THE UNIVERSITY
of EDINBURGH

Human Genome Interaction: Models for Designing DNA Sequences

Emily Scher

Doctor of Philosophy
Institute of Adaptive and Neural Computation
School of Informatics
University of Edinburgh
2020

to Alexis

Acknowledgements

This work was funded by the Darwin Trust of Edinburgh.

I would like to thank my supervisors, Guido Sanguinetti and Shay Cohen, for answering all of my questions, no matter how basic, and for teaching me so much about the field. None of this work could have happened without their guidance and patience.

I would also like to thank Joel Bader and Patrick Cai for their advice, the members of the Edinburgh Cai lab for their friendship and enthusiasm for science, and the members of the Sanguinetti group for their willingness to help and their expertise.

Also, thank you to Seth Toplin, Jordan Matelsky, Kevin Cryan, Jacqueline Morea, Annie Causey, and Kathleen Lewis, whose friendship and emotional support kept me going.

I would also like to thank my parents for making me take Calculus when I insisted I would never need it in my future life as a poet. It turns out they were right after all.

Finally, I would like to thank the Darwin Trust for giving me the opportunity to pursue my PhD.

Abstract

Since the turn of the century, the scope and scale of Synthetic Biology projects have grown dramatically. Instead of limiting themselves to simple genetic circuits, researchers aim for genome-scale organism redesigns, revolutionary gene therapies, and high throughput, industrial scale natural product syntheses. However, the engineering principles adopted by the founders of the field have been applied to Biology in a way that does not fit many modern experiments. This has limited the usefulness of common sequence design paradigms. As experiments have become more complex, the sequence design process has taken up more and more intellectual bandwidth, partially because software tools for DNA design have remained largely unchanged.

This thesis will explore software engineering, social science, and machine learning projects aiming to improve the ways in which researchers design novel DNA sequences for Synthetic Biology experiments.

Popular DNA design tools will be reviewed, alongside an analysis of the key conceptual metaphors that underlie their workflows. Flaws in the ubiquitous parts-based design model will be demonstrated, and several alternatives will be explored.

A tool called Part Crafter (partcrafter.com) will be presented, which aggregates sequence and annotation data from a variety of data sources to allow for rational search over genomic features, as well as the automated production of biological parts for Synthetic Biology experiments. However, Part Crafter's mode of part creation is more flexible than traditional implementations of parts-based design in the field. Parts are abstracted away from specific manufacturing standards, and as much contextual information as possible is presented alongside parts of interest.

Additionally, various types of machine learning models will be presented which predict histone modification occupancy in novel sequences. Current Synthetic Biology design paradigms largely ignore the epigenetic context of designed sequences. A gradient of increasingly complex models will be analysed in order to characterise the complexity of the combinatorial patterns of sequences of these epigenetic proteins. This work was exploratory, serving as a proof of concept for using a variety of increasingly complex

models to represent genomic elements, and demonstrating that the parts-based design model is not the only option available to us.

The aims of the field of Synthetic Biology become more ambitious every year. In order for the goals of the field to be accomplished, we must be able to better understand the sequences we are designing. The projects presented in this thesis were all completed with the aim of assisting Synthetic Biologists in designing sequences deliberately. By taking into account as much contextual information as possible, including epigenetic factors, researchers will be able to design sequences more quickly and reliably, increasing their chances of achieving the moon shot goals of the field.

Lay Summary

Every known organism contains DNA – a molecule which encodes the instructions for all cellular processes, and is sometimes called the "book of life." However, the processes through which these DNA instructions are followed are complicated and many-faceted. Systems of epigenetic proteins bind to DNA molecules, controlling which instructions are read, and which go unfollowed.

The field of Synthetic Biology was founded with the expectation that researchers could learn more about Biology not just by trying to read DNA instructions, but by writing them as well. A variety of techniques have been invented which allow scientists to design and insert novel DNA sequences into an organism's genome, causing its cells to behave in new ways. Synthetic Biology has led to important research into minimal genomes, gene and immunotherapies, as well as the production of useful products using cells as tiny factories.

Early Synthetic Biology experiments relied on engineering principles — such as the parts-based design paradigm — to make the DNA sequence design process more efficient. However, since the founding of the field at the turn of the century, Synthetic Biology projects have become dramatically more complicated, and the design paradigms from the early days of the field are not useful for some modern research projects.

This thesis will explore software engineering, social science, and machine learning projects aiming to improve the ways in which researchers design novel DNA sequences for Synthetic Biology experiments.

Popular software tools for DNA design will be reviewed. The underlying metaphors through which these tools attempt to represent DNA sequences will be analysed. Flaws in the ubiquitous parts-based design model will be demonstrated, and several alternatives will be explored.

A tool called Part Crafter (partcrafter.com) will be presented. This tool represents a more flexible way to generate the genetic "parts" which parts-based design relies upon. Descriptive information is gathered and associated with particular DNA sequences. This allows users to search for sequences using English queries. Parts are abstracted away

from specific manufacturing details, and as much contextual information as possible is presented alongside sequences of interest.

Additionally, various machine learning models will be presented that predict which epigenetic proteins — specifically histone modifications — will bind to newly designed sequences when they are inserted into a host genome. Current Synthetic Biology design paradigms largely ignore the epigenetic context of designed sequences. However, epigenetic proteins determine if and how an inserted sequence will be expressed. Inserting a new sequence into a genome can also alter the patterns of epigenetic proteins which bind to the entire genome, meaning that an insertion in one place along the genome can affect gene expression elsewhere.

A series of machine learning models will be presented which can each represent increasingly complex patterns of sequences of these epigenetic proteins. By using these increasingly complex models, we aimed to learn more about how complex the patterns of epigenetic protein sequences are. This work was exploratory, serving as a proof of concept for using a variety of increasingly complex models to represent genomic elements, and demonstrating that the parts-based design model is not the only option available to us.

The aims of the field of Synthetic Biology become more ambitious every year. In order for the goals of the field to be accomplished, we must be able to better understand the sequences we are designing. The projects presented in this thesis were all completed with the aim of assisting Synthetic Biologists in designing sequences deliberately. By taking into account as much contextual information as possible, including epigenetic factors, researchers will be able to design sequences more quickly and reliably, increasing their chances of achieving the moon shot goals of the field.

Contents

List of Figures xiii

List of Tables xv

1 Introduction 1
- 1.1 Contributions . . . 1
- 1.2 Thesis Layout . . . 2

2 Synthetic Biology, Molecular Biology, and Epigenetics 3
- 2.1 Synthetic Biology Prehistory . . . 3
- 2.2 Modern Synthetic Biology . . . 7
- 2.3 An Overview of Relevant Molecular Biology . . . 14
  - 2.3.1 DNA . . . 14
  - 2.3.2 DNA Replication . . . 15
  - 2.3.3 The Central Dogma . . . 15
  - 2.3.4 Splicing . . . 19
  - 2.3.5 Sequencing Technologies . . . 19
  - 2.3.6 Synthetic Biology Sequence Editing Techniques . . . 21
  - 2.3.7 Epigenetic Control of Gene Expression . . . 24
- 2.4 The Space Between Theory and Practise in Parts-Based Design . . . 31

3 An Overview of Relevant Modelling and Machine Learning Techniques 33
- 3.1 Linear and Logistic Regression . . . 34
- 3.2 N-Gram Models . . . 35
- 3.3 Hidden Markov Models . . . 36
- 3.4 Probabilistic Context Free Grammars . . . 40
- 3.5 The Chomsky Hierarchy of Languages . . . 44

4 Models and Metaphors for DNA Design **47**
4.1 Introduction . . . 48
4.2 DNA design tools: Different wrappings on (much) the same box . . . 51
4.2.1 Comparing features of DNA design tools . . . 51
4.2.2 Different features, same model . . . 54
4.3 Underlying limitations of current DNA design tools . . . 55
4.4 Rationalising the distance between DNA as model and DNA as molecule 57
4.4.1 Designing, not describing . . . 57
4.4.2 "Good enough" . . . 57
4.4.3 Constraints of the underlying metaphor . . . 58
4.5 Improving the metaphor for alternative models . . . 58
4.6 Conclusion: Reconceptualising "good enough" DNA design . . . 60

5 Part Crafter: Find, Generate and Analyze BioParts **63**
5.1 Abstract . . . 64
5.2 Introduction . . . 65
5.3 Workflow . . . 68
5.3.1 Organism Processing and Data Aggregation . . . 68
5.3.2 Search . . . 68
5.3.3 Parts Generation . . . 69
5.4 Example Use Case . . . 70
5.5 Validation . . . 72
5.6 Implementation . . . 73
5.7 Availability . . . 77

6 Histone Modification Occupancy Prediction Through Language Modelling **79**
6.1 Introduction . . . 79
6.2 Background . . . 80
6.3 Methods . . . 81
6.3.1 Data Processing . . . 81
6.3.2 N-gram Models . . . 82
6.3.3 Hidden Markov Models . . . 85
6.3.4 Probabilistic Context Free Grammars . . . 92
6.4 Discussion . . . 110

7 Conclusion **111**

Appendix A PartCrafter Workshop Worksheet **115**
A.1 PartCrafter Key Features 115
A.2 Exercises 115

Appendix B Additional N-Gram Model Statistics **119**

Appendix C Results for Additional HMM Tests **123**
C.1 HMMs Re-Parameterised by GC content 123
C.2 HMMs Re-Parameterised by ATCG Counts 126

Appendix D Additional HMM Matrix Heatmaps **131**

Bibliography **139**

List of Figures

2.1 Prehistory of Synthetic Biology. A selection of scientists and philosophers whose ideas have affected the development of the field of Synthetic Biology 6

2.2 A Brief History of Synthetic Biology: Highlighted events in the history of Synthetic Biology (Adler et al., 1958; Boeke et al., 2016; Cameron et al., 2014; Cello, 2002; Dymond et al., 2011; Elowitz and Leibler, 2000; International Human Genome Sequencing Consortium, 2001; Koppolu and Vasigala, 2016; Si and Zhao, 2016; Thiel, 2018; Watson and Crick, 1953) 8

2.3 The three modes of Synthetic Biology. A selection of scientists presented with examples of their work which fit into one of the three modes of Synthetic Biology. It is worth noting that each of these scientists has at some point worked on experiments in all three categories. (Boeke et al., 2016; Endy, 2005; Forster and Church, 2006b; Gibson et al., 2010; Kim and Benner, 2017; Nielsen et al., 2016; Richardson et al., 2017; Tamsir et al., 2011) 12

2.4 A timeline of software tool development in Synthetic Biology 13

2.5 An overview of the Central Dogma of Molecular Biology (Pierce, 2012). . 16

2.6 An overview of the structure of promoters and terminators (Blazeck and Alper, 2013). 18

2.7 An overview of the CRISPR/Cas9 system. A designed guide RNA targets a specific region in a gene. The Cas9 enzyme creates a double-stranded break. The cell then undergoes either nonhomologous end joining or homologous recombination. NHEJ tends to be used to disrupt genes, by introducing small insertions or deletions. HDR can be used to introduce a new, edited sequence. (Barrangou, 2014; Khan et al., 2018). 23

2.8 The original drawing of Waddington's Epigenetic Landscape, published in Waddington (1957). 26

2.9 An overview of the structure of a chromosome. 27

2.10 An early example of a Synthetic Biology gene circuit design. This particular design shows a toggle-switch. Two repressor genes (lacI and cI) are placed such that each represses transcription of the other. Heat can be used to disengage cI and IPTG is used to disengage lacI. By using these two environment inputs, it is possible to toggle the switch between two states. Once the switch is flipped with one of these environmental inputs, the transcriptional state remains through several generations (Gardner et al., 2000). Figure adapted from (Cameron et al., 2014). 32

3.1 A visual comparison of linear and logistic regressions. 35
3.2 The Chomsky Hierarchy of Languages. 45

4.1 A comparison of the user interfaces of several popular DNA editing tools. First row: Benchling, GenoCad; Second Row: SnapGene, Teselagen; Third Row: Genome Compiler; Genetic Constructor; Fourth Row: Geneious Prime 52
4.2 The anatomy of a standard BioPart. 53

5.1 The workflow of Part Crafter. 67
5.2 An overview of the interactions between web services. 75
5.3 The relationships between the database tables. 76

6.1 An overview of the data processing workflow. Beginning with sequence data, MNase-seq data for localising nucleosomes, and ChIP-seq data for localising histone modifications, we generated a series of vectors representing a nucleosome with it's associated histone modifications. 83
6.2 The precision-recall curve for the bigram model predicting H3K4ac occupancy. The no-skill line represents a model which always predicts 0. This particular model has both a high accuracy (98.3%) and a high precision-recall AUC (0.925). 85
6.3 The precision-recall curve for the bigram model predicting Htz1 occupancy. The no-skill line represents a model which always predicts 0. While this model has a high accuracy (98.2%), it has quite a low precision-recall AUC (0.093). 86
6.4 The two step model training process. We first trained a standard HMM. Then, using the transition and emission vectors from that model, trained our custom HMM where the transition matrix is re-parameterised by base counts. 89

6.5 The prediction accuracies for three histone modifications, compared between the standard and custom HMMs. All three of these accuracy rates improved when nucleotide data was incorporated into the model by re-parameterising the transition matrix. 93
6.6 The transition matrix for an HMM with 5 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices. 94
6.7 The emission matrix for an HMM with 5 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3. 94
6.8 The genomic weights learned for a custom HMM with 5 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring. 95
6.9 The precision-recall curves for 5-state standard (a) and custom (b) Hidden Markov models, when predicting H3K4ac occupancy. The no-skill lines represent models which always predict 0. The curves are similar to Figure 6.2. The custom HMM demonstrates slightly better performance than the standard. The precision-recall AUC for the standard model is 0.855, while the custom model had a slightly higher AUC of 0.875. 96
6.10 The precision-recall curves for 5-state standard (a) and custom (b) Hidden Markov models, when predicting Htz1 occupancy. The no-skill lines represent models which always predict 0. The curves are similar to Figure 6.3. The custom HMM demonstrates slightly better performance than the standard. The precision-recall AUC for the standard model is 0.025, while the custom model had a slightly higher AUC of 0.064. 97

6.11 An example of one of the simplest annotation trees generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors. 98
6.12 An example of an average-sized tree generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors. 98
6.13 An example of a more complicated tree generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors. 99
6.14 The log likelihood of the validation sequence for the Neural PCFG, averaged over 10 cross folds. This Neural PCFG was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. It is clear that this model was able to do only minimal learning, potentially indicating that not enough training data was provided. 104
6.15 The log likelihood of the validation sequence for the Compound PCFG, averaged over 10 cross folds. This Compound PCFG was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the Neural PCFG, it is clear that this model was able to do only minimal learning, potentially indicating that not enough training data was provided. 105
6.16 The corpus F1 score for the validation sequence with the Neural PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided. 106
6.17 The corpus F1 score for the validation sequence with the Compound PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided. 107

6.18 The sentence F1 score for the validation sequence with the Neural PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided. 108
6.19 The sentence F1 score for the validation sequence with the Compound PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided. 109

D.1 The transition matrix for an HMM with 2 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices. 132
D.2 The emission matrix for an HMM with 2 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications, though those associations appear to be much stronger for State 1. These associations are discussed further in Table 6.3. 132
D.3 The genomic weights learned for a custom HMM with 2 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring. 133
D.4 The transition matrix for an HMM with 3 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices. 134

D.5 The emission matrix for an HMM with 3 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3. 134
D.6 The genomic weights learned for a custom HMM with 3 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring. 135
D.7 The transition matrix for an HMM with 4 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices. 136
D.8 The emission matrix for an HMM with 4 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3. 136
D.9 The genomic weights learned for a custom HMM with 4 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring. 137

List of Tables

2.1 A summary of the Histone Code (Jiménez-Chillarón et al., 2014). 29

4.1 A comparison of a selection of popular DNA editing tools. 49

5.1 Comparison of Software Tools for Finding and Generating BioParts . . . 66
5.2 Fields aggregated from data sources 66
5.3 Summary of the top results for the query provided in Section 2.2. 71
5.4 An overview of the web services behind Part Crafter. 74

6.1 Statistics for the bigram and trigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors. These statistics are averaged over all 26 *Saccharomyces cerevisiae* histone modifications for which we have data. 82
6.2 This table shows the changes in test sequence log likelihood for each state number, and for the standard and custom HMMs. The final three columns show the computed description accuracies for each model. The custom HMMs' log likelihoods offer a modest improvement over those of the standard HMMs. Similarly, the DA slightly improves for the custom HMMs. 91
6.3 The histone modifications each state is associated with, as well as the effects those histone modifications are believed to have on transcription (Jiménez-Chillarón et al., 2014; Magraner-Pardo et al., 2014; Zhao and Garcia, 2015). 91

6.4 This table shows the log likelihoods and description accuracies of the neural, compound, and supervised PCFGs described in Section 6.3.4. The supervised model was trained by taking the rules and their frequencies directly from the input data. The unsupervised trees attempted to learn the rules and their frequencies from unlabelled parse trees. All three of these models were trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. Unsurprisingly, the supervised PCFG yielded both the highest log likelihoods and the highest description accuracies. 103

6.5 The PARSEVAL metrics of the various PCFGs generated using Evalb. 'Labelled' and 'unlabelled' refer to whether the annotation names were used when comparing the spans and generating the similarity scores. It was not possible to perform 'labelled' tests for the unsupervised PCFGs, as the parsed trees are inherently unlabelled. Each of these three models was trained using the same 8,000 annotation trees from chromosomes one through three of the human genome. However, the supervised model performed the best by far in regards to these metrics. 104

B.1 This table shows additional performance statistics for the Bigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors. 119

B.2 This table shows additional performance statistics for the Trigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors. 120

C.1 H2AS129ph, 137 Occurrences 124

C.2 H3S10ph, 74 Occurrences 124

C.3 H3K4ac, 80 Occurrences 124

C.4 H3K14ac, 75 Occurrences 125

C.5 H3K36me3, 91 Occurrences 125
C.6 H3K18ac, 138 Occurrences 125
C.7 H3K4me3, 170 Occurrences 126
C.8 126
C.9 H2AS129ph, 137 Occurrences 127
C.10 H3S10ph, 74 Occurrences 127
C.11 H3K4ac, 80 Ocurrences 127
C.12 H3K14ac, 75 Occurrences 128
C.13 H3K18ac, 138 Occurrences 128
C.14 H3K36me3, 91 Occurrences 128
C.15 H3K4me3, 170 Occurrences 129

Chapter 1

Introduction

The first recombinant DNA molecule was produced in 1972 – five years before the advent of Sanger Sequencing, and eighteen years before the start of the Human Genome Project (International Human Genome Sequencing Consortium, 2001; Jackson et al., 1972; Sanger et al., 1977). Synthetic Biology is a field built by scientists unafraid to run before they could walk, and, evidently, write before they could read. The field has celebrated numerous accomplishments, including the invention of genetic circuits, the production of antibiotics and biodiesel in yeast, the invention of novel cancer therapies, and the design and production of novel organisms.

These accomplishments are staggering, especially considering the technological advancements required to achieve them. However, these technological advancements have not been consistently matched by corresponding advancements in conceptual understanding. While DNA can now be synthesised by robotic assembly lines, and inserted into higher level organism chassis, researchers are still unable to decode even simple DNA sequences such that they can reliably be understood.

While we remain unable to read DNA, our ability to write it will always be limited.

1.1 Contributions

The aim of this work has been to analyse and improve the way Synthetic Biologists design novel DNA sequences. The contributions of this thesis are summarised below:

1. A social science analysis of the models and conceptual metaphors which underpin current DNA design software. This included a review of popular software tools, and a rhetorical analysis of the way in which DNA is represented. We also assessed the strengths and weaknesses of popular conceptual models.

2. The development of a novel software tool for discovering and editing biological parts for use in designing DNA sequences. This involved a rethinking about what researchers need out of their biological parts, as well as the development of a variety of web services which all work together to allow users to search for genomic features using plain text, and to turn the results into parts which are ready to use.

3. The development of several machine learning models with which we assessed the combinatorial patterns of histone modifications.

1.2 Thesis Layout

This thesis will explore software, social science, and machine learning projects to help researchers design Synthetic Biology experiments outside of the traditional parts-based design paradigm. Chapter Four presents a social science analysis of the conceptual metaphors behind popular DNA design software tools, proposing alternative models for representing DNA. Chapter Five presents Part Crafter, a software tool which allows researchers to search for and create biological parts in a flexible, context-aware manner. Chapter Six presents a variety of models with which we represented the combinatorial and relational patterns of histone modifications, with the aim of predicting which epigenetic proteins will bind to novel, designed sequences when inserted into an organism.

Chapter 2

Synthetic Biology, Molecular Biology, and Epigenetics

2.1 Synthetic Biology Prehistory

> The idea is now hovering before me that man himself can act as a creator, even in living nature, forming it eventually according to his will. Man can at least succeed in a technology of living substance. —Jacques Loeb

"Synthetic Biology" is a term which many have historically found hard to define. Sometimes a generously sprinkled buzzword, sometimes a phrase avoided like the plague, and sometimes an epitaph thought by some to be on the way out, the name refers to an amorphous collection of fields and skills, which have sometimes borne other names, and which may change name again at a moment's notice.

According to the Engineering Biology Research Consortium's "brief" definition, "Synthetic biology aims to make biology easier to engineer. Synthetic biology is the convergence of advances in chemistry, biology, computer science, and engineering that enables us to go from idea to product faster, cheaper, and with greater precision than ever before. It can be thought of as a biology-based 'toolkit' that uses abstraction, standardization, and automated construction to change how we build biological systems and expand the range of possible products. A community of experts across many disciplines has come together to create these new foundations for many industries, including medicine, energy and the environment" (EBRC).

The term "Synthetic Biology" appears in scientific literature as early as the turn of the 20th century, referring to the use of directed evolution to "design" new species (Campos, 2009). One of the earliest works using the term in reference to DNA manipulation is an

editorial by Waclaw Szybalski and Ann Skalka. They write "The work on restriction nucleases not only permits us easily to construct recombinant DNA molecules and to analyze individual genes but also has led us into the new era of 'synthetic biology' where not only existing genes are described and analyzed but also new gene arrangements can be constructed and evaluated" (Szybalski, 1978).

Many questions that Synthetic Biology hopes to answer – what is the nature of life? can we create new forms of life? – have been fundamental questions of philosophers and scientists since the dawn of humanity. Aristotle wondered about the "unity and persistence of the individual living being" (Lennox, 2019). Aristotle viewed life as a fundamental property of nature. His ideas on Biology were enduring, still influencing research today. Decartes' theories regarding the nature of life were radically different from those of Aristotle, as he viewed life as the interaction of mechanistic processes. Like several of his contemporaries, he split his questioning of life in two, separating his mechanistic understanding of the nature of life from the more abstract questions on the nature of the mind (Hatfield, 2018; Weber, 2018).

As Chemistry and Biology became formal fields, investigations into the nature of life turned specific, chemical, and mechanical. Scientists attempted to identify the biological and chemical features which were universal in all life forms, and thusly to define life itself. However, in the late 19th and early 20th century, the approaches taken by scientists remained anything but homogenous. Some searched for a fundamental vital force which was responsible for the existence of life itself. Others thought a universal chemical or physical explanation for the complex nature of life was impossibly simplistic. This second camp included J.S. Haldane, who wrote "It is life we are studying in biology, and not phenomena which can be represented by causal conceptions of physics and chemistry."

As time went on, and the zeitgeist of the Industrial Era took hold, it began to heavily affect the nature of experimentation. John Butler Burke, in an attempt to answer the question "could life be produced from nonlife?", completed what would be one of the most famous Synthetic Biology experiments of the early 20th century, though the field did not technically exist yet. Believing in the special, life-giving properties of radium – a common belief at the time – Burke combined the substance with microbes in some unknown way, creating something which was "half-radium and half-microbe," which he called "radiobes" (Campos, 2009). These beings were thought to be half-living, existing "on the frontiers of life, where they tremble between the inertia of inanimate existence and the strange throb of incipient vitality" (Campos, 2009).

Burke agreed with and built upon the work of another prominent scientist at the time, Jacques Loeb. However, Loeb was unimpressed by Burke's radiobes. His mechanistic

view on the fundamentals of life drove him to not only create new life forms in the laboratory, but to do so in a controlled way. His approach, also reminiscent of the zeitgeist of the time, relied on engineering principles to describe a future "technology of the living substance." He wrote, "It is in the end still possible that I find my dream realized, to see a constructive or engineering biology in place of a biology that is merely analytical." His overall goal was to develop a well understood set of steps for producing artificial life, and to, by doing so, learn about the fundamental laws of life itself. His conjecture that scientists need to domesticate biology into an engineerable discipline in order to learn its secrets is one which is still reverberating through biology labs across the world. At the time, many of his contemporaries described his creations as monstrous, their discomfort evoking images of Dr. Frankenstein in his laboratory of ill repute. That discomfort may well still underly critiques of the field today.

While the ideas and experiments of Burke and Loeb led to some sensationalist headlines, making a significant impression on people of the day, the overall field was less affected. Two decades after Burke's radiobes, the focus of many biologists turned away from creating new life from scratch, to altering life which already existed. "Gaining experimental control over evolution was seen as instrumental in such goods as improving crop yields or in developing new mutative varieties. Experiments in mimics of life, primitive life, or artificial life seemed less central" (Campos, 2009).

Throughout the early 20th century, several scientists worked to gain control over the process of evolution, harnessing it to create novel forms of life. Albert F. Blakeslee, in the 20s and 30s, was producing species he described as "made up to order...with definite plan and purpose" (Blakeslee et al., 1933). According to Campos, "Blakeslee's parts-based modular approach to chromosomal dynamics enabled him not only to characterise but to predict and to create novel types of species based on patterns of chromosomal rearrangement" and "far from being opposed to an 'engineering' approach, genetics in this period was much more than mere breeding — with the production of novel mutants, it was the site of some of the most interesting and enduring synthetic successes of the century."

Many of the ideas behind Blakeslee's experiments — his parts-based, modular approach to "engineering" species, the importance placed on playing with nature, as opposed to building from scratch in a test tube — still dominate the field of Synthetic Biology. These ideas, which are in many ways the foundation of almost every modern Synthetic Biology experiment, were first explored almost 100 years ago, yet are still touted as modern applications of engineering principles. This is despite 100 years of molecular biology and

genetics research, as well as 100 year of engineering innovation, including the advent of computing.

