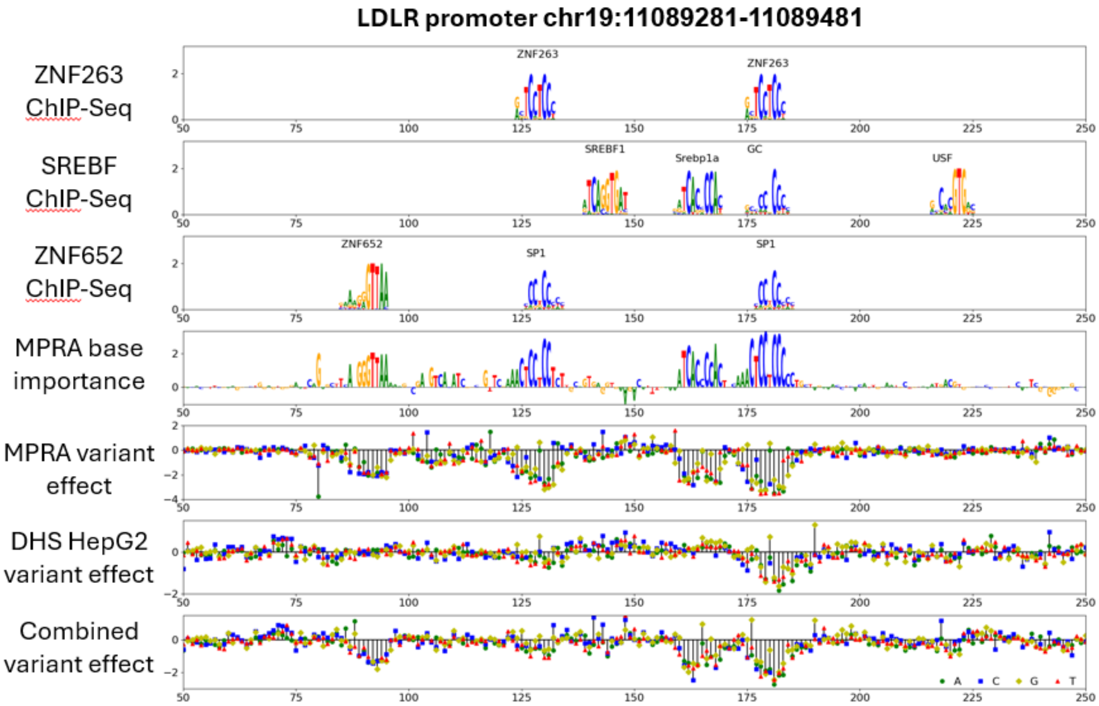


Figure 7.10 Datasets learn unique motifs to improve prediction accuracy of the LDLR promoter

Predicted TFBSs using mapTF and gkmPWM motifs from three ChIP-Seq models. Rows 1-3 contains the TFBS predictions from mapTF. The 4th row is the negation of the average of the variant effects from the 5th row. The 6th row shows variant effect predictions from a HepG2 promoter DHS model. The 7th row shows the least squares estimate of the variant effect predictions from the ChIP-Seq datasets. Pearson correlation of the 5th and 6th row is 0.39. Pearson correlation of the 5th and 7th row is 0.65.

gkmPWM motifs predict ChIP-Seq signal

Because MPRA does not directly identify the perturbed TF, I next evaluated whether mapTF motifs are directly predictive of TF ChIP-seq data in GM12878, where we have TF binding data for 6 factors identified by gkm-SVM (and corresponding motifs): RUNX1, EBF, OCT1, JUN (AP1), RELA (NFkB), and GABPA. I trained a GM12878 distal enhancer model and extracted motifs. We then mapped those motifs to the top 10,000 distal DHS peaks using mapTF. To create a prediction task, we classified each DHS as positive for TF binding if it overlapped a TF ChIP-seq peak for that factor, and negative if it did not. Then we generated a predicted TF binding score using the sum of mapTF occurrences within each peak. We calculated the AUPRC for the mapTF score's ability to predict TF ChIP-seq positive DHS peaks. The difference between the measured AUPRC and random assortment is shown in **Fig 7.11**, and all diagonal terms show predictive power, while off diagonal terms (negative controls) do not, as motif predictions for one factor are not predictive of another factors ChIP-seq binding.

