RICE UNIVERSITY

Population Regulomics: Applying population genetics to the cis-regulome

by

Troy Ruths

A THESISSubmitted
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
Doctor of Philosophy

APPROVED, THESIS COMMITTEE:

Dr. Luay K. Nakhleh
Associate Professor, Chair,
Computer Science

Dr. Chris Jermaine
Associate Professor,
Computer Science

Dr. Lydia E. Kavraki
Professor,
Computer Science

Dr. Michael Kohn
Associate Professor,
Ecology and Evolutionary Biology

Houston, Texas
May 2014
Abstract

Population Regulomics: Applying population genetics to the *cis*-regulome

Troy Ruths

Population genetics provides a mathematical and computational framework for understanding and modeling evolutionary processes, and so it is vital for the investigation of biological systems. In its current state, molecular population genetics is exclusively focused on molecular sequences (DNA, RNA, or amino acid sequences), where all application-ready simulators and analytic measures work only on sequence data. Consequently, in the early 2000s, when technologies became available to sequence entire genomes, population genetic approaches were naturally applied to mine out signatures of selection and conservation, resulting in the sub-field of population genomics. Nearly every present genome project applies population genomic techniques to identify functional information and genome structure.

Recent technologies have ushered in a similar wave of genetic information, this time focusing on biological mechanisms operating above the genome, most notably on gene regulation (regulatory networks). In this work, I develop a molecular population genetics approach for gene regulation, called population regulomics, which includes simulators and analytic measurements that operate on populations of regulatory networks. I conduct extensive data analyses to connect the genome with the *cis*-regulome, develop computationally efficient simulators, and adapt population genetic measurements on sequence to the regulatory network. By connecting genomic information to *cis*-regulation, we may apply the wealth of knowledge at the genome level to observed patterns at the regulatory level with unknown evolutionary origin.
I demonstrate that by applying population regulomics to the *E. coli* cis-regulatory network, for the first time we are able to quantitate the evolutionary origins of topological patterns and reveal the surprising amount of neutral signal in the bacterial *cis*-regulome. Since regulatory networks play a central role in cellular functioning and, consequently, organismal fitness, this new sub-field of population regulomics promises to shed the light of evolution on regulatory mechanisms and, more broadly, on the genetic mechanisms underlying the various phenotypes.
Acknowledgements

This thesis was a collaborative effort and is the result of much hard work and dedication of several individuals. I would like to thank my committee members, especially my advisor Dr. Luay Nakhleh, whose guidance, expertise, and interest were invaluable to the success of this research.

I would also like to thank my wife Ivy for her support and patience. I want to thank my brothers for expediting my growth by pointing out my flaws, but also for serving as strong role models on this academic path. To my parents - thank you for always, always supporting my ambitions and raising me into the person I am today, because I know it wasn’t easy.
Contents

1 Introduction .................................................. 1

2 Model ................................................................ 13
   2.1 Population genetic framework ......................... 13
   2.2 CRN genotype .............................................. 15

3 Boosting forward-time population genetic simulators through genotype compression ................................................. 22
   3.1 Using population genetic simulations as a null model ......................... 23
   3.2 Efficient genotype compression .................................. 24
   3.3 Compression in forward-time simulators ......................... 38

4 Binding site accumulation and topological features correlate with ncDNA .................................................. 47
   4.1 Distribution of TFBS ........................................ 49
   4.2 Distribution of TFBS length .................................. 49
   4.3 TFBS accumulation positively correlates with the amount of ncDNA ................. 51
   4.4 Regulatory complexity positively correlates with the amount of ncDNA ................. 53
   4.5 Enrichment of promoter length in *E. coli* CRN topology .......................... 54

5 Quantifying the role of neutral and adaptive forces in driving regulatory complexity and topology ................................................. 59
   5.1 ncDNA drives regulatory network complexity ......................... 60
   5.2 Building an evolutionary null (neutral) model for the *E. coli* CRN ......................... 65

6 Towards measures for population regulomics ................................................. 87
   6.1 Computational pipeline for inferring a population of CRNs ......................... 88
   6.2 Interaction alignments ........................................ 89
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3</td>
<td>Interaction diversity and segregating interactions</td>
<td>91</td>
</tr>
<tr>
<td>6.4</td>
<td>Building the Pan-Regulome</td>
<td>93</td>
</tr>
<tr>
<td>6.5</td>
<td>Measuring diversity in a population of networks at the gene, promoter, and interaction level</td>
<td>96</td>
</tr>
<tr>
<td>6.6</td>
<td>Estimating interaction mutation rates</td>
<td>98</td>
</tr>
<tr>
<td>6.7</td>
<td>Testing the independent interaction hypothesis</td>
<td>99</td>
</tr>
<tr>
<td>6.8</td>
<td>Inference of interaction gain and loss rate</td>
<td>101</td>
</tr>
<tr>
<td>6.9</td>
<td>Correlating interaction diversity with fitness</td>
<td>103</td>
</tr>
<tr>
<td>7</td>
<td>Discussion</td>
<td>105</td>
</tr>
<tr>
<td>7.1</td>
<td>Improving the performance of simulations</td>
<td>106</td>
</tr>
<tr>
<td>7.2</td>
<td>The significant role of ncDNA in CRN topology</td>
<td>108</td>
</tr>
<tr>
<td>7.3</td>
<td>Quantifying neutral signal in <em>E. coli</em></td>
<td>109</td>
</tr>
<tr>
<td>7.4</td>
<td>Revitalized non-adaptive perspective on CRN evolution</td>
<td>112</td>
</tr>
<tr>
<td>7.5</td>
<td>Future work</td>
<td>114</td>
</tr>
</tbody>
</table>
List of Figures

2.1 The population genetic lifecycle can be modeled as non-overlapping generations in which mutation, recombination, natural select, and drift operate. ......................................................... 14

2.2 A typical node in the CRN would correspond to co-transcribed genes or transcription units. Binding sites are denoted as short vertical rectangles surrounding the gene coding sequence. The coding sequence resides underneath the right angle arrows and ncDNA may be upstream (called the promoter) or downstream of the coding sequence. In eukaryotes, ncDNA within the gene coding sequence is called introns, which may harbor binding sites as well. .............................. 16

2.3 CRNs are modeled based on their sequence characteristics, including the length of promoter sequence and the transcription units in the network. In this diagram, I mapped the CRN model to an *E. coli* example, such that regulatory regions are promoters and transcription units are operons. Using the sequence information, it is possible to generate the network representation of the CRN. ................................. 17

3.1 Forward time simulators are useful for generating statistical null models for complex evolutionary scenarios or genotypes. A standard apparatus for investigating neutral bacterial evolution is diagrammed in this figure. Large populations of initially identical individuals evolve over long periods of time to give rise to a distribution of genotypes expected by neutral evolution. Running multiple simulations and running longer simulations help combat start condition biases. ............................... 24
3.2 The genotype compression algorithm works with any forward-time population genetic simulator that models the evolution of a collection of individuals over time (Population Simulation). The spontaneous process of mutation gives rise to novel genotypes through small modifications, which naturally induces a genotype space. However, rather than storing the genotypes themselves, we compress the genotype space by storing only the mutations of the genotype space, which we call the Operation Graph. It is possible to retrieve genotype data through decompression by applying the mutation operations as encoded by the topology of the operation graph. Decompression can be expedited by caching – as determined by the Compression Policy – the output of frequently applied operations. Thus, the simulation can access, at any point, the data for active genotypes.

3.3 An example execution of Greedy-Load is illustrated on an abstract operation graph. Each node a-l represents a distinct genotype (or allele) and each edge depicts evolutionary descent by mutation (one parent) or recombination (two parents). The actual genotype representation could be a sequence or pathway. Genotypes may be compressed (○), cached (□) or active (=). The load of each genotype is depicted as the background color, with darker colors corresponding to greater load. Sequential steps taken by the Greedy-Load algorithm are illustrated from left to write, showing the incremental changes that update the set of uncompressed genotypes from \{a, c\} in Pane 1 to \{e, g, h, i\} in Pane 4. Dashed arrows within each step illustrate which genotypes are compressed and uncompressed. For instance, in Pane 2, d is uncompressed (cached) and a is compressed. For this example, the total number of cached genotypes \(k\) is 4.


3.5 The tradeoff between Greedy-Load parameters \(k\) and \(t\) are presented as a heatmap of average execution times (log sec/gen), with a mutation scenario on the left and recombination on the right. Lighter colors are faster (better) simulations. The parameter \(k\) is given as the compression ratio \(k/N\), where \(N = 10^3\) is constant in all the simulations.
3.6 The average time per generation and memory usage required by each compression algorithm for replicate simulations of sequences (left) and pathways (right). For both time – measured in seconds per generation – and space – measured in total MB – lower values are better. The performance is measured across a range of genotype sizes: $10^5-10^7$ bases for sequences and $100-1,000$ genes for pathways. Larger genotypes require more space and longer execution times, hence a diagonal line in the space-time tradeoff. Solid lines connect a compression policy — top to bottom: Store-Active, Greedy-Load, Store-Root — and dashed lines connect genotype sizes (e.g., $10^5$ nt for each policy). Greedy-Load provides 95% compression for sequences and 90% for pathways.

3.7 The speed, measured in sec/gen, is plotted for Greedy-Load simulations across varying levels of compression (y-axis) and recombination (x-axis). Lighter colors are slower simulations, displayed in log-scale.

3.8 Left: We measured the ability of Greedy-Load to handle larger and larger genotypes while maintaining a memory ceiling of 100 MB. This memory ceiling was imposed by scaling the number of explicit genotypes (right axis, solid squares). Right: The time, measured as seconds per generation, and memory heap utilized by a simulation of 1000 individuals with sequences of length 50 Mb, using 95% compression.

4.1 Left. The TFBS density ($\#$TFBS/nt) was calculated for a sliding window along upstream ncDNA regions of TUs. TFBS density is normalized by the number of TUs and TFs in the study. Right. The distribution of TFBS broken out by TF is plotted for each organism. The distribution of all TFs is denoted by a solid black line.

4.2 The distribution of binding site width (right) and consensus sites (right) for *E. coli* (top) and several eukaryotes (bottom). The solid vertical lines display the mean length depicted in each plot.

4.3 The distribution of binding site width (right) and consensus sites (right) for *E. coli* (top) and several eukaryotes (bottom). The solid vertical lines display the mean length depicted in each plot.
4.4 Significant promoter length signal exists in the *E. coli* regulatory network on both the system and subsystem level. In A, each operon is plotted with its in-degree and promoter length; we report the Pearson correlation coefficient and significance above the plot. In B, the distribution of promoter length for operons which participate in feed forward loops (FFL) are presented as boxplots per node. The left axis provides the node label, which corresponds to the node in the subgraph diagram C (e.g., 0 or 1), along with the number of distinct operons represented in that distribution. The difference between the average promoter length in the node distribution minus the average promoter length in the network is listed on the right axis of B. Distributions with significant uplift, assessed using a non-parametric Wilcoxon ranksums test, are indicated with a gray background behind the boxplot (p-values for FFL0= $4 \times 10^{-11}$, FFL1= $5 \times 10^{-4}$).

4.5 Relevant distributions for genomic- and network-level properties of the *E. coli* regulatory network are presented. The top row provides information on distributions related to promoter length and the bottom row plots binding site gain and loss rate for each operon against their out-degree. The top left plot compares the distribution of promoter length of operons which encode a TF and do not encode a TF (‘non-TF’). The number of operons represented in each distribution are listed in the parentheses, with all operons in the regulatory network accounted for. Although TF encoding operons have an elevated average and median promoter length, it is not significant (Mann-Whitney non-parametric test, p-value= 0.18). In the top right, each operon is plotted with its in-degree and promoter length; we report the Pearson correlation coefficient and significance above the plot. In the bottom plots, the out-degree of TF-encoding operons is plotted against their spontaneous gain rate (left) and loss rate (left), as computed from their position weight matrices.
4.6 The distribution of promoter length for operons which participate in single input modules (SIM), feed forward loops (FFL), and bi-fans are presented as boxplots per node. The left axis provides the node label, which corresponds to the node in the subgraph diagram (e.g., 0 or 1), along with the number of distinct operons represented in that distribution. The difference between the average promoter length in the node distribution minus the average promoter length in the network is listed on the right axis. Distributions with significant uplift, assessed using a non-parametric Wilcoxon ranksums test, are indicated with a gray background behind the boxplot. The p-values for these distributions are: FFL0$= 4 \times 10^{-11}$, FFL1$= 5 \times 10^{-4}$, Bi-fan0$= 9 \times 10^{-10}$, Bi-fan1$= 8 \times 10^{-11}$.

5.1 a. Simulations using a 5-gene pathway under mutation and drift yield log-linear increases in the number of binding sites (#TFBS), degree, multiplicity, and redundancy with respect to the amount of ncDNA per transcription unit. Each line corresponds to a different binding site length, given in the legend. b. The results from running population genetic simulations on a random fitness landscape (y-axis) compared with the results from neutral - mutation and drift - simulations (x-axis). Deviations from neutrality are points off the diagonal. c. The pathway properties respond to the contributing factors $s$, which goes from insignificant ($s = 10^{-6}$) to dominant ($s = 1$). The x-axis denotes the strength of selection in a population, or $N s$. The rows denote the property under selection and the columns represent the high gain and high loss environments, respectively. The optimal value was chosen against the neutral bias and is denoted by a dashed line for degree. In the case of redundancy, the optimal value was 0 (high gain) and 1 (high loss). Shapes denote the regulatory complexity measurements of #TFBS (□), degree (○), multiplicity (×), and redundancy (△).

5.2 Results for simulations with $N = 10^9$ and $L = 100$ for varying binding site sizes.
5.3 The in-degree and out-degree distributions for the actual \textit{E. coli} network (A) and the random networks (B) are compared side by side. Scatter points indicate the 12 logarithmic bins used to plot the lines. The discontinuity in the out-degree line in the random plot indicates that there are no nodes in the bin with a degree of around 100. We used a two sample K-S test and found that all degree distributions did not differ significantly between the actual and random distributions.

5.4 Z-scores for 3 (A) and 4 (B) node subgraphs are ranked according to frequency in the \textit{E. coli} network. The left axis provides the scale for the z-score (bars) and the right axis measures the frequency (dashed line) of each subgraph. For each subgraph, the z-score for our model (black) and the edge-rewiring model (gray) are graphed side-by-side. Both models used 1,000 random networks to calculate significance. Important motifs are annotated along with the precise z-score found by each model (left, our model; right, edge-switching). Only the top 20 most frequent 4 node subgraphs are shown.

5.5 Z-scores for low frequency 4 node subgraphs are ranked according to frequency in the \textit{E. coli} network. The left axis provides the scale for the z-score (bars) and the right axis measures the frequency (dashed line) of each subgraph. For each subgraph, the z-score for our model (black) and the edge-switching model (gray) are graphed side-by-side.

5.6 For each operon in the \textit{E. coli} network, the distribution of significance (measured in z-score) is plotted for several local network properties: Clustering (A), Degree (B), In-Degree (C), and Out-Degree (D). Vertical bars indicate a z-score at ±3, separating each distribution into three categories. The percent of operons in each of these three categories — significantly below, expected by, and significantly above — are listed underneath the name of the property. For instance, 89\% of the operons have clustering coefficients as expected by the null model. In the Out-Degree distribution, only the 50 operons which encode TFs are included.

5.7 The distribution of number of edges at equilibrium under the alternate simulation scenarios.

5.8 The degree distributions — comparing actual to random — for the alternate simulation scenarios.

5.9 The distribution of clustering coefficients under the alternate simulation scenarios.
5.10 The joint distribution of in-degree and promoter length — comparing actual to random — under the alternate simulation scenarios.

5.11 The z-scores for 3-node subgraphs under the alternate simulation scenarios, sorted by the frequency in the *E. coli* network.

5.12 The z-scores for the most common 4-node subgraphs under the alternate simulation scenarios, sorted by the frequency in the *E. coli* network.

6.1 It is possible to infer the regulatory annotations of a related genome to a fully annotated reference genome using the PePPER tool coupled with additional data services.

6.2 The fifteen strains examined in this study, along with the outgroup *E. furgusonii* is depicted in this phylogeny. Branch lengths correspond to the number of interaction mutations occurring on each lineage and the color corresponds to inferred number of interaction gains (green) and losses (red) using PAML.

6.3 The pan-regulome is constructed by merging information from multiple *E. coli* strains into one network. The distribution of number of genes (gray) and edges (black) are determined by randomly sampling at most 200 combinations of each strain count from the 15 used in this study. Our pan-regulome focuses on genes within preserved operons, the number of which decreases as strains are added to the pan-regulome construction. The number of edges, as determined by predicted binding sites across the *E. coli* strains, increase as strains are added to the pan-regulome, although the number saturates at around 10 strains.

6.4 Cumulative degree distributions for interaction, promoter, and gene diversity reveal that average pairwise differences are most frequent between promoter sequences and least frequent between interactions. A strong correlation exists between interaction diversity and promoter diversity, while no correlation exists between interaction diversity and gene diversity.
6.5 The distributions for population mutation rate estimators (A) and mutation ratios (B) are plotted as box plots in the log scale. **A**: Population mutation rate estimates $\hat{\theta}$ can be calculated using nucleotide diversity (promoter and gene $\Pi$), segregating sites (promoter and gene $S/a$), interaction diversity (edge $\Pi$), and segregating interactions (edge $S/a$). Distributions are composed of estimates based on each operon coding, promoter, and interaction sequence. **B**: Distributions of mutation ratios per operon illustrate the relative rates between gene, promoter, and interaction mutations.