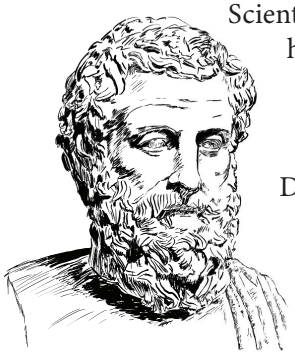
2.2 Modern Synthetic Biology

Like almost all other scientific disciplines, Synthetic Biology has been built upon the work of many researchers over hundreds of years. While, as demonstrated in the previous section, several of the "revolutionary" ideas which serve as the backbone of Synthetic Biology are surprisingly ancient, the field is generally considered to have been formalised around the turn of the millennium (Cameron et al., 2014). However, the dawn of Synthetic Biology could just as easily be ascribed to the 1960s when several early genetic engineering experiments took place, or to the 1950s when researchers discovered the mechanisms behind DNA synthesis. The definition of the beginning of Synthetic Biology depends on the definition of Synthetic Biology.

However, a number of scientific discoveries from the second half of the 20th century clearly affected the early development of the field. In 1961 Francois Jacob and Jacques Monod published their study on the *E. coli* lac operon, hypothesising that DNA regulatory circuits were allowing the organisms to react to its environment. Molecular cloning and PCR were developed in the 1970s and 1980s. In the 1990s, DNA sequencing became largely automated. This, along with the development of several useful computational tools including BLAST, allowed researchers to sequence and compare microorganism genomes with relative ease. The Human Genome Project launched in 1990 as well. At the same time various protocols were being developed to measure RNA, proteins, lipids, and metabolites (Altschul et al., 1990; Cameron et al., 2014; International Human Genome Sequencing Consortium, 2001).

The field of biology was awash in data. Additionally, there were also new ways to interact with an organism's genome. Scientists also viewed DNA's role in a cell in a somewhat different light, the lac operon experiments demonstrating that DNA circuits could be used to toggle between phenotypes. These conditions attracted computer scientists, physicists and engineers into the field, the new technologies available making it seem possible to finally turn life itself into a technology of living substance.

Initial Synthetic Biology experiments were influenced significantly by engineering principles. For example, in 2000, a genetic oscillatory circuit was built, and, in 2001, a genetic toggle box was constructed (Elowitz and Leibler, 2000; Hasty et al., 2001). The scientists involved in these experiments attempted to use model-based design, but the experimental output did not match the predicted output until after several

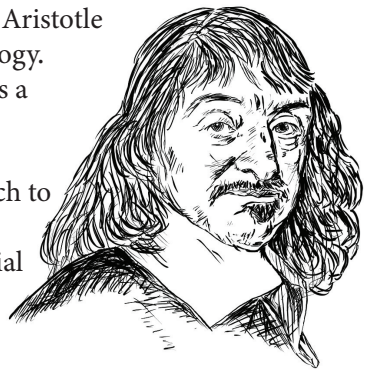


Aristotle
384 – 322 B.C.E.

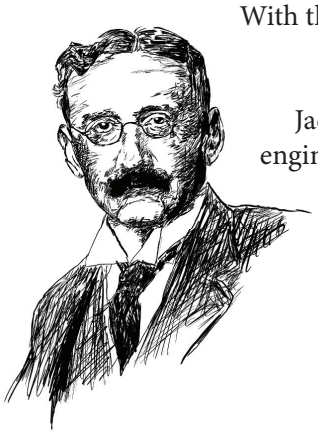
Scientists and philosophers from as far back as Aristotle have influenced the field of Synthetic Biology. Aristotle was the first to study life itself as a scientific discipline.

Descartes described a mechanistic approach to life, in which bodies are viewed, like machines, to be a combination of material and mechanical processes.

The ideas of these philosophers have been especially long lasting and pervasive, their ideas still influencing research today.



René Descartes
1596 – 1650

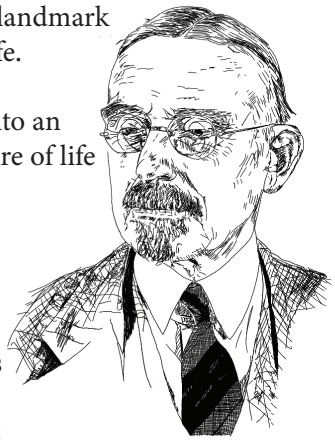


Jacques Loeb
1859 – 1924

With the late 19th and early 20th centuries came landmark experiments investigating the nature of life.

Jaques Loeb strove to domesticate Biology into an engineering discipline, hoping to learn the nature of life by changing it in a series of controlled, reproducible experiments.

Albert Francis Blakesee harnessed the power of directed evolution to produce made to order Jimson Weed strains. His approach to genetic engineering was modular and parts-based – as Synthetic Biology still is today.



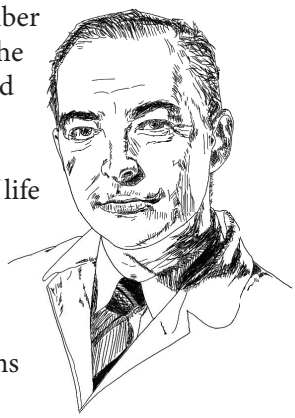
Albert Francis Blakeslee
1874 – 1954

Stanley Miller
1930 – 2007

The middle of the 20th century brought a number of important experiments which influenced the burgeoning fields of Genetic Engineering and Synthetic Biology.

Stanley Miller's experiments into the origin of life demonstrated that complex, organic molecules could be created from simpler, inorganic molecules.

Arthur Kornberg's discovered the mechanisms required for the synthesis of DNA.



Arthur Kornberg
1918 – 2007

Figure 2.1 Prehistory of Synthetic Biology. A selection of scientists and philosophers whose ideas have affected the development of the field of Synthetic Biology

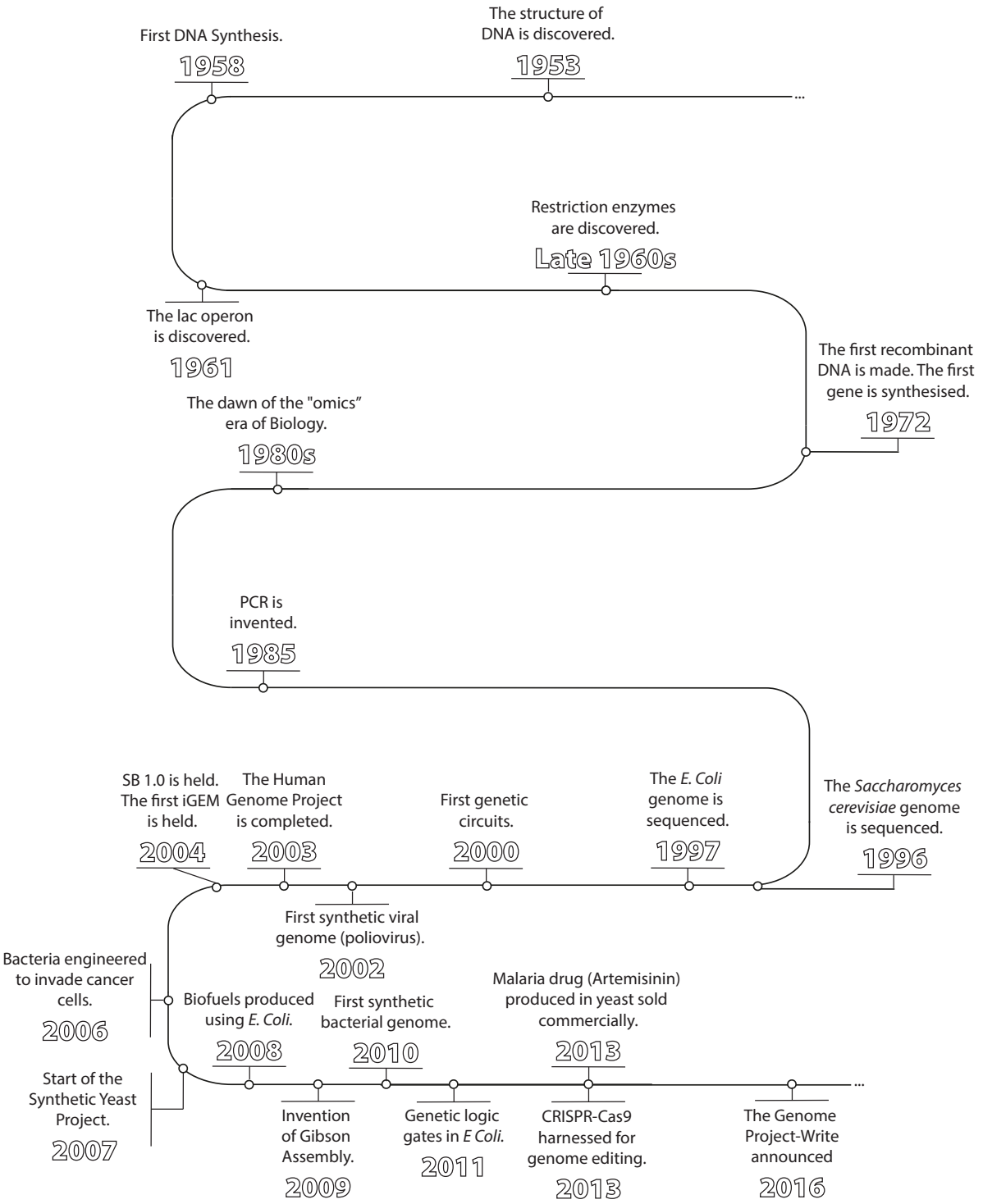


Figure 2.2 A Brief History of Synthetic Biology: Highlighted events in the history of Synthetic Biology (Adler et al., 1958; Boeke et al., 2016; Cameron et al., 2014; Cello, 2002; Dymond et al., 2011; Elowitz and Leibler, 2000; International Human Genome Sequencing Consortium, 2001; Koppolu and Vasigala, 2016; Si and Zhao, 2016; Thiel, 2018; Watson and Crick, 1953)

rounds of trial and error. After these initial experiments, there were a number of similar genetic constructions built, including various feedback modules. Logic gates were even constructed, using a small library of transcriptional regulators (Guet et al., 2002). However, the Synthetic Biology experiments in this early era were still very low-level, took place in a small number of organism chassis (largely *E. coli* and *Saccharomyces cerevisiae*), and focused on reproducing engineering ideas and mechanisms (Cameron et al., 2014).

Synthetic Biology 1.0, the first international Synthetic Biology conference, was held in 2004 at the Massachusetts Institute of Technology. Biologists, engineers, chemists, computer scientists, and physicists came together to discuss the practicalities of turning Biology into an engineer-able medium. Already, researchers were discussing abstraction, modularity, parts-based design, and standardisation.

The Registry of Standard Biological Parts was founded out of MIT in 2003. This was one of the first attempts to store biological parts – the DNA sequences scientists were using to create genetic circuits – in a standardised, easily accessible way. All parts in the registry were required to be compatible with the BioBrick Assembly protocol. However, due to the later inventions of Gibson and Golden Gate assemblies which rendered BioBrick assembly somewhat obsolete, as well as some quality control issues, the registry has been used almost entirely by students in the iGEM competition, as opposed to fully-fledged Synthetic Biologists.

The iGEM competition is a worldwide Synthetic Biology competition held each year by the iGEM Foundation. At the start of the competition each summer, students are given a set of "biobricks" — parts from the Registry of Standard Biological Parts — and are expected to build a new genetic circuit using these parts as well as new ones that they design. At the end of each competition, any newly designed parts are added into the registry, thus increasing the size of the registry with every competition. The following year's teams may then use the expanded registry to build their new designs. The first iGEM competition was held in 2004. Now, upwards of 6,000 students compete each year, contributing to a registry of over 20,000 parts.

Despite the large number of parts in the Registry of Standard Biological Parts, and the fact that every part in the registry has been used in at least one experiment, the registry parts are still insufficiently characterised. This is a problem which has plagued all of Synthetic Biology — not just the parts registry — since its inception. Despite decades of progress in DNA synthesis and genetics research, properly characterising a sequence remains time consuming and expensive. Additionally, if a sequence is characterised in

one genetic context, it is impossible to predict how it will behave when placed in another (Baldwin et al., 2015; Cameron et al., 2014).

Beginning in the mid-2000s, Synthetic Biology experiments saw an increase in both complexity and scope. While bioparts remained context-dependant, the number available and the quality of their characterisation improved. Designed sequences behaved more predictably, and DNA synthesis was becoming cheaper (Cameron et al., 2014; Gibson et al., 2009).

In this time, several relatively sophisticated engineering goals were accomplished in genetic circuits. In 2009, a genetic circuit was created which could count events. Recombinase-mediated DNA rearrangement was created to store a "memory" of the event of interest (Siuti et al., 2013). In 2011, logic gates were implemented in full in E. Coli (Tamsir et al., 2011). Scientists and engineers were building the components required to construct cellular computers. Indeed, many scientists at the time felt as if programmable organisms were easily within grasp.

By 2013, RNA circuits were being constructed, and the CRISPR-Cas9 system had been harnessed for genome engineering. Decreased DNA synthesis costs improved the feasibility of metabolic engineering projects. Synthetic Biologists became involved in drug discovery and development. Artemisinin, an anti-malarial drug, was produced in yeast, and sold commercially. Phage-based therapies built using Synthetic Biology began being investigated (Cameron et al., 2014).

As new technologies emerged, DNA sequencing and synthesis became even cheaper, and the field grew in both size and scope, Synthetic Biology research split into three categories. A. O'Malley et al. (2008) describes these three modes of Synthetic Biology as: DNA-based device construction, genome-driven cell engineering, and protocell creation.

DNA-based device construction consists of the assembly of genetic circuits from the bottom up using standardised components. This mode of Synthetic Biology was dominant during the beginning of the field, and much of the work discussed so far in this section would fall into this category. The design of these genetic circuits relies upon engineering principles including parts-based design, standardisation, and abstraction. Many of the experiments which fall into this category are those which attempt to replicate engineering concepts in cells, such as genetic logic gates, oscillators, and toggle-switches. However, this category would also include experiments in which, for example, a natural product's pathway is inserted into a chassis via a plasmid or CRISPR-Cas9.

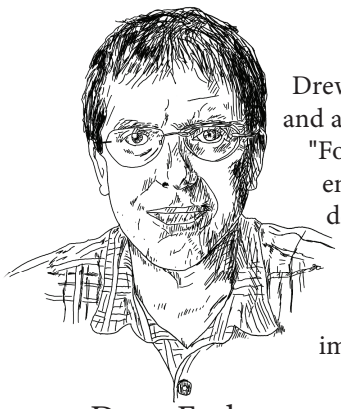
Genome-driven cell engineering involves synthesising an entire genome and inserting it into a living cell. Examples of this include *Mycoplasma laboratorium*, the first synthetic bacterial genome, and the Synthetic Yeast Project. Genomes are designed using both a

bottom-up and top-down approach. The goals of these experiments have varied greatly, and have sometimes been many faceted or amorphous. For example the goals of the Synthetic Yeast project are to "to answer a wide variety of profound questions about fundamental properties of chromosomes, genome organization, gene content, function of RNA splicing, the extent to which small RNAs play a role in yeast biology, the distinction between prokaryotes and eukaryotes, and questions relating to genome structure and evolution" (Sc2.0). However, one of the main goals of these projects is often to merely build the genome for the sake of building it, because, in doing so, new technologies will be developed.

The last category, protocell creation, seems to receive both the least attention and the least funding. A. O'Malley et al. (2008) describes the approach as an "'intermediate' strategy (neither top-down nor bottom-up, but 'reconstructive') [involving] a 'semi-synthetic' approach in which extant genes and enzymes are put into vesicles (often liposomes) to produce 'semi-artificial' cells." A. O'Malley et al. (2008) describes the goal of this mode of Synthetic Biology as the creation of minimal cellular systems using "a great deal of theoretical modelling, which attempts to capture underlying principles of cellular dynamics and molecular self-assembly in order to guide experimental work more effectively." However, since the publication of A. O'Malley et al. (2008), there has been quite a lot of development in cell-free systems for Synthetic Biology. While many aspects of these experiments may fall into this mode of Synthetic Biology — for example, the construction of subcellular fragments to isolate biological reactions from the influence of a larger cell system — the goals of these experiments tend to align more with other modes of Synthetic Biology than with the creation of minimal cells purely for the sake of biological research.

The growth of Synthetic Biology through the 2010s was enabled by the development of new technologies, protocols, and software tools. At the dawn of field, DNA was designed in word documents, and alignments often done by hand. Now, dozens of software companies have sprung up, their main goal to ease the DNA-design user experience, to allow for predictable experimental behaviour of sequences. However, despite the three modes of Synthetic Biology described above, these software tools largely cater to one category of experiments, DNA-based device construction. Additionally, these tools by and large rely upon the parts-based design model. This phenomenon, and why it is problematic, will be discussed in depth in Chapter 4.

Since at least as far back as Jaques Loeb, and likely before, scientists have longed to tame Biology, turning it into an engineering discipline, ostensibly to learn its secrets through tinkering. George Church and Anthony C. Forster once wrote "Until we can

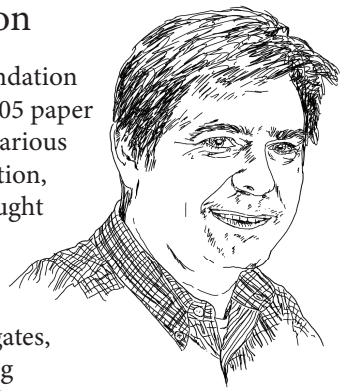


Drew Endy
Stanford

DNA-Based Device Construction

Drew Endy is both president of the BioBricks foundation and a co-founder of the iGEM competition. His 2005 paper "Foundations for engineering biology" laid out various engineering principles -- including standardisation, decoupling, and abstraction -- and how they ought to be applied to Synthetic Biology.

Christopher Voigt has worked both to implement genetic programs using DNA logic gates, as well as to implement a DNA programming language based on a set of several thousand *E. Coli* parts.



Christopher Voigt
MIT

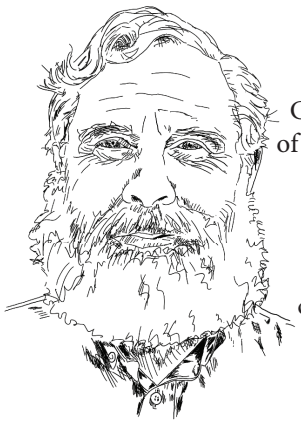
Jef Boeke
NYU

Genome-Driven Cell Engineering

Jef Boeke currently leads the Synthetic Yeast (Sc2.0) Project. This project involves the synthesis of an entire *Saccharomyces cerevisiae* genome which has been "optimised." He is also responsible for the newly begun Genome Project-Write consortium.

Craig Venter was a key player in the Human Genome Project, and was also one of the leaders of the synthesis of *Mycoplasma laboratorium*, the first synthesized bacterial genome.

Craig Venter
J. Craig Venter Institute

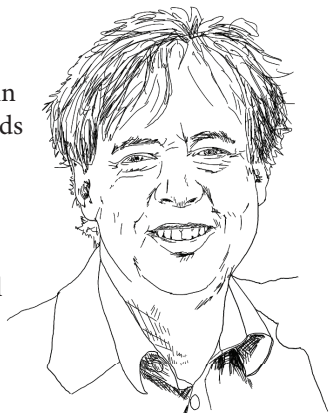


George Church
Harvard

Protocell Creation

George Church is part of Harvard's Origin of Life Initiative. In his 2006 paper "Towards synthesis of a minimal cell," he details his design of a minimal genome.

Steven Benner studies pre-biotic chemistry, the recreation of the chemical origin of life. He has recently shown that the building blocks for RNA can be made under prebiotic and plausible conditions.



Steven A. Benner
Foundation for Applied Molecular Evolution

Figure 2.3 The three modes of Synthetic Biology. A selection of scientists presented with examples of their work which fit into one of the three modes of Synthetic Biology. It is worth noting that each of these scientists has at some point worked on experiments in all three categories. (Boeke et al., 2016; Endy, 2005; Forster and Church, 2006b; Gibson et al., 2010; Kim and Benner, 2017; Nielsen et al., 2016; Richardson et al., 2017; Tamsir et al., 2011)

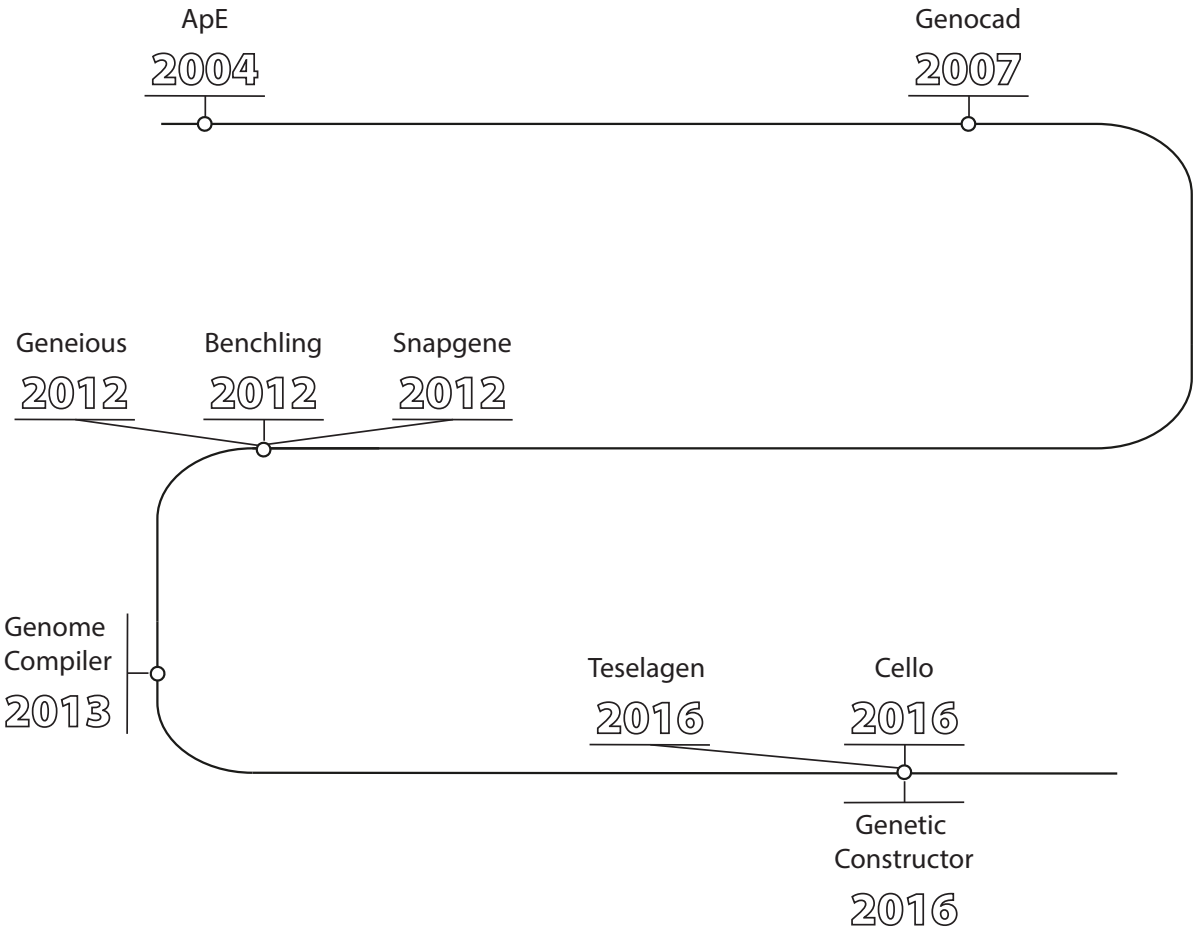


Figure 2.4 A timeline of software tool development in Synthetic Biology

assemble a form of life in vitro from defined, functionally understood macromolecules and small-molecule substrates, how can we say that we understand the secret of life?" (Forster and Church, 2006a). However, not all or even most modern Synthetic Biology experiments are striving to uncover a fundamental biological truth. Within the field exists a dichotomy between those experiments which strive to build something for the sake of building it, and those which strive to do "pure" biological research. According to Keller (2009), "What is of particular interest in many of these discussions is the repeated assertion that making is knowing, that the building of a machine is not only of obvious practical utility, but that that process is also, in itself, the royal route to an understanding of the machine. Making is knowing, and knowing is making." The "true" definition of Synthetic Biology, then, exists somewhere between engineering and science, practicality and abstraction. Between Rosalind Franklin, who said "I just want to look, I don't want to touch," and Richard Feynman, who said "What I cannot create, I do not understand" (Keller, 2009).

However, when something exists between many things, encompassing many concepts, fields, and philosophies all at once, how do you model it? How do you build one user interface through which to access the entire field of Synthetic Biology?

2.3 An Overview of Relevant Molecular Biology

The nature of Synthetic Biology and, in particular, whether it is science or engineering or technoscience, may still be up for debate (Keller, 2009). However, the genetic circuits and whole genome-based projects ultimately exist in real, living cells, and with that comes some baggage. When representing DNA as a collection of isolated pieces, Synthetic Biologists ignore many aspects of genetics which are intrinsic to Biological function.

2.3.1 DNA

DNA was first discovered in 1869 by chemist Friedrich Miescher (Dahm, 2008). While he was able to identify that this molecule, which he called nuclein, had chemical properties unlike any protein, another 50 years would pass before the importance of this molecule was realised by the scientific community. In 1944, Oswald Avery and colleagues published a paper describing "hereditary units," composed of DNA (Avery et al., 1944).

Relying on X-ray crystallography work by Rosalind Franklin and Maurice Wilkins, James Watson and Francis Crick eventually discovered the structure of a DNA molecule.

Their model, which remains largely the same today, has several important features (Pray, 2008).

- DNA consists of two strands, wrapped together in a helix. The two strands are connected using hydrogen bonds. Each strand is made of a sequence of nucleotides. Adenine is always bound to Thymine, and Guanine is always bound to Cytosine;
- The DNA helix is anti-parallel. Each strand has a 5' and a 3' end. The 5' end of one strand is paired with the 3' end of its partner strand;
- Each nucleotide is made of one of the four nitrogen-containing bases (Adenine, Thymine, Guanine, or Cytosine), a sugar molecule, and a phosphate group. Each of the outer edges of the nitrogen-containing bases are available for potential hydrogen bonding, while the inner edges are involved in base-pairing. This allows other molecules, such as an enzyme, to bond to the DNA molecule.

2.3.2 DNA Replication

Watson and Crick ended their famous publication describing the structure of the DNA double helix by saying "It has not escaped our notice that the specific pairing we have postulated immediately suggests a possible copying mechanism for the genetic material" (Watson and Crick, 1953). However, while the process of DNA replication may in some ways be intuitive given the structure of the DNA molecule itself, decades of research have shown the process to be more complicated and mysterious than Watson and Crick predicted.