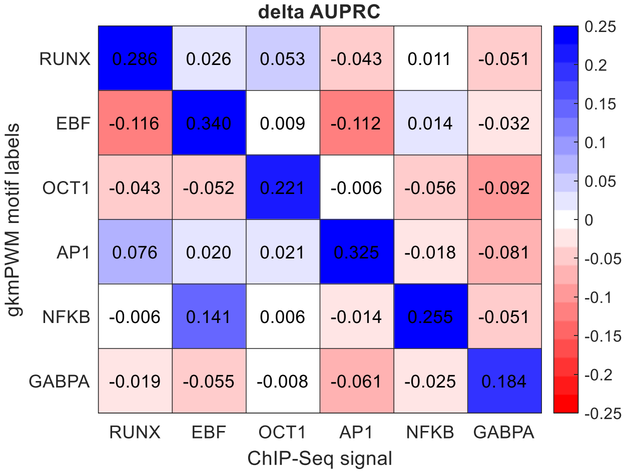


Figure 7.11 AUPRCs of predicting ChIP-Seq peaks in DHS peaks using motif counts

gkmPWM motif counts (of a particular TF) were used to classify DHS peaks as ChIP-Seq peak positive or negative. The value shown the difference in actual AUPRC and the AUPRC of random assortment, over 1 minus the AUPRC of random assortment. In other words, this is the increase in AURPC relative to the maximum possible increase. Diagonal elements are the TF's motif predicting ChIP-Seq peaks of the same TF. Off diagonal targets are TF motifs predict ChIP-Seq peaks of a different TF (negative controls).

In addition, instead of a prediction task, we can assess the actual TF ChIP-seq signal in the mapTF predicted peaks for each factor (TF+), compared to the signal in DHS peak regions without the mapTF motif (TF-). The ratio of these signals is shown in **Fig 7.12A**, normalized to the peak enrichment for each TF. In each case the diagonal elements, where the predicted TF matches the ChIP-seq TF, have the largest increase in signal, and the off-diagonal elements all show considerably less ChIP-seq signal enrichment (**Fig 7.12B**). Thus, the mapTF motif predictions are correlated with an increased ChIP-seq signal at those locations.

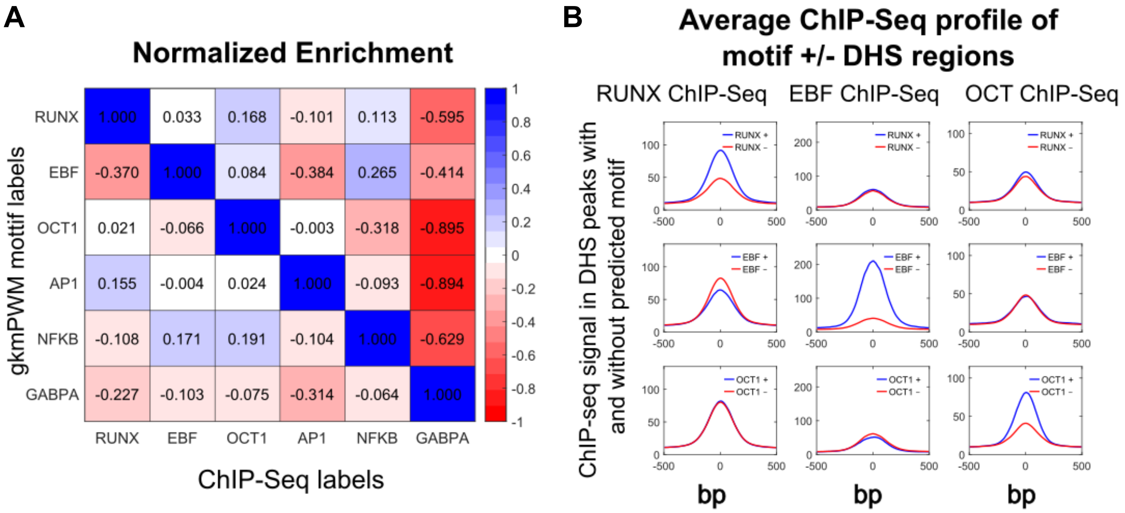


Figure 7.12 ChIP-Seq signal in DHS peaks is larger when the match TF motif is present

A) *The enrichment of ChIP-Seq signal (1-signal(motif-)/signal(motif+)) in peaks with a motif present versus not present. Diagonal elements have matching TF motif calls and ChIP-Seq experiments.* ***B)*** *The average read profile of ChIP-Seq in motif+ versus motif- regions. Rows and columns are formatted similarly to A.*

A collection of predicted binding sites for distal enhancers over many cell-types

At https://www.beerlab.org/gkmpwm, I provide the mapTF predictions of binding sites over all DNase/ATAC-Seq datasets as bed files. These predictions can be seen on an online Integrative Genomics Viewer (**Fig 7.13**). A higher quality example of these tracks is shown in **Fig 7.14** for multiple peaks in an intron of *STEAP1B* on chr7, which shows how many different combinations of TFs can bind at a peak.