6.6 The distribution of number of mutations per interaction in the pan-regulome is determined using a maximum-likelihood model of ancestral network reconstruction, along with the number of site mutations in promoters and genes. Any particular edge may be gained and lost several times during the evolutionary history of the *E. coli* population, which is reflected by the interactions with more than 1 mutation. The inset in the interactions plot presents the histogram for interactions that strictly exist in the pan-regulome (versus all the potential edges). Genes show the highest amount of conservation — nearly 90% of coding sequence is shared.
Chapter 1

Introduction

Gene regulation is a major cellular process underlying the central dogma of molecular biology, encoding the sophisticated biochemical cascades that drive cellular behavior and development. The process of gene regulation involves the binding of specialized proteins, called transcription factors (TFs), to binding sites in flanking non-protein coding DNA (ncDNA) regions. Bound transcription factors either recruit or inhibit the transcriptional machinery of the cell, which then ‘turns on’ or ‘turns off’ the target gene product. The links between TFs and their target binding sites form the cis-regulatory network (CRN) in the cell. Reconstructing a CRN from experimental data, elucidating its dynamic and topological properties, and understanding how these properties emerge during development and evolution are major endeavors in experimental and computational biology [2, 3, 4, 5, 6]. These CRNs can be small — connecting only a few genes in a pathway — or very large — encompassing the entire genome. While CRNs are a critical piece of the cellular puzzle, our
ability to formally model and understand them from an evolutionary perspective is nascent, due in large part to their complexity, current tools’ computational limits, and lack of models and measures on populations of CRNs. To this end, I have developed population regulomics: a collection of models, methods, and measures for the practical evolutionary analysis of CRNs and their properties. Because gene regulation is so central to modern genomic studies, synthetic biology, and drug design, a practical body of work like population regulomics could revolutionize how we examine, predict, and create CRN topology and behavior.

At the heart of population regulomics is the CRN model which captures the sequential underpinnings of the regulatory interactions. Because cis-regulatory interactions are encoded in the sequence as binding sites, I develop a model that captures interaction change through the currency of binding site turnover and gain. In doing so, sequence level effects and properties are captured at the network scale. In order to practically execute this CRN model within a simulation framework, novel tools are required to handle the inherent memory issues in simulating entire cis-regulomes in large populations over long periods of time. Lastly, I adapt sequence measures of diversity to the network level, which are useful for not only quantifying pairwise differences or segregating interactions between networks, but also for the inference of important evolutionary parameters like mutation rate.
A new extension to population genetics

Population genetics is ubiquitous in the fast-paced modern genetic world: it is a sophisticated math and framework used for — in the practical sense — interpreting evolutionary insights from real-world observations at the sequence level [7]. However, in its initial conception by Fisher, Wright, and Haldane, population genetics was strictly a theoretical math developed to model natural selection and Mendelian inheritance. Dobzhansky, an eminent field naturalist, was the first to apply the principles of population genetics to natural populations, setting in motion a cascade of intellectual revolutions which has resulted in modern population genetics as we use it today. Since its conception, population genetics has been revitalized by three revolutions: conceptual, empirical, and computational [7]. The theory of coalescent, which provides a framework for examining populations and their properties from a historical perspective, conceptually revamped population genetics, which was largely a math modeling processes forward in time. Secondly, the explosion of genetic sequence data in the forms of SNPs and DNA sequencing technologies provided needed data to which population genetic models may be applied — ushering in molecular population genetics. Lastly, enhanced computing power supported complex simulations and numerical analysis, making practical Markov-chain Monte Carlo simulators, Bayesian estimation, and maximum likelihood calculations. This resulted in the wide application of forward-time simulators as crucial evolutionary research tools. In its current form, population genetics has been extended to genomes (population genomics), proteins (population proteomics), and molecular sequences (molecular population genetics).
The success in these extensions is testament to the elegance and flexibility of population genetics to model evolutionary processes; however, population genetics has yet to be extended to non-sequential biological structures. Instead, complex biological structures are treated in the abstract, greatly reducing the practicality and confidence that we are able to adequately grasp the results of evolution operating on these structures. It is the goal of this thesis to successfully extend population genetics, in the very clear and practical sense, to cis-regulatory networks. In doing so, population regulomics may be a valuable extension to population genetics, allowing us to elevate our ability to model, understand, and experiment — from the evolutionary perspective — systems operating above the genome.

Fundamental role of binding sites and ncDNA in CRN topology

Binding sites are short DNA sequences to which transcription factors bind and regulate gene expression. However, due to their short length, which is generally between five and twenty base pairs, binding sites are hard to detect reliably based solely on sequence analysis [8]. Recent advances in biotechnologies, such as ChIP-seq and ChIP-chip, are enabling in vivo identification of binding sites, along with their affinity to transcription factors (TF), across entire genomes with high accuracy [9].

The availability of these data allows for new analyses and investigations into the role of transcription regulation in cellular decision making and, indeed, the entire functioning of an organism. For example, the location and TF affinity of binding
sites have been used to calculate properties of the regulatory network [10]. Such properties include the connectivity of a network (degree), the robustness of a network to edge manipulation (redundancy), and the robustness of regulatory sequence to mutations (multiplicity). In addition to informing about the topology of regulatory networks, they also have functional implications [11]. From a developmental perspective, numerous studies have shown that regulation of gene expression has either accompanied or even facilitated complexity in developmental systems, driven by the accumulation and proliferation of both transcription factors and their binding sites [12, 13, 14, 15, 16, 17].

Additionally, the availability of genomic data, along with annotations of functional features, is providing an opportunity for exploring the interplay between evolutionary forces and the amount of regulatory regions including binding sites. For example, a recent study comparing human and chimpanzee genomes revealed more signatures of adaptation, or rapid change, in regulatory regions than in coding regions [18]. Further, recent analyses have shown that conserved ncDNA, which harbors binding sites, is almost three times the amount of conserved protein-coding DNA in the *Drosophila* genome [19]. Other studies have shown that the amount of conserved ncDNA underestimates the amount of ncDNA with functional roles [20, 21]. In fact, the level of variation in TF binding sites between closely related species of yeast is substantially larger than that of their regulated orthologs [22].
Open debate on evolutionary origin of CRN properties

Adaptive arguments for binding site gain and loss across organisms are based not only on pathway dynamics, but also on the topological (static) properties of the pathway. For instance, properties such as robustness, modularity, redundancy, complexity, and evolvability can be measured by pathway connectivity and have been argued to convey adaptive value to the individual [11, 23, 24, 25]. The complexity of CRNs, coupled with observed “unexpected” trends in their properties, such as scale-freeness [26], high degree of clustering [27], and overrepresented subgraphs [11, 28, 4, 29], have led to several hypotheses of adaptive origins and explanations of CRNs and their properties. Central to most of these studies was the use of simplistic graph-theoretic models, such as randomly rewiring the connectivity of a biological network, to serve as a null model for CRN connectivity maps and their properties [30]. A recent study invoked fluctuating phenotypic selection to explain the evolution of scale-free distributions and pathway complexity [31]. However, whether adaptation can operate uniformly on the genome scale to have consistent effect on regulatory complexity is an open debate [3, 32]. Assuming regulatory complexity confers fitness, adaptation may select for more complex pathways, hence resulting in the accumulation of binding sites and interactions [15]. However, it has been shown in theory that when subjecting CRNs to the various neutral evolutionary forces and tracing their trajectory in time, many of these topological patterns and complexity may simply arise spontaneously due to the forces of mutation, recombination, gene duplication and genetic drift [29].
Recent experimental evidence also challenges the prevailing view that CRNs are highly adapted systems [33]. In an exhaustive study examining the knock-out (KO) fitness effect and expression level of roughly 75% of genes in *Shewanella oneidensis* MR-1, Morgan et al. found that the majority of genes are sub-optimally expressed: “many bacterial genes are expressed when they are detrimental to fitness, most genes are not upregulated when they are important for fitness, and genes with closely related functions often have different expression patterns.” Further, the authors reviewed other findings in related bacteria and across kingdoms in yeast and found the same sub-optimal pattern.

These studies call into question arguments that were made in favor of adaptive explanations for the emergence and conservation of CRN properties [34, 35, 11, 31], and identify important parameters which may significantly affect the evolution of CRNs from a neutral perspective. Specifically, [24] highlighted the role that promoter length (ncDNA), binding site size, and population size may play in forming certain topological patterns in CRNs.

**Population genetics and forward-time simulators**

Computer simulations have long played a central role in deriving and testing hypotheses in evolutionary biology and population genetics. A novel computational model that couples an individual’s CRN with its underlying genome would facilitate an evolutionary analysis of CRN properties. Thus far, population genetic simulations
have for the most part employed either abstract genotype constructs or very small
CRNs. In fact, previous work made many simplifying assumptions and lacked the
computational muscle to simulate the cis-regulome at scale \[24, 31, 29\]; consequently,
only theoretical “toy” models could be simulated and analyzed. An important insight
into improving the CRN model from previous work is that promoter length and
the spontaneous gain and loss rates of TFBS (TF binding sites) vary substantially
within a genome, and that reducing each distribution to one value potentially eclipses
important emergent properties and structure at the network level. To scale population
genetic simulators to handle these complex distributions, it is imperative to address
the computational requirements of existing simulators so as to enable handling such
large-scale genotypes and flexible genotype representations. Ultimately, a forward-
time population genetic simulator is needed that can handle the aggressive memory
requirements associated with simulating entire cis-regulomes.

Forward-time population genetic simulators are critical research tools in evolu-
tionary biology, as demonstrated by both the growing number of available simulators
and the collection of high-impact studies that employ them \[36\]. These simulators
allow for in-silico generation and testing of evolutionary hypotheses that would oth-
erwise be intractable to generate or test in a laboratory setting due in large part to
the nature of the process. Evolution “is a loose and complex process, the result of
a number of interacting, individually weak forces with many alternative outcomes”
\[37\]. Consequently, forward-time simulators are ideal for tinkering with these weak
forces—changing the ones that are modeled and their relative strengths or rates—in
order to observe the many alternative evolutionary outcomes. Unlike backward, or
coalescent, approaches to evolutionary analysis, forward-time simulators can handle the growing bevy of known evolutionary processes and environments [38].

But forward-time simulations have their limitations: a critical design pivot exists around execution speed, memory usage, and flexibility [36]. Available simulators necessitate a trade-off between flexibility and speed for realistic simulations to be feasible, and often require the user to adjust the evolutionary scenario to fit the capability of a certain simulator using scaling factors. This results in a large collection of simulators that require a decision flowchart to choose an appropriate simulator [39]. For example, methods were recently developed to increase the execution speed of simulations; however, these gains in speed come at the expense of reduced flexibility [40]. Because forward-time simulations track complete ancestral information, including all alleles which arose but were lost, the imposed computational burden limits the potential scope of the problem [38]. Even leveraging rescaling techniques that they employ, such as altering the input parameters to diminish the population size and number of generations, to improve computational efficiency does not evade this critical issue of computation time and memory usage. Simulating large sequences on the order of 10 Mb “tends to make forward simulators crash due to memory usage” [38], which is compounded by the stochastic and unpredictable nature of these simulations. Further, more complex genotypes, such as protein and RNA structures, regulatory pathways, and epigenetic mechanisms, are studied using forward time simulators [24, 41]. Although current simulators exist for efficiently simulating large genomic regions — FREGENE, SimuPOP, or GenomePop — the memory management techniques do not extend to arbitrary genotype representations like pathways or metabolic networks or
other mutation types like insertions or rearrangements [42, 43]. For instance, SimuPop provides a compression module which efficiently encodes long sequence regions with rare mutant variants [44]. In addition, general lossless data compression algorithms cannot scale to forward-time simulator scenarios where very large (> 100 MB) data strings must be compressed and decompressed thousands of times per generation for thousands or millions of generations. Compression and decompression that require on the scale of minutes — as is the case for general lossless compression algorithms — is completely infeasible as a general solution. Ultimately, the constraint on memory is a major roadblock to the application of forward-time simulators to both complex biological structures and processes and large problem scopes.

As part of population regulomics, I develop novel methods for addressing the memory issue inherent in forward time simulations by compressing, in real-time, active and ancestral genotypes [45]. I propose general algorithms which can be implemented in any current simulator and are independent of the evolutionary model or genotype representation. Specifically, novel contributions are two-fold: the operation graph, a compression data structure for forward-time simulators, and Greedy-Load, an algorithm for improving the decompression performance of the operation graph by managing a strategic cache. The algorithms I present work equally to compress the whole ancestral information or just the active alleles of a simulation. Compressing the ancestral information of extant genotypes retains important information that would otherwise be lost without our compression algorithm.
Contributions of the thesis: Population regulomics

Population genetics provides a mathematical and computational framework for understanding and modeling evolutionary processes, and so it is vital for the investigation of biological systems. In its current state, molecular population genetics is exclusively focused on molecular sequences (e.g., DNA, RNA, or amino acids), where measures of diversity and estimates of population size and mutation rate work only on sequence data. Consequently, when technologies became available to sequence entire genomes, population genetic approaches were naturally applied to mine signatures of selection and conservation, resulting in the subfield of population genomics. Nearly every present genome project applies population genomic techniques to identify regions undergoing conservation or adaptive selection.

In this work, I develop a subfield of population genetics for gene regulation, called population regulomics, which includes simulators and analytic measurements that operate on populations of regulatory networks. To this end, I conducted extensive data analyses to connect the genome with the cis-regulome [41], developed computationally efficient simulators [45], and adapted population genetic measurements on sequence to the regulatory network. By connecting genomic information to cis-regulation, we may apply the wealth of knowledge at the genome level to observed patterns at the regulatory level with unknown evolutionary origin. I demonstrate that by applying population regulomics to the E. coli cis-regulatory network, for the first time we are able to quantify the evolutionary origins of topological patterns and reveal the surprising amount of neutral signal in the bacterial cis-regulome [46]. I also show
that diversity measures can be calculated on a population of CRNs, which allows for
the inference of mutation rates for interactions and the comparison of relative evolu-
tionary rates of change between genes, promoters and regulatory interactions. Since
regulatory networks play a central role in cellular functioning and, consequently, or-
ganismal fitness, this new sub-field of population regulomics promises to shed the light
of evolution on regulatory mechanisms and, more broadly, on the genetic mechanisms
underlying the various phenotypes.
Chapter 2

Model

In this chapter, I describe the CRN genotype used within the population genetic framework.

2.1 Population genetic framework

The model I employ follows a Wright-Fisher life cycle wherein mutation, drift, and selection take place in non-overlapping generations, as shown in Figure 2.1 Simulations model a constant-sized, haploid, panmictic population of $N$ cells, evolved over $G$ generations. We simulate a Wright-Fisher model: within each non-overlapping generation, existing individuals are mutated to form a mutant pool which is then randomly sampled to select surviving individuals for the next generation. We used a scaling parameter of $10^6$ to improve the computational efficiency of the simulations.

The cycle begins with a population of zygotes (with size $2N$) that produce an
Figure 2.1: The population genetic lifecycle can be modeled as non-overlapping generations in which mutation, recombination, natural select, and drift operate. infinite number of gametes. Individuals are considered to be haploids, but diploids can be constructed by pairing the population. Mating is assumed to be random throughout; however, intricacies of mating systems, especially with regard to plant and animal distinctions, can be considered in future studies. During the creation of the gametes (or, depending on the organism, throughout the life of a zygote), mutations occur in the germ line, and recombination occurs between chromatids in a diploid organism or through some other recombination mechanism (transformation, conjugation, or transduction). The proportion of gametes in this population will be skewed from random mating by some selection function since fitter individuals will contribute more gametes, or genetic material, in comparison to individuals with lower fitness. Lastly, random genetic drift occurs in the finite sampling of haploid individuals from the infinite gamete pool to determine the allelic frequency for the next generation. For the most part, I consider only binary selection, where an individual is either fit (survives) or unfit (dies).
This lifecycle is a framework because it operates independently of the specific genetic material propagating through the population — whether it is DNA, networks, or proteins. Therefore, from a computer science perspective, the lifecycle is orthogonal to the choice of genotype, or type of genetic material; only an adequate definition of mutation, recombination, or fitness for a particular genotype representation is required to insert it into this simulation framework.

2.2 CRN genotype

The CRN model presented in this thesis takes a different approach from previous work which represented CRNs exclusively as a network (e.g., [24, 23, 17]) by encoding CRNs within their sequence context. I developed a reasonable data structure which mirrors the underlying genetic sequence of CRNs and derived mutation rates for binding sites which are functions of binding site motifs — the specific nucleotide sequence to which TFs bind, represented as PWMs.

2.2.1 Representation

The CRN model consists of transcription units (TUs), transcription factors, and binding site motif information. At the sequence level, transcription units are co-transcribed coding regions which share flanking regulatory regions (see Fig. 2.2). Depending on the organism, these TUs could be operons, genes, or gene clusters. Binding sites may arise in flanking ncDNA regions upstream or downstream of the coding sequence. In eukaryotes, binding sites can arise within the gene in non-coding
Figure 2.2: A typical node in the CRN would correspond to co-transcribed genes or transcription units. Binding sites are denoted as short vertical rectangles surrounding the gene coding sequence. The coding sequence resides underneath the right angle arrows and ncDNA may be upstream (called the promoter) or downstream of the coding sequence. In eukaryotes, ncDNA within the gene coding sequence is called introns, which may harbor binding sites as well.

regions called introns. Taken together, the amount of regulatory substrate per gene can be very high in eukaryotes (up to 100,000 bp) or very small in bacteria (around 100 bp). In this model, I assume that all regulatory material surrounding a gene is equivalent in terms of regulatory potency; this is a very good assumption for bacteria and an open issue of research for eukaryotes.