DNA replication is more complicated in eukaryotes than in bacterial cells, but even in these smaller organisms there are four major steps: initiation, unwinding, primer synthesis, and elongation. During the initiation and unwinding stages, the hydrogen bonds keeping the two DNA strands together are broken apart by DNA helicases. In the primer synthesis stage, short stretches of nucleotides, about 11 to 12 bases in length, are created by an enzyme called primase. Finally, in the elongation stage, DNA polymerases extend the short primer sequences by adding nucleotides one by one, following along the template strand. DNA replication always moves from the 5' to the 3' end of the sequence. Because A is known to always bind with T, and C always with G, the newly created DNA strand is exactly as the one before it (Alberts, 2008; Kornberg, 1960; Losick and Shapiro, 1998).

Through this process, one DNA molecule becomes two, which can then be passed on to daughter cells, which will inherit all of the genetic instructions contained therein.

2.3.3 The Central Dogma

DNA molecules are sometimes described as the blueprint of life, providing the instructions directing the behaviour of a cell throughout its lifetime. The ways in which these instructions are followed are many faceted. However, one of the most important ways in which DNA dictates the nature of a cell is by encoding instructions for producing all of the proteins a cell must create in order to perform just about every cellular function. The steps through which these proteins are produced, transcription and translation, are known as the Central Dogma of Molecular Biology (Alberts, 2008; Pierce, 2012). A summary of the Central Dogma is shown in Figure 2.5.

Transcription is the process through which RNA is produced. RNA is a molecule similar to DNA, in that it is also comprised of nucleotides. However, it is single-stranded, and thymine is replaced with its unmethylated form, uracil. RNAs serve important biological purposes outside of the process of transcription. However, messenger RNAs, often called mRNAs, are particularly important within the central dogma.

mRNAs are produced when RNA polymerase II binds upstream of a gene, in a part of the sequence called a promoter (Alberts, 2008; Crick, 1958; Logan et al., 1987). Promoters can be described as "strong" if they are associated with increasing the rate of transcription, or "weak" if they have the opposite effect. Eukaryotic genomes have enhancer sequences which can further increase the rate at which their associated genes are transcribed. Increased or decreased rates of transcription, and therefore gene expression, can majorly impact the cell. If these transcription rates are even slightly altered, this can lead to diseases, such as cancer.

Just like in DNA replication, once transcription is initiated, the DNA helix unwinds and one nucleotide is added at a time according to what is specified in the template strand. Transcription ends once an area of the sequence is reached called a terminator. Terminator sequences can affect the rate of mRNA production to the point that their influence can override the effect of a strong promoter on gene expression. Additionally, terminator sequences can affect the half-life of the mRNAs produced. Gene expression rates are dependent on the amount of available mRNA, which means that if less is produced, and what is produced decays faster, then a gene can effectively be silenced. The makeup of terminator sequences, much like promoter sequences, is not fully understood, but is known to be extremely important in the process of transcription (Alberts, 2008; Crick, 1958; Logan et al., 1987). The layout of the promoter and terminator sequences along a DNA sequence are shown in Figure 2.6.

Once an mRNA is created, it travels to a ribosome, which resides in the cytoplasm, to begin the process of being translated in protein. In eukaryotes, this requires the

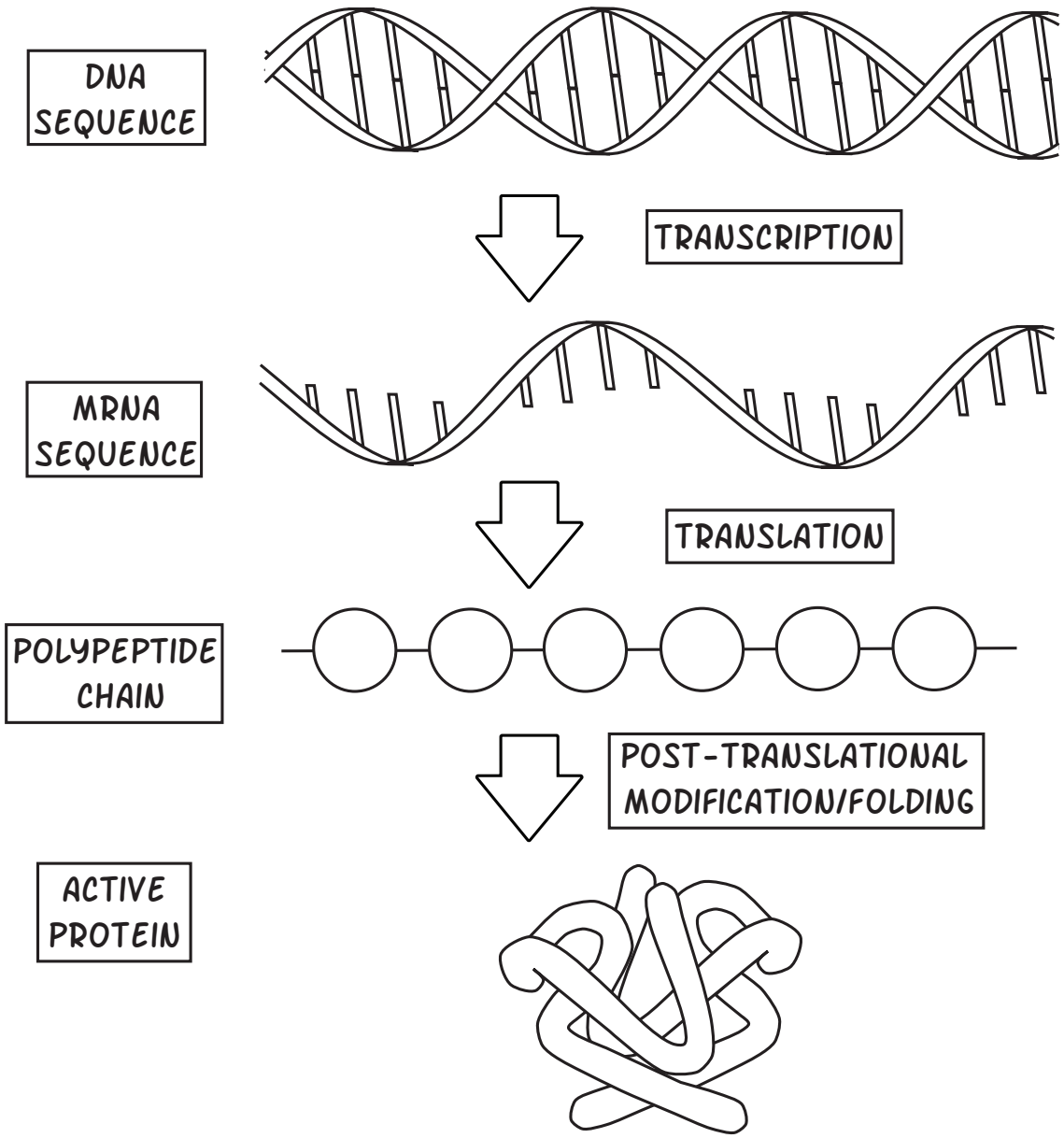


Figure 2.5 An overview of the Central Dogma of Molecular Biology (Pierce, 2012).

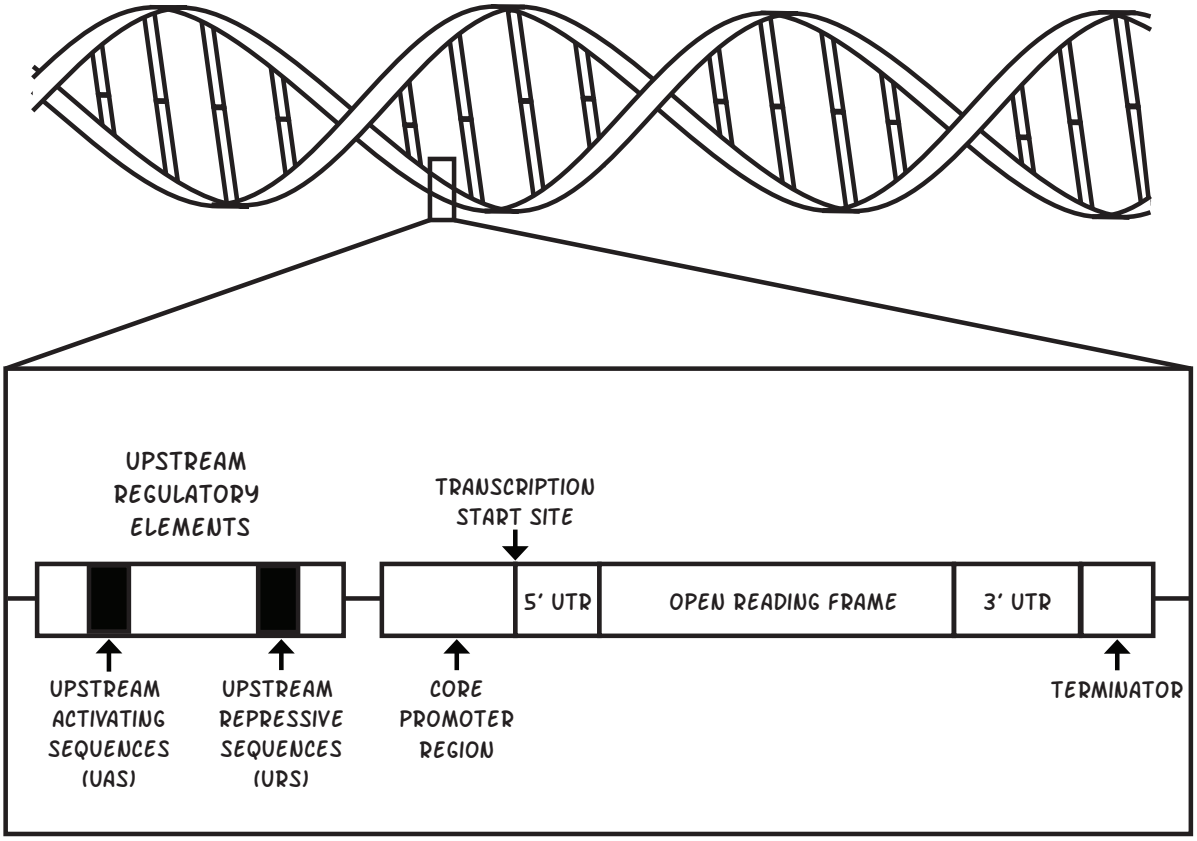


Figure 2.6 An overview of the structure of promoters and terminators (Blazeck and Alper, 2013).

mRNA to leave the nucleus. In prokaryotes transcription and translation both happen in the cytoplasm, and translation can occur on one end of an mRNA while transcription continues on the other.

The beginning of an mRNA sequence is known as a 5' UTR, or untranslated region. This piece of the mRNA contains the ribosome binding site.

Ribosomes are made of two separate pieces which join together on the mRNA sequence. These subunits contain tRNA and rRNA — transfer RNA and ribosomal RNA. mRNA is read in triplet, using a system called the Genetic Code. Each section of three bases is known as a codon, and each codon is associated with a particular amino acid. This Genetic Code is the same for both eukaryotes and prokaryotes. One end of a tRNA recognises a codon, while the other recruits its associated amino acid. The rRNA is responsible for attaching each new amino acid to the growing polypeptide chain.

The ribosome moves along the mRNA sequence from the 5' to the 3' end, the tRNA and rRNA working together to grow the polypeptide chain as specified by the mRNA sequence. This continues until one of three stop codons is reached, namely UAA, UAG, or UGA. Once one of these codons is reached, the mRNA is detached from the ribosome and translation is complete (Alberts, 2008; Crick, 1958; Logan et al., 1987).

2.3.4 Splicing

Between transcription and translation, an RNA molecule oftentimes goes through several processing steps. One important example of this, which occurs in most eukaryotic cells and some prokaryotes, is splicing.

Before a mature mRNA is produced, something called a pre-mRNA is first made. Several things must occur before the pre-mRNA is ready to be transcribed into protein. First, the transcript gains a 5' cap and a poly-A tail. These are necessary to both protect the transcript and to allow it to travel out of the nucleus to the ribosomes. Then, the spliceosome, a complex made of both RNA and protein, comes and removes specific sites, called introns, from the RNA. The remaining pieces of the mRNA, called exons, are then put back together to form a mature mRNA. Only the exons are eventually translated into protein (Alberts, 2008; Pierce, 2012).

Through a process called alternative splicing, one gene can be transcribed into multiple mRNAs. For example, once an mRNA's introns are removed, the exons may be stitched back together in a variety of ways, including potentially leaving behind one or more exons. Through this mechanism, one gene can potentially encode multiple proteins.

The exact purpose of splicing is still a biological mystery. However, alternative splicing seems to be an important mechanism by which one gene can lead to various protein

outputs. Additionally, introns can serve important roles in regulating gene expression. It is also hypothesised that the existence of splicing allows for more frequent mutation events which could lead to quicker development of evolutionary advantages (Alberts, 2008; Keren et al., 2010; Pierce, 2012).

2.3.5 Sequencing Technologies

The Human Genome Project was completed in 2003. Learning the exact sequence of nucleotides which lives in every cell of a human being has led to untold numbers of biological and biomedical advancements. This project, the product of 13 years of intensive work, relied upon Sanger Sequencing, which was invented by Fred Sanger in 1977.

The first step of Sanger Sequencing is generating DNA fragments of various lengths using purified DNA polymerase. During this process, dideoxynucleotide triphosphates (dATP, dGTP, dCTP, and dTTP) are incorporated. These are similar to standard nucleotides, except they lack the 3' hydroxyl on one end needed to form a bond with a subsequent nucleotide. Therefore, once one of these ddNTPs is incorporated into the DNA fragment by the polymerase, the fragment can grow no further. One of these ddNTPs can be included at any time, resulting in a variety of fragment lengths. This reaction is done four times, once for each of the four ddNTPs, which each take the place of a different standard nucleotide. Then, the fragments are separated by size. It is then possible to determine which base is at the end of each fragment, based on which ddNTPs caused the fragment to stop growing (Sanger et al., 1977).

Sanger Sequencing saw many reiterations and improvements over the thirty years it spent largely dominant. However, it remained expensive and in some ways inefficient. The next sequencing method to take hold, appropriately called Next-Generation Sequencing, was in many ways a vast improvement. Because it is both cheaper and faster, its development has made it possible for researchers to regularly sequence more and larger genomes, which has vastly improved our understanding of genetics. While there are a number of NGS techniques, they all rely on parallelisation to sequence a large number of sequence fragments at once (Behjati and Tarpey, 2013). Because NGS techniques produce shorter sequencing reads than Sanger Sequencing, bioinformatics techniques for the assembly of these reads into an entire genome sequence are especially important.

ChIP-sequencing

Sanger Sequencing and NGS are techniques for identifying the nucleotide sequence of a DNA molecule. However, DNA molecules do not exist in a vacuum. ChIP-sequencing

(ChIP-seq) is a technique for identifying which pieces of DNA interact with a specific protein. This technique is especially useful for researchers studying epigenetics, as it can be used to find the locations of transcription factors, histone modifications and RNA polymerases along the genome.

In general, the steps of ChIP-seq are:

- Proteins are cross-linked to their bound DNA. This is generally done using formaldehyde.
- Cells are homogenised and lysed.
- The chromatin is sheared using sonication.
- The chromatin is then immunoprecipitated with special magnetic beads bound to antibodies. This extracts the relevant DNA sequences.
- The resulting fragments are sequenced.

It is then possible to use bioinformatics tools called peak callers to identify the binding sites of the proteins of interest (Schmidt et al., 2009).

MNase-seq

MNase-sequencing is used to find the locations of nucleosomes. This is done in a way which is quite similar to ChIP-seq (Kelly et al., 2012; Yan et al., 2016).

- DNA is treated with MNase, an enzyme which breaks the sequence into fragments. Areas of the sequence which are bound to nucleosomes will be protected from the enzyme.
- The DNA fragments are sequenced.
- Bioinformatics tools are used to align the fragments to a template sequence, and identify the midpoint of nucleosome locations by averaging cut site locations.

2.3.6 Synthetic Biology Sequence Editing Techniques

The design of novel DNA sequences is largely done entirely computationally, through sequence editing software programs. However, once a new sequence is designed, is has to be inserted into its host cell. There are a variety of methods researchers use to insert these sequences into their organism of choice, and each of these methods places its constraints on the sequences which can be designed and the experiments it can be used for.

The 1970 discovery of restriction enzymes allowed researchers to, for the first time, isolate the DNA sequences of specific genes (Smith and Welcox, 1970). This, along with the 1967 discovery of DNA ligases, made it possible for researchers to create recombinant DNA sequences consisting of several genes of interested copied and pasted together (Roberts, 2005; Weiss and Richardson, 1967). Plasmids, which had been discovered almost two decades earlier, became widely used for inserting new DNA sequences into cells. This was done using transformation. While this method of copying and pasting genes onto plasmids and inserting those plasmids into new organisms was the first gene editing technique, it is one that is still used in biology labs today. Many synthetic biology experiments which take place in single cellular organisms rely on plasmids to deliver designed sequences.

However, since 1970, researchers have developed myriad ways of performing larger-scale, more precise, and more efficient genome editing experiments, even in multicellular organisms. Modern DNA editing techniques typically rely upon engineered nucleases. There are four types of engineered nucleases: meganucleases, Zinc-finger nucleases (ZFNs), transcription activator-like effector nucleases (TALENs), and CRISPR/Cas9 (clustered regularly interspaced short palindromic repeat/CRISPR associated protein). However, meganucleases are less commonly used.

These nucleases are capable of making double-stranded breaks in the DNA sequence at specific, pre-determined places in the genome. When one of these breaks occur, the cell naturally attempts to repair it. This process is then exploited in order to introduce changes to the DNA sequence (Carlson et al., 2012; Grizot et al., 2009; Li et al., 2011; Shukla et al., 2009).

There are various types of edits which can be made. For example, researchers sometimes make small DNA changes, often inserting or deleting just a handful of nucleotides. This small change can turn off or repress the function of a gene. It is also possible to remove a larger stretch of DNA using nucleases, by making cuts at either end of the piece to be removed, and allowing the cellular machinery to stitch the broken ends back together. Longer stretches of DNA can also be inserted using a piece of DNA with special sequences similar to the cut site to repair the break.

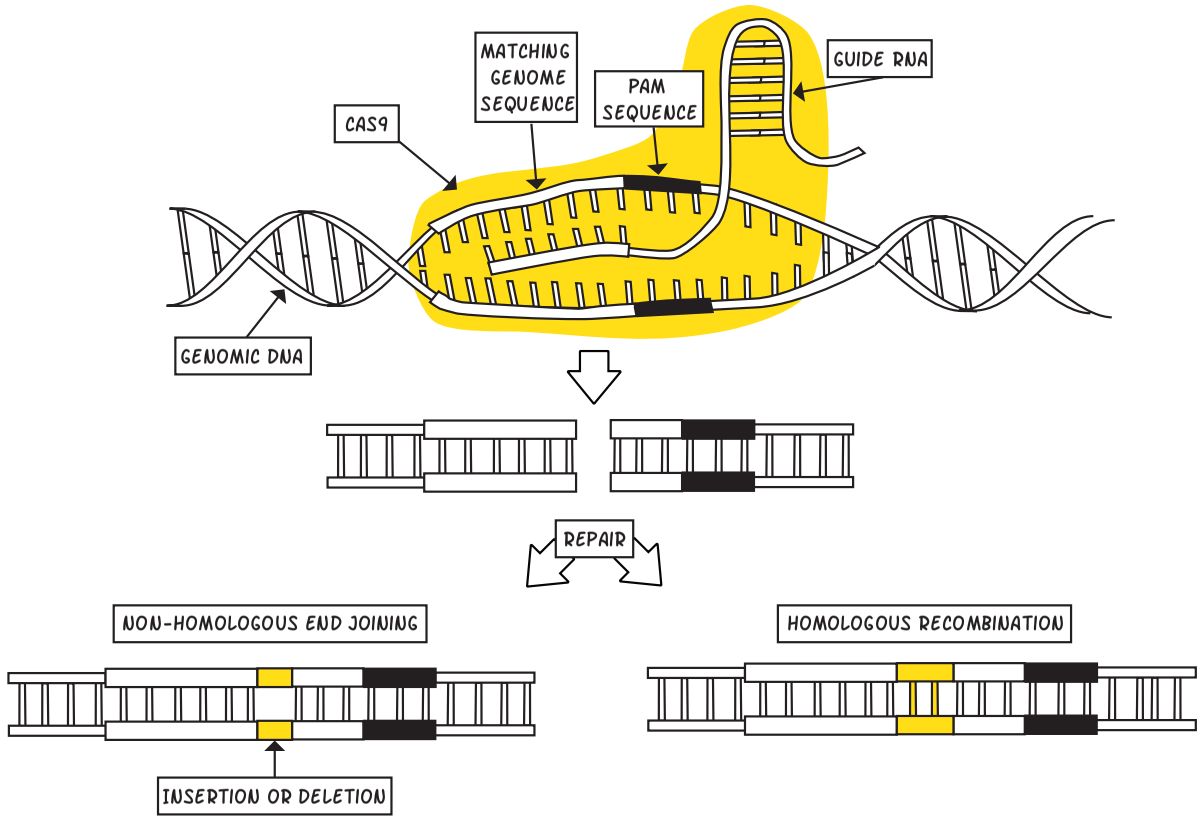


Figure 2.7 An overview of the CRISPR/Cas9 system. A designed guide RNA targets a specific region in a gene. The Cas9 enzyme creates a double-stranded break. The cell then undergoes either nonhomologous end joining or homologous recombination. NHEJ tends to be used to disrupt genes, by introducing small insertions or deletions. HDR can be used to introduce a new, edited sequence. (Barrangou, 2014; Khan et al., 2018).

Zinc-Finger Nucleases

Zinc-finger nucleases consist of the FokI nuclease, along with a number of zinc-finger proteins, which each bind to a small number of nucleotides. It is possible to target a particular sequence by making different combinations of zinc-finger proteins, though this often requires some trial and error. A pair of ZFNs must be made, as two FokI molecules are required to come together to make a double stranded cut (Carroll, 2011).

TALENs

Similarly to ZFNs, TALENs rely on a pair of FokI nucleases to make double stranded cuts. TALENs are engineered to bind to specific DNA sequences using transcription activator-like effector (TALE) domains. There is a different TALE domain for each nucleotide, so it is relatively easy to design a set of TALE domains which will find the correct sequence. As with ZFNs, two TALENs must be made, one for each strand (Joung and Sander, 2013).

CRISPR/Cas9

CRISPR/Cas9 is the most commonly used system for genome editing. CRISPR is the part of the system which targets a specific DNA sequence. It consists of an RNA molecule, which is often called a guide, which binds to a particular sequence based on complementary base-pairing. Cas9 is the nuclease responsible for cutting the DNA. While CRISPR/Cas9 is cheap, efficient, and effective, it is thought by some to be prone to off-target effects. Additionally, the Cas9 gene is large enough that it is difficult to use in gene therapies using adeno-associated viruses (Pennisi, 2013; Tachibana, 2019).

2.3.7 Epigenetic Control of Gene Expression

Every cell in the human body has the same DNA. Despite that, liver cells look and behave completely differently than brain cells, skin cells, and heart cells. While human DNA contains the blueprint for every possible human cell, other biological mechanisms control which plans are actually built.

Epigenetics is the study of heritable biological changes which control which genes are expressed and which are silenced, without involving any change in genome sequence. The word “epigenetics” comes from C. H. Waddington’s 1942 paper The Epigenotype (Waddington, 2012). In 1957 Waddington would go on to describe cell differentiation, the process by which relatively neutral stem cells turn into specific cells (such as liver,

skin, or heart cells), as unidirectional. Like a marble rolling down a hill, once the process moves in one direction, it was, according to Waddington, unable to return to any previous condition. While this is still generally thought to be true for the natural progression of differentiation, the idea has been formalised and refined by more recent biologists.

The actual molecular processes through which differentiation as well as other epigenetic changes occur are not fully understood. One intuitive aspect of epigenetics is DNA packaging. Figure 2.9 shows an overview of DNA packaging in eukaryotes. If all of the DNA in one diploid, human cell was stretched out, it would be over 2 metres long. In order for it all to even fit in our cells, DNA must be significantly compacted, folded on itself again and again. This condensed mass of DNA, along with the proteins bound to it, is called chromatin.

At the lowest level, the epigenetic proteins responsible for chromatin folding are called histones. Histones, with a slight positive charge, bond closely to DNA, which has a slight negative charge. Histones group together into nucleosomes, which are made of two each of the histones H2A, H2B, H3, and H4. Each nucleosome is wrapped approximately twice in DNA, taking up about 150 base pairs. Nucleosomes appear semi-regularly throughout a eukaryotic genome, resembling "beads on a string" (Bednar et al., 1998; Kornberg, 1974).

On top of the folding around nucleosomes, DNA is further folded into more complex structures. However, the exact structure of these 30-nanometer fibres is not currently known. Additionally, the structure of the chromatin changes during the cell cycle. For example, the chromatin is especially compacted during metaphase.

Because for transcription to occur the relevant cellular machinery must be able to access the DNA sequence, chromatin structure is directly related to gene expression levels. At any given moment, the vast majority of a cell's DNA is tightly coiled, preventing access. Only when and where the chromatin is "open" is transcription possible. Because all cells in an organism contain the same DNA, varying gene expression levels is fundamental to the process of differentiation. Additionally, these changes in gene expression level are largely inherited in mitosis (Gibney and Nolan, 2010).

However, there are more well-studied mechanisms through which gene expression is controlled epigenetically. Most steps in the process of gene expression are influenced in some way by these mechanisms. These include:

1. cytosine methylation
2. post-translational modification of histone proteins and remodelling of chromatin
3. RNA-based mechanisms.