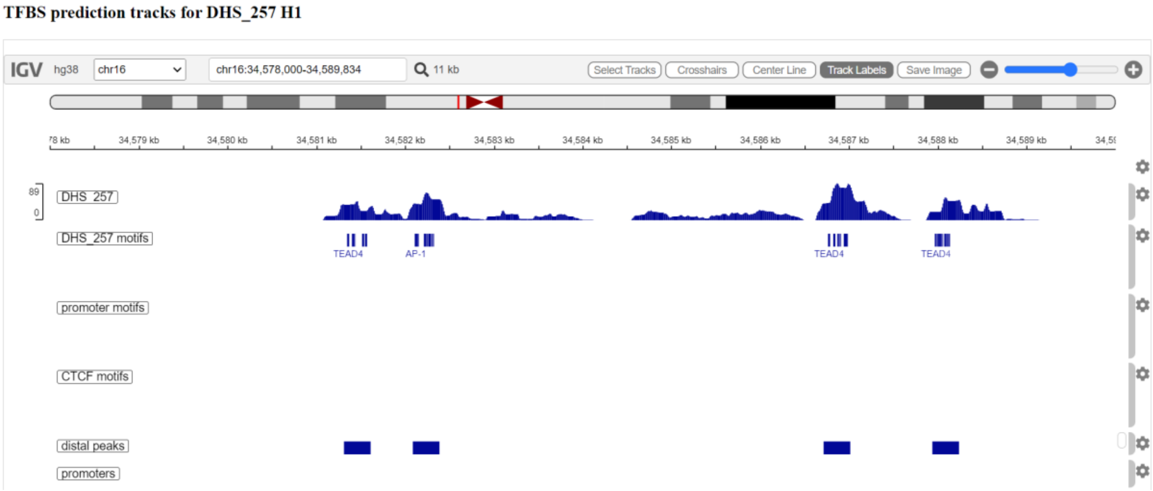


Figure 7.13 Visualizing mapTF predictions of TF binding sites in distal enhancers

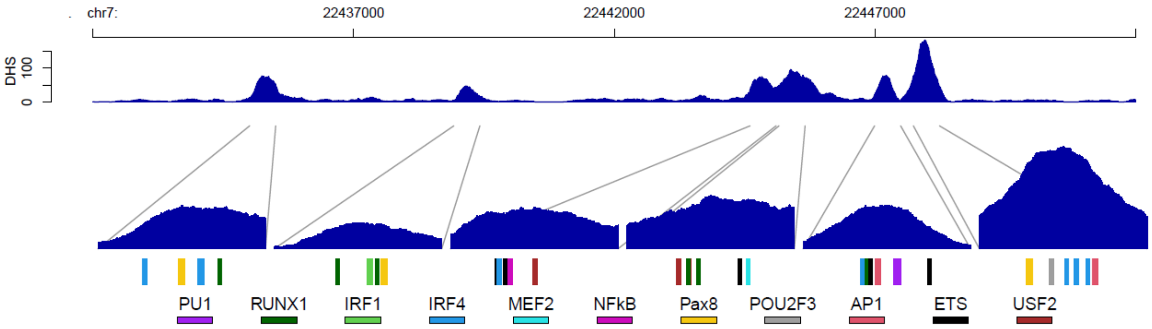


Figure 7.14 GM12878 motifs predictions in an intron of STEAP1B on chr7

7.4 Discussion

Compared to the systematic evaluation of the model performance presented in previous chapters, a rigorous assessment of specific TFBS predictions made by mapTF is made difficult by the relative scarcity of assays that target the exact position of TFBSs within enhancers, and I believe that additional analysis should be done to test the quality of the mapTF predictions presented. Nevertheless, mapTF predictions of TF binding are well aligned with saturation mutagenesis assays and predict ChIP-Seq peaks and enrichment in signal well. As mentioned in the introduction to this chapter, the predicted locations of TFBSs will be necessary when constructing models of gene regulatory networks. Identifying the correct TFs that bind to the enhancers involved in the network is an essential component of the model.