For each TU, a non-zero ncDNA length that corresponds to the amount of base pairs which may harbor binding sites acting exclusively on the TU, and, if the TU encodes one or more TFs, the TFs encoded by the TU. For each transcription factor, binding site motifs are provided in IUPAC code. IUPAC code describes ambiguous sites in a sequence motif, where each IUPAC character may correspond to one, two, three, or four nucleotide combinations.

The regulatory region (call it promoter) is encoded as an array of locations, where each location may or may not be occupied by a binding site (see Fig. 2.3). The number of locations is equal to the base-pair length of the promoter provided as input, and the location of a binding site is interpreted as the distance from the transcription
**Figure 2.3:** CRNs are modeled based on their sequence characteristics, including the length of promoter sequence and the transcription units in the network. In this diagram, I mapped the CRN model to an *E. coli* example, such that regulatory regions are promoters and transcription units are operons. Using the sequence information, it is possible to generate the network representation of the CRN.

Start site to its center position. We allow binding sites to overlap but not share the same center position. Each location in a promoter is modeled independently and identically.

Given a list of TF binding motifs (here we use TF and the TU encoding the TF synonymously), I represented regulatory networks as a collection of promoter sequences, such that each location in the promoter was either empty (‘0’) or contained a binding site (non-zero index of the matching TF sequence motif). All binding sites are assumed to be *n* base pairs in length, such that they overlap with \(\lceil(n - 1)/2\rceil\) neighboring binding sites on each side. For instance, the average length of *E. coli* PWMs used in this study is 19, and so we used a 9 bp overlap with neighboring sites. The regulatory network, given the collection of promoter sequences, can be
constructed by adding a node for each TU and an incoming edge for each binding site to the corresponding TF.

2.2.2 Mutation

Given a CRN, I model mutation as follows. I assume that there are sufficient locations in the promoters such that in one round of mutation, gain and loss do not conflict, which is a safe assumption with the size of the network for which this model is designed (e.g., 70,963 promoter locations for E. coli). I calculate the probability of gain and loss of motifs given a base pair mutation in the binding site for each sequence motif provided to the simulator. Because I calculate the gain and loss rates given a base pair mutation, I begin the mutation process by calculating the number of base pair mutations in the promoter regions. For each base pair mutation, a random center location $x$ is chosen from all promoters, such that each location is equally likely. Then, for each site $y$, with length $n$, such that $x - (n - 1)/2 \leq y \leq (n - 1)/2 + 9$ and $y$ is in the bounds of the promoter, a binding site gain is calculated for each TF and, if $y$ corresponds to a location occupied by a binding site for TF $i$, a binding site loss is calculated for TF $i$.

Calculating binding site gain and loss rate from PWMs. For each TF sequence motif, provided in IUPAC code, I calculated their spontaneous gain and loss rates. Standard IUPAC Nucleotide code describes ambiguous sites in a sequence motif, where a single IUPAC character may represent more than one nucleotide. The gain rate is calculated as: given a base pair mutation in the sequence, what is the
probability that the sequence deviates from the PWM at one site and that the mutation mutates it to a sequence which matches its motif. Let $M$ be the IUPAC code for a TF motif of length $N$ and let $M[i]$, for $1 \leq i \leq N$, denote the character at position $i$ in the sequence $M$. We assume a simple model where all nucleotides and mutations between nucleotides are equally likely. We define a score function that returns the number of ambiguous nucleotides for each IUPAC character.

$$c(x) = \begin{cases} 
1, & x \in \{A,C,T,G\} \\
2, & x \in \{M,R,W,S,Y,K\} \\
3, & x \in \{B,D,H,V\} \\
4, & x \in \{N,\} 
\end{cases}$$

Provided the score of an IUPAC character $x$, the probability that one site matches it is $c(x)/4$ and the probability that the site does not match is $1 - c(x)/4$. The probability that the sequence deviates from the PWM at only the first site (in the equation, $M[1]$) is:

$$\frac{4 - c(M[1])}{4} \prod_{j \neq 1}^{N} \frac{c(M[j])}{4}$$

Then, provided we know a mutation occurs at the first site, the probability that it mutates to a nucleotide which matches the IUPAC character is $c(x)/3$. So, the probability that the sequence deviates at only one site and that that site mutates to
a matching nucleotide is:

$$\text{gain}(M) = \frac{1}{N} \sum_{i}^{N} \frac{c(M[i])}{3} \frac{4 - c(M[i])}{4} \prod_{j \neq i}^{N} c(M[j])/4.$$ 

The loss rate is calculated as follows: given a base pair mutation within a binding site, the probability that the base pair mutation dissolves agreement with the TF motif is:

$$\text{loss}(M) = \frac{1}{N} \sum_{i}^{N} \frac{4 - c(M[i])}{3}$$

Since we know the mutation occurs in the binding site, all base pairs match their corresponding IUPAC character. The probability that the mutation occurs at site $i$ is $1/N$, and the probability that the mutation causes the nucleotide to not match the IUPAC sequence character $x$ is $(4 - c(x))/3$.

Because both loss($\cdot$) and gain($\cdot$) are probabilities conditional on a base pair mutation, multiplying by the base pair mutation rate $u$ gives the individual loss and gain rates for each TF motif $M$: $u\text{loss}(M)$ and $u\text{gain}(M)$.

### 2.2.3 Simplifying assumptions

Although many evolutionary factors are simultaneously at play in shaping the topology of the regulatory network, I chose to focus on properties that were well studied with strong empirical support, accurately quantifiable, and altered regulatory interactions through clear mechanisms. Because our goal was to create a null model that provided quantitative, rather than qualitative, results, the ability to accurately
quantify rates was paramount. For this reason I constrained our model to sequence-level base pair mutation, high-confidence sequence annotation (e.g., promoters and operons in *E. coli*), and non-combinatorial transcription factors (i.e., no complexes). In addition, although I am simulating over time scales in which gene duplication and loss might occur, I only simulated the evolution of interactions while keeping the gene content unchanged. Similarly, this model assumes that the lengths of promoter regions are constant and that only binding sites within promoter regions change over time. The neutral processes which expand and contract promoters are known but difficult to quantify (e.g., the rate of transposon insertion and length). In essence, I assume that promoter length and coding DNA are under purifying selection, which is in-keeping with neutral evolutionary theory.
Chapter 3

Boosting forward-time population genetic simulators through genotype compression

Population regulomics, due to the complex and heterogeneous nature of CRNs, heavily leverages forward-time simulators. Because both the population context and the CRN model require substantial memory resources, it is actually not feasible with current computational tools. Therefore, to improve the memory performance of forward time simulators, I devised a novel compression and decompression algorithm specifically designed to greatly reduce the memory overhead of a simulation without impacting its speed [45]. This chapter covers the specifics of this compression algorithm and how forward time simulations can be used to test evolutionary hypotheses. I then assess the performance of the real-time genotype compression algorithm in a
variety of evolutionary scenarios.

3.1 Using population genetic simulations as a null model

Using population genetic forward-time simulations as a null model is standard practice in the domain (e.g., \cite{7,29,24}), most notable because running evolutionary studies is not feasible within the time constraints of standard experiments. So, armed with a reasonable forward-time simulator, scientists can ask evolutionary questions by constructing a specific scenario and understanding the evolution of a population within the confines of those conditions. Forward time simulators are therefore used to calculate the neutral evolutionary expectation, which is really a distribution of simulated results. A simple apparatus is diagrammed in Fig. 3.1 which explains how an experiment would use population genetic simulations.

Because simulations are identical and independent, the results from parallel simulation runs can be used as a statistical null model for empirical properties. For instance, simulators for DNA sequence have long been used to identify regions of the genome that have evolved differently than expected by neutral evolution provided the particular demographics, mating patterns, and migration of that population \cite{7}. Critical to applying this apparatus correctly is the choice of starting conditions, number of runs, and number of generations simulated. Indeed, as depicted in Fig. 3.1 running these simulations can be extremely taxing from a computational perspective, especially with respect to memory issues.
Figure 3.1: Forward time simulators are useful for generating statistical null models for complex evolutionary scenarios or genotypes. A standard apparatus for investigating neutral bacterial evolution is diagrammed in this figure. Large populations of initially identical individuals evolve over long periods of time to give rise to a distribution of genotypes expected by neutral evolution. Running multiple simulations and running longer simulations help combat start condition biases.

3.2 Efficient genotype compression

I propose a real-time compression algorithm for reducing the memory footprint of a forward-time population genetic simulation, composed of two components: the compression technique (operation graph) and the decompression accelerator (Greedy-Load). The operation graph represents each genotype by the sequence of evolutionary events that gave rise to it, and Greedy-Load maintains a “small” set of explicit genotypes that accelerates the decompression of compressed genotypes in the operation graph. Whenever the simulation or analysis requires access to the genotype information, genotypes can be retrieved on-the-fly by applying the evolutionary events to an explicitly represented genotype. I now describe the algorithm and data structure we use in detail, including the decision on which genotypes to represent explicitly, how to decompress a genotype, and how to build/augment the compression data structure.
Let’s begin with the compression technique, which we call the operation graph (see Fig. 3.2).

### 3.2.1 The operation graph

As evolutionary operations — such as mutation or recombination — occur in the population genetic simulation, the dependency of each operation on the previous genetic history is encoded in the operation graph (OG). Operations are stored as nodes in the OG, a directed acyclic graph (DAG) structure, where operations with
one incoming edge correspond to mutations and with two incoming edges correspond to recombinations. Each operation that arises over the course of the simulation is encoded as a distinct node in the OG, along with the genetic material produced by the operation.

Let \( F \) denote the set of evolutionary operations allowable in a simulation, and let \( G \) denote the set of genotypes that arise during a simulation. For mutational evolutionary events, each element \( op \in F \) is a function \( op : G \times \Phi \rightarrow G \), where \( op(A, \phi) = C \) denotes that genotype \( C \) is the result of applying evolutionary event \( op \) to genotype \( A \) with parameters \( \phi \). However, for recombination, \( op \in F \) is a function \( op : G \times G \times \Phi \rightarrow G \), where \( op(A, B, \phi) = C \) denotes that \( C \) is the result of a recombination event involving genotypes \( A \) and \( B \) with parameters \( \phi \).

For example, if we take \( \phi = \langle \text{base-pair-mutation}, 3, T \rangle \) and apply it to genotype \( A = ACCAAAT \), we obtain genotype \( C = ACTAAAT \), since the operation applied to \( A \) is a base-pair mutation that substitutes nucleotide \( T \) in the third position. Since different evolutionary events have different types of parameters, in addition to the “input” genotypes \( A \) and \( B \), we abuse notation, for the sake of simplicity, and use \( op \) as a function from \( G \) to \( G \) for mutation and \( G \times G \) to \( G \) for recombination—additional parameters \( \phi \) for applying \( op \) should be clear from the context.

The operation graph \((OG)\) is a rooted, labeled, weighted DAG \( OG = (V, E, \ell, f, w, c) \), where

1. \( V \) is the set of nodes;

2. \( E \subseteq V \times V \) is the set of edges;
3. \( \ell : V \rightarrow (G \cup \{nil\}) \) is the genotype labeling function with the constraint that 
\( \{v \in V : \ell(v) \neq \text{nil}\} \neq \emptyset \);

4. \( f : V \rightarrow \mathcal{F} \) is the operation labeling function;

5. \( w : V \rightarrow \mathbb{R} \) is the weight function such that \( w(v) \), for node \( v \in V \), is the frequency of the genotype \( \ell(v) \); and

6. \( c : V \rightarrow \mathbb{R} \) is the cost function such that \( c(v) \), for node \( v \in V \), is the non-negative computational cost of applying the operation \( f(v) \).

A node \( v \) is called \textit{explicit} if \( \ell(v) \neq \text{nil} \). That is, an explicit node corresponds to a genotype that is not compressed. For a node \( x \in V \), we denote by \( \text{Anc}(x) \subseteq V \) the set of all \textit{lowest} explicit nodes between \( x \) and the root of \( OG \), where a node \( y \) is lowest if no explicit node \( z \) (\( z \neq x \) and \( z \neq y \)) resides on a path between \( y \) and \( x \). In particular, if \( x \) is explicit, then \( \text{Anc}(x) = \{x\} \). The set of \textit{active} nodes in an \( OG \), denoted by \( A(OG) \), is all nodes whose corresponding genotypes have non-zero frequency; that is, \( A(OG) = \{v \in V : w(v) \neq 0\} \).

**Novelty of the operation graph.** The \( OG \) is a compression technique similar to LZ77 with edit operations and uses a structure similar to the Ancestral Recombination Graph (ARG), a phylogenetic structure that describes the evolutionary history of a set of genetic samples [48, 49, 50]. The LZ77 algorithms replace repeated occurrences of data with references to a single copy of that data existing earlier in the input data stream. In our case, instead of repeated occurrences, we replace “evolutionary related occurrences”, such that we keep track of homologous, rather than identical,
genotypes. For instance, if “ACCCT” evolved from “ACCGT”, only one instance is explicitly saved. Further, the operation graph is implicitly produced by forward time population genetic simulators, whether or not it is explicitly stored; whereas for LZ77, the identification of previous, similar strings is the bulk of the computational work in its implementation. Lastly, while LZ77 is a general compression scheme, the operation graph is biologically motivated, and in general, applies to scenarios where data evolves in a population, so that occurrences of data can be related to each other through evolution and this relatedness is used in the compression. For instance, it is not clear how LZ77 would handle the forking replacement dependencies incurred through processes like recombination.

While both the OG and ARG employ a DAG, the similarity between the two almost ends there. An ARG provides an explicit model of the evolution of a set of genetic sequences, mainly under point mutations and recombination [50]. The mutational model is often assumed to be the infinite sites, but more recent work has considered finite-site models as well [51]. On the contrary, the OG is an implicit representation of a set of related genetic information, where mutations and recombinations can be general (ranging from point mutations to insertions/deletions to genomic rearrangements). Further, while ARGs model the evolution of genetic sequences in a population setting, the OG is defined for arbitrary genotypes. A case in point is the analysis in this thesis of regulatory networks in *E. coli*, where the OG was defined over genotypes consisting of regulatory networks.
Updating the operation graph. Whenever a new genotype $C$ arises from existing genotypes $A$ and $B$ through a recombination operation $op$, the operation graph is updated by (1) adding a new node $u$ to $V$, (2) adding new edges $e_1 = (x,u)$ and $e_2 = (y,u)$ to $E$, where $x$ and $y$ are the nodes that correspond to genotypes $A$ and $B$, respectively, and (3) setting $f(u) = op$. In terms of $\ell(u)$, it can be set to $nil$ or to the new genotype $C$; we discuss below the choice we make in our algorithm. If the operation is a mutation, then only a single new edge is added in Step (2). The cost of $op$, or $c(u)$, can be set based on the type of operation (e.g., insertion, base mutation, deletion, recombination) or the input to the operation $\phi$. In the case of recombination, the ordering of the two parents is handled at the implementation level.

Whenever a genotype $A$ is lost from the population, the operation graph is updated only when the node $x$ that corresponds to genotype $A$ is a leaf node in $OG$. In this case, the algorithm identifies the set $Y$ where each node $y \in Y$ is the lowest node on a path from the root to $x$ that is either active, of out-degree 2, or the root of $OG$. Once node set $Y$ is identified, all nodes on the path from $y \in Y$ to $x$, excluding $y$, and all edges on that path, are deleted from $OG$. If $x$ is not a leaf node, no update is done, since some active genotypes may be “under” it.

Measures of the operation graph quality. Given the graph $OG$, the genotype in every node can be decompressed; that is, for every node $x$ with $\ell(x) = nil$, the explicit value of $\ell(x)$ can be computed by traversing the path, or paths, from $x$ to nodes in $Anc(x)$ and applying the corresponding operations. The decompression cost
for a given node \( x \), denoted by \( \text{cost}(x) \), is

\[
\text{cost}(x) = \sum_v c(v),
\]

where the sum is taken over all nodes that resides on paths between nodes in \( \text{Anc}(x) \) and \( x \). For a pair of nodes \( x \) and \( y \), where \( y \) is on the path from a node in \( \text{Anc}(x) \) to \( x \), we define the cost of decompressing node \( x \) by using information on the way from \( y \) to it, as \( \text{cost}(x, y) = \sum_v c(v) \), where \( v \) ranges over all nodes on the path from \( y \) to \( x \) (\( \text{cost}(x, y) = 0 \) if \( y \) is not on any path from a node in \( \text{Anc}(x) \) to \( x \)).

Further, the load of a node \( x \) (or, the corresponding genotype) is given by

\[
\text{load}(x) = \sum_{y \in U(x)} w(y) \cdot \text{cost}(y, x),
\]

where \( U(x) \) denotes the set of all nodes in \( OG \) that are under node \( x \) and require node \( x \) for decompression. Notice that for two operation graphs \( OG_1 \) and \( OG_2 \) whose underlying graphs are isomorphic and node labelings are identical, it may be the case that \( \text{cost}(x) \) based on \( OG_1 \) is different from \( \text{cost}(x) \) based on \( OG_2 \).

If we denote by \( C(V) = \{ v \in V : \ell(v) \neq \text{nil} \} \), which is the set of uncompressed genotypes, then no compression is achieved when \( C(V) = V \), and maximum compression is achieved when \( C(v) = \{ r \} \) for the root node \( r \) of graph \( OG \). The time it takes to access the explicit genotypes is effectively the time it takes to decompress all the compressed genotypes.
3.2.2 Compression algorithms

The set $C(V)$ of an operation graph $OG$ is at the core of the space-time trade-off here: the larger $C(V)$, the more space is consumed and the less time is required to access the explicit genotypes, and the smaller $C(V)$, the less space is consumed and the more time is required to access the explicit genotypes. Therefore, a central task here is to determine the set $C(V)$ that would minimize the load of an operation graph. Here, we describe several compression algorithms for this task, one which is the main contribution of this paper — Greedy Load — and the others which are used for performance comparison.