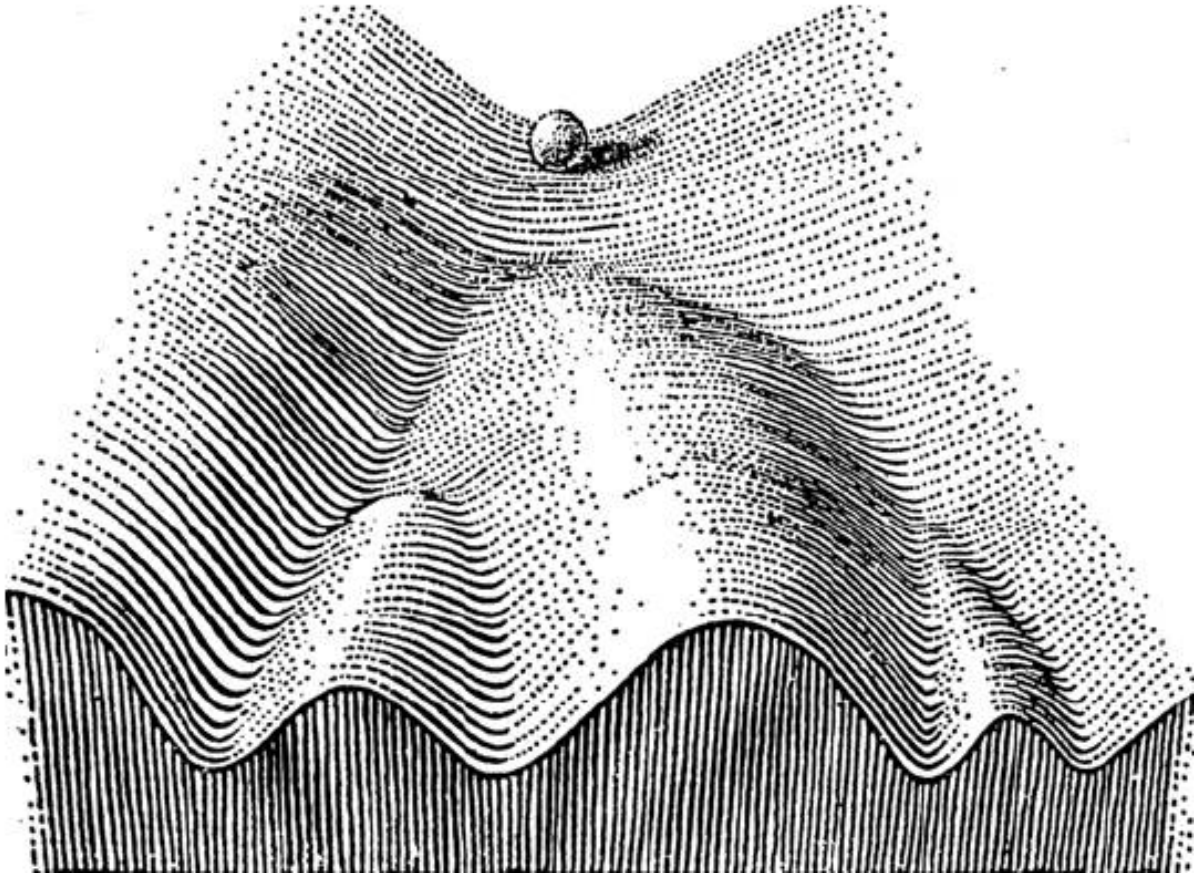


Figure 2.8 The original drawing of Waddington's Epigenetic Landscape, published in Waddington (1957).

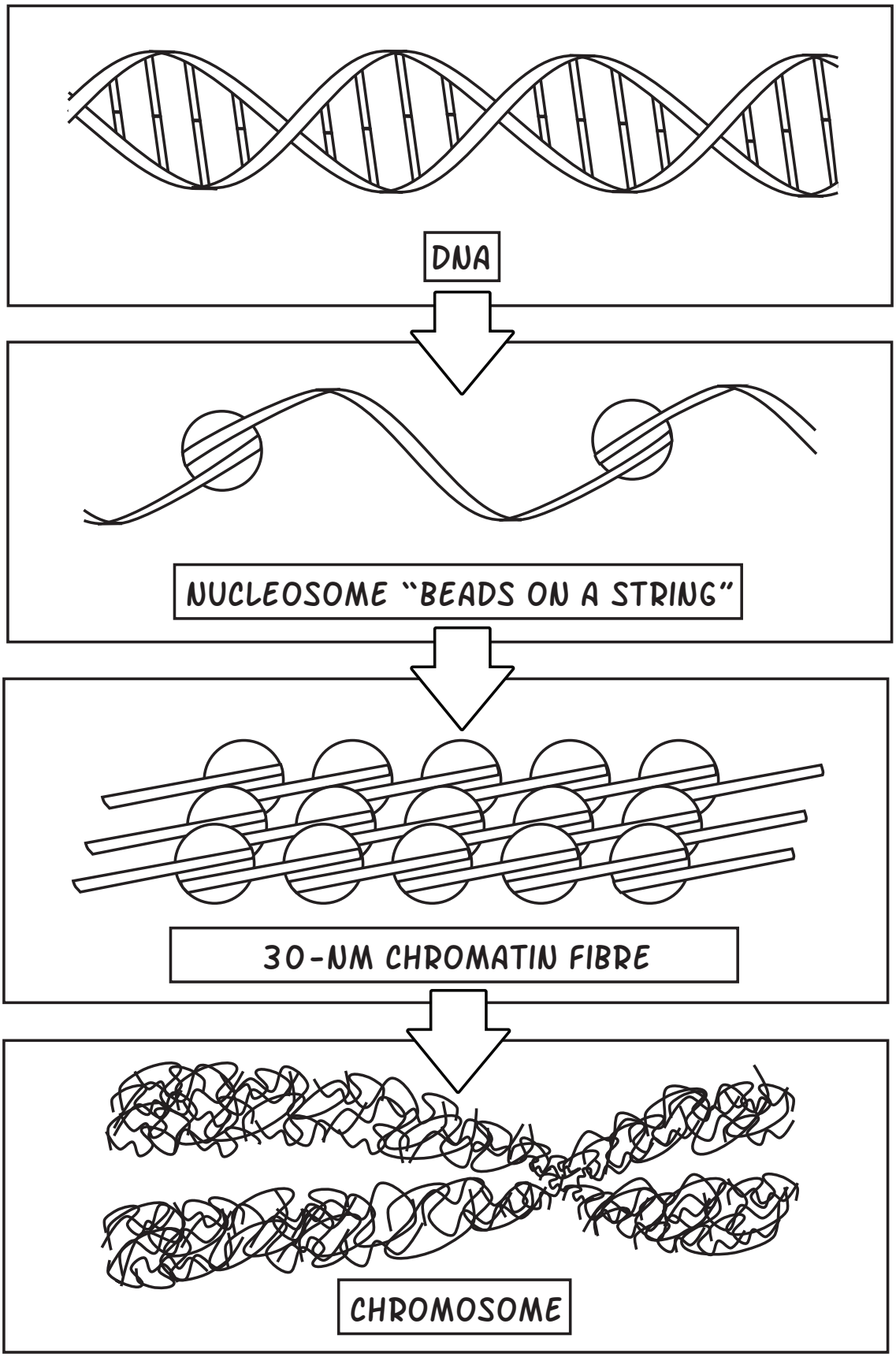


Figure 2.9 An overview of the structure of a chromosome.

Methylation

Broadly, methylation is the addition of methyl groups to DNA molecules. While it is possible for both adenine and cytosine to be methylated, cytosine methylation has been studied far more than adenine methylation. Approximately 3% of cytosines in the human genome are methylated, and in general 3-8% of cytosines in mammalian genomes are thought to be methylated (Nafee et al., 2008). However, yeast strains have much lower cytosine methylation rates, ranging from 0.014 to 0.364% (Tang et al., 2012).

In mammals, cytosine methylation only occurs when the cytosine comes before a guanine. These instances of methylation are commonly referred to as CpGs, as the p represents the phosphodiester bond linking the two nucleotides. Methylation is in general associated with the repression of gene expression. This repressed state is generally inherited by any daughter cells.

In mammalian genomes, the combination of a C nucleotide followed by a G is rarer than one would expect in entirely random conditions. However, there are some locations in the genome where this combination happens more often than expected (Gardiner-Garden and Frommer, 1987). These regions are known as CpG islands. About half of CpG islands are associated with promoter regions. It is theorised that the other half are associated with the transcription start sites of non-coding RNAs. Several experiments have been conducted which showed that gene expression levels could be measurably affected by the methylation and demethylation of specific CpGs within specific genes (Gibney and Nolan, 2010).

While not everything is known about DNA methylation, it is widely accepted to play a major role in the regulation of gene expression (Illingworth et al., 2008). DNMTs, the group of enzymes responsible for methylation, are thought to interact with transcription factors. This interaction then leads to methylation in promoter regions, which in turn is thought to influence the chromatin and transcriptional machinery (Gibney and Nolan, 2010).

Various methyl-binding proteins (MBPs) exist which preferentially bind to CpGs. These MBPs repress trascription through a variety of mechanisms, including the recruitment of co-repressors and histone deacetylases, which cause chromatin compaction. Some MBPs serve as a physical barrier preventing transcriptional machinery from accessing the DNA (Gibney and Nolan, 2010).

Post-Translational Modification of Histone Proteins and Remodelling of Chromatin

At the lowest level, the basic unit of the chromatin is the nucleosome, which is made of eight histones. Each of these histones has a protruding 'tail.' Each of these tails can experience post-translational modifications, known as histone modifications. These modifications are thought to change over time, and are thought to be highly influential on gene expression. Their influence is thought to be exerted through three mechanisms. First, histone modifications affect chromatin structure. Second, they prevent or disrupt the binding of proteins to the chromatin. Third, they attract effector proteins (Gibney and Nolan, 2010).

Various types of histone modifications have been experimentally correlated with changes in gene expression. These associations between particular modifications and their effects on gene expression are known as the histone code, of which Table 2.1 shows a summary.

The mechanisms through which these histone modifications affect gene expression are many and few are fully understood. However, some modifications have been thoroughly characterised. For example, H3K4, H3K36, and H3K79 are all associated with activation. Meanwhile, H3K9, H3K27, and H4K20 are all associated with repression.

More specifically, H3K4me3 and H3K36me3 are associated with genes actively being transcribed, H3K4me3 in the promoter regions and H3K36me3 in the exons. H3K9me3 seems to appear in the promoters and bodies of repressed genes. These correlations have shown to be true in a variety of genomes, including human, mouse, and yeast (Gibney and Nolan, 2010; Heintzman et al., 2007). Many of these associations have been found using ChIP-sequencing using modification-specific antibodies, which allows researchers to study histone modifications across entire genomes.

Table 2.1 A summary of the Histone Code (Jiménez-Chillarón et al., 2014).

Histone	Modification Type	Site	Function
H1	Methylation	K26me	Binding of HP1
	Phosphorylation	T10ph	Activation expression H1B
		S17ph	Activation expression H1B
		T137ph	Activation expression H1B
		S172ph	Activation expression H1B
	Ribosylation	E2ar1	Neurotrophic activity
		K213ar1	Neurotrophic activity
H2A	Acetylation	K5ac	Transcriptional activation

		K7ac	Transcriptional activation
	Biotinilation	K9bio	Cell proliferation and gene silencing
		K126bio	Cell proliferation and gene silencing
	Phosphorylation	S121ph	Telomere silencing
		T125ph	Telomere silencing
	Ubiquitylation	K119ub1	Polycomb silencing
	Sumoylation	K126su	Transcriptional repression
H2B	Acetylation	K5ac	Transcriptional activation
	Phosphorylation	S33ph	Transcriptional activation
	Sumoylation	K16su	Transcriptional repression
		K17su	Transcriptional repression
	Ubiquitylation	K120ub	Transcriptional activation
		K123ub	Telomeric silencing
H3	Acetylation	K9ac	Transcriptional activation
		K14ac	Transcriptional activation
		K23ac	Transcriptional activation
	Methylation	K4me	Transcriptional activation
		K9me	Transcriptional repression
		K27me	Polycomb repression
		K79me	Telomeric silencing
	Phosphorylation	S10ph	Transcriptional activation of early genes
		S28ph	Mitotic chromosome condensation
H4	Acetylation	K8ac	Transcriptional activation
	Methylation	R3me	Transcriptional activation
		K20me	Transcriptional silencing
		K59me	Silent chromatin formation

RNA-Based Mechanisms

Non-coding RNAs are known to serve both infrastructural and regulatory roles. The majority of the genome encodes for RNA molecules which are never translated into protein. However, the mechanisms by which these non-coding RNA transcripts affect gene expression are less clearly understood than those related to methylation and histone

modifications (Gibney and Nolan, 2010). There are a variety of ways of classifying non-coding RNAs (ncRNAs), but they are commonly grouped by length.

There are a variety of small ncRNAs, but the most well characterised are microRNAs (miRNAs), short interfering RNAs (siRNAs), PIWI-interacting RNAs (piRNAs), and repeat-associated RNAs (rasiRNAs). In general, these small ncRNAs are formed when longer RNA molecules are cleaved by RNAse III-family enzymes. miRNAs base-pair with specific mRNAs. When they pair perfectly with the targeted mRNAs, they cause it to be degraded. When they pair imperfectly, they inhibit translation of the mRNA.

Similarly, siRNAs base-pair with target mRNAs, and can cause them to be degraded when paired perfectly. When they pair imperfectly, they repress translation. siRNAs have also been shown to silence genes by increasing DNA methylation in specific areas (plants), by recruiting histone methyltransferases (yeast), or by generating heterochromatin (yeast).

piRNAs interact with proteins in the PIWI family. rasiRNAs, which are thought to be a subspecies of piRNAs, are involved in control of transposable elements in the germ lines of various organisms, and seem important for germ-line viability. A study showed that when piRNAs were added to Drosophila oocytes, the phenotype of the progeny was changed in a way which was later inherited by a future generation. It is therefore hypothesised that piRNAs may be important for epigenetic inheritance (Gibney and Nolan, 2010; Malone and Hannon, 2009).

The function of long non-coding RNAs (lncRNAs) is not entirely known. It is thought that their transcription could increase or decrease transcription of a nearby, downstream region, that they can affect splicing of other transcripts, that they can affect protein activity, or that they can lead to the production of smaller RNAs. Additionally, lncRNAs interact with complexes which go on to change chromatin structure, often by changing histone modifications, and, by doing so influence gene expression (Gibney and Nolan, 2010).

2.4 The Space Between Theory and Practise in Parts-Based Design

In his 2005 paper "Foundations for Engineering biology," Drew Endy laid out a list of engineering principles which he viewed as essential for engineering biology (Endy, 2005). These included standardisation, decoupling, and abstraction. He described a future (now, in some ways, a reality) of pre-existing standard biological parts, which could be arranged by any novice into complex biological circuits. His list of ideals serves as the foundation of parts-based design as applied to Synthetic Biology.

The NIH Genetics Home Reference defines a gene as a "the basic physical and functional unit of heredity". In many cases, the gene has served as a basic building block of parts-based design. It is generally accepted that genes are sequences of DNA which encode some specific "functionality," whether or not they are protein-coding.

However, given the field's understanding of the complex relationship between the genome and the epigenome, as well as the heritability of epigenetics, a gene has become "a slippery concept to define" (Hopkin, 2009). Evelyn Fox Keller has argued, "No longer does it make sense to think of the genome as a starting point, as the beginning of a causal chain that takes us from genotype to phenotype. Instead, I claim, we need to reconceptualize it as itself a fundamentally reactive system, a sub-system of the cell composed of DNA that has been designed over the course of evolution to sense and to respond to the signals impinging on it" (Keller, 2014).

This is not to say that there are no legitimate units of heredity, or of functionality, only to demonstrate that defining such units is not straightforward. With that in mind, Endy's goals of standardisation, decoupling, and abstraction when designing DNA sequences seem less realistic, and potentially less desirable. If genes are anything but discrete and self-sufficient, then perhaps there is a better model with which to represent them.

Figure 2.10 shows a circuit design for a genetic toggle switch from the early days of Synthetic Biology. Modern Synthetic Biology experiments sometimes take a similar form, but sometimes take forms impossible to represent through simple circuits, including genome-wide redesigns, precision immunotherapies, and pathway optimisations through directed evolution. While much of the field's software and modelling infrastructure is based around parts-based design, much of the research has outgrown it.

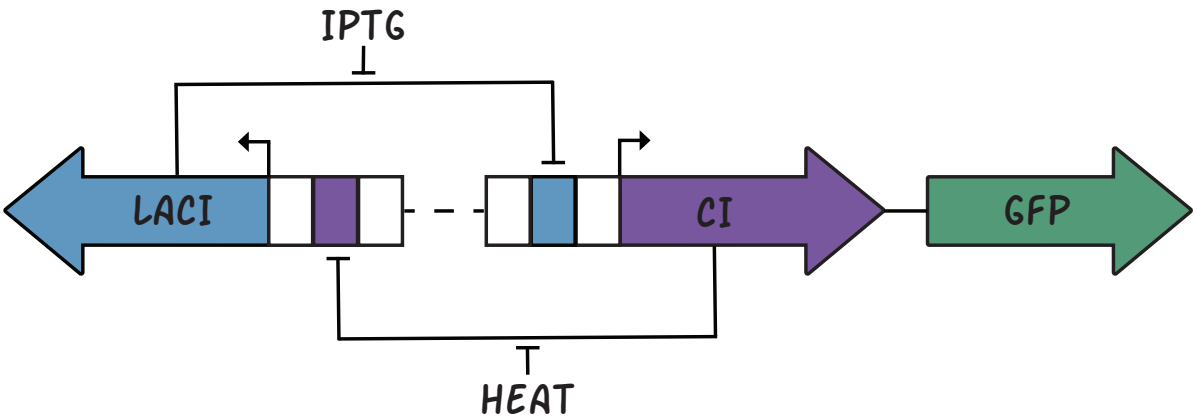


Figure 2.10 An early example of a Synthetic Biology gene circuit design. This particular design shows a toggle-switch. Two repressor genes (lacI and cI) are placed such that each represses transcription of the other. Heat can be used to disengage cI and IPTG is used to disengage lacI. By using these two environment inputs, it is possible to toggle the switch between two states. Once the switch is flipped with one of these environmental inputs, the transcriptional state remains through several generations (Gardner et al., 2000). Figure adapted from (Cameron et al., 2014).

Chapter 3

An Overview of Relevant Modelling and Machine Learning Techniques

Biology is complicated. The relatively simplified versions of important processes in the previous section hint at far more complex biological realities. High throughput assays allow researchers to capture huge amounts of data, representing the behaviours of the innumerable players that make up a living cell. Turning that data into biological discoveries often amounts to searching for a pattern in a sea of seemingly illegible signals.

Machine learning is the scientific discipline dedicated to the algorithms that can be used to find meaning in otherwise meaningless masses of data. A variety of different problem-solving techniques can fall under the umbrella of machine learning. The vast majority of these loosely fall into one of two categories: supervised learning, and unsupervised learning.

Supervised learning problems are those where a mathematical model is built through which an input variable x is mapped to an output variable y. Classification and regression, two very common machine learning problems, fall into the category of supervised learning. In classification, the algorithm attempts to map an input variable (or set of input variables) to a category. A good example of this is a spam filter, which decides whether an email is spam or not. Regression tries to map input variables to continuous, numerical values (Ayodele, 2010).

In contrast, unsupervised learning problems rely on input data to learn more about the underlying structure of the data itself. In unsupervised models, an input variable x can also be mapped to an output variable y, but, importantly, the y value is not observed during training. An intuitive example of unsupervised learning is clustering, in which related data is associated together forming sub groups (Ayodele, 2010).

"Solving" each of the above problems involves building and training a model based on a dataset. One has many options to choose from when designing such a model, and it is out of scope to describe each and every one of them. However, those models and algorithms which are directly relevant to this thesis are described below.

3.1 Linear and Logistic Regression

A linear regression model assumes that the relationship between x, an explanatory variable, and y, a dependant variable, is linear. In the simplest case, the model takes the form

$$y = X\beta + \epsilon \tag{3.1}$$

where y is a vector of observed values, X is a vector of input variables (regressors), β is a parameter vector (regression coefficients), and ϵ is the error term. ϵ is usually sampled from a Gaussian distribution with a zero mean, and is meant to capture all other variables which influence y other than the regressors.

In general, regression lines are fit using the method of least-squares. The best fitting line for the observed data is found by minimising the sum of the squares of the vertical deviations from each data point to the proposed line of best fit.

Similarly, logistic regression models are used to model the relationship between an explanatory and a dependant variable. However, unlike in linear regression, in logistic regression the dependant variable is categorical. Therefore, logistic regression models are used as a classifier. In the simplest case, a logistic regression can distinguish between two classes. However, it is possible to build a multinomial logistic regressions which decides between more than two classes.

Instead of fitting a linear model to the data, in a logistic regression the sigmoid function is fit to the data:

$$\theta(x) = \frac{1}{1 + e^{-z}}. \tag{3.2}$$

The general logistic function, for the case where y is a function of one explanatory variable, x, is given as:

$$\theta(x) = \frac{1}{1 + e^{-(x\beta+\epsilon)}}. \tag{3.3}$$

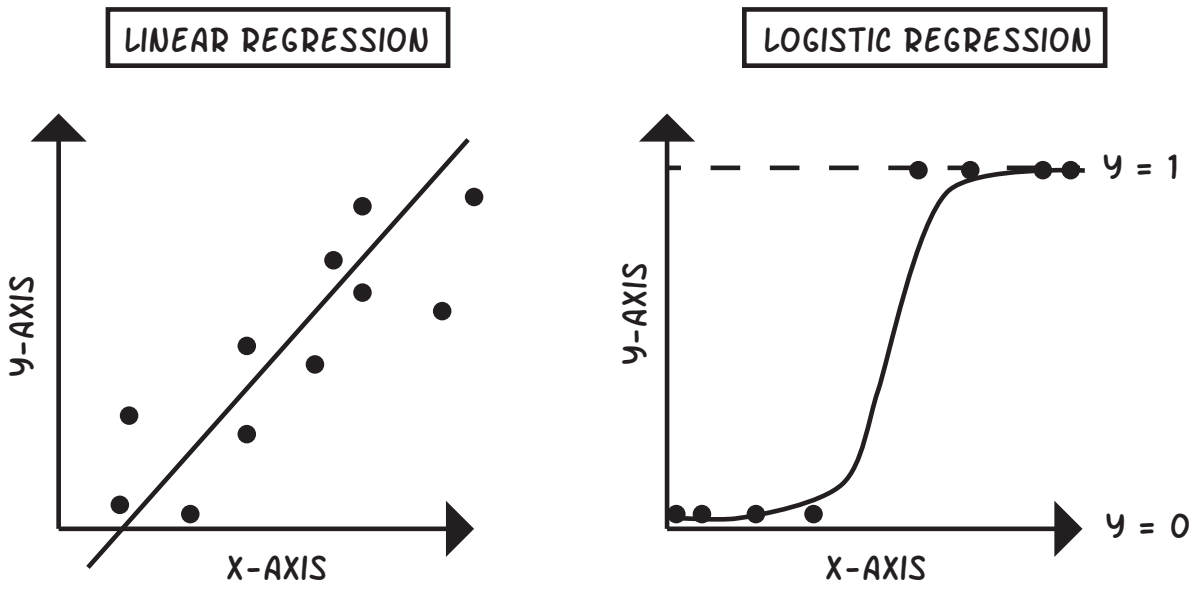


Figure 3.1 A visual comparison of linear and logistic regressions.

This function allows us to predict whether an observation belongs in one of two classes. Thus, while linear regression allows us to map input variables to numerical output values, logistic regression allows us to map input variables to classifications.

3.2 N-Gram Models

Language models are models which assign probabilities to a sequence of words. The simplest and most intuitive of these models is the n-gram, which predicts the next word based on what came before it, for instance

$$P(w|h) \tag{3.4}$$

where h is the 'history' of w, the word. For example, assuming we have a trigram model where h is "my name is," one could represent the probability of the next word being "Emily" with

$$P(\text{Emily}|\text{my name is}). \tag{3.5}$$

The simplest way of finding this probability is by counting the number of times "my name is Emily" occurs in a corpus, and diving it by the number of times "my name is" occurs in the corpus

$$P(\text{Emily}|\text{my name is}) = \frac{C(\text{my name is Emily})}{C(\text{my name is})}. \tag{3.6}$$

The n of the n-gram refers to the length of history we rely upon to help us predict the next word. However, when the n is larger, the corpus must also be larger in order to capture the variety of possible combinations.

If our model was to capture the entire history preceding every prediction, the joint probability of a sequence of words would be calculated as follows

$$P(w_1, ..., w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)...P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \tag{3.7}$$

where w_i^j represents $w_i, ..., w_j$. Instead of this, however, n-gram models attempt to estimate a word's history using only a small part of it, ie

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}). \tag{3.8}$$

Given the above approximation, the joint probability of an entire word sequence is given as

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1}). \tag{3.9}$$

In practise, these probabilities are often estimated using maximum likelihood estimation (MLE). This algorithm is discussed more in Jurafsky et al (Jurafsky and Martin, 2009).

3.3 Hidden Markov Models

Both Hidden Markov Models and N-gram Models rely upon the Markov Assumption which, in the case of language models, implies that the probability of a word depends only on the probability of the previous $n-1$ words. A Markov Chain is a model which predicts the future state based on only the current state. This is of course analogous to an n-gram model, where $n = 2$ ("bigram").

Markov chains are represented using the following parameters:

- A set of states
- A matrix representing the probabilities of transitioning from state i to state j

- An initial probability distribution representing the likelihood of starting in a particular state

While a Markov Chain intuitively represents a sequence of observed events, a Hidden Markov Model represents a series of events which cannot be directly observed — they can only be observed through another variable.

The parameters of an HMM are very similar to those of a Markov Chain, though they allow for the representation of both unobserved states and observed sequences:

- A set of states
- A matrix representing the probabilities of transitioning from state i to state j (Transition Matrix)
- An initial probability distribution representing the likelihood of starting in a particular state
- A sequence of observations, drawn from an alphabet
- A matrix representing the probabilities of emitting a particular observation while in a particular state (Emission Matrix)

The two defining rules of an HMM are, first, that the probability of being in a particular state depends only on the state that came before it

$$P(q_i|q_1...q_{i-1}) = P(q_i|q_{i-1}), \tag{3.10}$$

and, second, that the probability of an output observation being emitted depends only on the state that emitted it

$$P(o_i|q_1...q_i, ..., q_r, o_1, ..., o_i, ..., o_r) = P(o_i|q_i). \tag{3.11}$$

A dynamic programming algorithm known as the Forward-Backward Algorithm is used to estimate the posterior probabilities over the latent states (Baum, 1972). This algorithm is described in detail below. The Forward-Backward algorithm can also be used as the E-step in a special case of the Expectation Maximisation algorithm, called the Baum-Welch algorithm, which is used to learn the parameters above (Dempster et al., 1977).

At each point in the sequence of states, each state emits only one output. Therefore, the length of the output sequence (O) is the same as the length of the state sequence

(Q). Therefore, the probability of an observation sequence is simple to compute when the state sequence is known, and is given by

$$P(O|Q) = \prod_{i=1}^{T} P(o_i|q_i). \tag{3.12}$$

However, things are more complicated when the state sequence is unknown. The joint probability of seeing an observation, O, and a state sequence Q is given by

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{T} P(o_i|q_i) \times \prod_{i=1}^{T} P(q_i|q_{i-1}) \tag{3.13}$$

given that q_0, which is not part of the sequence, represents the initial probabilities.

The total probability of an observation sequence where the sequence of hidden states is unknown is the same as equation 3.13, but summed over all possible hidden state sequences:

$$P(O) = \sum_{Q} P(O,Q) = \sum_{Q} P(O|Q)P(Q). \tag{3.14}$$

However, computing the total likelihood of an observation for a task where the number of states, the number of observations, and the number of potential emissions are all non-trivially large can be extremely computationally intensive. Therefore, the forward-backward algorithm takes advantage of a dynamic programming table.