Chapter 8
Discussion

I have presented a method to extract TF binding motifs from enhancer prediction models and showed that our method, gkmPWM outperforms previous feature detection algorithms. Our method uses Lasso regression to find the embedded frequency of TFBS PWM motifs in a set of background sequences to reproduce the score function of the machine learning model. While gkmPWMlasso uses a known set of PWM motifs as input, I also presented an algorithm which learns de novo PWM motifs by Lagrange optimization. I demonstrated that gkmPWM identifies a compact set of cell-type specific core TFs that can accurately classify the cell type and tissue of held out datasets. By contrast, I identified a common set of tissue and cell type independent promoter motifs. I mapped gkmPWM motifs to DHS peaks at nucleotide resolution and showed that motifs are predictive of TF binding. I made comparisons of linear (gkm-SVM) and non-linear (CNN) enhancer prediction methods and found that they are likely learning different features, with gkm-SVM able to learn weaker motifs than non-linear methods. Finally, motivated by these systematic comparisons across all ENCODE4 chromatin accessibility datasets, I developed a novel hybrid gkm-DNN model that outperforms both gkm-SVM and DNNs.

Identifying cell-type specific TFs and their binding sites within enhancers is essential for the development of predictive gene regulatory network models of cell state transitions. For example, cell-state transitions in development and cancer are likely driven by differential activation or repression of a set of TFs which alter the regulatory state of the cell. I envision gkmPWM as a critical component to model dysregulation in disease or in response to mutation or environmental stimuli. Altered TF activity in such situations can be identified by training gkm-SVM on differentially accessible ATAC-seq peaks and extracting motifs with gkmPWM, or by comparing motifs extracted from models trained on ATAC-seq in disease vs. healthy tissue control. One drawback of our method is that I cannot distinguish between TFs within families with similar binding domains (e.g. GATA1 vs GATA2). However, this can be overcome in some cases by using RNA-seq in parallel or using additional tissue-specific biological knowledge. In other cases, the activity of the TF inferred by gkmPWM can reflect an active state of a TF (e.g. phosphorylated or ubiquitinated) that is not detectable from mRNA levels. Finally, I anticipate that gkmPWM will aid the construction of dynamic gene regulatory network models by identifying specific binding sites of TFs in enhancers flanking TF genes. These regulatory connections (gene-to-TFBS in enhancer) and (enhancer-to-gene) specify the wiring of the genetic networks that specify stable cell states and transitions in development and disease. Our gkmPWM predictions can specify the inputs to a particular enhancer, but do not necessarily specify its gene targets. However, much recent progress has been made in identifying mechanisms of enhancer-promoter interactions, and the observation that CTCF looping constrains these interactions considerably reduces this search space. In our recent work, identification of TFBS for DE TFs in CRISPRi responsive enhancers in the transition from ESC to DE allowed the development of a detailed dynamic model of the transition, which explains the delay phenotype and hysteresis of enhancer perturbation[23] which may be applicable to hysteresis in other regulatory model systems and may explain much of the difficulty in validating disease associated regulatory variants in human disease models.

Bibliography

1. Grove, J. *et al.* Identification of common genetic risk variants for autism spectrum disorder. *Nat Genet* **51**, 431–444 (2019).
2. Pasquali, L. *et al.* Pancreatic islet enhancer clusters enriched in type 2 diabetes risk-associated variants. *Nat Genet* **46**, 136–143 (2014).
3. Huang, Q. *et al.* A prostate cancer susceptibility allele at 6q22 increases RFX6 expression by modulating HOXB13 chromatin binding. *Nature Genetics* **46**, 126–135 (2014).
4. Lettice, L. A. *et al.* A long-range Shh enhancer regulates expression in the developing limb and fin and is associated with preaxial polydactyly. *Human Molecular Genetics* **12**, 1725–1735 (2003).
5. Lettice, L. A., Devenney, P., Angelis, C. D. & Hill, R. E. The Conserved Sonic Hedgehog Limb Enhancer Consists of Discrete Functional Elements that Regulate Precise Spatial Expression. *Cell Reports* **20**, 1396–1408 (2017).
6. Ghandi, M., Lee, D., Mohammad-Noori, M. & Beer, M. A. Enhanced Regulatory Sequence Prediction Using Gapped k-mer Features. *PLOS Computational Biology* **10**, e1003711 (2014).
7. Alipanahi, B., Delong, A., Weirauch, M. T. & Frey, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nature Biotechnology* **33**, 831–838 (2015).
8. Lee, D. *et al.* A method to predict the impact of regulatory variants from DNA sequence. *Nature Genetics* **47**, 955–961 (2015).

9. Shigaki, D. *et al.* Integration of Multiple Epigenomic Marks Improves Prediction of Variant Impact in Saturation Mutagenesis Reporter Assay. *Human Mutation* **40**, 1280–1291 (2019).

10. Yan, J. *et al.* Systematic analysis of binding of transcription factors to noncoding variants. *Nature* **591**, 147–151 (2021).