Greedy-Load

In Greedy-Load, the inputs, in addition to the operation graph $OG$, are $k$, which is a pre-specified bound on the desirable size of $C(V)$, and $t$, which is the number of generations elapsed between updates of the set $C(V)$. This algorithm assumes that $load(x)$ for all $x \in V$ is implicitly calculated and updated whenever the membership of $C(V)$ changes.

In a nutshell, Greedy-Load seeks to advance the set $C(V)$ towards the leaves and active alleles of the $OG$ by greedily caching genotypes with high levels of load. We define the utility function $advance(x)$ which maximally “advances” the decompression from $x$ towards the leaves of the $OG$:

1. let node $y \in U(x) \cup \{x\}$ be the highest node that is either:

(a) a leaf,
(b) has non-zero weight (frequency), or

(c) has at least two children each of which has non-zero load and is not in $C(V)$;

2. decompress the genotype corresponding to node $y$ and set $\ell(x) = nil$.

The Greedy-Load algorithm applies the following two steps on a given operation graph $OG$ every $t$ generations in the simulation (in the first application of this algorithm, we set $C(V) = \{r\}$). In the first step, nodes that are no longer needed for decompression — $load(x) = 0$ — are compressed, otherwise the decompression is advanced towards the leaves of the OG. In the second step, nodes are added to $C(V)$ by decompressing the max-load child of the max-load cached node.

1. For each node $x \in C(V)$:

   (a) if $load(x) = 0$ and $|C(V)| > 1$, set $\ell(x) = nil$, or

   (b) if $load(x) > 0$, perform $\text{advance}(x)$.

2. Add nodes to $C(V)$ until $|C(V)| = k$ or no other nodes may be added. Let node $x \in C(V)$ have maximum load in $C(V)$ and node $y$ be the max-load child of $x$, at each iteration

   (a) decompress the genotype corresponding to node $y$, and

   (b) perform $\text{advance}(y)$ and $\text{advance}(x)$. 
Figure 3.3: An example execution of Greedy-Load is illustrated on an abstract operation graph. Each node a-l represents a distinct genotype (or allele) and each edge depicts evolutionary descent by mutation (one parent) or recombination (two parents). The actual genotype representation could be a sequence or pathway. Genotypes may be compressed (○), cached (□) or active (=). The load of each genotype is depicted as the background color, with darker colors corresponding to greater load. Sequential steps taken by the Greedy-Load algorithm are illustrated from left to write, showing the incremental changes that update the set of uncompressed genotypes from \{a, c\} in Pane 1 to \{e, g, h, i\} in Pane 4. Dashed arrows within each step illustrate which genotypes are compressed and uncompressed. For instance, in Pane 2, d is uncompressed (cached) and a is compressed. For this example, the total number of cached genotypes k is 4.

Example execution. Assume an OG as illustrated in Fig. 3.3, composed of 12 operations labeled a to l connected by 12 edges. Node a is the root and nodes j, k, h, e, and l are leaves. All nodes are mutation operations except for d, which is a recombination operation with inputs b and c.

Panel 1 in Fig. 3.3 depicts the OG prior to the execution of Greedy-Load. All
leaves correspond to genotypes that are active in the population in addition to the internal node $i$. This example walks through the application of Greedy Load with $k = 4$.

In Panel 2, the first step of Greedy-Load ‘advances’ the decompression from $a$ towards the leaves. In this case, node $d$ has two children, $g$ and $h$, each of which has non-zero load and is compressed. Because node $c$ does not require $a$ for decompression, it is not in the set of nodes considered in $\text{advance}(a)$. Because node $c$ has two compressed children with non-zero load, it is not possible to advance the decompression from $c$ towards the leaves, so nothing is done.

In Panel 3, assume $\text{load}(c) > \text{load}(d)$ and $\text{load}(f) > \text{load}(e)$, so $f$ is decompressed and $\text{advance}(f)$ is performed, which results in decompressing $i$. Because $i$ corresponds to a genotype that is active in the population, $i$ may generate decompression requests, and so decompression cannot progress down the $OG$. In addition to $\text{advance}(f)$, $\text{advance}(c)$ is also performed, which results in the decompression of $e$ because $c$ has only one child with non-zero load.

In Panel 4, because $C(V) < 4$ and $\text{load}(d) > \text{load}(i)$, node $g$ is decompressed and $\text{advance}(g)$ and $\text{advance}(d)$ are performed. Because $g$ has two compressed children with non-zero load, decompression cannot be advanced further down the $OG$; however, because $d$ only has one compressed child with non-zero load (since $g$ is now decompressed), then $d$ is compressed and $h$ is decompressed. At this point, $C(V) = \{e, g, h, i\}$ and the application of Greedy-Load is complete.

In more realistic simulation scenarios, the $OG$ is both much wider and taller than presented in this simple example execution, so we visualized the execution of
Greedy-Load on more complicated OG topologies (see Supplementary Movie). In this animation, the evolution of the OG is visualized along with the set $C(V)$ for scenarios with low and high recombination rates.

### Other compression algorithms

In order to measure the performance of Greedy-Load, we defined two additional compression policies that make fast, but potentially poor (in terms of memory and execution speed), explicit representation decisions. Unlike Greedy-Load, these simple comparison compression algorithms or policies do not require knowledge of the entire operation graph to select the explicitly stored genotypes. Current simulators store active genotypes that arise during the course of a simulation; we refer to this policy as *Store-Active*. The alternative is to store only the root genotype(s) in the operation graph, which we call *Store-Root*. More formally, for an operation graph $OG = (V, E, \ell, w)$, we have:

- **Store-Active**: set $C(V) = A(OG)$.
- **Store-Root**: set $C(V) = \{r : r$ is a root node in $OG\}$.

### 3.2.3 Implementation

I implemented a population genetic simulator and the compression algorithms in C++, which can be used as a development library or a command line tool. It is important to note that I used explicit memory management, rather than *garbage collection*, for genotype data structures, so memory usage metrics are honest measurements of
allocated memory. The emphasis in this work is on the compression algorithm rather than the implementation of a memory-bounded forward-time population genetic simulator. I did not find any existing simulator with a software architecture that allows for integrating (without completely overhauling the implementation) a memory management policy, such as the ones we propose here: hence, our choice to implement our algorithms independently of existing simulators. However, we still provide a command line tool which, in addition to taking flexible input parameters, provides an example for how the compression techniques in this paper may be integrated into a pre-existing simulator.

To improve the performance of the population genetic simulation with a memory-managed genotype heap, I implemented both partial and batch decompression. In partial decompression, rather than uncompressing a 100,000 bp sequence to access only 10 bp, we implemented intelligent decompression which could retrieve randomly accessed locations without decompressing the entire sequence. Because each operation in the OG stores meta-data associated with its application (such as locations and mutations), we implemented operations such that they can be applied on the entire sequence or directly on a given index. In batch decompression, we implemented the population genetic simulator such that it reduces the data requests of a particular genotype. For instance, during a mutation event involving multiple base-pair changes, the genotype is uncompressed once and used repeatedly rather than uncompressed with each base pair change.

Because calculating load on the OG may be a costly exponential calculation, I tracked the number of data requests per operation as a proxy for load. For all oper-
ations in the OG, the number of data requests are set initially to zero and increment during the population genetic simulation. The number of data requests increments by one when the population genetic simulator requires the decompression of its corresponding genotype, which may occur during the calculation of a mutation event, recombination event, or fitness value. Data requests on compressed genotypes propagate up the OG to the most recent uncompressed operations. Consequently, genotypes with higher frequencies in the population will tend to generate more data requests than low frequency genotypes, and so we can use the number of data requests as a proxy for load. However, there may be operations with non-zero load but no data requests: for instance if during time period \( t \) an active genotype is not mutated or if partial decompression does not propagate to both parents of a recombination event. Therefore, I maintain a boolean flag that indicates if a particular operation is required for the decompression of some active genotype, which we use in place of ‘non-zero load’. It is important to note that the calculation of this boolean flag requires \( O(n) \), where \( n \) is the number of nodes which are required for the decompression of some active genotypes. Lastly, during the execution of Greedy-Load, the number of data requests for an operation may be reset (step 1) or decremented (step 2), accordingly.

To demonstrate that this approach is generally applicable to various choices of genotypes, I implemented two very different genotype models: a DNA sequence (represented by strings) and the CRN model presented in this thesis. In terms of memory allocation, a DNA sequence of length \( L \) occupies \( L \) bytes and a pathway of \( k \) genes occupies roughly \( k^2 \) bytes. For the DNA model, we implemented four evolutionary events (that is, operations in the set \( \mathcal{F} \)): point mutations (\( u \)), sequence insertions (\( u_i \)),
sequence deletions ($u_d$), and sequence crossover ($c$). Consequently, over the course of a simulation, the actual length of a DNA sequence may changes due to insertions and deletions. To our knowledge no other SNP-based compression techniques (FREGENE or SimuPop) handle length variation. For the pathway model, we implemented binding site loss ($u_l$) and gain ($u_g$), as described in the Model chapter.

I verified the execution of the simulator using the DNA sequence genotype by comparing the input mutation and recombination rates to the estimated mutation and recombination rates inferred by the output sequences. In addition, I verified the measured sequence polymorphism and diversity using the input population, sequence length, and mutation rate. All simulations were run on a MacPro with two 2.26 GHz Quad-Core Intel Xeon processors and 16 GB 1066 MHz DDR3 memory.

3.3 Compression in forward-time simulators

To evaluate the performance of the genotype compression algorithms—Greedy-Load and Store-Root—against the current memory management technique implemented in modern simulators, Store-Active, I ran population genetic simulations under a variety of scenarios. These scenarios were chosen to test the memory and time performance of each algorithm, measured in terms of mega bytes (MB) and seconds per generation, respectively. Except for the time scaling experiments below, the time and memory usage of each simulation were recorded after an initial burn-in period, which is a standard technique employed to remove start-condition biases. I also used scaled population, generation, mutation, and recombination parameters to increase
the time efficiency of the simulations [38]. The data compression ratio for a simulation is calculated as $\frac{\text{Compressed Size}}{\text{Uncompressed Size}} = \frac{k}{N}$, reported as a ratio, and the space savings is $1 - \frac{k}{N}$, reported as a percentage. Thus, a Greedy-Load representation that compresses a simulation from 100 MB to 5MB has a compression ratio of 1:20 (0.05) and space savings of 95%.

3.3.1 Time scaling

The goal of the compression algorithm is to constrain the memory footprint of a population-genetic simulation such that as simulation time increases, memory usage remains constant, which can be trivially achieved by swapping unconstrained memory and constant time for unconstrained time and constant memory. Indeed, if decompression decisions are poor, then the latter may be the case. We measured the scaling of time (seconds per generation) as a function of simulation time over 1000 generations; results are shown in Fig. 3.4.

For both the sequence and pathway genotypes, Store-Root exhibited log-linear (poor) scaling with respect to simulation time, whereas Greedy-Load showed constant execution time throughout the simulation. The sawtooth pattern of Greedy-Load results from the repetitive application (every $t$ generations) of the algorithm.

3.3.2 Parameterizing $k,t$ in Greedy-Load

Greedy-Load requires two parameters: $k$, the maximum number of explicitly represented genotypes (the set $C(V)$), and $t$, the number of elapsed generations between
applications of Greedy-Load on the operation graph. Although $k$ constrains the memory footprint used by the simulation, both $k$ and $t$ can have a combined effect on its speed, which calls for careful choice of their values. I ran multiple simulations across a dual parameter sweep of $k$ and $t$ under both mutation and recombination scenarios and recorded the average seconds per generation; results are shown in Fig. 3.5.

Under a mutation-only simulation, the speed performance of Greedy-Load improves by increasing $k$ and/or decreasing $t$. Except for low ($< 0.02$) compression ratios, Greedy-Load is ‘robust’ to $k$ and $t$ values in that performance does not significantly degrade across the parameter space. In contrast, under simulations which employed both recombination and mutation, a linear tradeoff exists between $k$ and $t$: as $k$ increases $t$ should increase as well. Because recombination introduces significant complexity to the OG topology — in fact, under mutation the OG is a tree — compression levels achieved by performant recombination simulations are near an order
Figure 3.5: The tradeoff between Greedy-Load parameters $k$ and $t$ are presented as a heatmap of average execution times (log sec/gen), with a mutation scenario on the left and recombination on the right. Lighter colors are faster (better) simulations. The parameter $k$ is given as the compression ratio ($k/N$), where $N = 10^3$ is constant in all the simulations.

of magnitude less than the compression levels for mutation scenarios.

3.3.3 Space/Time performance of policies

In this experiment, I measured the performance of each compression algorithm in terms of both time, reported as the average seconds per generation, and space, reported in MB used by the genotype heap. The memory footprint is dominated by the explicitly represented genotypes, but also counts the operations stored in the operation graph, which account for less than 0.1% of the total reported memory for all policies except Store-Root.

Time and space values were averaged over multiple simulations for both sequence and pathway genotype models. The results for both sequence and genotype models
are shown in Fig. 3.6, and depict similar performance patterns despite drastically
different underlying representations.

I compared the performance of Greedy-Load to uncompressed (Store-Active) and
maximum compression (Store-Root) bounds for varying genotype sizes. As the size
of the genotype increases, the space used by the simulation increases as well; how-
ever, this quantity is dependent on the level of compression. In the case of the
upper bound, no compression is imposed (Store-Active). The lower bound has max-
imum compression — only storing one genotype, at a compression rate of 1:N or
1:1,000 (Store-Root). Greedy-Load provides a ‘performance knob’ between these two
bounds, allowing for high levels of lossless compression without imposing significant
time penalties. For the upper and lower bounds on compression, certain genotype
sizes failed to complete for either space (upper bound) or time (lower bound) limita-
tions. Sequences ranging logarithmically in size from $10^5$ to $10^7$ bp were simulated at
95% compression. Pathways ranging in size from 100 to 1,000 genes were simulated
at 90% compression. For both genotype representations, Greedy-Load performed at
competitive levels of space and time in comparison to the upper and lower bounds
and completed simulations otherwise intractable to Store-Active and Store-Root.

3.3.4 Greedy-Load performance in high recombination rate
simulations

Recombination introduces multiple inheritance to the operation graph and so
presents a unique challenge beyond a mutation-only model. Further, the rate of
Figure 3.6: The average time per generation and memory usage required by each compression algorithm for replicate simulations of sequences (left) and pathways (right). For both time – measured in seconds per generation – and space – measured in total MB – lower values are better. The performance is measured across a range of genotype sizes: $10^5 - 10^7$ bases for sequences and $100 - 1,000$ genes for pathways. Larger genotypes require more space and longer execution times, hence a diagonal line in the space-time tradeoff. Solid lines connect a compression policy – top to bottom: Store-Active, Greedy-Load, Store-Root — and dashed lines connect genotype sizes (e.g., $10^5$ nt for each policy). Greedy-Load provides 95% compression for sequences and 90% for pathways.

recombination directly relates to the amount of genotypes with multiple inheritance — or complexity of the operation graph topology. Consequently, the performance of Greedy-Load may be sensitive to the rate of recombination in a simulation.

In this experiment, I measured the performance of Greedy-Load across a range of compression rates with respect to $c/u$, the ratio of per-base pair recombination over the mutation rate, by running a logarithmic parameter sweep of $c/u$ from $10^{-2}$ to 10 (Fig. 3.7). The mutation rate $u$ is held constant at $10^{-4}$ and $c$ is determined from the sweep parameter. The population size is $10^3$ and the sequence length is $10^4$.

Because the complexity of the operation graph increases with respect to recombi-
Figure 3.7: The speed, measured in sec/gen, is plotted for Greedy-Load simulations across varying levels of compression (y-axis) and recombination (x-axis). Lighter colors are slower simulations, displayed in log-scale.

 nation rate — moving right on the x-axis in Fig. 3.7 — higher recombination rates require higher compression ratios (lower space savings). In fact, a phase shift exists in terms of execution time between sufficient and insufficient explicit genotypes (k, or compression ratio) for a given recombination rate. This decision boundary imposes limitations on the level of compression supported by Greedy-Load for high levels of recombination (c >> u). Although Greedy-Load performs correctly at any compression rate, execution time is potentially sacrificed for memory-savings.
3.3.5 Imposing a memory ceiling using Greedy-Load

Imposing a memory ceiling constrains memory potentially at the cost of time. To investigate this tradeoff, I measured the ability of 100 MB memory-constrained simulations to handle genotypes of growing size. Sequences were scaled logarithmically from $10^5$ to $10^7$ nucleotides, where it is possible to calculate the maximum number of explicit genotypes with $k = \left\lfloor \frac{100}{(\text{MB/genotype})} \right\rfloor$, with (MB/genotype) being roughly $L/10^6$ for sequences. The execution speed for simulations under 100 MB memory constraints are shown in Fig. 3.8 along with the maximum number of explicit genotypes, $k$, for each genotype size.

The execution time scales log-linearly with respect to the size of the data, showing that even for low $k$ values, Greedy-Load performs consistently with the size of the
genotype representation and does not perform arbitrarily poorly when $k$ is low or genotypes are large. Although 100 MB is a threshold chosen primarily for demonstrative purposes, this experiment highlights the ability of Greedy-Load to threshold memory usage and prevent unexpected program crashes due to memory limitations.