Each cell in the forward table represents the probability of being in a particular state, j, after observing the first t observations. This is given by the sum of the probabilities of every path that leads to that particular outcome.

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t) \tag{3.15}$$

Where $\alpha_{t-1}(i)$ is the forward path probability from the previous time step, a_{ij} is the transition probability from the previous state q_i to the current state q_j, and $b_j(o_t)$ is the observation likelihood of o_t given j.

At each time step, the table is extended via two paths. The formal definition of this recursion is given below, where $forward[s,t]$ is equivalent to $\alpha_t(s)$ above, and λ is the HMM.

Initialization:

$$\alpha_1(j) = \pi_j b_j(o_1) 1 \leq j \leq N \tag{3.16}$$

Recursion:

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t); 1 \leq j \leq N, 1 < t \leq T \tag{3.17}$$

Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.18}$$

The algorithm above describes the steps for generating the forward probabilities, sometimes called the forward message. However, we're still missing the backward part of the forward-backward algorithm. The backward message is calculated very similarly to the forward message. The recursion for doing so is the following:

Initialization:

$$\beta_T(i) = 1, 1 \leq i \leq N \tag{3.19}$$

Recursion:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1})) \beta_{t+1}(j), 1 \leq i \leq N, 1 \leq t < T \tag{3.20}$$

Termination:

$$P(O|\lambda) = \sum_{j=1}^{N} \pi_j b_j(o_1) \beta_1(j). \tag{3.21}$$

The forward and backward probabilities shown above are used in the calculation of the transition and emission matrices which are the core of the Hidden Markov Model. These calculations are done iteratively, using expectation-maximisation. This involves two steps, the expectation step, and the maximisation step. In the expectation step, the likelihood of the observation sequence given the model is calculated. In the maximisation step, the γ (the expected state occupancy count) and the ξ (the expected state transition count) are used recompute the transition and emission matrices.

The calculations made in the expectation step are as follows:

$$\gamma_t(j) = \frac{\alpha_t j \beta_t(j)}{\alpha_T(q_F)} \ \forall \text{ t and j} \tag{3.22}$$

$$\xi(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\alpha_T(q_F)} \ \forall \text{ t, i, and j} \tag{3.23}$$

The calculations for the maximisation step are as follows:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)} \tag{3.24}$$

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{3.25}$$

.

The HMM parameters are initialised, and the E and M steps are run iteratively (Jurafsky and Martin, 2009). The likelihood of the observation sequence is guaranteed to improve with each EM step, until convergence at a local minimum of this likelihood.

3.4 Probabilistic Context Free Grammars

The previously mentioned language models, the n-gram and the HMM, demonstrate increasing ability to represent sequence structure. While n-grams can capture only simple and local patterns, HMMs are capable of representing more complex combinatorial patterns. Meanwhile, context free grammars are able to capture complex, hierarchical patterns within a sequence, while the mathematics behind them represent a natural development from HMMs (Manning et al., 1999).

A context free grammar is defined by four attributes:

- N, a set of non-terminal symbols. This is analogous to the set of states in an HMM
- V, a set of terminal symbols. This is analogous to the vocabulary in an HMM
- R, a set of rules, each of the form $N^i \rightarrow \zeta^i$, where ζ^i is a sequence of terminals and non-terminals
- N^1, the start symbol

A probabilistic context free grammar has all of the above, and also

- P, a set of rule probabilities, such that $\forall_i \sum_j P(N^i \rightarrow \zeta^j) = 1$

The language L_G generated by a grammar G is the set of all sequences composed of terminals which can be generated from the start symbol N^1.

From these attributes, one can see that PCFGs represent sentences in a language using a tree structure. A sentence thusly represented by a tree of terminals and non-terminals is called a parse tree.

PCFG models rely upon a number of conditions:

- Place invariance: a subtree's probability does not depend upon where in the string its associated words are. This is analogous to time invariance in HMMs.

- Context Free: The probability of a subtree does not depend on words not included in the subtree
- Ancestor Free: The probability of a subtree does not depend on nodes outside of the subtree

Given said conditions, calculating the probability of a tree simply requires multiplying the probabilities of the rules that built its local subtrees.

In order to train a PCFG, one must already have the set of non-terminals, the set of terminals, and the start symbol in advance. There must also already be a set of rules in place. However, this can be agnostic, meaning that all rules are possible, or, alternatively, a known structure can be represented up front.

The algorithm used to train the PCFG rule probabilities is called the Inside-Outside algorithm. As was the case for the HMM, this is a special case of expectation maximisation. The Inside-Outside algorithm assumes that the best grammar is the one which maximises the likelihood of the sentences in the training corpus.

In order for the Inside-Outside algorithm to be used, the grammar must be in Chomsky Normal Form, which requires all production rules to take one of the following forms:

$$\begin{aligned} A &\to BC, \text{or} \\ A &\to a, \text{or} \\ S &\to \varepsilon \end{aligned} \tag{3.26}$$

where S is the start symbol, A, B, and C are non-terminals, a is a terminal symbol, and ε is an empty string. However, there are other versions of the Inside-Outside algorithm which do not require grammars to be in Chomsky Normal Form, and which work with similar efficiency.

The probability of the set of rules is:

$$\hat{P}(N^J \to \zeta) = \frac{C(N^J \to \zeta)}{\sum_\gamma C(N^J \to \gamma)} \tag{3.27}$$

where the function C gives the total number of times that a rule is used.

If our training data is already parsed, then these probabilities are simple to calculate. This is similar to Markov chains. However, when the training data is unparsed, the problem is analogous to the hidden data problem in HMMs. In that case, the probability functions on rules have to be determined based on the probabilities of sentences.

Because, like with HMMs, it is impractical to calculate the probability of a sequence by summing the probabilities for all possible paths to that string, dynamic programming

is used to calculate these probabilities. The inside probabilities are given by:

$$P(W_{1m}|G) = P(N^1 \stackrel{*}{\Rightarrow} w_{1m}|G) = P(w_{1m}IN :, , G) = \beta_1(1, m). \tag{3.28}$$

The inside probability of a subsequence is generated by induction on its length:

Base Case: To find $\beta_j(k, k)$, which represents the probability of a rule $N^j \rightarrow w_k$

$$\beta_j(k, k) = P(w_k|N^j_{kk}, G) = P(N^j \rightarrow w_k|G). \tag{3.29}$$

Induction: this step requires the calculation of $\beta_j(p, q)$, for $p < q$. This is the step that requires the grammar to be in Chomsky Normal Form as that allows the string can be divided in two in various places. The first rule must be of the form $N^j \rightarrow N^r N^s$.
Then, $\forall j, 1 \leq p < q \leq m$,

$$\beta_j(p, q) = P(w_{pq}|N^j_{pq}, G) = \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s)\beta_r(p, d)\beta_s(d + 1, q). \tag{3.30}$$

This calculation represents all of the ways that a certain constituent can be built out of two smaller constituents.

To calculate the outside probabilities, for an $k, 1 \leq k \leq m$,

$$P(w_{1m}|G) = \sum_j \alpha_j(k, k)P(N^j \rightarrow w_k). \tag{3.31}$$

The induction which is used to calculate the outside probabilities relies upon the inside probabilities, so the inside probabilities must be calculated first.

Base Case: The probability of the root node having nothing outside of it

$$\alpha_1(1, m) = 1 \tag{3.32}$$

$$\alpha_j(1, m) = 0 \text{ for } j \neq 1 \tag{3.33}$$

Inductive Case: The node we're interested in might be on either the right or the left. Therefore, we have to sum over both possibilities, without double counting.

$$\begin{aligned}\alpha_j(p,q) = [\sum_{f,g\neq j}\sum_{e=q+1}^{m}\alpha_f(p,e)P(N^f \rightarrow N^j N^g)\beta_g(q+1,e)]+ \\ [\sum_{f,g}\sum_{e=1}^{p-1}\alpha_f(e,q)P(N^f \rightarrow N^g N^j)\beta_g(e,p-1)]\end{aligned} \tag{3.34}$$

For more information about how these inside and outside probabilities are derived, consult Manning et al (Manning et al., 1999).

In order to use these inside and outside probabilities to train our PCFG, we first calculate the probability of each parse of a training sentence. Then, we sum the probabilities of each rule being used in each place. This gives us an expectation of how many times each rule was used. We use these expectations to maximise the likelihood of our training data.

With $P(N^1 \stackrel{*}{\Rightarrow} w_{1m}$ denoted as π, the E step of the inside outside algorithm is as follows:
$E(N^j \rightarrow N^r N^s, N^j$ used)

$$= \frac{\sum_{p=1}^{m-1}\sum_{q=p+1}^{m}\sum_{d=p}^{q-1}\alpha_j(p,q)P(N^j \rightarrow N^r N^s)\beta_r(p,d)\beta_s(d+1,q)}{\pi}. \tag{3.35}$$

Assuming that there is a corpus with multiple training sentences which are considered to be independent, let f_i, g_i, and h_i be the common sub-terms for use of a nonterminal at a branching node, at a pre-terminal node, and anywhere respectively. Given the above, the maximisation step calculations are given by:

$$\hat{P}(N^J \rightarrow N^r N^s) = \frac{\sum_{i=1}^{w}\sum_{p=1}^{m_i-1}\sum_{q=p+1}^{m_i} f_i(p,q,j,r,s)}{\sum_{i=1}^{w}\sum_{p=1}^{m_i}\sum_{q=p}^{m_i} h_i(p,q,j)} \tag{3.36}$$

$$\hat{P}(N^J \rightarrow w^k) = \frac{\sum_{i=1}^{w}\sum_{h=1}^{m_i} g_i(h,j,k)}{\sum_{i=1}^{w}\sum_{p=1}^{m_i}\sum_{q=p}^{m_i} h_i(p,q,j)}. \tag{3.37}$$

Like in the EM algorithm for HMMs, the Inside Outside algorithm is run iteratively until convergence. However, the Inside Outside algorithm is comparatively slow. Additionally, local maxima are more likely to cause problems, and, in general, in order to wind up with a grammar that resembles expectations, the algorithm often requires the initial grammar rules to be pre-constrained (Manning et al., 1999).

3.5 The Chomsky Hierarchy of Languages

The three language models presented are each capable of representing increasingly complex patterns than the one that came before it. A useful paradigm through which to compare language models is the Chomsky Hierarchy. Figure 3.2 shows a visual representation of the hierarchy. The various types of grammars are summarised below. Notably, N-grams, HMMs, and PCFGs can all be formulated as formal grammars.

- Type-3: Languages which can be represented by a regular grammar, or accepted by a deterministic or non-deterministic finite automata;
- Type-2: Languages which can be represented by a context free grammar;
- Type-1: Languages which can be represented by context sensitive grammars, or a linear-bounded automaton. This is similar to a DFA/NFA, except it is able to store symbols in a finitely long list;
- Type-0: All languages which can be represented by a Turning Machine, which includes all former grammars

Regular grammars, including n-grams, fall into the Type-3 category. Hidden Markov models are also associated with Type-3 grammars, but offer an extension over them in a similar way to which PCFGs offer an extension over CFGs (Johnson, 2010; Searls, 2002).

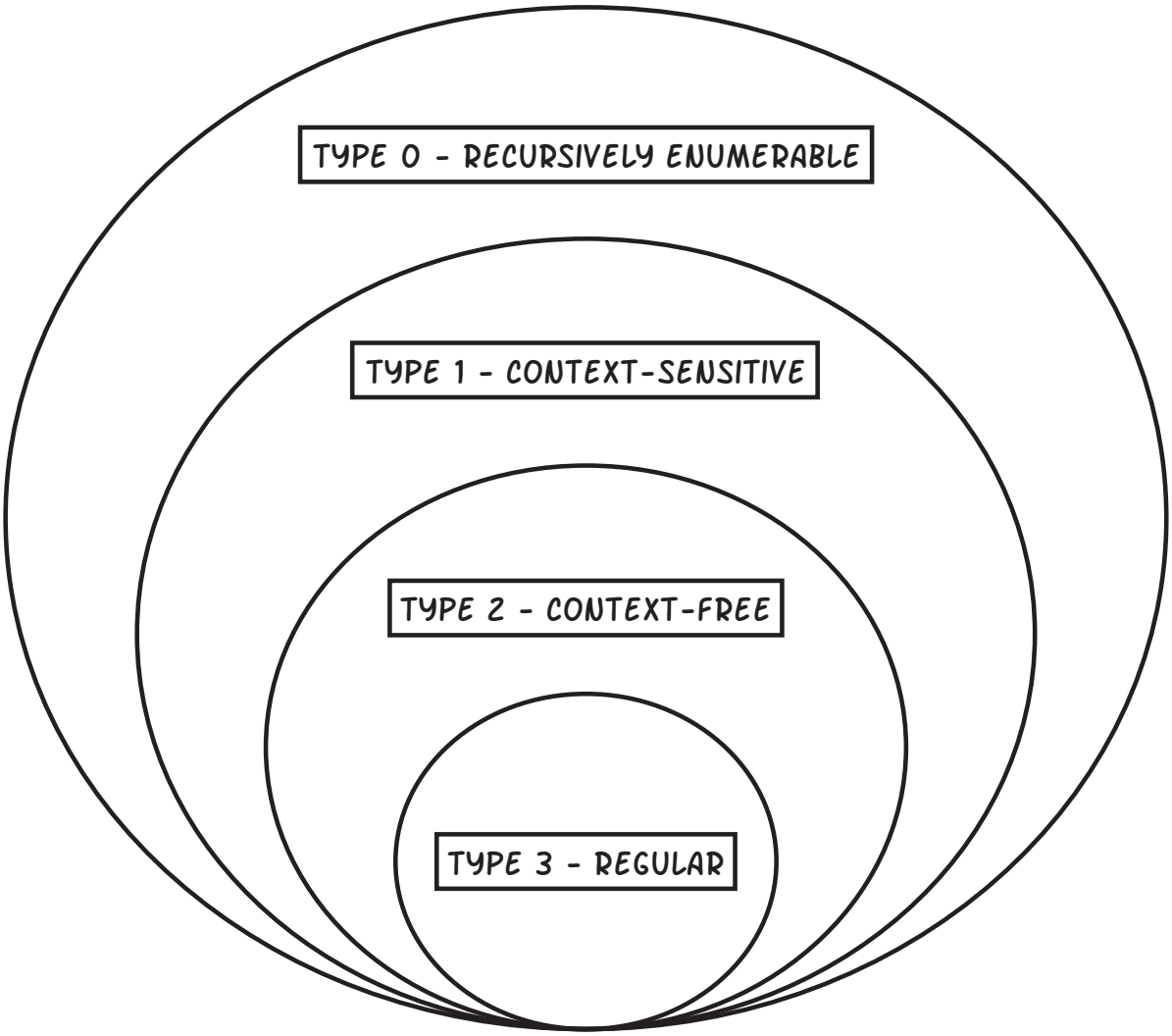


Figure 3.2 The Chomsky Hierarchy of Languages.

Chapter 4

Models and Metaphors for DNA Design

The previous chapters have introduced the main concepts and techniques relied upon when researchers design Synthetic Biology experiments.

However, there is a meaningful disconnect between these Synthetic Biology design paradigms and molecular biology reality. The parts-based design paradigm has been particularly pervasive. Almost all popular DNA design tools were designed around it. Yet, it is unclear how a genetic unit of functionality ought to be defined, and it is unclear how a genome can be separated into entirely decoupled parts. When built upon this model which is not quite fit to purpose, software tools in turn to do not meet expectations. Designing DNA sequences still requires a large amount of expertise, and often several iterations when sequences do not behave as expected.

The way in which parts-based design has been implemented for Synthetic Biology does not provide a reliable enough version of abstraction to be useful for many experiments.

In the following chapter, we will review existing DNA design tools, identifying underlying similarities in how they represent the structure and function of DNA sequences. We will suggest that iterating the existing parts-based model is unlikely to overcome limitations in matching software applications to design aspirations.

While all of the tools we review represent DNA solely as linear text, and often as a collection of separate parts, we will argue that such a representation captures too little of the known complexity of gene expression and DNA function. New models able to account for more of that complexity and therefore to enable more ambitious DNA design goals are likely to call for new underlying representations of DNA — a need that may

be addressed by rethinking DNA in terms of human languages rather than computer languages or circuits.[1]

4.1 Introduction

Since at least 2004, synthetic biologists and others who design and build genetic constructs have been aided by DNA design tools, software applications offering massive advantages over using word processors to manipulate text files (Kelwick et al., 2014; Marchisio and Stelling, 2009). These tools have improved substantially and now standardise routine operations, sense-check assemblies in silico, and incorporate data storage and teamwork functions. Without exception, however, all general-use tools (i.e., those not designed to work only with a limited range of assemblies) rely on the same essential underlying assumptions about the structure and function of DNA and units of heredity — their models assume that functional genetic units can be equated with "parts," conceptually equivalent to words in the "book of life" and comprised solely of linear text. On the one hand, this model has enabled remarkable progress in molecular genetics. On the other, it accounts for relatively little of what is now known about the densely interactional nature of genetic structure and function, epigenetic modifications and inter- and intra-molecular interactions — the genome's capacity to be an organ of environmental response rather than a static conductor of cellular operations, or what philosopher of science Evelyn Fox Keller has termed "the reactive genome" (Keller, 2014). For recent reviews, see Lappalainen and Greally (2017); Nicoglou and Merlin (2017); Pinel et al. (2018).

We suggest that what was once an adequate model of DNA function for DNA design is becoming inadequate, less because the model has become invalid than because what the model is expected to do has changed. Notwithstanding many successes, synthetic biology is characterised by communal frustration that current design strategies do not efficiently ensure that in silico design becomes in vivo function (Davidsohn et al., 2015; Dietz and Panke, 2010; Kwok, 2010; Nielsen and Keasling, 2016). Responding to this frustration calls for new tools grounded in a different underlying model rather than a series of more sophisticated user interfaces for working with the established model. That new model should ideally reconsider the fundamental units of DNA used for design ("parts") and incorporate more contextual information into how these units are meaningfully combined, modelling parts as more than non-interacting segments of linear text. As a computational biologist and a rhetorician of science, we have arrived at the same

[1] Much of the text in this chapter comes from (Szymanski and Scher, 2019), which was written by myself and Erika Szymanski (joint first authors).

conclusion from different disciplinary angles—synthetic biology would be well-served by a model accounting for more of what is currently known about DNA structure and function and such models require a different underlying conceptual metaphor, beyond DNA solely as linear text, to have more than a superficial effect on DNA design.

Table 4.1 A comparison of a selection of popular DNA editing tools.

Name (release date)	Default assembly view	Unique features
ApE (2004) Desktop Application Free	Linear text box	BLAST integration, digestion simulation, automatic annotation based on a custom library
Benchling (2012) Web Application Free for personal use; paid subscription for enterprise users	Linear text box OR parts-based design alongside text	Integrated lab notebook and other collaboration tools, CRISPR guide, history tracker, automatic annotation
Snapgene (2012) Desktop Application Paid; choice of annual subscription or permanent license; discount for academics	Sequence view OR plasmid/linear parts-based design feature map	Custom feature library; auto-annotation; all design is done through the sequence view, not through the feature map; useful, intuitive history-tracking feature
Geneious (2012) Desktop Application Paid; discount for academics; version for biopharmaceutical drug discovery	Sequence view OR plasmid/linear parts-based design feature map	All design is done through the sequence view, which is the plasmid map; supports custom algorithms/plugins; API; potential to link to LIMS; NGS Analysis; tree building

Genome Compiler (2013) Web Application or Desktop Application Online version is free; paid custom versions available for enterprise	Text view alongside parts-based design view	Sanity-checks DNA to check it for simple errors; integrated with Twist Bioscience, can order DNA from Twist from Genome Compiler; special iGem addition; automatic annotation
Genocad (2007) Web Application Free	Sequences are represented using SBOL parts and "grammars"	Lets users "design" sequences using libraries of parts and set "grammars" — sets of rules dictating how parts can be arranged; users can only create sequences using these grammars; combinatorial design; custom libraries; limited library of grammars
Cello (2016) Web Application, command line tools, API Free	DNA is designed using a custom programming language with an emphasis on inputs and outputs	Genetic circuits are designed using the Verilog programming language; can generate sequences for bacterial cells only; when sequences were carefully insulated from their genetic contexts, they generally performed as expected

Teselagen (2016) Web Application Free for individuals; paid for enterprise	SBOL, parts-based representation of sequences	DNA is designed by dragging and dropping parts from a library onto a design pane; combinatorial design; users can define "rulesets" based on annotation tags or sequence-matching regex
Genetic Constructor (2016) Web Application Free for academics; paid for enterprise	SBOL, parts-based representation of sequences; hierarchical representation of parts	DNA is designed by dragging and dropping parts from a library onto a design pane; combinatorial design; parts can be hierarchical, aka made of other parts; supports "sketch" elements, aka placeholders which allows the user to store some information about what goes in a particular spot, without specifying a specific part; public and private parts libraries

4.2 DNA design tools: Different wrappings on (much) the same box

4.2.1 Comparing features of DNA design tools

Veteran researchers speak of the early days of synthetic biology as a Wild West of pioneers and outlaws, where DNA was written in Microsoft Word, alignments were manually constructed with dot matrix-printed pages, and cloning a single gene could be a graduate

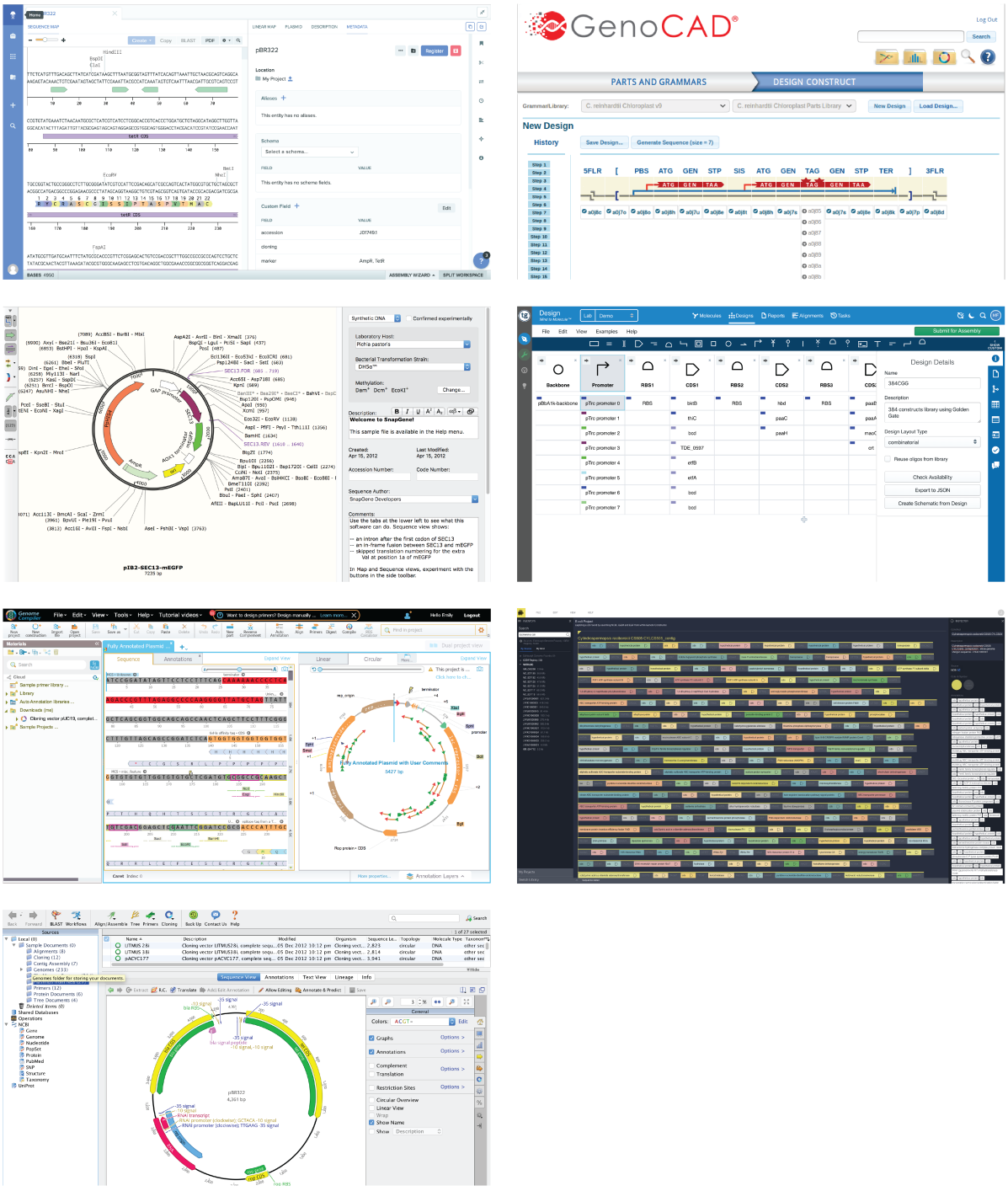


Figure 4.1 A comparison of the user interfaces of several popular DNA editing tools. First row: Benchling, GenoCad; Second Row: SnapGene, Teselagen; Third Row: Genome Compiler; Genetic Constructor; Fourth Row: Geneious Prime

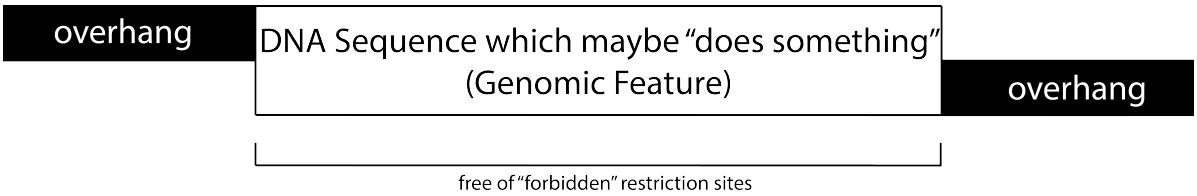


Figure 4.2 The anatomy of a standard BioPart.

student project. This landscape has been domesticated, not only through game-changing wet lab techniques such as Golden Gate assembly and CRISPR, but via computer-assisted tools that radically change both conceptual and practical strategies for designing DNA.