11. Patel, Z. M. & Hughes, T. R. Global properties of regulatory sequences are predicted by transcription factor recognition mechanisms. *Genome Biology* **22**, 285 (2021).

12. Dunham, I. *et al.* An integrated encyclopedia of DNA elements in the human genome. *Nature* **489**, 57–74 (2012).

13. Pang, B. & Snyder, M. P. Systematic identification of silencers in human cells. *Nat Genet* **52**, 254–263 (2020).

14. Vierstra, J. *et al.* Global reference mapping of human transcription factor footprints. *Nature* **583**, 729–736 (2020).

15. Kelleher, R. J., Flanagan, P. M. & Kornberg, R. D. A novel mediator between activator proteins and the RNA polymerase II transcription apparatus. *Cell* **61**, 1209–1215 (1990).

16. Yang, C., Bolotin, E., Jiang, T., Sladek, F. M. & Martinez, E. Prevalence of the Initiator over the TATA box in human and yeast genes and identification of DNA motifs enriched in human TATA-less core promoters. *Gene* **389**, 52–65 (2007).

17. Kircher, M. *et al.* Saturation mutagenesis of twenty disease-associated regulatory elements at single base-pair resolution. *Nature Communications* **10**, 1–15 (2019).

18. Banerji, J., Rusconi, S. & Schaffner, W. Expression of a β-globin gene is enhanced by remote SV40 DNA sequences. *Cell* **27**, 299–308 (1981).

19. Hansen, J. L., Loell, K. J. & Cohen, B. A. A test of the pioneer factor hypothesis using ectopic liver gene activation. *eLife* **11**, e73358 (2022).

20. Hansen, J. L. & Cohen, B. A. A quantitative metric of pioneer activity reveals that HNF4A has stronger in vivo pioneer activity than FOXA1. *Genome Biology* **23**, 221 (2022).

21. Rao, S. S. P. *et al.* A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping. *Cell* **159**, 1665–1680 (2014).

22. Xi, W. & Beer, M. A. Loop competition and extrusion model predicts CTCF interaction specificity. *Nature communications* **12**, 1046 (2021).

23. Luo, R. *et al.* Dynamic network-guided CRISPRi screen identifies CTCF-loop-constrained nonlinear enhancer gene regulatory activity during cell state transitions. *Nat Genet* **55**, 1336–1346 (2023).

24. Johnson, D. S., Mortazavi, A., Myers, R. M. & Wold, B. Genome-Wide Mapping of in Vivo Protein-DNA Interactions. *Science* **316**, 1497–1502 (2007).

25. Robertson, G. *et al.* Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nat Methods* **4**, 651–657 (2007).

26. Visel, A. *et al.* ChIP-seq accurately predicts tissue-specific activity of enhancers. *Nature* **457**, 854–858 (2009).

27. Barski, A. *et al.* High-Resolution Profiling of Histone Methylations in the Human Genome. *Cell* **129**, 823–837 (2007).

28. Wang, Z. *et al.* Genome-wide Mapping of HATs and HDACs Reveals Distinct Functions in Active and Inactive Genes. *Cell* **138**, 1019–1031 (2009).

29. O'Geen, H., Echipare, L. & Farnham, P. J. Using ChIP-Seq Technology to Generate High-Resolution Profiles of Histone Modifications. in *Epigenetics*

Protocols (ed. Tollefsbol, T. O.) 265–286 (Humana Press, Totowa, NJ, 2011). doi:10.1007/978-1-61779-316-5_20.

30. Villar, D. *et al.* Enhancer Evolution across 20 Mammalian Species. *Cell* **160**, 554–566 (2015).

31. Heintzman, N. D. *et al.* Histone modifications at human enhancers reflect global cell-type-specific gene expression. *Nature* **459**, 108–112 (2009).

32. Oh, J. W. & Beer, M. A. Gapped-kmer sequence modeling robustly identifies regulatory vocabularies and distal enhancers conserved between evolutionarily distant mammals. *Nature Communications* 2024.10.06.561128 Preprint at https://doi.org/10.1101/2023.10.06.561128 (2023).

33. Mikkelsen, T. S. *et al.* Genome-wide maps of chromatin state in pluripotent and lineage-committed cells. *Nature* **448**, 553–560 (2007).

34. Li, Z. *et al.* Identification of transcription factor binding sites using ATAC-seq. *Genome Biology* **20**, 45 (2019).

35. Klemm, S. L., Shipony, Z. & Greenleaf, W. J. Chromatin accessibility and the regulatory epigenome. *Nat Rev Genet* **20**, 207–220 (2019).