### 3.3.6 Simulating big data

I simulated a population of 1000 individuals each with 50 Mb DNA sequence using base pair mutation ($u = 10^{-4}$), sequence insertion and deletion for 1000 generations. These parameters leveraged a scaling factor of $10^5$, so, in effect, an equivalent population of $10^8$ for $10^8$ generations with a base pair mutation rate of $10^{-9}$ was simulated. The Greedy-Load algorithm with parameters $k = 50$ (95% compression) and $t = 0$ managed the compression. This simulation completed successfully, using around 1.6 GB of memory and on average 20 sec/gen (see Fig. 3.8).

According to recent reviews, no forward-time population genetic simulator can handle this computationally demanding, yet biologically reasonable, parameter set \cite{38, 39}.  

\[ \text{[38, 39]}. \]
Chapter 4

Binding site accumulation and topological features correlate with ncDNA

Central to the CRN model presented in this work is that the amount of ncDNA upstream of a gene has a strong effect on the number of binding sites which govern its regulatory interactions — and so also the network topology. Indeed, although this connection has been posited in previous work (e.g., [24]), no clear relationship in data has been presented to corroborate the phenomenon. To this end, I analyzed binding sites upstream of identified transcription units in five diverse model organisms to investigate the correlation between the length of upstream intergenic regions and the accumulation of binding sites [41]. Then focusing solely on the E. coli CRN, I examined in more detail the enrichment of intragenomic variability ncDNA length
with respect to topological patterns like degree and specific subgraph topologies [46].

I curated data from five data warehouses for *E. coli* [52], *S. cerevisiae* [53], *C. elegans* [54], *D. melanogaster* [55], and *A. thaliana* [56], representing binding site data for at least 28 TFs per species. Only binding sites residing in the upstream intergenic region and 5' untranslated region (UTR) were used; downstream or intronic binding sites with putative effect were not counted in this study. In *E. coli*, each operon was treated as an independent transcription unit (TU), whereas in the eukaryotic genomes I treated each set of overlapping genes as a single TU. When possible, I used experimentally verified, rather than computationally predicted, binding sites. All binding site data for *D. melanogaster* and *C. elegans* contain only experimentally verified sites, *E. coli* and *S. cerevisiae* contain both experimentally verified and predicted sites, and *A. thaliana* is comprised largely of predicted binding sites. In this regard, I minimize the potential effect of false positives in large ncDNA regions by curating mostly experimentally verified sites. I include *A. thaliana* for enhancing the completeness of the study. I matched binding sites with their closest downstream target, similar to the method in [22], and recorded the length of upstream intergenic DNA region as the amount of regulatory substrate available to the particular TU. In the case where forward and reverse strand genes share the same upstream region, binding sites were matched to both genes.
4.1 Distribution of TFBS

I calculated the distribution of binding sites as a function of distance from the TSS. For a sliding window, the TFBS density was calculated as the number of TFBS within the window normalized by: (1) the window size, (2) number of promoters which at least the length of the extent of the window, and (3) the total number of TFs. The results are presented in Fig 4.1. The distribution of TFBS density is non-uniform for all organisms, following an exponential distribution: binding sites are more likely to occur closer to the TSS, which is corroborated by previous work [57]. TFBS density does not diminish to 0, but rather plateaus around $5 \times 10^{-5}$ #TFBS/nt.

In addition, the distribution of binding sites per TF is heterogeneous. I calculated the number of binding sites in a given window upstream of the gene for each TF and presented the results in Fig 4.1.

4.2 Distribution of TFBS length

The length of binding sites varies between both prokaryotes and eukaryotes and between transcription factors. To better understand this distribution, I downloaded the distribution of binding sites for *E. coli* from RegulonDB and several eukaryotes (*D. melanogaster*, *C. elegans*, and *S. cerevisiae*) from JASPAR (shown in Fig 4.2) [58]. Because we parameterize our simulations with the length of a binding site, we are interested in determining the ‘effective’ binding site length which is somewhere between the number of consensus sites and the total length of the binding site motif. I calculated the average consensus sites and binding site length for *E. coli* and select
Figure 4.1: **Left.** The TFBS density (#TFBS/nt) was calculated for a sliding window along upstream ncDNA regions of TUs. TFBS density is normalized by the number of TUs and TFs in the study. **Right.** The distribution of TFBS broken out by TF is plotted for each organism. The distribution of all TFs is denoted by a solid black line.
eukaryotes in this study and find the effective binding site to exist between 10-20 bp for *E. coli* and 5-10 bp for eukaryotes.

### 4.3 TFBS accumulation positively correlates with the amount of ncDNA

Across an entire genome, binding site accumulation correlates with the size of the intergenic region, although this correlation may be as low as 0.23 and as high as 0.89 (see Fig 4.3b). Because binding site formation is subject to many diverse, heterogeneous factors (e.g. binding site length, TF promiscuity, conservation) which vary between species, I do not expect this correlation to be high within the genome.
of an organism or consistent across organisms; however, it’s important to note that the general trend — longer ncDNA upstream regions tend to harbor more binding sites — is indeed supported by empirical evidence. Despite the lower number of TFs sampled in *C. elegans* and *D. melanogaster* in comparison to the other organisms, the distribution of TFBS count per TU matches that of *S. cerevisiae* and *E. coli*, following a scale-free pattern (see Fig 4.3a). The frequency of intergenic regions harboring a given amount of binding sites log linearly decreases as the number of binding sites increase. I observe this pattern regardless of the number of transcription factors, the number of intergenic regions, or the length of the intergenic regions.

The data repository for binding sites in *A. thaliana* constrained upstream intergenic regions to be at most 3,000 nt, although larger upstream regions exist in the genome. I suspect that this truncation of intergenic regions in the *A. thaliana* data repository skews the distribution of TFBS count/TU from that of the other organisms and improves the correlation between TFBS count and ncDNA length (because ncDNA close to transcription start sites have higher TFBS density). In addition, because the binding site data for the other organisms is largely experimentally verified, I suspect that the strong correlation in *A. thaliana* may be due to a systematic false positive rate in the identification of its binding sites.
Figure 4.3: The distribution of binding site width (right) and consensus sites (right) for *E. coli* (top) and several eukaryotes (bottom). The solid vertical lines display the mean length depicted in each plot.

### 4.4 Regulatory complexity positively correlates with the amount of ncDNA

As the amount of ncDNA correlates with the accumulation of TFBSs, and that latter governs the emergence of regulatory interactions, a question naturally arises as to the complexity patterns that emerge in regulatory networks as a function of the amount of ncDNA. To investigate this question, I considered four properties of regulatory pathways: for each TU, the average number of upstream binding sites (I
refer to it as ‘#TFBS’), the average number of binding sites per TF that regulates the TU (I call it ‘multiplicity’), the number of unique TFs that regulate the TU (I call it ‘degree’), and the fraction of TUs with more than one regulating TF (I call it ‘redundancy’).

To account for the varying amount of TFs for each organism, each reported property was scaled according to the contribution of each TF (by dividing by the number of TFs per organism in this study). I present these values as a function of upstream ncDNA in Fig 4.3c, and find strong positive correlations between the amount of ncDNA upstream of TUs and the regulatory properties under study (shown in Fig. 4.3c). Although these properties are derived from binding site accumulation, their values are not completely determined by the number of binding sites. For instance, binding sites accumulation may increase degree without affecting multiplicity. Similarly, if unique binding sites accumulate, then multiplicity may increase without affecting degree.

As the amount of ncDNA per TU increases, an arbitrary TF would increase TFBS and multiplicity by binding to more sites upstream of a gene, degree by interacting with more genes, and redundancy by redundantly regulating more genes.

### 4.5 Enrichment of promoter length in *E. coli* CRN topology

I compiled the *E. coli* regulatory network using data readily available from RegulonDB [52]. Nodes represent operons and directed edges represent regulatory interactions between operons, where the source operon must encode a TF. Operons which
encoded one part of a heterodimer TF (e.g., IHF) were merged together as one node in the network to avoid representation issues. The length of promoter sequences was measured by the amount of contiguous ncDNA which may harbor functional binding sites upstream of an operon, which was determined per operon by the maximum distance from transcription start site for all identified functioning binding sites. I only included operons that had a clear contiguous upstream region and previously identified binding sites. TF sequence motifs are also available from RegulonDB for 50 TFs, provided in IUPAC format \[59\]. I only included interactions in the regulatory network for which there existed a corresponding PWM.

Each node (operon) in the *E. coli* regulatory network is annotated with the length of its promoter region and the sequence motifs for any TFs encoded by the operon, where a node with outgoing edges corresponds to a TF-encoding operon. With this annotated network, I investigated correlations between genomic and network level properties (Figure 4.4A and Figure 4.5). I found that only promoter length correlated with in-degree (Pearson correlation coefficient $r = 0.48$), but otherwise the loss and gain rates of TFs poorly correlated with out-degree and with each other.

To further understand the role of promoter length in the *E. coli* network, I enumerated the operons that participated in subgraphs that have been analyzed and studied extensively for their functional roles: feed-forward loop (FFL), single input module (SIM), and bi-fan. For each of these three subgraph types, I calculated the distribution of promoter length at each node in the subgraph (Figure 4.4B for FFL, and Figure 4.6 for SIM and bi-fan). Although operons may arise multiple times in the enumeration of subgraphs (e.g., the regulator in the SIM), I only count each operon
Figure 4.4: Significant promoter length signal exists in the *E. coli* regulatory network on both the system and subsystem level. In A, each operon is plotted with its in-degree and promoter length; we report the Pearson correlation coefficient and significance above the plot. In B, the distribution of promoter length for operons which participate in feed forward loops (FFL) are presented as boxplots per node. The left axis provides the node label, which corresponds to the node in the subgraph diagram C (e.g., 0 or 1), along with the number of distinct operons represented in that distribution. The difference between the average promoter length in the node distribution minus the average promoter length in the network is listed on the right axis of B. Distributions with significant uplift, assessed using a non-parametric Wilcoxon ranksums test, are indicated with a gray background behind the boxplot (p-values for FFL0 = 4 × 10^{-11}, FFL1 = 5 × 10^{-4}).
Figure 4.5: Relevant distributions for genomic- and network-level properties of the E. coli regulatory network are presented. The top row provides information on distributions related to promoter length and the bottom row plots binding site gain and loss rate for each operon against their out-degree. The top left plot compares the distribution of promoter length of operons which encode a TF and do not encode a TF ("non-TF"). The number of operons represented in each distribution are listed in the parentheses, with all operons in the regulatory network accounted for. Although TF encoding operons have an elevated average and median promoter length, it is not significant (Mann-Whitney non-parametric test, p-value = 0.18). In the top right, each operon is plotted with its in-degree and promoter length; we report the Pearson correlation coefficient and significance above the plot. In the bottom plots, the out-degree of TF-encoding operons is plotted against their spontaneous gain rate (left) and loss rate (left), as computed from their position weight matrices.

Downstream nodes in the subgraph (that is, nodes with in-degree greater than 0) tended to be significantly overrepresented by operons with longer promoter regions (using Wilcoxon rank-sums test against all 545 operons). It is important to note that the operons which encode TFs, and would naturally be upstream in the motif, tend to have longer promoter regions than non-TF encoding operons, although not signif-
Figure 4.6: The distribution of promoter length for operons which participate in single input modules (SIM), feed forward loops (FFL), and bi-fans are presented as boxplots per node. The left axis provides the node label, which corresponds to the node in the subgraph diagram (e.g., 0 or 1), along with the number of distinct operons represented in that distribution. The difference between the average promoter length in the node distribution minus the average promoter length in the network is listed on the right axis. Distributions with significant uplift, assessed using a non-parametric Wilcoxon ranksums test, are indicated with a gray background behind the boxplot. The p-values for these distributions are: FFL0 = 4 × 10^{-11}, FFL1 = 5 × 10^{-4}, Bi-fan0 = 9 × 10^{-10}, Bi-fan1 = 8 × 10^{-11}.

icantly (Mann-Whitney non-parametric test, p-value= 0.18). Consequently, the fact that downstream genes have longer promoter regions is not due to a predetermined bias. In the feed-forward loop, the operons that share a common regulator (nodes 0 and 1 in Figure 4.4C) tend to have longer promoter regions than the common regulator. The downstream nodes in the FFL have promoter regions that are, on average, 73 and 81 bp longer than the overall average. Similarly, the nodes that share common regulators in the bi-fan (nodes 0 and 1) also tend to have longer promoter regions than the regulators, on average 67 and 62 bp longer. However, the single input module is not enriched for longer promoters at the downstream nodes, due to the fact that nearly all operons (498 of the 545) participate in this pattern.
Chapter 5

Quantifying the role of neutral and adaptive forces in driving regulatory complexity and topology

In the previous chapter, I showed that inter-genomic and intra-genomic variability in the amount of ncDNA correlates with important regulatory and topological features. In this chapter, I leverage population genetic simulations to quantify the role that neutral evolution plays in crafting certain regulatory properties and topologies based on the inter-genomic and intra-genomic variability of ncDNA [46].
5.1 ncDNA drives regulatory network complexity

With the average ncDNA per TU and binding site length, it’s possible to examine the average regulatory complexity expected under neutral forces. While the specific PWM for the TFs are not known between these species, I parameterized the binding site length, $n$, with a value somewhere between the number of sites which must match exactly (consensus sites) and the total length of the binding site motif. To account for this, I previously calculated the average number of consensus sites and binding site motif length for *E. coli* and the eukaryotes in the study. I found that reasonable values of $n$ for *E. coli* are in the range 10-20 bp and in the range 5-10 bp for eukaryotes.

I simulated the evolution of a 5-gene pathway, with each gene as its own TU, in a population of size $N$ for $10^9$ generations across values of ncDNA ($L$) ranging from $10^2$ to $10^4$ and binding site size ($n$) ranging from 10 – 20 (prokaryotes) and 6 – 10 (eukaryotes), under models of binding site mutation (gain/loss) and genetic drift.

I set the population size $N$ as a function of $L$, based on the empirical correlation between genome size and population size—as genomes increase in length, populations tend to shrink—such that $N = 10^2(uL)^{-1}$, where the base pair mutation rate $u$ remained constant at $10^{-9}$ for all simulations [60, 61].

For each $N$, amount of ncDNA $L$, and binding site length $n$, results were averaged over 80 replicate stochastic simulations. The results (Fig. 5.1a) show that the four properties under investigation log-linearly increase with respect to the amount of ncDNA, following the pattern found in the empirical data. The binding site length has an effect on the slope and position of this increase: shorter binding sites sponta-
neously arise more frequently than longer ones, leading to higher values of binding site accumulation, degree, multiplicity and redundancy. The binding site length (8 bp) that elicits a similar pattern to the empirical data is the length between the average number of consensus sites (6 bp) and the length (10 bp) for eukaryotes. In comparison, a binding site length of 10 bp shows minimal response to increasing amounts of ncDNA and a length of 6 bp ‘saturates’ the degree, multiplicity, and redundancy values of the simulated pathway. For *E. coli*, I performed further simulations to understand the effect of increasing the binding site size to 20 bp when \( L \approx 100 \). Under these simulations, I found a minimal response in regulatory complexity to binding site lengths beyond 10 bp (see Fig. 5.2). Because we implement a viability constraint that all genes must remain regulated, the effect of increasing the binding site length, at some point, is masked by the inability of pathways to lose any more binding sites. We find that this point, under our simulation settings, is around 10 bp for short upstream ncDNA regions.

In [24] it was shown that redundancy increases with increasing amounts of ncDNA; however, my results show that this is a result of increasing binding site accumulation which also drives degree and redundancy. Large amounts of ncDNA in the genome result in higher complexity of the regulatory pathways in the population. Thus, the log-linear increase of the binding site accumulation previously identified in the empirical data is explainable by mutation and genetic drift.
Figure 5.1: a. Simulations using a 5-gene pathway under mutation and drift yield log-linear increases in the number of binding sites (#TFBS), degree, multiplicity, and redundancy with respect to the amount of ncDNA per transcription unit. Each line corresponds to a different binding site length, given in the legend. b. The results from running population genetic simulations on a random fitness landscape (y-axis) compared with the results from neutral - mutation and drift - simulations (x-axis). Deviations from neutrality are points off the diagonal. c. The pathway properties respond to the contributing factors, which go from insignificant ($s = 10^{-6}$) to dominant ($s = 1$). The x-axis denotes the strength of selection in a population, or $N_s$. The rows denote the property under selection and the columns represent the high gain and high loss environments, respectively. The optimal value was chosen against the neutral bias and is denoted by a dashed line for degree. In the case of redundancy, the optimal value was 0 (high gain) and 1 (high loss). Shapes denote the regulatory complexity measurements of #TFBS (□), degree (○), multiplicity (×), and redundancy (△).

Figure 5.2: Results for simulations with $N = 10^9$ and $L = 100$ for varying binding site sizes.
5.1.1 Selection and TFBS accumulation

While simulation results thus far confirm an essential role of neutral evolutionary forces in shaping TFBS accumulation, they do not rule out the role of adaptive forces. To investigate whether selection aids in explaining the observed trends of TFBS accumulation, I modeled fitness in two ways: acting on the regulatory pathway function via an abstract phenotype fitness function (see Methods), and acting on the regulatory pathway complexity via pathway property fitness functions properties using properties defined above.

In employing the abstract phenotype fitness function, an initially monomorphic population evolves on a soft fitness – meaning the fitness function is not binary – landscape where each phenotype class is assigned a random selection coefficient. A phenotype class is defined as a vector of the ‘up’ or ‘down’ regulation of each gene in steady state [23] (see Methods). The population evolves on this landscape for $10^9$ generations. The results from these simulations are given in Fig. 5.1.