Table 4.1 compares DNA design tools in current use. Figure 4.1 juxtaposes screenshots of the user interfaces of these tools. Both their underlying principles and user interfaces are remarkably similar, with a key distinction being the level of abstraction at which the user is encouraged to view and manipulate an assembly. One group — including ApE, Benchling, SnapGene, and Genome Compiler — uses the main editing area to display sequence information as linear text alongside a graphical representation highlighting coding sequences, restriction sites, and other biologically relevant landmarks. A second group — including GenoCad, Teselagen, and Genetic Constructor — instead uses the main editing area to display a linear assembly of abstracted "parts," typically denoted by corresponding SBOL symbols. While users of the second group can still see textual sequence information, that level of resolution is not part of the default view. Some tools such as Benchling allow users to toggle between text and parts views, but still require the user to edit in only one of the two views (text, in Benchling's case). The rationale for focusing on parts, the creators of Genetic Constructor argue, is that "focus on sequence hampers concentration on function, centralisation of design and experimental intent, and the encapsulation and reuse of parts" in large multi-part projects (Bates et al., 2017). A second apparent point of difference amongst these tools is more subtle: some assemble "features;" others assemble "parts." "Feature" generically applies to any sequence a user wants to delineate for potentially any reason. In DNA design, features are often sequences annotated with putative functions on the basis of sequence homology or limited experimental data. In contrast, parts (sometimes called bioparts) are more narrowly defined as sequences in one of several standardised assembly formats empirically demonstrated to yield a particular function — to "do something." The standard anatomy of a part is outlined in Figure 4.2. In theory, assembling parts rather than features places more constraints on user design but decreases the need for in-depth sequence knowledge and increases the likelihood of a successful assembly. In practice, distinctions between parts and features blur because researchers often find that parts verified in one

experimental setting do not perform as expected in other contexts. In addition, the range of parts successfully implemented in vivo across varied contexts is small, particularly in comparison with the current aspirations of the field. In practice, whether a design tool relies on parts or features makes little difference to the user.

These tools differ in other respects for which prospective users may prefer one over another. Some are free; most have paid subscription-only versions, but differ in the features available to non-paying users. Web-based and downloadable options vary. Each integrates various extra features such as Benchling's integrated online lab notebook, SnapGene's version history tracker, or Genome Compiler's sanity checks to avoid misplaced stop codons or sequences not divisible by three. Each relies more or less heavily on built-in part/feature libraries and provides different options for users to input custom libraries, with some (e.g. ApE) relying entirely on the latter.

4.2.2 Different features, same model

Behind their distinctive features, each of these tools takes the same parts-based approach to DNA design, relying on end-to-end assembly of discrete sections of linear DNA treated as sufficient to produce context-independent functions. Consequently, they also make many of the same assumptions about how DNA functions, including that:

- DNA is modular, comprised of independent linear sections which can be abstracted and manipulated as "parts."
- DNA parts, as both structural and functional units, are context-independent, associated with a consistent function across varied assemblies.
- DNA parts are sufficient to elicit their associated functions.

It is well-known that DNA does not in fact work this way. DNA molecules are not one-dimensional linear texts, but three-dimensional physical polymers. DNA function is not necessarily contained within a discrete, linear segment of DNA, but may (and often does) involve interactions with diverse other molecules and non-linear interactions with distant sections of the same DNA molecule. The function of a section of DNA may (and often does) vary with its context. DNA parts, in other words, are technologies constructed for the convenience of DNA designers, not descriptive features of evolved genomes. Like that other four-letter word, "gene," "part" is a curiously crude way of abstracting how nucleic acid sequence relates to cell behaviour (Pearson, 2006).

In light of these discrepancies, what is most remarkable is just how good a "part" can be, and how useful the text metaphor is for working with DNA. Many philosophers

of biology have pointed out how this and other metaphors central to synthetic biology are theoretically inaccurate (Boudry and Pigliucci, 2013; Kay, 2000; Kogge and Richter, 2013), yet in practice they are remarkably useful. What now appears to be changing is that the scope and scale of synthetic biology have grown and now more often encounter the rare instances in which they are insufficient.

4.3 Underlying limitations of current DNA design tools

Ideally, DNA design tools generate DNA sequences which perform an intended function when inserted into the user's organism of choice. In practice, success is rarely this easy. DNA designers routinely go through multiple cycles of designing in silico, ordering DNA, loading the construct into an organism, testing function, and returning to in silico design before in silico function matches in vivo function (Davidsohn et al., 2015; Kelwick et al., 2014). As Kelwick et al. and others have observed (Kelwick et al., 2014), the bottleneck in DNA design is not in designing and building constructs but in predicting how constructs will function and getting them to work.

Synthetic biology often relies on the design-build-test cycle wherein success is the product of iteration and refinement. Iteration, however, requires resources — graduate student time, grant dollars, and scientific expertise necessary to design, build, and test numerous constructs. Proponents of synthetic biology, however, have long envisioned a more efficient process in which the design tool does more of the work. To make custom organisms economically viable for a wider range of applications, generating them needs to require less time and money (Chari and Church, 2017), which requires that design tools output constructs that require less troubleshooting in vivo. And democratising agendas that call for DNA design to become accessible to nonspecialists (Schmidt, 2009) can only be achieved if design tools do not necessitate substantial scientific expertise.

Abandoning parts-based design altogether is one option. Ground-breaking synthetic biology projects — of which the *Saccharomyces cerevisiae* 2.0 or "synthetic yeast" project is the largest and most ambitious to date — have treated the genome as a conceptual whole and applied design principles across that whole rather than subdividing the genetic construct into discrete functional units (Annaluru et al., 2015; Richardson et al., 2017). The synthetic yeast genome incorporates SCRaMbLE —Synthetic Chromosome Recombination and Modification by LoxP-mediated Evolution (Dymond et al., 2011; Shen et al., 2016) — which might be called a YAD, or yeast-aided design tool. SCRaMbLE employs yeast-directed, semi-random, large-scale genome recombination to generate genomic

diversity for directed evolution experiments. The potential sequences of SCRaMbLE recombinants depend on the location of Cre recombinase recognition sites coded into the redesigned genome; however, functional genetic units are identified in observing which recombinant sequences are viable, not an a priori assumption of the design. SCRaMbLE is a leading example of a non parts-based design strategy: SCRaMbLE-ing may lead to characterising new parts, but does not rely on them.

Parts-based design nevertheless remains a mainstay of the synthetic biology imagination. Many aspirations rest on the idea of inserting custom pathways into standard host cell "chassis" to create custom organisms for individual tasks (Adams, 2016). Whole genome approaches are unlikely to satisfy those aspirations. Other projects have retained the idea of parts as genetic units, but reimagined what designing with those parts might involve. Directed evolution and other means of screening large, diverse libraries for desired functionalities, for example, acknowledge that the vast majority of constructs will fail but trust that at least one of many alternatives will succeed (Bassalo et al., 2016; Cobb et al., 2013; Kelwick et al., 2014).

Cello, a design tool which we have not included in our comparison because it presently applies only to a very small design space (a limited number of logic circuits in E. coli), takes a third approach (Nielsen et al., 2016). While Cello uses a different vocabulary — replacing the agnostically defined parts of other design tools with more tightly defined logic operators and reconfiguring functional genetic units in terms of the Verilog hardware description language rather than construction blocks — it rests on essentially the same parts-based approach employed by other tools. However, two important respects in which Cello distinguishes itself highlight specific problems with relying solely on linear DNA sequence to determine genetic function. First, accounting for genetic function also depending on interactions between target DNA sequences and other cellular components, Cello requires a user constraint file specifying the organism (species and strain) and growth conditions for which an assembly (or "circuit") is being created. Second, Cello restricts users to parts rigorously tested for those specific constraints, massively improving on the success rate of most DNA design tools but also severely restricting the design space to which the "programming language" currently applies (Nielsen et al., 2016). Whether or not any of these tools is sufficient is largely a function of what the tool is expected to do. If each part is characterised by some degree of uncertainty about its annotated function, the likelihood that any assembly will work as intended decreases as the assembly involves more parts. This decrease may be more than additive because long-range interactions across sequence units and the ability of parts to function in the context of any particular as-yet-untested assembly is also uncertain. Moreover, it may

be sufficient for a tool to output a sequence which initially fails, but which can be made to work after iterating the design-build-test cycle. And yet, as DNA design projects are becoming more ambitious, involving larger constructs with more and more important intra- and inter-molecular interactions and dependencies, what was once sufficient may no longer be.

Goals for DNA design are moving from getting a small number of things to work at all to getting a comparatively large number of things to work efficiently. To reduce the cost and increase the speed of building custom organisms, make design accessible beyond small circles of highly trained scientists, and expand the limited range of functional DNA designs realised to date, the direct outputs of DNA design tools need to do more of the work of achieving a functional physical assembly. Building such tools, we argue, likely requires an improvement on the "part" that incorporates more of the non-linear, long-range, and more-than-sequence based complexity of genetic function. Doing so will call for a fundamentally different metaphor for conceptualising what DNA is and how it works — one that rethinks the productive but imperfect reduction of nucleic acid sequences to linear text.

4.4 Rationalising the distance between DNA as model and DNA as molecule

Why do DNA design tools continue to model DNA through assumptions of modularity, linearity, sufficiency, and context-independence? We suggest three additional interdependent factors.

4.4.1 Designing, not describing

DNA design involves making heritable genetic material as you want it to be, not as you found it. Even if a well-fitting descriptive model for DNA function must involve far more than linear text, a constructive model for DNA design may ignore complicating features in an effort to identify a simpler system that still "works." The past fifteen years of synthetic biology, it could be argued, have been an experiment in testing how much simpler DNA can be made to be.

4.4.2 "Good enough"

Models for DNA design need not perfectly encompass DNA function; they need only be good enough to enable the applications for which they are used. That is not necessarily to say that the field should lower its expectations, but that tools should be evaluated on how well they function. For a software output to be judged successful, does its output need to immediately function as expected once realised in physical DNA? Or will the software still have "succeeded" even if the sequence it outputs must be manually adjusted to deliver a functional outcome? DNA designers have generally accepted that design tools do only part of the work actually necessary to build constructs that function as desired in vivo.

4.4.3 Constraints of the underlying metaphor

The model of DNA structure and function in a design tool is the set of rules that structure the behaviour of the in silico representation of the physical DNA molecule. The metaphor that model employs is part of the underlying conceptual framework guiding what kinds of assumptions and choices are made in determining those rules. Two deeply entrenched metaphors structure how DNA is conceptualised in most scientific work and society more broadly. DNA function is understood through metaphors identifying DNA as the blueprint, director, or set of instructions governing cell behaviour (Condit et al., 2002; Lindee and Keller, 1997; Nerlich and Hellsten, 2009). DNA structure is understood as linear text, whereby a genome is "the book of life" and DNA is read, written, and edited. Together, they encourage understanding DNA as linear and modular with parts sufficient to elicit functions.

Metaphors are not merely poetic devices or tools for explaining scientific phenomena to non-experts; they are intrinsic to everyday "ordinary language," shaping fundamental cognitive patterns, and are part of the theoretical equipment for doing science (Falkner, 2016; Lakoff and Johnson, 2003; Nietzsche, 1896). Metaphors for what DNA and genes are thought to be like frame how scientists (and others) understand what DNA can and cannot do. Understanding DNA as text and units of DNA as words is so much the norm that it is easy to forget that they are not simply inevitable. The physical structure of DNA does not demand that we understand it as something that can be read, written, and edited like a human language; on the contrary, knowledge about what DNA is and what it does has been constructed through the use of these metaphors (Falkner, 2016). They are apt, but not all-encompassing. What DNA becomes is a function of the tools

used to visualise and work with it such that DNA is iteratively made increasingly in the image of the metaphors used to understand it.

4.5 Improving the metaphor for alternative models

Improving the model underpinning design tools seems to call for a new metaphor. Historians and philosophers of science have called for this move, or at least pointed to the internal contradictions and historical idiosyncrasies embedded in calling DNA a "genetic code" and likening it to written language in the first place. And yet, the panoply of scientific developments enabled by reading and editing "genetic code" should be seen as testimony to the productivity of this metaphor — though, of course, these accomplishments cannot be compared to what might have been enabled by some other metaphor in some alternate historical trajectory.

Regardless, language metaphors are deeply entrenched in molecular genetics. Scientific tools are constructed around the idea of reading and editing genetic text. To reimagine DNA design without discarding this infrastructure, we might think in more detail about what kind of words DNA design units are understood to be, and in what kind of language the proverbial book of life is written.

DNA might be conceived as a machine language, with units belonging to one of a restricted set of operational categories and assembled through logics analogous to electronic circuits — the model behind Cello, for which empirically tested grammars are built-in for the limited design space handled by the tool. DNA might be conceived as a programming language, assembled per a fixed grammar — the model behind GenoCad, requiring users to input their own custom grammars and thus already know the relevant logics for their particular constructs. But DNA might also be conceived as a human or natural language, in which units belong to an open set, are assembled per a complex and fluid grammar, and only become meaningful in the wider context of the "text" and the environment for "reading" it (Cai et al., 2009; Fish, 2000). We advocate for this third option.

It is easy to imagine how intra- and inter-molecular context could be included in future DNA design tools by modelling long-range dependencies and epigenetic factors, respectively — moving models for DNA design away from problematic determinism and closer to recapitulating Keller's reactive, context-sensitive genome. Rather than revising the metaphor of DNA as text, adding context calls for rethinking relatively simple ideas of DNA as a machine or programming language — reinforced by current talk of "programming" DNA — and instead more seriously investigating what might

be learned by thinking of DNA as a more nuanced and context-sensitive language. For example, Stanley Fish's widely accepted theory of interpretive communities argues that (human-language) texts do not contain their meanings, nor are their meanings fixed. Instead, meaning is constructed amongst a text and its reader(s) in a particular context, changing with how, where, and by whom it is read (Fish, 2000). This interpretive flexibility does not mean that any text can mean anything, nor that all interpretations are equally valid at all times; on the contrary, the single most appropriate interpretation is usually obvious in a given situation. "Programming" the "genetic code" encourages pursuing how units of sequence — genes or parts — produce effects (Keller, 2014). Interpreting a context-sensitive text might instead encourage pursuing how the meaning of a (perhaps longer) sequence is produced in a particular setting through the interaction of text and context.

Capturing such additional factors for more sophisticated models could mandate an enormous set of new measurements. To highlight only one example, describing the epigenetic state of a cell requires documenting histone modifications, methylation, and chromatin state, potentially across a variety of cell types, DNA sequences, and cell states. Because epigenetic states vary across time and environment and are sometimes but not always inherited, generalising resulting models to other populations may also require accepting uncertainty about their outputs. The results, however, will model a system of which genetic information is one significant component — the text in its context — rather than the directing or controlling element of a system of effects. In higher-order organisms, novel constructs are often silenced through epigenetic means. Incorporating more information in design tools could enable predicting when and where inserts may be silenced, allowing users to redesign before wasting time and lab resources.

4.6 Conclusion: Reconceptualising "good enough" DNA design

Keeping DNA words but rethinking the genetic language might not resolve the logical issues raised by historical and philosophical critique of the "genetic code" (Falkner, 2016; Lindee and Keller, 1997). However, whether doing so might improve the efficacy of DNA design tools is a different matter. Can a better "good enough" model be built through the same basic assumptions behind current tools? Perhaps. Can an even better "good enough" model be built through rethinking those assumptions? We think so.

As disappointing as it may seem, it is impossible to ascertain how much better DNA design models must become to be good enough to achieve DNA design goals, not least

because those goals are an endlessly moving target; what will be good enough tomorrow will surely not be good enough twenty years from now. Twenty years from now, we hope that the field will also have substantially more knowledge and more experience with DNA design to inform new models. Even so, so long as the conceptual metaphor of DNA as non-interacting linear text remains unchanged, changes to the model built on top can only go so far. At some point, improving models will require a conceptual metaphor that accounts for more genetic complexity, permits looking at strings of base pairs as something other than individual linear words, and motivates new ways of parsing genetic material.

It might be argued that scientists should attempt to escape the constraints of metaphorical language altogether, but this is neither possible nor desirable. DNA cannot be explained as it “really is” because knowledge about DNA does not exist outside of language, and all language is metaphorical in describing things in terms of their selective resemblance to other things. Instead, being aware of the work metaphors enables the possibility of improving them, developing models that are not continually bound by the same framing assumptions embedded in the same metaphor. We should ask: can synthetic genetic assemblies be made to function in accord with the blueprint and text metaphors, creating systems in which DNA governs cell behaviour in terms of modular functions conveyed by discrete units of sequence? But this question leads to another — whether the kinds of DNA assemblies that can be made to function in this way are sufficient to satisfy the ambitions of DNA designers. We think not, but a different way is within the field’s metaphorical grasp.

Chapter 5

Part Crafter: Find, Generate and Analyze BioParts

The founding fathers of Synthetic Biology hoped to one day develop a catalog of standard biological parts which could be designed into complex structures even by someone without significant biological knowledge (Endy, 2005).

In previous chapters, this thesis has already discussed the complex mix of ideas behind how researchers view and design DNA. In particular, Chapter 4 explored some of the problems associated with parts-based design. However, these problems do not necessarily have to do with the idea of parts themselves. All of the alternative conceptual models for DNA design discussed — including the human language model, the SCRaMbLE YAD model, and the Cello Verilog model — involve some unit of genetic function, their differences lying in how these units relate to each other conceptually and how they are assembled together into genetic designs.

Abstraction to some extent is necessary for the progression of the field. Highly complex projects cannot feasibly be designed base pair by base pair. However, previous implementations of DNA parts have not worked well for all purposes.

For example, while Cello's Verilog programming language allows for the complex combination of a variety of well-characterised parts, the number of parts available and the limited contexts in which they can be applied makes them less than optimally relevant for researchers hoping to take on entirely novel experiments (Nielsen et al., 2016).

Meanwhile, the Parts Registry contains a larger variety of parts, but they have been less well characterised. Additionally, very similar sequences which represent the same genetic function — those which have maybe been, for example, codon optimised for a different host, or had the overhangs of a different assembly standard applied — are listed as entirely separate from each other. Instead of being flexible, abstract DNA building

blocks, these parts are only meant to be used in a very narrow context, and, if this context changes slightly, an entirely different part may be required.

Without a clear cut, well-working paradigm with which to use biological parts, the line between these parts and genomic features is blurred. Given that characterising these parts is expensive and often impractical, experiments will, at least in the near future, continue to draw mostly on annotated features instead of a catalogue of standard biological parts. Meanwhile, the vast majority of DNA design software tools rely on the parts-based design paradigm, ignoring the problem of identifying useful features and generating specific part sequences.

The following chapter presents Part Crafter [1], a software tool which allows users to search for genomic features and turn them into biological parts. However, rather than ignoring contextual information, Part Crafter relies on data aggregation from a variety of sources to provide users with as much information about their designed sequences as possible. Additionally, rather than creating ultra-specific parts which can only serve the narrowest of functions, Part Crafter allows sequences to be adapted for any host and any manufacturing standard.

The predictability of a designed biopart is entirely dependant on the amount and quality of its associated data. Part Crafter gathers as much data as possible about each part, giving users a better chance of designing predictable sequences. These parts are not only relevant in a traditional parts-based design context, but in any design paradigm which relies upon genes as a fundamental unit of genetic function.

5.1 Abstract

The field of Synthetic Biology is both practically and philosophically reliant on the idea of BioParts — concrete DNA sequences meant to represent discrete functionalities. While there are a number of software tools which allow users to design complex DNA sequences by stitching together BioParts or genetic features into genetic devices, there is a lack of tools assisting Synthetic Biologists in finding BioParts and in generating new ones. In practice, researchers often find BioParts in an ad hoc way. We present Part Crafter, a tool which extracts and aggregates genomic feature data in order to facilitate the search for new BioParts with specific functionalities. Part Crafter can also turn a genomic feature into a BioPart by packaging it according to any manufacturing standard, codon

[1] Much of the text in this chapter comes from Scher et al. (2019), which was written by me, with support from Guido Sanguinetti and Shay Cohen.

optimising it for a new host, and removing forbidden sites. Part Crafter is available at partcrafter.com.

5.2 Introduction

Parts-based design has been a central tenet of Synthetic Biology since the field's inception. Endy (2005) described how sequences should be designed using an abstraction hierarchy, where devices could be built from parts, and systems could be built from devices. Almost all of the popular DNA design tools for Synthetic Biology are built around the parts-based model, including SnapGene, Genome Compiler, and Benchling. These tools provide a library of features or parts — sequences of DNA that encode for a specific biological function (Baldwin et al., 2015), sometimes called BioParts. Using these libraries, or parts of their own, users can easily design complex genetic systems.

However, DNA design tools rely on existing part libraries and do not provide an automated way of finding and generating parts. This is not surprising: "it is currently easier to assemble multi-part genetic circuits consisting of several BioParts, or even entire genomes, than it is to reliably predict how these BioParts will interact in the final system" (Kelwick et al., 2014). Many existing BioParts rely on disparate pieces of a genome, contextual conditions, and luck for their 'expected' functionality to come to light. Unless characterization experiments have been performed for a part in a wide variety of circumstances, it is impossible to know how the part will behave in vivo.

Part Crafter was built to help users make informed decisions about which genomic features would make sensible BioParts for their experiments. We have enabled rational search of genomic features by leveraging existing annotated data from YeastMine (Balakrishnan et al., 2012), SynBioMine, ThaleMine (Krishnakumar et al., 2016), The Saccharomyces Genome Database (Cherry et al., 1998), UniProt (Consortium, 2014), various NCBI databases, PubMed, and DOOR (Mao et al., 2013). Unlike other, hand-curated parts libraries, like the Registry of Standard Biological Parts, Part Crafter is not limited to a certain number of organisms, manufacturing standards, or a certain subset of parts, but can handle a theoretically unlimited number and variety of genomic features.

Other tools exist which allow users generate to BioParts, such as J5 (Hillson et al., 2011) and GeneDesign (Richardson et al., 2006). However, these tools require that the user already knows what genomic feature they want to turn into a part. Part Crafter allows users who do not have a genomic feature in mind to find and generate the BioParts that they need. Additionally, unlike other Synthetic Biology search tools, Part Crafter does not require the user to provide sequence or annotation data. Our extensive data

Table 5.1 Comparison of Software Tools for Finding and Generating BioParts

	Unlimited Number of Parts/Features	Part Generation	Search Capabilities	User Must Provide the Data	Free
Parts Registry	Only the 20,000 parts in the database	Yes	Full search of parts in the database	No	Yes
J5	Yes	Yes	No search capabilities	Yes	Yes
GeneDesign	Yes	Yes	No search capabilities	Yes	Yes
BioPartsBuilder	Yes	Yes	Full text search and filtering of the GFF file data	No	Yes
Archetype	Yes	No	Full text search of user-provided data	Yes	No
SynBioHub	Yes	No	Full text search of user-provided data	No, though all data is user provided	Yes
Part Crafter	Yes	Yes	Full text search and filtering of extensive aggregated descriptive data	No	Yes

aggregation allows users to search quickly and easily for features, and to find more illuminating results. The differences between Part Crafter and several other related tools are documented in Table 5.1.

While many databases allow users to search for sequences using feature identifiers, scientists are hindered by being unable to link these sequences with functional meaning. Synthetic Biologists especially need a tool which can link sequence text with functional characteristics if they are to be able to design complex systems with a reasonable level of accuracy.

Table 5.2 Fields aggregated from data sources

Database	Fields
SynBioMine	description, feature.description, feature.identifier, feature.name, protein.name
YeastMine	briefDescription, description, name, phenotypeSummary, functionSummary
ThaleMine	briefDescription, computationalDescription
NCBI (protein)	comment, description, keywords
NCBI (nucleotide)	comment, description, keywords
DOOR	species, size, synonyms, symbols
PubMed	ArticleTitle, AbstractText

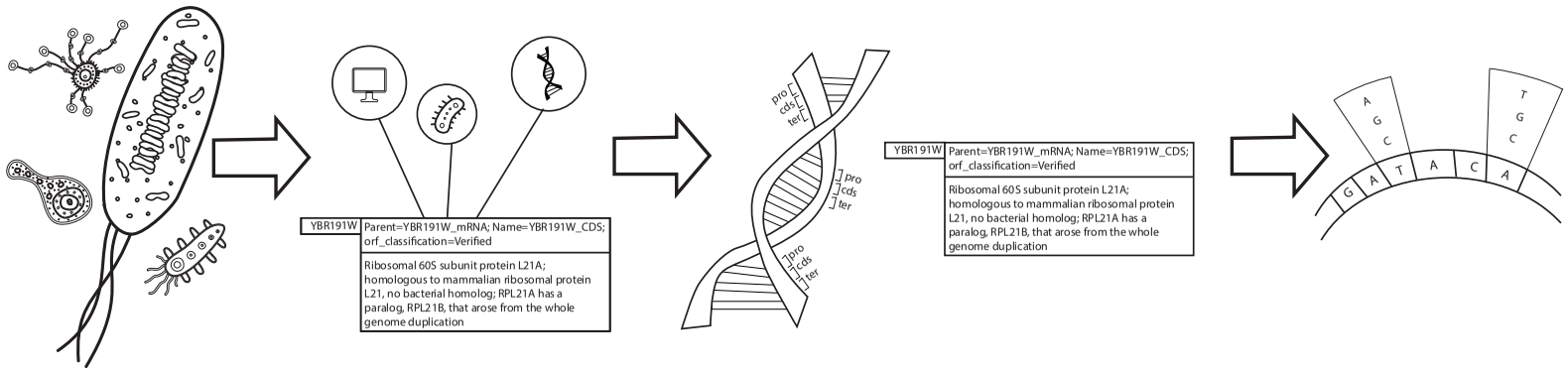


Figure 5.1 The workflow of Part Crafter.

5.3 Workflow

The Part Crafter workflow consists of four steps: Organism Processing, Data Aggregation, Search, and Part Generation. These steps are summarized in Figure 5.1, and described further below.

5.3.1 Organism Processing and Data Aggregation

Part Crafter can process any genome, but it comes pre-loaded with four model organisms: *Saccharomyces cerevisiae*, *Escherichia coli*, *Arabidopsis thaliana*, and *Schizosaccharomyces pombe*. A genome is added by uploading a standard GFF file containing the genome sequence. When a new genome is uploaded to the application, the application first extracts all of the genome's features, and then aggregates descriptive data about each of these features. This data comes from a number of sources, documented above. The specific fields included are documented in Table 5.2. Part Crafter then uses this aggregated data to build an index through Lucene (McCandless et al., 2010). To build an index, Lucene first breaks the text up into terms, and then associates each term with the documents which contain it. This inverted index — so called because it is the inverse of the more natural relationship between documents and terms — allows Lucene to quickly return all documents related to the search terms inputted by the user.

Uploading a new genome involves processing the file, extracting out the annotated features, and, for each of those features, aggregating data from our variety of sources. This involves hundreds of thousands of requests in total, and, because these requests are made with delays in between them to prevent overloading the servers of our data sources, this takes several days for each genome. Because this is a long and computationally intense process taking up significant amounts of memory, only administrative users of Part Crafter can upload new organisms themselves. However, the tool includes a form which allows users to request that a new organism is added.