36. Thurman, R. E. *et al.* The accessible chromatin landscape of the human genome. *Nature* **489**, 75–82 (2012).

37. Inoue, F. & Ahituv, N. Decoding enhancers using massively parallel reporter assays. *Genomics* **106**, 159–164 (2015).

38. Tewhey, R. *et al.* Direct Identification of Hundreds of Expression-Modulating Variants using a Multiplexed Reporter Assay. *Cell* **165**, 1519–1529 (2016).

39. Arnold, C. D. *et al.* Genome-Wide Quantitative Enhancer Activity Maps Identified by STARR-seq. *Science* **339**, 1074–1077 (2013).

40. Klein, J. C. *et al.* A systematic evaluation of the design and context dependencies of massively parallel reporter assays. *Nat Methods* **17**, 1083–1091 (2020).

41. Thakore, P. I. *et al.* Highly specific epigenome editing by CRISPR-Cas9 repressors for silencing of distal regulatory elements. *Nat Methods* **12**, 1143–1149 (2015).

42. Fulco, C. P. *et al.* Activity-by-contact model of enhancer–promoter regulation from thousands of CRISPR perturbations. *Nat Genet* **51**, 1664–1669 (2019).

43. Reilly, S. K. *et al.* Direct characterization of cis-regulatory elements and functional dissection of complex genetic associations using HCR–FlowFISH. *Nat Genet* **53**, 1166–1176 (2021).

44. Mirny, L. A. Nucleosome-mediated cooperativity between transcription factors. *PNAS* **107**, 22534–22539 (2010).

45. Berg, O. G. & von Hippel, P. H. Selection of DNA binding sites by regulatory proteins. Statistical-mechanical theory and application to operators and promoters. *J Mol Biol* **193**, 723–750 (1987).

46. Bailey, T. L. & Elkan, C. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *Proc Int Conf Intell Syst Mol Biol* **2**, 28–36 (1994).

47. Jolma, A. *et al.* Multiplexed massively parallel SELEX for characterization of human transcription factor binding specificities. *Genome Res.* **20**, 861–873 (2010).

48. Fornes, O. *et al.* JASPAR 2020: update of the open-access database of transcription factor binding profiles. *Nucleic Acids Res* **48**, D87–D92 (2020).

49. Jolma, A. *et al.* DNA-Binding Specificities of Human Transcription Factors. *Cell* **152**, 327–339 (2013).

50. Reddy, T. E. *et al.* Effects of sequence variation on differential allelic transcription factor occupancy and gene expression. *Genome Res* **22**, 860–869 (2012).

51. Fonseca, G. J. *et al.* Diverse motif ensembles specify non-redundant DNA binding activities of AP-1 family members in macrophages. *Nat Commun* **10**, 414 (2019).

52. Roider, H. G., Kanhere, A., Manke, T. & Vingron, M. Predicting transcription factor affinities to DNA from a biophysical model. *Bioinformatics* **23**, 134–141 (2007).

53. Lee, D., Karchin, R. & Beer, M. A. Discriminative prediction of mammalian enhancers from DNA sequence. *Genome Res.* **21**, 2167–2180 (2011).

54. Patel, Z. M. & Hughes, T. R. Global properties of regulatory sequences are predicted by transcription factor recognition mechanisms. *Genome Biology* **22**, 285 (2021).

55. Zhou, J. & Troyanskaya, O. G. Predicting effects of noncoding variants with deep learning–based sequence model. *Nat Methods* **12**, 931–934 (2015).

56. Ghandi, M., Mohammad-Noori, M. & Beer, M. A. Robust k-mer frequency estimation using gapped k-mers. *J. Math. Biol.* **69**, 469–500 (2014).

57. Lee, D. *et al.* A method to predict the impact of regulatory variants from DNA sequence. *Nat Genet* **47**, 955–961 (2015).

58. Beer, M. A. Predicting enhancer activity and variant impact using gkm-SVM. *Human Mutation* **38**, 1251–1258 (2017).

59. Mo, A. *et al.* Epigenomic landscapes of retinal rods and cones. *eLife* **5**, e11613 (2016).

60. Kreimer, A. *et al.* Predicting gene expression in massively parallel reporter assays: A comparative study. *Human Mutation* **38**, 1240–1250 (2017).

61. Amanchy, R. *et al.* Identification of Novel Phosphorylation Motifs Through an Integrative Computational and Experimental Analysis of the Human Phosphoproteome. *J Proteomics Bioinform* **4**, 22–35 (2011).

62. Pampari, A. *et al.* Bias factorized, base-resolution deep learning models of chromatin accessibility reveal cis-regulatory sequence syntax, transcription factor footprints and regulatory variants. Zenodo https://doi.org/10.5281/zenodo.10396047 (2023).