Although the fitness is based on the pathway steady state dynamics, there are no significant deviations from the patterns found in non-adaptive simulations for the number of binding sites and degree across all binding site and upstream ncDNA sizes. For a 10 bp binding size length, selection on steady state dynamics tended to increase the frequency of multiplicity and redundancy in the pathway across all sizes of ncDNA over the results found by neutral simulations. In sparse pathways, loss of a binding site often leads to loss of function — so selecting for a certain steady state dynamics would enhance this effect over the viability constraint imposed in the neutral simulations. Thus, in sparse networks with long binding sites, one would expect to
find signal for conserved binding sites. On the other side of the spectrum, when networks are not sparse, and the spontaneous gain rate of binding sites is relatively equivalent to their loss rate, selecting on a specific pathway function does not affect regulatory complexity. Therefore, constraining a population to evolve within certain pathway dynamics restrictions (as imposed by the fitness landscape) has minimal effect on binding site accumulation beyond the neutral bias, but may have an affect on multiplicity and redundancy in prokaryotic genomes. So, the general application of selection on pathway dynamics does not seem to affect binding site accumulation on the genome scale.

If selection acted not entirely on pathway function, but instead to some extent on complexity itself, then we can model such a relationship with a selection coefficient directly on a topological pathway property associated with complexity. In this experiment, the fitness of a pathway genotype is a function of its redundancy and average degree. Here, I ran simulations on both ends of the ncDNA amount spectrum (ncDNA amount = 100 and ncDNA amount = 10^6) to understand the effect of selection operating directly on pathway properties, using the pathway properties fitness model. Unlike the biologically realistic rates and population sizes used in the previous sections, s—the contribution factor of the complexity property—cannot be readily estimated from biological evidence. So, in these experiments, we performed a logarithmic parameter sweep of s from insignificant (s = 10^{-6}) to dominant (s = 1). The results of this experiment are shown in Fig. 5.1c.

The strength of selection is relative to the population size, where the magnitude of genetic drift overpowers selection when s < 1/N or Ns < 1. In Fig. 5.1c, there is a
noticeable phase shift when \( Ns > 1 \) in the left column (ncDNA= \( 10^6 \) and \( N = 100 \)). For the right column (ncDNA= \( 10^2 \) and \( N = 10^6 \)), the minimum value of \( s \) is \( 10^{-6} \), which is equivalent to \( 1/N \), so selection overpowers genetic drift.

Unlike the experiments above, where fitness had no significant effect on pathway properties, direct selection on pathway properties can force deviations from the pattern caused by mutation and drift. However, selection on average degree effects the other complexity properties differently than selection on redundancy. When average degree is under selection, all other properties respond in accordance by covarying with the change in average degree. On the other hand, when redundancy is under selection, the other properties retain the neutral bias.

These experiments demonstrate that specific pathway properties can be driven by adaptive forces when selection operates as a function of that property; and that the signature of selection on primary or derived complexity properties are different. We may expect that if selection were to operate on complexity, it would be on a derived property, like redundancy or multiplicity.

5.2 Building an evolutionary null (neutral) model for the *E. coli* CRN

At this point, I have investigated the role of neutral evolution on driving regulatory complexity across diverse genomes; now, in this section, I zoom into a single genome in order to understand the role that intra-genomic variability of ncDNA region length may have on promoting or inhibiting the development of regulatory network topology.
I use the intra-genomic variability within the *E. coli* CRN to predict the network topology under neutral expectations alone. I leverage population genetic simulations on the entire *E. coli* CRN — made possible by the Greedy-Load algorithm — to build a distribution of null (neutral) CRNs. Armed with this null model, it becomes possible to quantify the likelihood of certain subgraphs and topological features arising from neutral evolution acting on ncDNA (without invoking adaptation).

Using the *E. coli* operon regulatory network with the contents of promoters expunged, I generated initial random networks with a minimal binding site set, where each promoter contains only one binding site. I used initial conditions similar to those in [24], such that TF-encoding operons are auto-regulatory and non-TF encoding operons are regulated by a randomly chosen TF. This initial random network seeds a clonal population of $10^9$ cells which evolves for $5 \times 10^{10}$ generations. I observed that at about $10^{10}$ generations, the number of edges in the simulated network plateaued. The specific value at which the number of edges plateaus is governed by the total amount of ncDNA represented in the promoters, the relative binding site loss and gain rates, and stochastic forces. At the end of each simulation, I take the CRN that occurs with highest frequency in the extant population as the overall result for that simulation. I performed 1,000 replicate simulations to develop the null distribution of 1,000 regulatory networks according to the evolutionary model.

I studied important properties of the *E. coli* regulatory network at the system (regulatory network), subsystem (regulatory patterns, or subgraphs), and operon level. To obtain statistically significant results, our analyses are based on 1,000 random networks generated by our model.
All the initial networks which seeded the evolutionary simulations begin with 545 edges whereas the real E. coli network has 1,039. The number of nodes (545) in both the random and actual networks are identical. The clustering coefficient for all initial networks, because all TFs are auto-regulatory, is 0. The only subgraph present in the initial network is a single input module, because TFs regulate themselves and other random non-TF encoding operons. Each operon has an in-degree of 1, regardless of its promoter length. Therefore, any topological signal at the system, subgraph, and operon level occurs during the course of the evolutionary simulations.

To validate the genotype model and evolutionary simulation settings, I compared the expected number of edges in the networks generated by our evolutionary model with the number of edges in the E. coli network. For the 545 operons represented in the E. coli network, the actual number of interactions is 1,039, which is within two standard deviations (z-score=1.73, p-value=.04, N=1,000) of the 989.5 interactions expected based on the model. Because the model does not take into account many variables and processes that affect the evolution of interactions, it is reasonable to believe that the null distribution generated by our model matches the E. coli network precisely; nonetheless, the low z-score shows that the sequence-level parameters provided to the model may explain a substantial portion of the network topology.

5.2.1 System level properties

Two properties that are often investigated at the system, or network, level are the degree distribution and clustering coefficient of the network. Figure 5.3 shows the in-degree and out-degree distributions of the actual E. coli network and the networks
Figure 5.3: The in-degree and out-degree distributions for the actual *E. coli* network (A) and the random networks (B) are compared side by side. Scatter points indicate the 12 logarithmic bins used to plot the lines. The discontinuity in the out-degree line in the random plot indicates that there are no nodes in the bin with a degree of around 100. We used a two sample K-S test and found that all degree distributions did not differ significantly between the actual and random distributions.

generated based on the CRN model.

As the figure demonstrates, the in-degree and out-degree distributions of the actual *E. coli* network match the distributions found by the model (Kolmogorov-Smirnov test; in-degree: $D=0.029$, $p$-value=0.75; out-degree: $D=0.031$, $p$-value=0.65). Thus, my model provides an explanation to both degree distributions observed in the *E. coli* network.

Clustering coefficient is a graph-theoretic measure of the *transitivity* of the network. I compared the clustering coefficient of the actual network to the distribution expected by our model, and found a strong agreement between the two. Specifically, the actual network has a clustering coefficient of 0.189 and the distribution based on our model has mean of 0.162 and variance of 0.05 ($z$-score of 0.526; $p$-value of 0.3).

I also compared the discretized joint distribution of in-degree and promoter length
by subtracting the distribution under our model from that of the actual \textit{E. coli} network. The in-degree of an operon is the number of unique regulatory interactions (equivalently, the number of distinct binding sites and their affinities in the promoter). I find that the distribution under our model accounts for about 91\% of the interactions present in the actual network, leaving only 9\% of the interactions to fall outside the model.

\subsection{Subgraph level topologies}

Network \textit{motifs} are subgraphs that are significantly overrepresented in the actual network when compared to networks generated under a null model. In a seminal study [11], Alon and colleagues reported on motifs and their distribution in the \textit{E. coli} regulatory network. For their null model, the authors rewired the actual network randomly, while maintaining the in-degree and out-degree distribution, to obtain random networks. To identify certain subgraphs as motifs, the frequency of each subgraph (up to a certain subgraph size) in the actual network is compared to the mean and variance subgraph frequency found in random networks, resulting in a \textit{z}-score and p-value for each subgraph. For the evolutionary model presented in this thesis, I calculate the mean and variance frequency by counting each subgraph topology in the 1,000 simulated networks.

I used the Kavosh program with default parameters to both enumerate subgraphs and calculate motifs using the random rewiring model, chosen for its speed, command line interface, and accessible output format [62]. Significance scores for the random rewiring model used 1,000 random networks.
Figure 5.4: Z-scores for 3 (A) and 4 (B) node subgraphs are ranked according to frequency in the E. coli network. The left axis provides the scale for the z-score (bars) and the right axis measures the frequency (dashed line) of each subgraph. For each subgraph, the z-score for our model (black) and the edge-rewiring model (gray) are graphed side-by-side. Both models used 1,000 random networks to calculate significance. Important motifs are annotated along with the precise z-score found by each model (left, our model; right, edge-switching). Only the top 20 most frequent 4 node subgraphs are shown.

For each subgraph, the z-scores, using both the random rewiring model of the original study and my evolutionary model, are compared for 3- and 4-node subgraphs (Figure 5.4). There is poor agreement between the two models, which is expected due to their fundamental differences. For 3-node subgraphs, the feed forward loop (FFL), is highly significant according to the random rewiring model (z-score of 11.7) but is highly insignificant according to our model (z-score of 1.1). The single-input module (SIM) and the bi-fan subgraphs both occur with low z-scores under our model as well (-0.7 and -0.5, respectively). Many high-frequency subgraphs occur at signifi-
Figure 5.5: Z-scores for low frequency 4 node subgraphs are ranked according to frequency in the *E. coli* network. The left axis provides the scale for the z-score (bars) and the right axis measures the frequency (dashed line) of each subgraph. For each subgraph, the z-score for our model (black) and the edge-switching model (gray) are graphed side-by-side.

Sufficient levels according to our null model, including the 3-node linear pathway and the feed-back loop, and other subgraph topologies. Plots for other less-frequent 4-node subgraphs can be found in Figure 5.5.

### 5.2.3 Operon level properties

Since the null model makes use of the promoter length and PWMs for each operon in the *E. coli* network, it is possible to build distributions of network properties for any operon of interest. For instance, the null model can provide expected values for out-degree, in-degree, and clustering coefficient for the *fis* operon. I use this approach to identify operons which fit the model used in this study. I calculated the z-score for the clustering coefficient, in-degree, out-degree, and degree (sum of in- and out-degree)
per operon and plot their distributions in Figure 5.6. It is important to note that these network properties are dependent on one other, but in this analysis we decouple them by taking the distribution per operon across the 1,000 random networks. This is why, for example, the clustering coefficient plot in Figure 5.6A presents only negative z-scores per operon but the z-score is positive for the average clustering coefficient per network.

For each property, I classify each operon into three categories to gauge the agreement with the null model: having z-score < $-3$ (underrepresented), between $-3$ and 3 (expected), > 3 (overrepresented). The various operon sequence and network properties used in this study, including the empirical and null model values, are reported in [46]. The clustering coefficient, in-degree, and degree for operons have high agreement with the null model, with 89%, 94% and 90% in the expected category, respectively. The few operons which are overrepresented in degree are the same operons which are overrepresented in out-degree. Out-degree only has 38% agreement with the null model, with 50% being overrepresented; however, out-degree only applies to operons which encode TFs.

I investigated operons which had an absolute z-scores greater than 10. This set consists of 16 operons, only 1 of which did not encode a transcription factor, lending itself to the poor fit of out-degree. The operon $ubiCA$, which does not encode a transcription factor, has two interactions inferred from gene expression analysis which are not found in the promoter sequence, and so potentially fall outside the regulatory model used in this study. This list also includes important global or pleiotropic regulators like H-NS, Fis, Fnr, CRP, and I-HF, all of which have sequence motifs with
low gain rates but nonetheless interact with many operons. On the other hand, MalT, important for maltose metabolism, and MetJ, a common repressor, have binding motifs with high gain rates but low out-degree compared to the null model. Both results are explainable by poor PWMs, functional conservation (in the case of the global regulators), or removal of detrimental binding sites by selection.

5.2.4 Homogeneous versus heterogeneous promoter lengths

I performed additional simulations to measure the effects of the parameterization and initialization of our null model on the results. If instead we parameterize our model with homogeneous (average) promoter length and PWMs for all operons to generate a null distribution of 1,000 random networks, then all the results presented in this study are in fact reversed (see Figures 5.7, 5.8, 5.9, 5.10, 5.11, 5.12). On the system level, the homogenous null model resulted in 5,232 interactions (z-score of -71.4), average clustering coefficient of 0.552 (z-score of -14.6), and significantly different in- and out-degree distributions. The distribution of promoter length and in-degree differed by 97%. Among many other differences in the subgraph distribution, the bi-fan and feed forward loop subgraphs had z-scores 10x and 3x larger than the non-homogeneous model. At the operon level, the majority of operons fell significantly outside neutral expectations, which emphasizes the significance of incorporating the promotor length distribution in the model.

I also investigated the robustness of the model to “biologically reasonable” changes in parameterization, specifically, an alternative initial condition and shorter binding site length (7 bp). These alternate simulations yielded insignificant deviations from
Figure 5.6: For each operon in the E. coli network, the distribution of significance (measured in z-score) is plotted for several local network properties: Clustering (A), Degree (B), In-Degree (C), and Out-Degree (D). Vertical bars indicate a z-score at ±3, separating each distribution into three categories. The percent of operons in each of these three categories — significantly below, expected by, and significantly above — are listed underneath the name of the property. For instance, 89% of the operons have clustering coefficients as expected by the null model. In the Out-Degree distribution, only the 50 operons which encode TFs are included.
the results of the main study. However, when I performed additional simulations to measure the role of genetic drift, by using a smaller population size ($10^6$) and random walks instead of population genetic simulations, we found large differences at the system, subgraph and operon level. This suggests that genetic drift is an important force for the patterns observed in this study. The details and results of these alternate simulations are described in the following section.

5.2.5 Alternate simulation scenarios

This study leveraged carefully parameterized simulations to understand neutral patterns which may arise in the *E. coli* cis-regulatory network over evolutionary time scales. While all parameters in the main study derive from empirical values of the *E. coli* genome and regulome, a critical question remains as to the effect—whether weak or strong—that deviations in these parameters may have on the results. Because the empirical parameters, like binding site size, may present small or biased samples, it is important to understand the strength of the results provided noise in the parameterization. Specifically, we measure the effect of promoter heterogeneity, initialization condition, binding site length, and population size on the results of the study.

In the following sections, we describe the other simulations we ran and discuss their results. The figures provide an all-against-all comparison of the different experiments, split out by system, subgraph, and operon level (see Figures 5.7, 5.8, 5.9, 5.10, 5.11, 5.12).

Across all these experiments, we found that the results from the main study are not only robust to perturbations in the parameter values or starting conditions but also modify the results in a systematic fashion. For instance, decreasing the
binding site length increases the number of edges at equilibrium, which elevates the clustering coefficient, and does not significantly affect the results in the subgraph or operon level. In fact, we find that “small” changes in the parameters still yield results wherein FFLs and other previously identified ‘adaptive’ properties are not statistically significant from the null model. Only when we use homogeneous values for promoter lengths and mutation rates or use random walks instead of population genetic simulations do we notice drastically different results. Thus, the use of actual distributions to parameterize the null model—despite the minor differences in which these empirical distributions are derived from sequence-level data—is a significant and novel contribution to the identification of neutral patterns in the E. coli regulatory network.

**Empirical parameterization.** I parameterized the evolutionary simulations with empirical distributions of:

- promoter length - inferred from coding regions on the E. coli genome
- binding site PWMs - downloaded from RegulonDB [52]
- average binding site length - the average length of binding site PWMs
- population size - estimated in previous studies

These empirical distributions were retrieved from RegulonDB, which is kept up-to-date with the latest findings on the E. coli regulatory network. The population size for E. coli is a rough estimate for bacterial populations. For each operon, the results
of the main study including the parameters used — promoter length and PWM — are provided in the supplemental table of operon results.

**Homogeneous promoter lengths and mutation rates.** In this experiment, we investigated the significance of using actual distributions of promoter lengths and binding site gain and loss rates derived from PWMs by comparing to evolutionary simulations parameterized with average values only. To test the affect of using a homogeneous promoter length, used in previous studies like [24], we parameterized each operon with the same promoter length, set to the average of the actual distribution in *E. coli*. In a similar manner, we replaced the heterogeneous distribution of binding site gain and loss rates—derived from their respective PWMs—with the average gain and loss rate. All other parameters were left unchanged, including the average binding site length (20 bp) and population size ($10^9$).

Under this model, all of the results of the main study are effectively annulled. On the system level, the homogenous null model expects 5,232 interactions ($z$-score of -71.4), average clustering coefficient of 0.552 ($z$-score of -14.6), and the in-degree and out-degree distributions were significantly different. The distribution of promoter length and in-degree differed by 97%. Among many other differences in the subgraph distribution, the bi-fan and feed forward loop subgraphs had $z$-scores 10x and 3x larger than the non-homogeneous model. At the operon level, the majority of operons fell significantly outside neutral expectations.