5.3.2 Search

There are two ways to search for a feature with Part Crafter. First, a user can input a description of the features that they would like. Part Crafter will then output the features in the database whose aggregated texts best match the requested description. In this way, users can find parts associated with a particular functionality. Users can also filter their searches using tags. For example, to search for all features related to cell death, a user would simply search for "cell death." However, to search for all features

related to cell death that occur in *Saccharomyces cerevisiae*, they would use the following query.

```
"cell death" AND organism_name:"Saccharomyces cerevisiae"
```

The tags which can be used with Part Crafter are documented fully on the website. This particular query has 42 results, the top ten of which are summarised in Table 5.3.

Additionally, users can search for features which are maximally similar to a feature of interest. The user inputs the name of their feature of interest, and Part Crafter outputs the features whose descriptive data is the most similar to the descriptive data of the specified feature.

All search results are based entirely on annotation data. This is largely because experimental data is incredibly sparse. However, previously, this annotation data has been disparate, and impossible to search centrally (Mısırlı et al., 2017).

The search functionality was built using the search engine library Lucene (McCandless et al., 2010). The "Search by Function" feature uses full-text search to find matches to the query string. The "Find Similar Features" feature uses the Lucene "More Like This" query, which searches for documents most similar to a selected document of interest.

5.3.3 Parts Generation

Once the user has found their feature or features of interest, Part Crafter allows them to generate the specific BioParts they need.

Part Crafter extracts the feature sequence from its genome, along with its promoter and terminator sequences, if applicable, as specified by the user. Then, the user is able to specify the manufacturing standard they would like to use to package their part, or create their own. The user can then search for forbidden restriction sites, and add their manufacturing standard's required overhangs. However, finding forbidden sites does not automatically remove them — the user can choose to do so by codon optimising their part to remove the sites. Additionally, they can codon optimise the part for any host. If the user already has the strain the feature comes from in their lab, there is no need for them to synthesize the sequences de novo. Instead, they can use the primer generation form to generate primer sequences which will allow them to PCR the sequences out of the host genome. This functionality is all based on that of Genome Carver (Scher et al., 2014). The primers are generated using Primer3 (Rozen and Skaletsky, 2000). The codon optimisation is done using DNAChisel (Zulkower, 2018). The DnaChisel codon

optimisation algorithm uses a dynamic programming approach and codon usage tables for each organism to build sequences which meet desired constraints. These constraints can be, for example, to optimise the sequence for a particular organism, or to remove unwanted sequences, such as restriction sites.

Finally, the user can download their parts in CSV format using the download button. Currently, only CSV format is supported.

5.4 Example Use Case

We illustrate the use of Part Crafter in a simple and generic scenario.

A researcher is investigating programmed cell death in *Saccharomyces cerevisiae*. In order to design synthetic DNA circuits using a common DNA design software tool, they first need to find genes related to cell death, and turn them into BioParts.

First, the researcher navigates to **partcrafter.com**, and then to the "Find Features" section. The search for features using the following query:

```
"cell death" AND organism_name:"Saccharomyces cerevisiae"
```

This query searches for all features in the database related to cell death, limiting the results to those in *Saccharomyces cerevisiae*.

Part Crafter now displays the results, including several genes the researcher would like to turn into BioParts. One such gene is YMR074C, a homologue of Human PDCD5 protein which promotes programmed cell death. The researcher turns this feature into a BioPart by pressing the "Make into a Part" button, which brings up the "Generate a Part" form. This form pulls out the relevant feature sequence, along with the promoter and terminator sequences. The researcher then edits the sequences, adding the relevant manufacturing standard overhangs to the transcriptional unit, and removing forbidden sites.

As the researcher would like to generate a number of parts, they then navigate to the "Bulk Query" tab of the "Generate Parts" screen. There, they are able to generate and edit several parts at once.

The researcher realises that they do not need to synthesize all of the features. They have *Saccharomyces cerevisiae* in their lab strain collection, and can generate some of their required sequences through PCR. For these features, the researcher generates cloning primers using the Primer Generation form.

Table 5.3 Summary of the top results for the query provided in Section 2.2.

Result Number	Systematic Name	Feature Name	Summary of Descriptive Data
1	YNR074C	AIF1	Mitochondrial cell death effector
2	YGL203C	KEX1	Cell death protease essential for hypochlorite-induced apoptosis
3	YNL305C	BXI1	Protein involved in apoptosis
4	YLR011W	LOT6	FMN-dependent NAD(P)H:quinone reductase. Role in apoptosis-like cell death.
5	YMR074C	SDD2	Protein with homology to human PDCD5. PDCD5 is involved in programmed cell death.
6	YGL231C	EMC4	Member of conserved ER transmembrane complex.
7	YHR179W	OYE2	Conserved NADPH oxidoreductase containing flavin mononucleotide (FMN). May be involved in sterol metabolism, oxidative stress response, and programmed cell death.
8	YKL184W	SPE1	Ornithine decarboxylase. Deletion decreases lifespan, and increases necrotic cell death and ROS generation.
9	YPL171C	OYE3	Conserved NADPH oxidoreductase containing flavin mononucleotide (FMN). Has potential roles in oxidative stress response and programmed cell death.
10	YKR042W	UTH1	Mitochondrial inner membrane protein. Implicated in cell wall biogenesis, the oxidative stress response, life span during starvation, and cell death.

Without Part Crafter, this simple pipeline would have required several different databases and tools. For example, to find the features of interest, the researcher would have potentially had to search SGD, NCBI, and Yeastmine. Once they found their list of features, they would have had to add the overhangs by hand, or turned to one of several part generation tools, for example GeneDesign or J5.

In contrast Part Crafter, is a one stop shop. It is possible to find genomic features which would be useful for an experiment, edit them as necessary to turn them into BioParts, and generate primer sequences which will allow the features to be amplified out of an organism. Further, the final generated sequences can be outputted in CSV format for easy use with part-based design tools.

As shown in Table 5.1, several tools exist which allow users to search for genomic features, and several tools exist which can turn specific DNA sequences into BioParts. However, Part Crafter is the only data aggregation and search platform built with the specific aim of helping biologists find and build the BioParts that they need for their experiments. Part Crafter offers a streamlined alternative to using a various other disjointed databases and tools, while also providing more illuminating search results.

5.5 Validation

The validation of our tool was two-pronged.

First, we held a workshop at the UK Centre for Mammalian Synthetic Biology Research, an EPSRC funded centre at the University of Edinburgh. Each of our participants were researchers — PhD students and postdocs — in a Synthetic Biology lab at the University of Edinburgh, and were familiar with popular DNA design tools.

Each participant was asked to choose any *Escherichia coli* or *Saccharomyces cerevisiae* gene, and write a short description of its function. They were then asked to search Part Crafter using their short description. For each of these searches, the gene of interest occurred in the top 2 results 5/5 times, and as the top search result 3/5 times.

Each participant was also asked to rate each of the top ten search results for their query as either "not relevant," "somewhat relevant," or "very relevant." Over all of the queries, 86% of the top 10 results were at least somewhat relevant to the query, and 32% of the results were very relevant to the query. 88% of the top 5 results were at least somewhat relevant, and 40% were very relevant.

The worksheet used in this workshop is available on the Part Crafter website, under the "Help" section. It is also provided in Appendix A. It provides some quick exercises to help users learn how to use Part Crafter. Users are able to submit their completed

worksheets to us, which will help us to continually verify that our search results are of good quality.

Additionally, we programmatically validated our search results by comparing them to another database. WikiGenes (Hoffmann, 2008) is a collaborative database for genetic annotation data. Uniquely for this type of data aggregation, it offers an API, and the data can be edited by anyone, with the intent that researchers will be able to crowdsource their expertise.

In order to validate the Part Crafter search results, we identified 468 genes which have entries in both of these databases. These genes came from *Escherichia coli*, *Saccharomyces cerevisiae*, and *Schizosaccharomyces pombe*, as WikiGenes does not have entries on genes from *Arabidopsis thaliana*. For each of these genes, Part Crafter was queried using the data from the WikiGenes entry as the query string. Our search results were then scored using the mean reciprocal rank, a standard metric to evaluate information retrieval systems. This gave us a score of approximately 0.569, indicating that, on average, the correct gene was listed second in the Part Crafter search results.

Interestingly, there was a significant difference in this figure when looking at genes from the individual organisms. The mean reciprocal rank was 0.668 for just the *Saccharomyces cerevisiae* genes, 0.410 for the *Escherichia coli* genes, and 0.546 for the *Schizosaccharomyces pombe* genes. This variability likely speaks to the comparative quality of annotation data for these different organisms, either in the WikiGenes database, Part Crafter database, or both.

In total, the desired result appeared in the top 5 search results 70.1% of the time. For *Escherichia coli* genes, the desired result was in the top five 48.2% of the time, for *Saccharomyces cerevisiae* 81.6% of the time, and for *Schizosaccharomyces pombe* 69.9% of the time.

These metrics do not offer a perfect comparison between the two data sources. For instance, there are many genes listed in Part Crafter which are not in WikiGenes, which may well have affected the search results. The two databases also, of course, have differing annotation data, which means that we cannot expect a reciprocal rank of 1. That being said, these results are quite encouraging, as they demonstrate that, in general, Part Crafter highly ranks relevant entries.

5.6 Implementation

Part Crafter is implemented as a web application. It consists of seven web services, built around a main user interface written in Meteor. Each of the web services is summarised

Table 5.4 An overview of the web services behind Part Crafter.

Web Service	Implementation	Responsibilities
User Interface	Meteor	The user interface of the application. It also is responsible for the simpler bioinformatics tasks, authentication, file uploads/downloads
Database	MongoDB	Where all of the data associated with Part Crafter is stored
Search Index	Lucene (Java)	An index over the data in the database. This enables full text search. (Jakarta, 2004)
Database/Index Go-Between	Python	Responsible for keeping the search index up to date when the data in the database changes
API	Flask (Python)	Allows users to query the search index
Data Aggregation	Flask (Python)	Pulls data from all listed sources when new organisms are uploaded, as well as periodically to keep data from getting stale
Codon Optmization	Flask (Python)	Performs codon optimisation of sequences (Zulkower, 2018)

in Table 5.4. Figure 5.2 shows how the web services interact. Figure 5.3 shows the relationships between the MongoDB database tables, in UML format as MongoDB is schemaless.

5.7 Availability

Part Crafter is available at **partcrafter.com**. It is not open source, however, docker images of the Part Crafter services are publicly available. Instructions for setting up a Part Crafter server are available on the website. An API is available with instructions for use on the website help page. Additionally, the authors agree to maintain the application for at least two years from the date of publication.

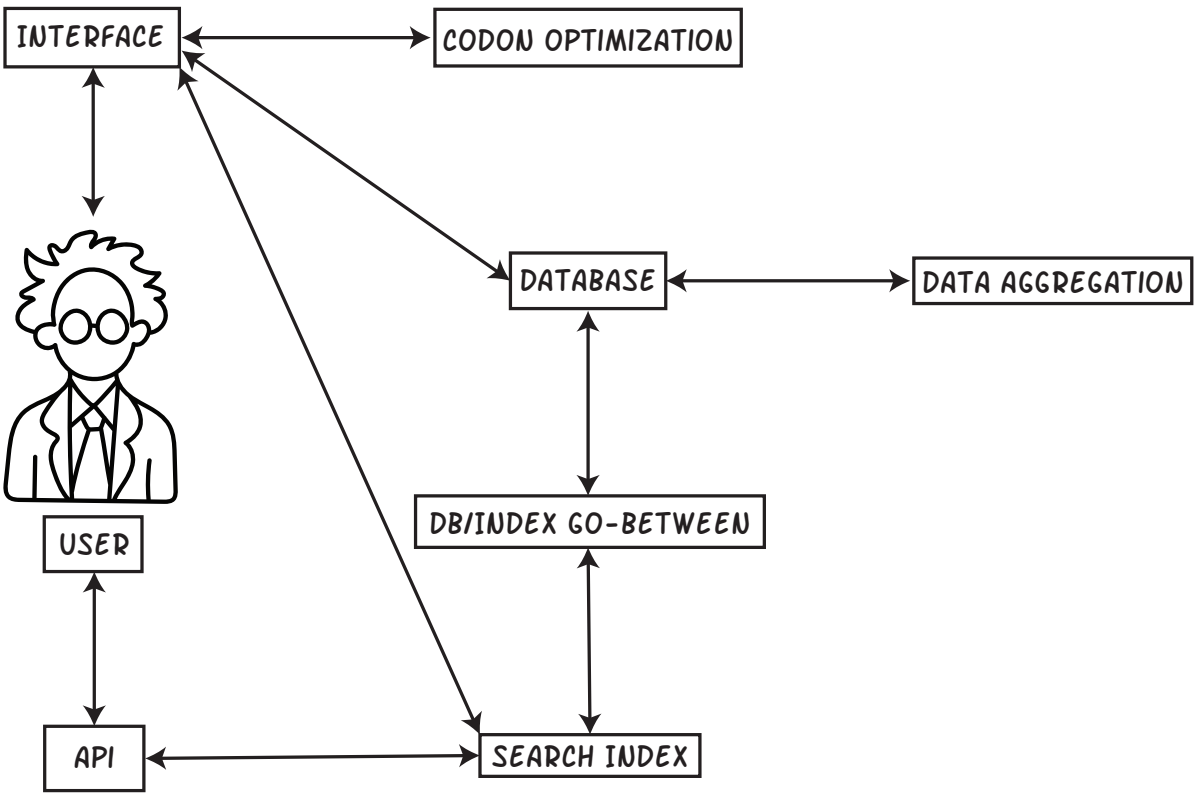


Figure 5.2 An overview of the interactions between web services.

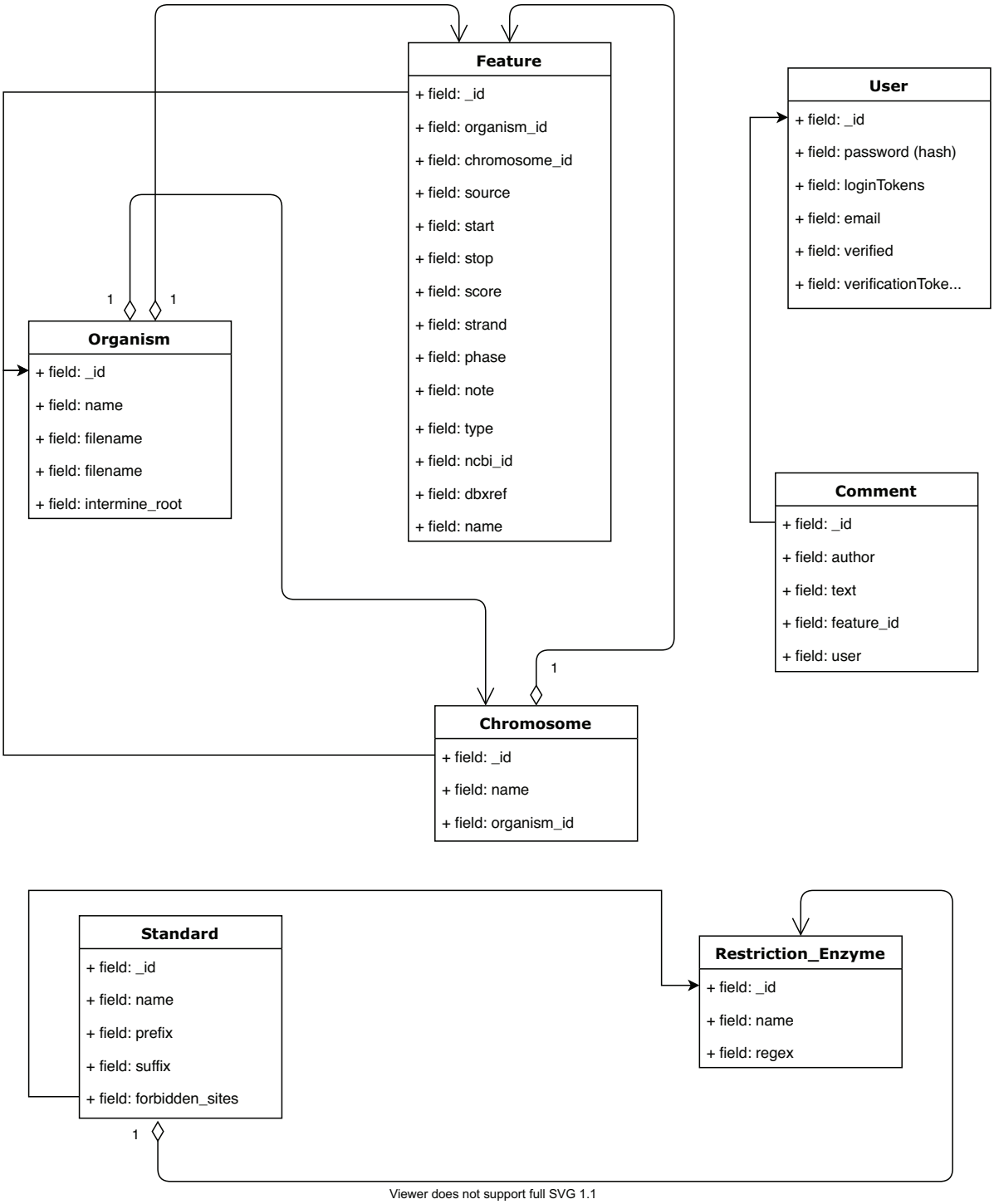


Figure 5.3 The relationships between the database tables.

Chapter 6

Histone Modification Occupancy Prediction Through Language Modelling

6.1 Introduction

The previous chapters of this thesis have discussed various problems with current Synthetic Biology design paradigms, specifically parts-based design. One important flaw in the parts-based design paradigm as applied to Biology is that it offers no way to incorporate epigenetic information into sequence designs. Parts are almost always represented as DNA sequence units, which can be stuck together like DNA building blocks.

However, foreign DNA inserted into higher order organisms has been shown experimentally to alter patterns of methylation and transcription across the entire genome (Doerfler, 2016). It is therefore potentially impossible to design genetic circuits for these organisms using simple parts-based design without eliciting unknown phenotypic changes due to epigenetic alterations at other genomic locations. In order to design DNA sequences deliberately for higher order organisms, it is important to be able to predict what epigenetic changes will occur when the DNA is inserted.

It is also important to understand the nature of the epigenetic patterns at play. By determining the complexity of these patterns, we may be able to classify just how much context needs to be taken into account in a new version of parts-based design, which allows for the incorporation of epigenetic information.

The following chapter will present a sequence of increasingly complex models, each used to predict histone modification occupancy from the local sequence content and

the epigenetic context, with the aim of understanding how a sequence inserted in a certain epigenetic context would behave. By using a series of models which can represent increasingly complex combinatorial patterns, we aim to characterise the complexity of the patterns these histone modifications make.

By modelling histone modifications — one of the simplest and most well understood aspects of epigenetics — we hope to establish a proof of concept for further explorations into the best ways to usefully capture epigenetic information for DNA design tools. Instead of predicting histone modification occupancy for the sake of characterising wild-type chromatin profiles, we hope to lay the groundwork for future builders of software tools, who may choose to use complex language models to represent genomic features, in order to predict how an inserted sequence will disrupt epigenetic control of gene expression.[1]

6.2 Background

As discussed in Chapter 2, DNA sequences are bound to various protein complexes. The combination of DNA and these bound proteins is called chromatin. The chromatin can be in either an opened or closed state, indicating whether the DNA itself is exposed to the cellular machinery responsible for transcription, or coiled up and hidden away by proteins.

The most basic structural unit of the chromatin in eukaryotes is the nucleosomes, which consist of lengths of DNA wrapped around bundles of protein. These proteins are made of eight subunits called histones. Some of these histones can be "modified," meaning that various molecules can be bound to them. It has been shown experimentally that where these modifications occurs affects gene expression.

Several attempts have previously been made to predict chromatin state and gene expression levels using histone modifications. Each histone modification's locations are identified using a separate ChIP-seq experiment, so it is useful to be able to predict the locations of a larger number of modifications based on the locations of a subset of modifications.

The majority of tools available for histone modification modelling aim to use histone modifications as input to classify the overall chromatin state. The most commonly used of these tools is ChromHMM of Ernst and Kellis (2012). Using a multivariate Hidden Markov Model and ChIP-seq data, ChromHMM is able to identify a genome's major

[1]The relevant source code for this chapter is available at https://github.com/emilyscher/Histone-Modification-Language-Models.

reoccurring spatial and combinatorial histone modification patterns. These patterns are harnessed to predict chromatin states across the genome.

EpiCSeg, has a similar goal. Using histone maps, it segments the genome into cell-specific, epigenetic landscapes. These landscapes are then classified so that they can be used to predict gene expression levels (Mammana and Chung, 2015).

Segway uses various types of epigenetic data, including histone modifications and transcription factor binding sites, to train a dynamic Bayesian network which again segments the genome into various chromatin profiles. It can also be used to identify patterns associated with various genomic regions of interest, such as transcription start sites, gene ends, enhancers, CTCF-elements, and repressed regions (Hoffman et al., 2012). Segway's DBN has base pair resolution, which increases training time considerably.

These tools all rely on similar methods and data sources to predict chromatin state. However, our goal has been to approach the problem from another direction — instead of predicting genomic chromatin state from histone modification data, we aim to predict which histone modifications will bind to inserted sequences, and to thus predict where a novel sequence ought to be inserted to minimise unwanted epigenetic effects. While for our goal it is still important to attempt to understand the combinatorial patterns which dictate histone modification occupancy, we are aiming to predict this occupancy for novel sequences, not to classify wild-type gene expression levels. Because of this, it was important to incorporate both epigenetic and sequence information into our models, as we expect the novel sequence to have a particularly meaningful influence on the epigenetic factors at play.

6.3 Methods

6.3.1 Data Processing

The following work was undertaken using *Saccharomyces cerevisiae* and *Homo sapiens* genomes. When generating testing and training datasets, *Saccharomyces cerevisiae* nucleosome and histone modification locations were taken from Weiner et al. (2015). For *Homo sapiens*, nucleosome locations were predicted using NSeq (Nellore et al., 2013). Histone modification CHIP-seq data was taken from ENCODE (Davis et al., 2018) and was aligned to the reference genome using Bowtie 2 (Langmead and Salzberg, 2012). Specific histone modification locations were identified using the MACS 2 peak caller (Zhang et al., 2008).

Table 6.1 Statistics for the bigram and trigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors. These statistics are averaged over all 26 *Saccharomyces cerevisiae* histone modifications for which we have data.

Model	Accuracy	F1	Precision	Recall	AUC (ROC)	AUC (Precision-Recall)
Bigram	0.981	0.779	0.851	0.745	0.745	0.619
Trigram	0.981	0.784	0.859	0.748	0.748	0.624

For each of the identified nucleosomes for each genome, a vector of length n was generated, where n is the number of possible modifications present in that nucleosome. In the case of *Saccharomyces cerevisiae*, $n = 26$, and in the case of *Homo sapiens*, $n = 12$. These n values do no represent all histone modifications present in these genomes in vivo, only the number for which we have CHIP-seq data.

Each entry in these vectors is either a 0 or 1, representing the absence or presence of the associated modification respectively. The order of each of the histone modifications within these sets is arbitrary as the structure of histones within a nucleosome does not translate to a sequential order. Additionally, a second vector of length four was generated for each nucleosome, representing the number of each type of base (A, T, C, G) which occurs in the genomic region associated with the nucleosome.

This data processing workflow is represented in Figure 6.1.

The final output of this data processing is a sequence of vectors for each organism. Each vector represents a nucleosome, and each entry in the vector represents the presence or absence of a particular histone modification. Each vector is associated with a particular place along the genome, and with sequence information related to that particular genomic position.

6.3.2 N-gram Models

N-gram models are some of the simplest commonly used language models. For that reason, they were chosen as the first in our gradient of models. Using the sequence of vectors described above, a series of bigram and trigram models were built for *Saccharomyces cerevisiae*. The occupancy of each histone modification was predicted individually. The average statistics for these models are shown in Table 6.1.

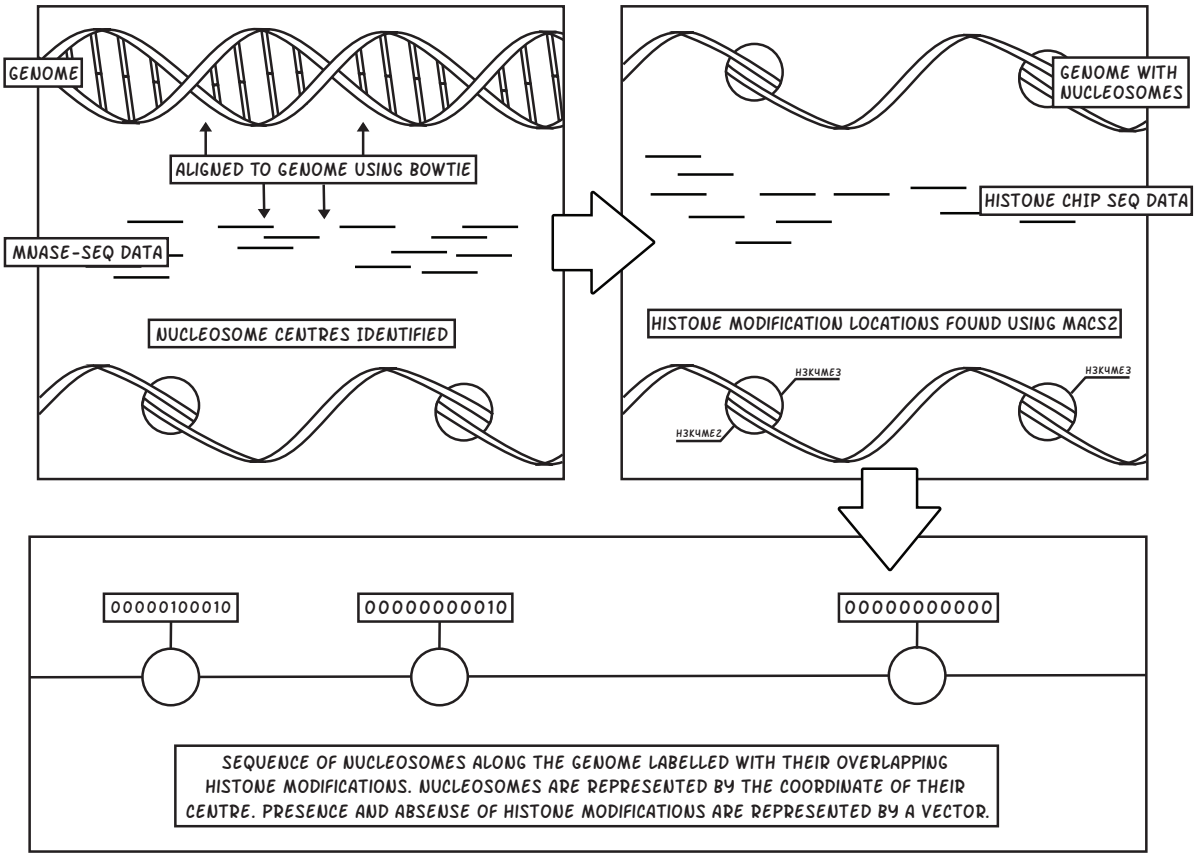


Figure 6.1 An overview of the data processing workflow. Beginning with sequence data, MNase-seq data for localising nucleosomes, and ChIP-seq data for localising histone modifications, we generated a series of vectors representing a nucleosome with it's associated histone modifications.