63. Moore, J. E. *et al.* Expanded encyclopaedias of DNA elements in the human and mouse genomes. *Nature* **583**, 699–710 (2020).

64. Kundaje, A. *et al.* Integrative analysis of 111 reference human epigenomes. *Nature* **518**, 317–330 (2015).

65. Xi, W. & Beer, M. A. Local epigenomic state cannot discriminate interacting and non-interacting enhancer–promoter pairs with high accuracy. *PLOS Computational Biology* **14**, e1006625 (2018).

66. Reports from the fifth edition of CAGI: The Critical Assessment of Genome Interpretation - Andreoletti - 2019 - Human Mutation - Wiley Online Library. https://onlinelibrary.wiley.com/doi/10.1002/humu.23876.

67. Dong, S. & Boyle, A. P. Predicting functional variants in enhancer and promoter elements using RegulomeDB. *Human Mutation* **40**, 1292–1298 (2019).

68. Kreimer, A., Yan, Z., Ahituv, N. & Yosef, N. Meta-analysis of massively parallel reporter assays enables prediction of regulatory function across cell types. *Human Mutation* **40**, 1299–1313 (2019).

69. Heinz, S. *et al.* Simple Combinations of Lineage-Determining Transcription Factors Prime cis-Regulatory Elements Required for Macrophage and B Cell Identities. *Molecular Cell* **38**, 576–589 (2010).

70. Avsec, Ž. *et al.* Base-resolution models of transcription-factor binding reveal soft motif syntax. *Nat Genet* **53**, 354–366 (2021).

71. Beer, M. A., Shigaki, D. & Huangfu, D. Enhancer Predictions and Genome-Wide Regulatory Circuits. *Annu. Rev. Genom. Hum. Genet.* **21**, 37–54 (2020).

72. Ghandi, M. *et al.* gkmSVM: an R package for gapped-kmer SVM. *Bioinformatics* **32**, 2205–2207 (2016).

73. Lee, D. LS-GKM: a new gkm-SVM for large-scale datasets. *Bioinformatics* **32**, 2196–2198 (2016).

74. Ho, S. W. T. *et al.* Regulatory enhancer profiling of mesenchymal-type gastric cancer reveals subtype-specific epigenomic landscapes and targetable vulnerabilities. *Gut* gutjnl-2021-326483 (2022) doi:10.1136/gutjnl-2021-326483.

75. Sheng, T. *et al.* Integrative epigenomic and high-throughput functional enhancer profiling reveals determinants of enhancer heterogeneity in gastric cancer. *Genome Med* **13**, 158 (2021).

76. Xing, M. *et al.* Genomic and epigenomic EBF1 alterations modulate TERT expression in gastric cancer. *J Clin Invest* **130**, (2020).

77. Xu, C. *et al.* Comprehensive molecular phenotyping of ARID1A-deficient gastric cancer reveals pervasive epigenomic reprogramming and therapeutic opportunities. *Gut* **72**, 1651–1663 (2023).

78. Razavi-Mohseni, M. *et al.* Machine learning identifies activation of RUNX/AP-1 as drivers of mesenchymal and fibrotic regulatory programs in gastric cancer. *Genome Res.* **34**: 680–695 (2024).

79. Razavi-Mohseni, M., Shigaki, D. & Beer, M. A. Machine Learning Sequence Modeling Identifies Gene Regulatory Responses to Bone Marrow Stromal Interactions in Multiple Myeloma. *Blood* **142**, 4144 (2023).

80. Li, Q. V. *et al.* Genome-scale screens identify JNK–JUN signaling as a barrier for pluripotency exit and endoderm differentiation. *Nat Genet* **51**, 999–1010 (2019).

81. Yao, D. *et al.* Multicenter integrated analysis of noncoding CRISPRi screens. *Nat Methods* **21**, 723–734 (2024).

82. Shin, J. Y. *et al.* Epigenetic activation and memory at a TGFB2 enhancer in systemic sclerosis. *Science Translational Medicine* **11**, eaaw0790 (2019).

83. McClymont, S. A. *et al.* Parkinson-Associated SNCA Enhancer Variants Revealed by Open Chromatin in Mouse Dopamine Neurons. *The American Journal of Human Genetics* **103**, 874–892 (2018).

84. IGVF Consortium. The Impact of Genomic Variation on Function (IGVF) Consortium. Preprint at https://doi.org/10.48550/arXiv.2307.13708 (2023).

85. Jain, S. *et al.* CAGI, the Critical Assessment of Genome Interpretation, establishes progress and prospects for computational genetic variant interpretation methods. *Genome Biol* **25**, 53 (2024).