**Alternate initial condition.** In the main study we seeded the simulations with a random initial network, such that all TF operons are auto-regulatory and any non-
TF encoding operon is regulated by a random TF. Such a configuration guarantees a minimally viable network such that the loss of any binding site would render the network non-viable. Further, this initial condition has no ‘interesting’ motifs besides a single input module, because no TFs can regulate another TF, so we do not bias the subgraph results with the initial random network. However, there are other methods for generating such random networks. To test the effect of the initial conditions on the results of the main study, we defined an alternative random initial network strategy such that both non-TF and TF encoding operons are regulated by a randomly chosen TF. Under this strategy, we do not force TFs to be auto-regulatory, but all operons still have only one incoming regulatory interaction. So, although there may be linear pathways and feedback loops in this initial random network, few other subgraphs, like FFL or bifan, exist in the network. In fact, this bias is present in the results: the second most common subgraph, a three-node linear pathway is much more common in the alternate initial condition than in the main study; however, z-scores for all other motifs are nearly identical.

Under this experiment, the results of the main study were not only repeated and validated, but the values themselves were closely matched. Therefore, the topology of the initial condition, evidenced by the agreement of the results from these two different approaches, does not have a significant effect on the results of the main study.

**Shorter binding site length.** While the average binding site length is calculated from actual *E. coli* PWMs, the length of these PWMs spans a broad distribution.
Consequently, the average used in the main study, may be a poor approximation of the ground truth. Although the specific gain and loss rates of each PWM are calculated from their IUPAC sequence, we simulate all binding sites as the same length, for computational purposes. Thus, by performing the same simulations with a shorter binding site length (7 bp), we may examine the extent to which (1) the average binding site and (2) the use of a homogeneous binding site length may have on the simulations. In regards to the latter, the effect of using heterogeneous binding site lengths could be extrapolated from the effect that changing the length from 20 bp to 7 bp has on the overall resulting networks.

Decreasing the binding site length reduces the loss rate of binding sites overall, because smaller binding sites present a smaller target for point mutations. Consequently, using a 7 bp binding site increases the average number of edges in equilibrium to 1,127 (z-score of -3.77) from 1,039 (results from Empirical) and the average clustering coefficient to 0.231 (z-score of -0.7) from 0.162. Despite these slightly elevated numbers, decreasing the binding site corroborated the results of the main study overall, especially including the subgraph and operon level results. Thus, the binding site length plays a ‘minor’ role in the results of the main study, and so we may expect that enhancing the simulator to support heterogeneous binding site length would also result in minor changes to the overall results.

**Smaller population size.** While the population size used in the main study is widely accepted as a reasonable effective population size for bacteria over the evolutionary timescale under investigation in this study, the role that the effective popu-
lation size has in the null model is of significant interest to this study. To investigate
the role of population size on the null model, we performed simulations with an ef-
fective population size of $10^6$. Altering the population size affects the mechanics of
genetic drift: reducing the population size in fact makes genetic drift a more powerful
evolutionary force (all else staying equal).

The smaller population size resulted in substantially (around 300) more edges in
equilibrium, which in turn increased the clustering coefficient and degree distributions.
Smaller populations allowed for more binding site gains to occur in long promoter
regions, increasing the number of regulatory edges in the network. However, even
under the smaller population size, FFLs occurred with a z-score of 1.3, too low to
reject the null model.

**Random walk.** While an evolutionary perspective is at the core of our study, a
question arises as to whether full-blown population genetic simulations are required to
reproduce the results. In fact, previous work which leveraged a neutral evolutionary
approach to replicate clustering patterns in eukaryotic enhancers opted for random
walks versus population genetic simulations [63]. Because random walks are much
simpler from a computational and modeling perspective, they are indeed the preferred
choice if they accurately represent the evolutionary dynamics of the study.

We simulated network evolution using random walks instead of a population ge-
netic lifecycle. Each random walk is initialized identically to the main study and each
step in the random walk corresponds to a binding site mutation (gain or loss). If the
mutation results in a non-viable network, the step is rejected and a new mutation is
sampled. This is the same process used in [63], where they found “no appreciable difference” between the results of population genetic simulations versus random walks. We measured the number of steps until an equilibrium in number of edges is reached (similar to the population genetic approach), and let that determine the number of steps in each walk.

Under random walks, the number of edges in equilibrium nearly doubles to 1,918 (z-score of -49) and the clustering coefficient also increases to 0.73 (z-score of -9). Like the smaller population size, there is a strong linear correlation between the length of a promoter and its in-degree; in both the main study and the real E. coli network this relationship is not so distinct. The degree distributions of the random walk networks is skewed to higher degrees. At the subgraph level, the subgraph distributions are indeed different than those produced by population genetic simulations (even under smaller population sizes), but FFLs, bifans, and SIMs still occur at levels similar to the actual E. coli network, although still not at equivalent values to the other simulations.

These results show that random walks, where non-viable mutations can be resampled, do not accurately represent the random walk performed by a population guided under genetic drift. Intuitively, the more accessible areas will continue to be explored by a large population whereas inaccessible areas (high non-viability or against the “mutation bias”) will not be explored. In this case, because there is a loss bias for binding sites, continuing to gain sites (and edges in the network) operates against the mutation bias present in the E. coli genome. Hence, population genetic simulations gives us a different answer than random walks.
Figure 5.7: The distribution of number of edges at equilibrium under the alternate simulation scenarios.

Figure 5.8: The degree distributions — comparing actual to random — for the alternate simulation scenarios.
Figure 5.9: The distribution of clustering coefficients under the alternate simulation scenarios.
Figure 5.10: The joint distribution of in-degree and promoter length — comparing actual to random — under the alternate simulation scenarios.
Figure 5.11: The z-scores for 3-node subgraphs under the alternate simulation scenarios, sorted by the frequency in the *E. coli* network.
Figure 5.12: The z-scores for the most common 4-node subgraphs under the alternate simulation scenarios, sorted by the frequency in the *E. coli* network.
Chapter 6

Towards measures for population regulomics

While forward-time simulations are useful for investigating complex evolutionary hypotheses, analytic measures exist that check for agreement to standard neutral models. I adapt these molecular population genetic measures of diversity to the pathway level by constructing interaction alignments between pathways in a population. Central to the application of these measures is the ability to sample multiple pathways in a population. In this chapter, I develop a pipeline for the inference of a population of pathways using a reference genome.
6.1 Computational pipeline for inferring a population of CRNs

The inference of an organism’s cis-regulatory network is standard practice in epigenetic studies and can be automated using high-throughput technologies like Chip-ChIP and Chip-Seq. Once a sufficient number of epigenetic studies have produced cis-regulatory annotations on genomes, it is possible to leverage this information using comparative computational techniques which infer the regulatory map for a related sequenced genome. Consequently, whether experimental or computational in origin, it is now possible to study a population of genomes from the same species. As an example, I build a method for inferring a population of *E. coli* CRNs using the well studied K-12 network, where each individual network represents a particular *E. coli* strain.

I used the comparative regulatory annotation tool PePPER to identify regulatory elements in each *E. coli* strain genome [1] (see Fig. 6.1). PePPER is a web tool which takes as input a set of transcription factor binding site motifs and outputs the locations of each promoter sequence, binding site per motif, and coding regions; it has been shown, in its original paper, to perform well on gamma protobacteria due to the quality of motif mapping and length of bacterial binding site motifs. Further, PePPER uses orthology and known, experimentally verified binding sites from a reference genome to provide additional hints to the binding site detection process. We used the well studied and experimentally verified motifs from the *E. coli* K-12 genome as input to detect the regulatory elements in each *E. coli* strain. Due
to the high conservation of coding sequence between genes in the *E. coli* population, we assume that the binding site domains in K-12 are an accurate representation of the domains in each strain.

With the regulatory and genome annotations — promoters, binding sites, and ORFs — I compiled a CRN for each strain where each node corresponds to a gene and each edge corresponds to a *cis*-regulatory relationship encoded by at least one binding site (in case there are redundant sites). I annotated each gene with its ortholog family downloaded from OrthologueDB, which uses RBB to identify pairwise orthologs between bacterial species and strains, its promoter sequence, and its operon. Each edge is annotated with the binding site(s) that drives the transcription of the gene.

### 6.2 Interaction alignments

Using the orthology mapping between genes across individuals (or strains), it is possible to build multiple interaction alignments across a population. An interaction alignment is built by aligning each potential interaction between a transcription factor and target gene that can be encoded in a promoter. Because the transcription factors and target genes are known, this is a finite alignment between the potential sources and targets of the *cis*-regulatory network. In the interaction alignment, I make the assumption that each interaction is independent and identical despite the fact that binding sites, which encode the interaction, may overlap or participate in combinatorial regulation. Much like the same assumption in studies focused on DNA or amino
**Figure 6.1:** It is possible to infer the regulatory annotations of a related genome to a fully annotated reference genome using the PePPER tool [1] coupled with additional data services.
acid sequence, this is a standard starting assumption for foundational models. For a population of CRNs encoding the set of transcription factors $T$ and target genes $G$, there are $|T| \times |G|$ potential interactions in the alignment. At each potential interaction $t \rightarrow g$ in an individual, $t \in T, g \in G$, it is encoded as ‘1’ if the interaction exists and ‘0’ if it does not exist. Again, an interaction exists if an ortholog of $t$ regulates an ortholog of $g$; however by way of construction, it is a good idea to restrict to the subset of genes with only one ortholog.

6.3 Interaction diversity and segregating interactions

A common molecular population genetic technique for measuring diversity in a population is to count the average number of pairwise differences between multiple aligned sequences, called average pairwise differences or $\Pi$. Nucleotide site diversity — the length-normalized form of $\Pi$ — is the probability that a given site will differ between two randomly sampled individuals in the population. For each site in the sequence, $\Pi$ is quantified as the number of sequence pairs with a non-matching versus matching site, and $\pi = \Pi/L$, where $L$ is the length of the sequence alignment. If sequence diversity is 1, then all sites are guaranteed to be different. If it is 0, then a population is isogenic.

I define “Interaction Diversity” as the average number of interaction differences between two randomly sampled individuals. I define two normalized measures of interaction diversity called gene-interaction and regulator-interaction diversity, where the
average number of differences is normalized by the number of TFs (gene-interaction) or number of gene targets (regulator-interaction). The former normalized measure corresponds to molecular differences in a gene promoter and the latter corresponds to the binding pattern across a genome for a particular TF or gene regulator. If we represent the cis-regulatory network as an adjacency matrix between regulators (columns) and gene targets (rows), then these two diversity measures correspond to a row-major (gene-interaction) or column-major (regulator-interaction) calculation.

Another common molecular population genetic measurement for diversity is segregating sites, which counts the number of polymorphic sites in aligned sequences. I define “Segregating Interactions” as the number of polymorphic interactions in an interaction alignment. An interaction is polymorphic if at least one individual does not share the same character (0 or 1) at a site in the interaction alignment.

Under neutral evolutionary scenarios, nucleotide diversity and segregating sites can be used as estimators for the population mutation rate, or \( \hat{\Theta} = 4Nu \). Because the frequency of mutations per generation is a driving variable behind the level of diversity expected in a population, it is possible to infer this population mutation rate using the measured level of sequence diversity. Likewise, it is possible to use the interaction diversity and segregating interactions as estimates for the population interaction mutation rate \( 4Nu \), where \( u \) is the rate of interaction change per individual per generation.
6.4 Building the Pan-Regulome

The pan-genome is the set of all genes present in a population; in the case of *E. coli*, the pan-genome has been well studied in terms of its functionality and adaptive evolution [64], scope [65], metabolic behavior [66], and lateral gene transfer [67]. In this work, we build a similar structure to the pan-genome, but at the regulatory level, which encompasses all regulatory interactions present in a population: the pan-regulome. Similar to the pan-genome, the pan-regulome represents important substrate from which strains may share functional capability encoded as identifiable genome units. For the pan-regulome, these transferable regulatory elements are the promoters and binding sites which mediate cis-regulatory transcription, and, much like coding sequences, can be inherited by or transferred to other strains by the same vehicles which diffuse coding DNA.

I inferred a population of *E. coli* CRN where each individual in the population corresponds to an *E. coli* strain using the pipeline as described in the Methods chapter. The fifteen strains examined in this study are shown in the phylogeny in Fig. 6.6. The tree was constructed using FastML2 on concatenated coding sequences for genes with exactly one ortholog present in each genome (cite). The inferred phylogeny corroborates previous work [65].

To build the pan-regulome for *E. coli*, I aligned each CRN using the annotated gene orthology for each node. For each pair gene A and gene B, we add an edge if an ortholog of A regulates an ortholog of B in at least one strain and annotate that edge with the set of strains that harbor this interaction. I considered the subgraph
Figure 6.2: The fifteen strains examined in this study, along with the outgroup *E. fergusonii* is depicted in this phylogeny. Branch lengths correspond to the number of interaction mutations occurring on each lineage and the color corresponds to inferred number of interaction gains (green) and losses (red) using PAML.

represented by all strains as the core pan-regulome. Further, because regulatory interactions in a CRN are affected by operon structure and gene duplication, I applied a rigorous data quality filter, using only those genes that have clear, non-duplicated orthologs in each strain and are conserved in the same operon. As strains are added to the pan-regulome, this filter reduces the number of ‘core’ genes but increases the number of interactions (see Figure 6.3). The number of unique interactions in the core pan-regulome saturates at around 10 strains, and is in fact quite sparse in comparison to the potential density of the regulatory network (roughly 6% density). Also, the core pan-regulome, based on its degree distribution, matches the topological properties of the pan-regulome itself, which means that the patterns found in the core pan-regulome may scale well to the entire genome.
Figure 6.3: The pan-regulome is constructed by merging information from multiple *E. coli* strains into one network. The distribution of number of genes (gray) and edges (black) are determined by randomly sampling at most 200 combinations of each strain count from the 15 used in this study. Our pan-regulome focuses on genes within preserved operons, the number of which decreases as strains are added to the pan-regulome construction. The number of edges, as determined by predicted binding sites across the *E. coli* strains, increase as strains are added to the pan-regulome, although the number saturates at around 10 strains.

The core pan-regulome for *E. coli* consists of 7,651 interactions, the majority of which are unique to a strain (3,548 interactions). While the majority of interactions are unique, or shared by another strain, there is an ancestral interaction set shared between 13-15 species which correspond to the several long internal branches at the root of the *E. coli* strain tree.
6.5 Measuring diversity in a population of networks at the gene, promoter, and interaction level

The pan-regulome is a powerful data structure which supports the analysis of regulatory networks from the population level. In this section, I apply the population regulomic measures of interaction diversity and segregating interactions.

For each gene in the pan-regulome, I calculated gene-interaction diversity at the regulatory level and at the sequence level calculated $\pi$ for both the upstream promoter and gene coding sequences. I report these values per gene as interaction, promoter, and gene diversity (we elide the word “sequence” from promoter and gene diversity); their distributions are plotted in Figure 6.4. In the *E. coli* population, differences within promoters are more common than differences at the interaction level (by roughly a factor of 10), and gene coding polymorphisms occur roughly twice as much as interaction differences. I examined the correlation between these different diversity measures at the gene level by comparing interaction diversity to sequence diversity in the promoter and interaction diversity to sequence diversity in the gene coding region. A statistically significant correlation exists between interaction and promoter diversity (Pearson correlation coefficient 0.54; p-value=$2 \times 10^{-9}$; N=108) but no correlation exists between interaction and gene diversity (-0.01; p-value=0.88; N=113).

Promoter length, as noted in the previous sections, can play a causal role in determining binding site accumulation and interaction degree. In the pan-regulome, there
was a higher correlation between interaction diversity (0.72; p-value=5 × 10^{-19}; N=108) than promoter diversity (0.45; 9 × 10^{-7}; N=108) with respect to promoter length.

The in-degree of a gene in the pan-regulome corresponds to the number of unique transcription factors which regulate that gene across the entire population. Consequently, it is also an upper-bound on the number of interaction polymorphisms within a promoter. So, there is a strong correlation between the interaction diversity and the in-degree of a gene (coefficient 0.85; p-value=3 × 10^{-34}; N=113), but a weak correlation between the promoter sequence diversity and its in-degree (0.38; 3 × 10^{-5}; N=113).

For the 54 TFs in the pan-regulome considered in the study, I calculated the regulator interaction diversity (mean=0.00842, median=0.00654, var=3.3 × 10^{-5}). The mean can be interpreted as the average number of interaction differences between TFs and their targets between two randomly sampled individuals in the population. Provided the genome contains about 4,000 genes, this number indicates that there are roughly 32 interaction differences per TF between any two random individuals in the *E. coli* population. The regulator interaction diversity showed an extremely strong positive correlation with the out-degree of the TF (Pearson coefficient 0.986; p-value=1 × 10^{-42}). So, global regulators, or TFs with high out-degree like CRP, also tended to have high interaction diversity; whereas, pathway specific TFs with low out-degree tended to have lower interaction diversity.
Figure 6.4: Cumulative degree distributions for interaction, promoter, and gene diversity reveal that average pairwise differences are most frequent between promoter sequences and least frequent between interactions. A strong correlation exists between interaction diversity and promoter diversity, while no correlation exists between interaction diversity and gene diversity.

6.6 Estimating interaction mutation rates

Under neutral evolutionary scenarios, nucleotide diversity and segregating sites can be used as estimators for the population mutation rate, or $\hat{\Theta} = 4Nu$. Because the frequency of mutations per generation is a driving variable behind the level of diversity expected in a population, it is possible to infer this population mutation rate using the measured level of sequence diversity. Now that we have adapted population genetic measures for the diversity of interactions, we can estimate the population interaction mutation rates $Nu$, where $u$ is the rate of interaction change per individual per generation. Figure 6.5A plots the distribution of $\hat{\theta}$ for interactions, promoter sequences, and gene sequences in the *E. coli* pan-regulome.

Because there are reasonable estimates for the mutation rate in coding regions ($\hat{u} = 3 \times 10^{-9}$), we can calculate the effective population size $\hat{N}_e = N_e u/\hat{u}$ of the population in this study. Then, we can use the estimate for $N_e$ to determine the
interaction mutation rate. I found the population size to be roughly $2 \times 10^8$, with an interaction mutation rate of $5.4 \times 10^{-10}$ and a promoter mutation rate of $1.4 \times 10^{-8}$.

Because the effective population size is consistent across population genetic measures, the ratio of population mutation rates is equivalent to the ratio of mutation rates. Consequently, for the *E. coli* pan-regulome, we can calculate the relative mutation rates between coding and non-coding sequence rates and interaction changes (see Figure 6.5B). For instance, by calculating the ratio of interaction over promoter mutation rate, I quantified the interaction per base pair change in the promoter sequence. In this study, I found that on average 10.5 bp changes correspond to one interaction mutation (median 5.9 bp/interaction), which is corroborated by the average number of consensus sites in an *E. coli* binding site motif (9.8 bp). I found a large amount of variation within the mutation rates across operons, which is largely driven by the variance in the distribution of nucleotide diversity in promoter regions; the average mutation ratio between interactions and genes is near 1 (0.93) and has relatively low variance (1.9 variance; N=113) compared to ratios including the promoter sequence (variance over 110).

### 6.7 Testing the independent interaction hypothesis

Binding sites which encode the same regulatory interaction — that is, bind the same TF — may arise in the same promoter, leading to binding site multiplicity and redundancy. At the network level, these repeated binding sites are condensed
under one interaction link between the encoded TF and the downstream operon. Consequently, network interactions are very much dependent on the number and configuration of their underlying binding sites. So, it is in fact not clear whether an independent interaction hypothesis — that each interaction is mutated only once — holds within populations. I tested the independent interaction hypothesis on the strain tree by reconstructing likely evolutionary histories and counting the number of mutations per network interaction.

I inferred the evolutionary histories of the *E. coli* strain networks by applying PAML to interaction sequences (recall these are two character sequences encoding the network) [68]. Each location in the interaction sequence corresponds to a directed edge in the regulatory network between a TF and target gene. This process produces ancestral regulatory networks along with interaction gains and losses along the strain
tree, which I then use to count the number of mutations per interaction in the \textit{E. coli} population (see Fig. 6.6). Implicit in this modeling is an assumption of strictly vertical evolution within the \textit{E. coli} strains, which has been challenged by other evidence based on genome gene content [67]; however, the majority of interactions have been either mutated only once or not at all, which would support an independent interaction hypothesis. With that said, there still are a small fraction — about 2% — of interactions in the pan-regulome that do not fit the independent interaction hypothesis. Because the pan-regulome is sparse, non-existent interactions remain non-existent from a likelihood perspective. If we run the same analysis on only interactions that exist in the pan-regulome (672 of 11,187 interactions), about 35% account of interactions with more than one mutation. If we compare the number of interaction mutations to the number of site mutations in promoters and genes, we see that roughly 20% of promoter sites do not follow the independent site hypothesis (compared to about 3% for coding sequence).

6.8 Inference of interaction gain and loss rate

Although theoretical and experimental approaches have attempted to determine the gain and loss rate of binding sites, inferring an effective gain and loss rate at the interaction level is best accomplished looking backwards in time. I reconstructed the ancestral history of the \textit{E. coli} regulatory networks using a two parameter — corresponding to interaction gain and loss — maximum likelihood approach. As such, PAML inferred gain and loss rates from a coalescent perspective for regulatory
**Figure 6.6:** The distribution of number of mutations per interaction in the pan-regulome is determined using a maximum-likelihood model of ancestral network reconstruction, along with the number of site mutations in promoters and genes. Any particular edge may be gained and lost several times during the evolutionary history of the *E. coli* population, which is reflected by the interactions with more than 1 mutation. The inset in the interactions plot presents the histogram for interactions that strictly exist in the pan-regulome (versus all the potential edges). Genes show the highest amount of conservation — nearly 90% of coding sequence is shared.

interactions. The loss-to-gain mutation rate ratio is an informative number which correlates with topological network properties, including redundancy, motif distributions, degree distribution, and complexity [24][31], and has been previously estimated forward in time for binding sites in *E. coli* somewhere between 10-1000. The inferred loss-to-gain interaction mutation ratio based on the data set in this study is 1.4. Because a coalescent approach does not include edges that existed in an ancestral population but were lost, the inferred ratio is no doubt a lower bound of the true ratio; however, our results suggest that although *E. coli* is at a higher risk for losing interactions versus gaining them, it may not be to the extent theorized in previous work.
6.9 Correlating interaction diversity with fitness

Because interaction diversity quantifies the regulatory differences between individuals in a population, we may expect a correlation interaction diversity of a particular gene and its fitness knock-out (KO) effect. Fitness KO data is captured by silencing each gene individually and measuring the relative growth rate of each mutant against all other KO strains. With this approach, it is possible to relate the transcription product of a particular gene with its quantifiable effect on growth rate in various mediums and stresses. In this study, I merged the fitness KO experiments from [69] with the interaction diversity measures calculated on the present E. coli population. Intuitively, we might expect that genes with high interaction diversity — that have undergone more regulatory changes than other genes — would be less critical for fitness. However, the subset of genes used in this study indicated that there is little correlation between interaction diversity and fitness.

To assess whether fitness and interaction diversity correlated, I merged the results of [69] with the interaction diversity measures at the locus level, resulting in 83 loci over 324 conditions, where each condition is organized into groups (e.g., stress, amino acid). I found that these 83 genes were a reasonable statistical sub-sample of the larger set reported in [69] (Mann-Whitney, p-value=0.22, using average fitness across conditions). For each gene, I calculated its average KO fitness across all conditions, over each condition family, and within each condition. I compared each of these fitness calculations to interaction diversity using linear regression, and found no significant relationship at any level. I then calculated correlations against nucleotide diversity.
in coding regions and fitness, which produced minimal correlations as well.

To understand if gene and interaction changes could be working in concert, I correlated a linear model composed of nucleotide diversity and interaction diversity against fitness KO effect. Again, I found no signal connecting fitness KO effect and measurements of diversity. In fact, nucleotide diversity within the promoter was the best correlate with fitness ($r=0.22$, p-value=.04), albeit a weak correlation. Because the subset of genes for which we could calculate interaction diversity corresponds to some of the most conserved genes in the *E. coli* pan-genome, poor correlations between interaction diversity and fitness may be an artifact of the construction of the study sample.
In this work, I have created a set of models, measures, and methods for the practical analysis of CRNs that I call population regulomics. A major emphasis of population regulomics is to provide solutions for quantifying and testing neutral evolutionary hypotheses on real CRN data. In particular, I designed a CRN genotype representation that captures the underlying sequence and defined a model of mutation which can handle actual binding site motifs and variability in intra-genomic ncDNA regions. I validated the model against relevant data across many organisms, demonstrating the valid connection between sequence features and regulatory properties. I was able to apply this model within a population genetic apparatus to quantify that certain topological properties in the *E. coli* CRN evolved from neutral forces alone. Lastly, I measured diversity in a population of networks and connected interaction diversity to its sequence counterparts.


7.1 Improving the performance of simulations

A critical facilitating technology for executing the CRN genotype model are forward-time simulators. The operation graph (OG) defined in this work presents a general and efficient data structure for lossless compression of genotypes in real-time, for the main purpose of constraining the memory footprint of forward-time population genetic simulations. By itself, the OG is capable of decreasing the memory footprint by several orders of magnitude, making possible large-scale simulations that would otherwise crash the system. However, without explicitly representing a subset of the genotypes in the OG, the time cost of decompression grows with simulation time; this amounts to trading “simulations that crash” for “simulations that never end.” Therefore, the constant-time scaling of Greedy-Load with respect to simulation time is crucial for the viability of the operation graph as a general solution. Further, the OG and Greedy-Load leverage only inheritance topology to perform compression, which means our approach is general not only to genotype representation but also to implementation details of evolutionary operations.

The performance of the Greedy-Load algorithm is robust to \( t \), the frequency of its execution, and \( k \), the maximum number of explicit genotypes. When \( k \) is low, which, (compression rate < 0.02), there was a significant drop in the time performance of Greedy-Load; otherwise, \( k \) and \( t \) had little effect on the execution time of the algorithm. I recommend fine-tuning \( k \) and \( t \) using shorter simulations to determine which parameters to use for longer simulations.

Because Greedy-Load diminishes the strain on the memory system while still
efficiently minimizing the decompression cost of active genotypes, Greedy-Load consistently performed on-par and with less memory than Store-Active in all of our experiments. Reducing the amount of memory that is allocated and freed has a significant impact on the efficiency of the memory hierarchy. For example, reducing the overall memory overhead reduces cache misses and page faults, which, over time, has a significant impact on the speed of a simulation. So, not only does Greedy-Load constrain the memory footprint, it can do so without sacrificing speed.

Although constraining the memory footprint of a simulation can increase the execution time, providing a performance knob that tunes between space savings/time cost and space cost/time savings is not only a useful tradeoff but crucial for simulations with large genotypes or large populations. For example, constraining the memory footprint enables more parallel, independent simulations to run on the same node. In a recent review of forward-time simulators, sequences of length 10 Mb caused many simulators to crash [38]; in contrast, we showed that Greedy-Load could handle a population of 1000 sequences of length 10 Mb while constraining the genotype heap to 100 MB. And these benefits are not just for sequences; our compression technique also facilitated the simulation of 1,000-gene pathways while still constraining the memory to under 1 GB.
7.2 The significant role of ncDNA in CRN topology

By analyzing data from five model organisms, I found that the mutational bias imposed by the amount ncDNA, in addition to population size, fosters or inhibits the accumulation of binding sites neutrally. Further, purifying selection on pathway function or on regulatory complexity itself shows minor effect over the mutational bias. These results do not diminish the role of selection in conserving functional binding sites; instead, they demonstrate that selection must operate within the bounds of complexity determined by the amount of regulatory substrate, or ncDNA. These results are corroborated by convincing empirical evidence—the strong correlation between ncDNA and binding site accumulation—from a diverse group of model organisms.

To validate our hypotheses, we performed realistic population genetic simulations which were parameterized with accurate and empirically-calculated rates. We used a model of pathway genotype that encodes binding sites instead of regulatory interactions \cite{23, 47, 24}, which we believe is more biologically accurate and leads to different results. For instance, our binding site based model demonstrated that recombination between binding sites is a minor force compared to the mutation bias in developing redundancy; this result is considerably different than what was found previously under an interaction-based model \cite{24}. Our results are supported by the empirical data which shows that the levels of redundancy in \textit{E. coli} seem to agree with the log-linear trend against ncDNA. Furthermore, our results depict the significant effect binding site length can have on their accumulation when the length of ncDNA is large: an
expansion of 2 bp in length (from 8 bp to 10 bp) can decrease the number of binding sites, degree, multiplicity, and redundancy by almost an order of magnitude. When there is a small amount of ncDNA, regulatory complexity is less affected by binding site length. This observation is especially interesting because binding sites are known to be longer in prokaryotes (10-20 bp) than in eukaryotes (6-10 bp). Due to this large difference in length, the short binding sites in eukaryotes, working in concert with long stretches of ncDNA, may play a significant role in the emergence of regulatory complexity which characterizes ‘higher organisms.’

7.3 Quantifying neutral signal in *E. coli*

The results in this study demonstrate that, taking only a few important sequence characteristics and neutral evolutionary processes into account, it is possible to generate random networks that resemble the actual *E. coli* network at the system, subgraph, and operon levels. Due to their large effective population size and short reproductive cycles, bacteria are thought to be molded primarily by natural selection on the sequence and network level [11, 70, 71]; however, our model, which takes into account population genetic mechanisms, predicts that important features of the regulatory network for *E. coli* follow neutral patterns.

These results show that the degree distribution, clustering coefficient, and number of interactions all follow neutral patterns. Furthermore, the ability to quantify these neutral trends revealed the staggering portion of the *E. coli* network—nearly 90% for several important network properties—that occur at frequencies expected by non-
adaptive evolution. However, this does not mean that only 10% of the *E. coli* network derives from selective forces. Instead, our results serve as a guide for identifying more informative network properties that are enriched with adaptive signal. For instance, 94% of operons have in-degree within three standard deviations of the null model, but only 38% of operons have out-degree that agree with non-adaptive expectations. Thus, although the number of unique regulatory interactions per operon may be neutral, transcription factors are wired considerably differently (about 62% according to out-degree) than expected by chance. Identifying this deviation is another strength of our model, since it guides the user to go back and understand the biology of the underlying system and highlights areas of further investigation.

The neutral trends in the local wiring of regulatory networks also challenge the prevailing adaptive perspective: namely, are commonly accepted ‘motifs’, such as feed-forward loops, really *motifs*? A network rewiring model which preserves system-level properties was used to identify motifs—which have since been shown to execute highly functional temporal programs. These results have been adopted in the systems biology community to the point that ‘feed forward loops’, ‘single input module’, and ‘bi-fan’ are synonymous with ‘motif’. Subsequent work has linked the ubiquity of these motifs to convergent evolution driven by functional requirements [35, 72]. However, when using our evolutionary model, which accounts for events at the genomic level, it emerges that FFLs, SIMs, and bi-fans all occur within frequencies expected by neutral evolution. Without taking into account pathway dynamics, our null model accurately predicts the frequency of three important and highly functional subgraphs. Therefore, according to our model, these three subgraph topologies are not motifs,
but rather topological patterns that would be expected to emerge by non-adaptive forces operating on sequences.

The functionality of motifs, although an enabler of sophisticated cellular behavior, is not necessarily the cause nor the explanation of their origin. This is not to say that regulatory network motifs do not exist; in fact, our model identified several 3- and 4-node subgraph topologies which were significantly over- or under-represented in the *E. coli* network. However, accounting for true evolutionary processes, such as mutation and drift, rather than synthetically rewiring a network, might lead us in the direction of the true motifs.

I also demonstrated the ability to calculate population genomic measures at the regulatory level by using interaction diversity, which presents a different approach for quantifying neutral signal. Because interaction diversity measures the relative regulatory differences between individuals within a population, it can be used, like nucleotide diversity, to infer mutation rates. Further, because interaction diversity is calculated at the locus level, it is possible to build the distribution of interaction diversity within a CRN, allowing us to identify which genes potentially fall within and outside a neutral model. Using *E. coli*, I showed that interaction diversity poorly correlated with fitness KO effect, which supports the findings that most regulatory interaction changes are neutral.

These results in general agree with other recent perspectives on regulatory evolution. Within a promoter, complex binding site patterns like clustering, which were once thought to be adaptive, may in fact emerge from neutral evolution [63]. At the subgraph level, commonly accepted motifs are poorly conserved across homologous
genes [73], thus, convergent evolution of subgraphs [73, 74] is more simply explained as the result of neutral forces acting similarly in divergent species rather than as an adaptive response to environmental changes. Within a system, genome-wide expression and fitness experiments in bacteria have identified the sub-optimal control at the genome level, which raises questions as to the efficacy of selection to mold regulatory networks beyond direct interactions [75]. The ability of selection to optimize the structure of a regulatory network is well studied in simulations, e.g., [35, 72], but lacking in any in vivo observations beyond a few regulatory interactions [76]. In fact, protein abundance is determined only in part by transcription, so regulatory network topology is significantly obfuscated from organismal-level selection, which is rarely taken into account in simulation studies [77]. Indeed, a simpler explanation to all these observations, and in keeping with the results from our study, is that bacterial regulatory patterns are mostly explainable through neutral evolution acting on genomic properties.

7.4 Revitalized non-adaptive perspective on CRN evolution

The cis-regulome is not a clean functional network crafted by adaptive forces alone, but instead a data source filled with the noise of non-adaptive forces. From a regulatory perspective, this evolutionary noise manifests as complexity on both the binding site and pathway level, which has significant implications on many directions in microbiology, genetics, and synthetic biology.
If the accumulation of binding sites follows neutral trends, then most binding sites may actually be neutral to function, and therefore encode very little ‘meaningful’ information in terms of understanding the transcriptome. Consequently, the identification of binding sites would yield minimal functional insight and actually mining functional signal from their identification would be muddied by over-complicated regulatory interactions. Ultimately, the cis-regulome is not a clean functional network, but instead a ‘messy’ data source filled with the effects of non-adaptive forces.

A revised, neutral understanding of how regulatory complexity and topology evolves has a dramatic impact on many directions in microbiology and genetics. The spontaneous gain rate of binding sites in ‘higher organisms’ would lead to unknown polymorphisms in the regulatory regions of genes, between individuals, and even between cells in a body, which is a significant complicating factor to treating non-Mendelian diseases like cancer. Tools like population regulomics will be vital for identifying the interactions and pathways which follow neutral trends versus those that have been conserved or positively selected. These results transfer to synthetic biology, where design principles of regulatory circuits must accommodate the neutral evolutionary forces that will wreak havoc on ‘industrial’ (wild) populations. Only pathways that can withstand the neutral bias of their host population will remain as designed; pathways that do not have adequate evolutionary ‘supports’ in place may either spontaneously decompose or augment, depending on the amount of ncDNA and population size.
7.5 Future work

Much like population genetics itself, population regulomics is ripe for extension and enhancement. In its initial conception, I made many simplifying assumptions for practicality sake, but now that successful results have been attained, it is possible to layer in more complicated and sophisticated processes which may play a significant role in shaping CRN topology. Also, much like the response of population genetics to the explosion of empirical sequence data, I believe population regulomics will need to continue to take into account the variety of representations and outputs from modern high-throughput technologies that generate regulatory annotations.
Bibliography