86. Xi, W. & Beer, M. A. Local epigenomic state cannot discriminate interacting and non-interacting enhancer–promoter pairs with high accuracy. *PLOS Computational Biology* **14**, e1006625 (2018).

87. Gschwind, A. R. *et al.* An encyclopedia of enhancer-gene regulatory interactions in the human genome. 2023.11.09.563812 Preprint at https://doi.org/10.1101/2023.11.09.563812 (2023).

88. Inoue, F. *et al.* A systematic comparison reveals substantial differences in chromosomal versus episomal encoding of enhancer activity. *Genome Res.* **27**, 38–52 (2017).

89. Wang, X. *et al.* High-resolution genome-wide functional dissection of transcriptional regulatory regions and nucleotides in human. *Nat Commun* **9**, 5380 (2018).

90. Liu, Y. *et al.* Functional assessment of human enhancer activities using whole-genome STARR-sequencing. *Genome Biology* **18**, 219 (2017).

91. Melnikov, A. *et al.* Systematic dissection and optimization of inducible enhancers in human cells using a massively parallel reporter assay. *Nature Biotechnology* **30**, 271–277 (2012).

92. de Almeida, B. P., Reiter, F., Pagani, M. & Stark, A. DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of synthetic enhancers. *Nat Genet* **54**, 613–624 (2022).

93. Sahu, B. *et al.* Sequence determinants of human gene regulatory elements. *Nat Genet* **54**, 283–294 (2022).

94. Huang, P. *et al.* HIC2 controls developmental hemoglobin switching by repressing BCL11A transcription. *Nat Genet* **54**, 1417–1426 (2022).

95. Villarejo, A., Cortés-Cabrera, Á., Molina-Ortíz, P., Portillo, F. & Cano, A. Differential Role of Snail1 and Snail2 Zinc Fingers in E-cadherin Repression and Epithelial to Mesenchymal Transition*. *Journal of Biological Chemistry* **289**, 930–941 (2014).

96. Qin, K. *et al.* Dual function NFI factors control fetal hemoglobin silencing in adult erythroid cells. *Nat Genet* **54**, 874–884 (2022).

97. Huang, D.-Y., Kuo, Y.-Y. & Chang, Z.-F. GATA-1 mediates auto-regulation of Gfi-1B transcription in K562 cells. *Nucleic Acids Res* **33**, 5331–5342 (2005).

98. Zhu, X. *et al.* NF-Y Recruits Both Transcription Activator and Repressor to Modulate Tissue- and Developmental Stage-Specific Expression of Human γ-Globin Gene. *PLOS ONE* **7**, e47175 (2012).

99. Kuo, Y.-Y. & Chang, Z.-F. GATA-1 and Gfi-1B interplay to regulate Bcl-xL transcription. *Mol Cell Biol* **27**, 4261–4272 (2007).

100. Gomes, A. M. *et al.* Cooperative Transcription Factor Induction Mediates Hemogenic Reprogramming. *Cell Reports* **25**, 2821-2835.e7 (2018).

101. Bruce, A. W. *et al.* Functional diversity for REST (NRSF) is defined by in vivo binding affinity hierarchies at the DNA sequence level. *Genome Res.* **19**, 994–1005 (2009).

102. Rockowitz, S. *et al.* Comparison of REST Cistromes across Human Cell Types Reveals Common and Context-Specific Functions. *PLOS Computational Biology* **10**, e1003671 (2014).

103. Seo, J. *et al.* AP-1 subunits converge promiscuously at enhancers to potentiate transcription. *Genome Res.* **31**, 538–550 (2021).

104. Phanstiel, D. H. *et al.* Static and dynamic DNA loops form AP-1 bound activation hubs during macrophage development. *Mol Cell* **67**, 1037-1048.e6 (2017).

105. Beagan, J. A. *et al.* YY1 and CTCF orchestrate a 3D chromatin looping switch during early neural lineage commitment. *Genome Res.* (2017) doi:10.1101/gr.215160.116.

106. Weintraub, A. S. *et al.* YY1 Is a Structural Regulator of Enhancer-Promoter Loops. *Cell* **171**, 1573-1588.e28 (2017).

107. Diao, Y. *et al.* A new class of temporarily phenotypic enhancers identified by CRISPR/Cas9-mediated genetic screening. *Genome Res.* **26**, 397–405 (2016).

108. Grant, C. E., Bailey, T. L. & Noble, W. S. FIMO: scanning for occurrences of a given motif. *Bioinformatics* **27**, 1017–1018 (2011).