A logistic regression was trained using sequences of n nucleosome vectors, as well as the nucleotide frequencies associated with each nucleosome, as it is established that histone modifications correlate in some way with the makeup of their associated genetic region (Benveniste et al., 2014). For example, the bigram model predicting H3K4me3 occupancy makes its predictions using:

- The presence or absence of all histone modifications for the given nucleosome, other than H3K4me3
- The presence or absence of all histone modifications for the previous nucleosome in the sequence
- The nucleotide counts for the given nucleosome
- The nucleotide counts for the previous nucleosome

after being trained on all of the histone modification and nucleotide data from the training sequence of histone modification vectors, incorporating the counts of each bigram which occurs in that sequence.

Similar bigram and trigram models were trained for all 26 histone modifications for which we had occupancy data.

N-gram models fall into the Regular Grammar category of the Chomsky Hierarchy (Lüdeling and Kytö, 2008). While they are able to represent simple, localised patterns, they are unable to capture long range dependancies. Experimental research has shown that various epigenetic proteins demonstrate genome-wide patterns (Doerfler, 2016). Therefore, the high accuracy of these simple models is somewhat surprising. However, it is uncertain how such a model would fare when tested on an entirely novel sequence inserted into a host organism. In such a situation, where a poorly represented histone modification or n-gram might be more commonly represented than in the host genome, an n-gram model would undoubtably adapt poorly.

Additionally, because the histone modification data is particularly sparse, the accuracy and area under the ROC curve are inflated for some histone modifications. Figure 6.2 shows the precision-recall curve for a bigram predicting H3K4ac occupancy. This particular model has both a high accuracy (98.3%) and a high precision-recall AUC (0.925). Figure 6.3 shows the same curve for a bigram predicting Htz1 occupancy. While this model also has a high accuracy (98.2%), it has quite a low precision-recall AUC (0.093). AUC statistics for each histone modification are shown in Appendix B.

While these n-gram models offer a high accuracy, they do not, in all cases, perform better than a no-skill model which always predicts empty histone modification vectors.

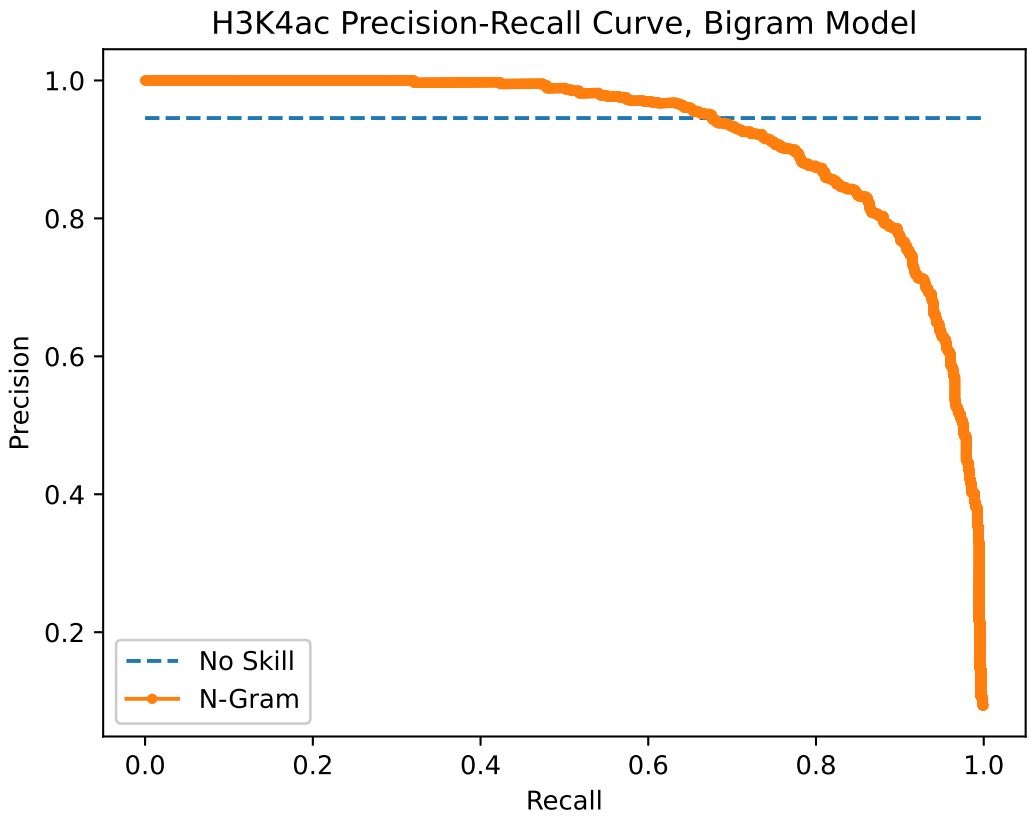


Figure 6.2 The precision-recall curve for the bigram model predicting H3K4ac occupancy. The no-skill line represents a model which always predicts 0. This particular model has both a high accuracy (98.3%) and a high precision-recall AUC (0.925).

These simple models are not capable of representing complex structures. While they often predict correctly in situations they have seen before, their efficacy is tightly tied to the quality and quantity of training data, and they would likely perform poorly when tested on data even moderately dissimilar from what they have been trained on.

6.3.3 Hidden Markov Models

Hidden Markov Models can be viewed as a stochastic extension to the regular grammar. The complexity offered by the HMM over an n-gram model is intuitive, as n-gram models can be built using simple, constrained HMMs. However, the hidden state sequence allows these models to represent more complex combinatorial patterns. Even when they have Markovian dynamics, the latent states can create long-range correlations in observed sequences. By training the model parameters on histone modification data, it is possible

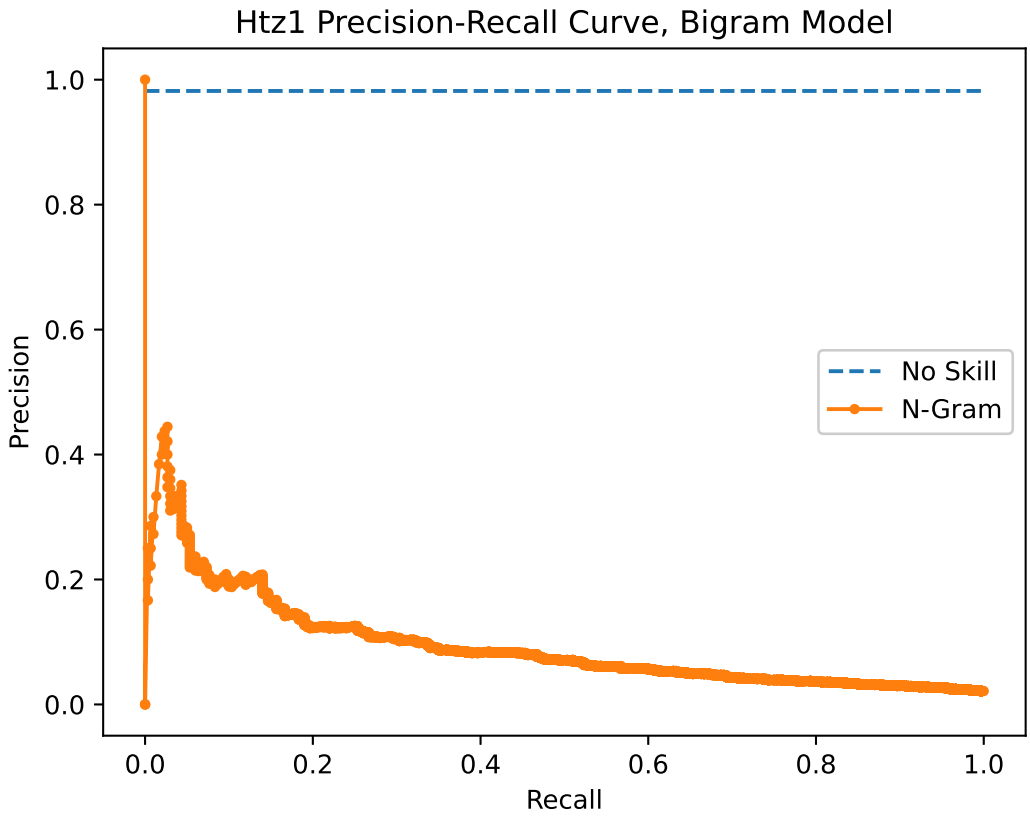


Figure 6.3 The precision-recall curve for the bigram model predicting Htz1 occupancy. The no-skill line represents a model which always predicts 0. While this model has a high accuracy (98.2%), it has quite a low precision-recall AUC (0.093).

to learn about the underlying structures we are attempting to represent. For more information about these models, see Chapter 3.

For the following models, we aimed to train the HMM model parameters such that the hidden states in some way represent different chromatin states. In different chromatin states — for example, in an open, highly expressive state — we would expect specific histone modifications to be more commonly observed than in closed, highly silenced states.

In addition, for some of the models constructed, we included a covariate. Because chromatin state and, therefore, histone modification occupancy has been shown to be correlated with various DNA sequence features (Zacher et al., 2017), we chose to build Hidden Markov Models which relied upon both histone modification data and genomic sequence data to make their predictions.

For the rest of the following section, "standard" HMMs will refer to those models which are not reparameterized by DNA sequence information. Meanwhile "custom" HMMs are those which have been reparameterized by sequence data.

Using a Pytorch implementation of the Forward-Backward algorithm, a series of standard Hidden Markov Models were trained. The sequences used for training were sequences of histone modification vectors, produced through the process described above. For these tests, we used the *Saccharomyces cerevisiae* chromosome one, as well as data for 26 histone modifications. As is typical for a Hidden Markov Model, we trained three matrices per model:

- A transition matrix, T, representing the probabilities of moving from one state to another. This matrix is of the form $S \times S$ where S is the number of states.
- An emission matrix, E, representing the likelihood of a particular output being emitted from a particular state. For these models, the emission matrix has the dimensions $S \times 26$, where 26 is the number of potentially present histone modifications. The likelihood of seeing a particular histone modification vector emitted from a particular state is the product of the relevant probabilities.
- An initial transition vector, $T0$, representing where the state sequence is likely to begin.

These standard HMM parameters were trained using the Baum-Welch algorithm. Once learned, these parameters were used to build the custom HMMs.

In the custom models, when T would normally be referenced, instead a wrapper function is called. Given $t_{\alpha\beta}$, the probability of the transition from state α to state β as learned from the first HMM, the custom HMM transition probability is given as:

$$t'_{\alpha\beta} = \frac{\exp\left(w_\alpha^\beta x_i\right)}{\sum_\beta \exp\left(w_\alpha^\beta x_i\right)} \tag{6.1}$$

where i represents an index in the nucleosome vector sequence. The weights are initialised at the beginning of the optimisation such that this function will yield the same values as the standard HMM transition matrix before the weights are trained.

We then learned the series of w values referenced above using stochastic gradient descent. The parameters of this model are a $3 \times S \times S$ matrix, in which the last $S \times S$ matrix represents the bias terms, which were initialised to the learned standard transition matrix values. The first two $S \times S$ matrices are dependant on the vector gv, which represents the absolute values of the differences between the C and G counts in the current nucleosome as compared with the mean.

The two-part model building process is shown in Figure D.9.

The above system was tested using state numbers 2 through 5, using 10-fold cross-validation. The sequence data used as the covariate in these tests was GC content, the percentage of the sequence which is made of Cytosine and Guanine. This covariate was chosen because it is relatively simple to represent, and because high GC content has been shown to be correlated with higher levels of mRNA production, and thus higher levels of gene expression (Kudla et al., 2006). However, any sequence covariate could potentially be used.

Table 6.2 shows the test sequence log likelihood values for both the standard and custom Hidden Markov Models, as well as the Description Accuracy for the two models. Description Accuracy is a metric which is used to show how well a model describes the data it is meant to represent. Description Accuracy is defined as:

$$DA = f\left(1 + \frac{\log_s P(Y|D)}{l}\right) \tag{6.2}$$

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \frac{1-e^{-0.25x}}{1+e^{-0.25x}} & \text{if } x < 0 \end{cases} \tag{6.3}$$

where l is the sequence length, s is the number of outputted symbols, Y is the data, and D is the model.

A description accuracy > 0 indicates that the model predicts the stochastic sequence more accurately than random, and a description accuracy of 1 indicates that it has been predicted perfectly (Srinivasan et al., 2017).

The results in Table 6.2 indicate that the custom HMMs offer a modestly improved log likelihood. However, it is encouraging to see that the custom HMMs also offer improved

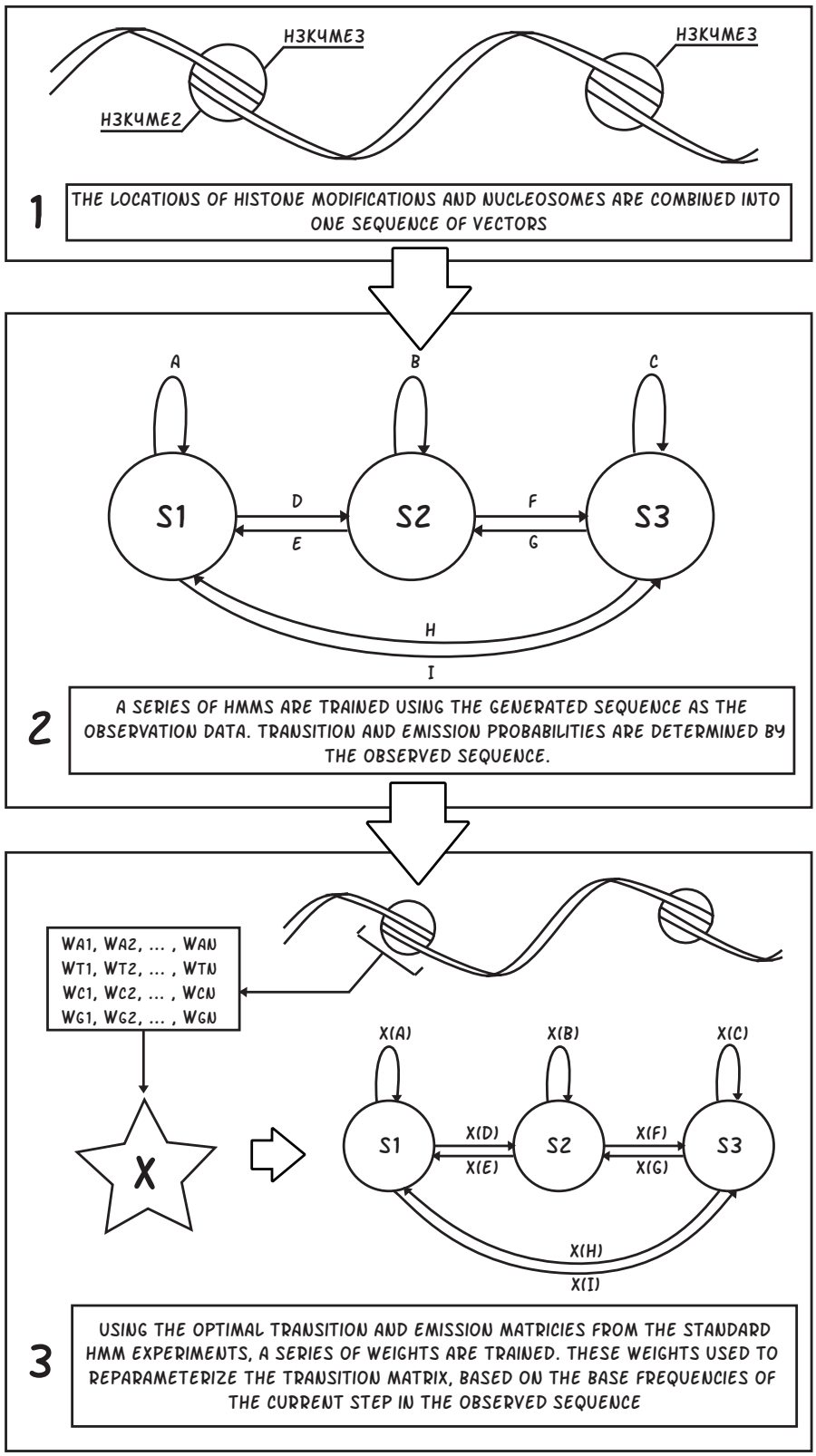


Figure 6.4 The two step model training process. We first trained a standard HMM. Then, using the transition and emission vectors from that model, trained our custom HMM where the transition matrix is re-parameterised by base counts.

description accuracy, indicating that the custom HMMs do indeed represent the problem at hand more accurately.

Figure 6.5 compares the prediction accuracies for three histone modifications between the standard and custom HMMs. Further statistics for more histone modifications — all of them which occur at least 50 times in the dataset — are available in Appendix C.

For the majority of the histone modifications shown, there is little difference in accuracy and F1 between the standard and custom HMMs. However, In the case of H2AS129ph, the custom HMM offers an 8% improvement in accuracy, while for H3S10ph it offers a 4% improvement in accuracy. Interestingly, H3K4me3 predictions had some of the worst accuracies. H3K4me3 is a relatively well studied histone modification, and is associated with high levels of gene expression. It may well be that H3K4me3 occupancy is significantly predictive of other histone modifications, while being difficult to predict in and of itself.

These same tests were run for another version of the custom HMM, which used A, T, C, and G counts as the covariates, instead of only G and C counts. The results for these other tests are available in Appendix C.

Figures 6.6 through 6.8 show the learned parameters from this two-step training process, given a model with five states. The same figures for the other state numbers are available in Appendix D.

Figure 6.7 shows the emission matrix for the model, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Table 6.3 shows the histone modifications associated with each state, as well as the effects said modifications are expected to have on transcription. State 0 and State 2 appear to be extremely similar, indicating that they actually represent the same latent state. In the four state model in Appendix D, these two states are indeed combined into one. Both of these states seem to be likely to emit histone modifications associated with transcriptional activation, meaning that they may well mimic an open chromatin state.

State 1 emits histone modifications associated with a variety of functions — DNA repair, transcriptional repression, transcriptional activation, telomeric silencing, genome stability, and DNA repair. State 1 could therefore represent a regulatory chromatin state, which can both increase and decrease gene expression depending on other factors.

State 3 could potentially represent a chromatin state related to telomeric silencing. However, the state is not strongly associated with any particular histone modification, meaning that it is likely that this association is an artifact of the training data.

Table 6.2 This table shows the changes in test sequence log likelihood for each state number, and for the standard and custom HMMs. The final three columns show the computed description accuracies for each model. The custom HMMs' log likelihoods offer a modest improvement over those of the standard HMMs. Similarly, the DA slightly improves for the custom HMMs.

# of States	Standard Test Seq LL	Custom Test Seq LL	Delta	Standard DA	Custom DA	Delta
5	-263.018	-260.235	2.783	0.473	0.479	0.006
4	-281.485	-269.974	11.511	0.440	0.449	0.009
3	-724.580	-720.448	4.132	0.356	0.360	0.004
2	-366.148	-358.424	7.723	0.352	0.376	0.025

State 4 could represent a chromatin state which activates only specific genes, but for this state the associations are again somewhat weak.

These states in some ways match biological expectations, but some pieces of the puzzle are missing. For example, we would expect to see a heterochromatic state, but unfortunately the histone modification data available to us lacks modifications associated with heterochromatin (according to Zhao and Garcia (2015), H3K9me2, H3K9me3, H3K56me1, H3K56me3, and H3K64me3). Tests with higher numbers of states may yield more granular representations of chromatin states, though given that it seems States 0 and 2 represent the same chromatin state in the tests above, this may not be the case.

Table 6.3 The histone modifications each state is associated with, as well as the effects those histone modifications are believed to have on transcription (Jiménez-Chillarón et al., 2014; Magraner-Pardo et al., 2014; Zhao and Garcia, 2015).

State	Histone Modification	Expeted Effect
0	H3K14ac	Transcriptional activation
	H3k18ac	Transcriptional activation
	H3K23ac	Transcriptional activation
	H3K4ac	Transcription activation at some promoters
	H3K4me3	rDNA/telomeric silencing, transcriptional activation
	H3K9ac	Transcriptional activation
	H4K8ac	Transcription regulation
1	H3K36me3	Regulation of DNA damage repair and the maintenance of genomic stability after DNA damage
	H3K4me	Transcriptional activation
	H3K79me3	Telomeric silencing

	H4K16ac	Both transcriptional activation AND repression activities
	H4R3me2s	Transcriptional activation
2	H3K14ac	Transcriptional activation
	H3k18ac	Transcriptional activation
	H3K23ac	Transcriptional activation
	H3K4ac	Transcription activation at some promoters
	H3K4me3	rDNA/telomeric silencing, transcriptional activation
	H3K56ac	Transcriptional activation; DNA damage
	H3K9ac	Transcriptional activation
	H4K8ac	Transcription regulation
3	H2AS129ph	Telomere silencing
4	H3K4me3	rDNA/telomeric silencing, transcriptional activation
	H3S10ph	Transcriptional activation of early genes

6.3.4 Probabilistic Context Free Grammars

While N-gram models fall into the Type-3 (regular grammar) category of language models, and Hidden Markov models fall into the Type-2 (context-free) category, PCFGs are most similar to Type-1 language models. In a similar way to how HMMs offer a stochastic extension to regular grammars, PCFGs extend Context Free Grammars. Just as it is possible to represent n-gram models as constrained HMMs, it is possible to represent HMMs through constrained PCFGs.

Several different PCFGs were constructed using the sequence of histone modification vectors from the *Homo sapiens* genome, as described above. In these models, non-terminals represent features annotated along the genome sequence. A tree of non-terminals was built for each gene in the human genome based on the structure of the annotated features it contains. Each leaf node in these trees is associated with a particular genome sequence, and, therefore, a set of histone modification vectors.

It is very common for each human gene to be associated with several transcripts, and for each of those transcripts to contain several exons. This structure, among other less common ones, is captured by these trees. The relationships between annotations are recursive — a gene can contain another gene, an exon can contain another exon, etc. For this reason, PCFGs represent them well, given that these models are particularly adept at handling recursive data.

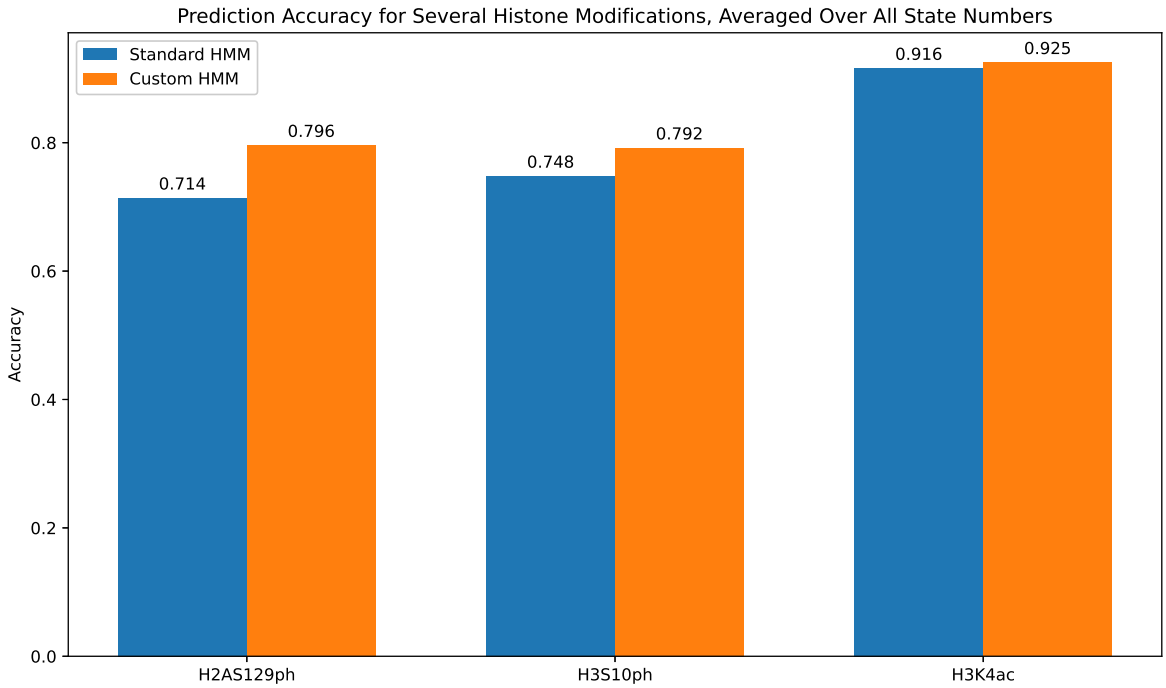


Figure 6.5 The prediction accuracies for three histone modifications, compared between the standard and custom HMMs. All three of these accuracy rates improved when nucleotide data was incorporated into the model by re-parameterising the transition matrix.

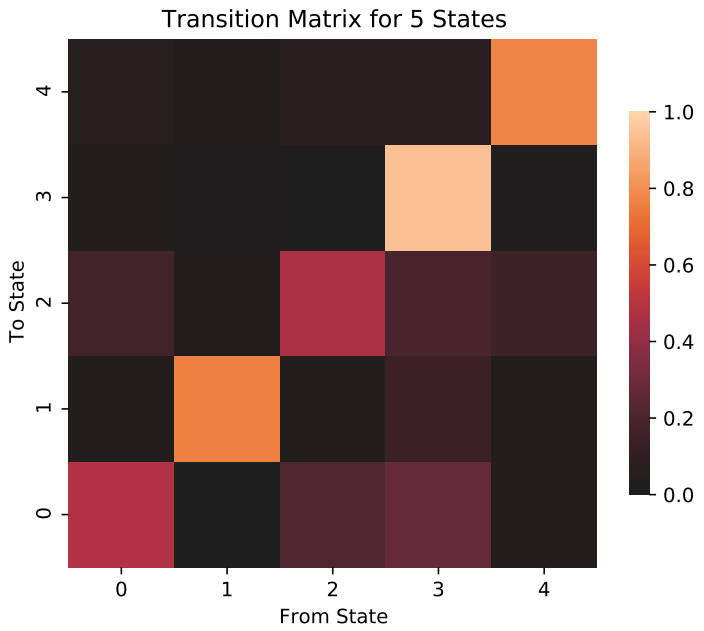


Figure 6.6 The transition matrix for an HMM with 5 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices.

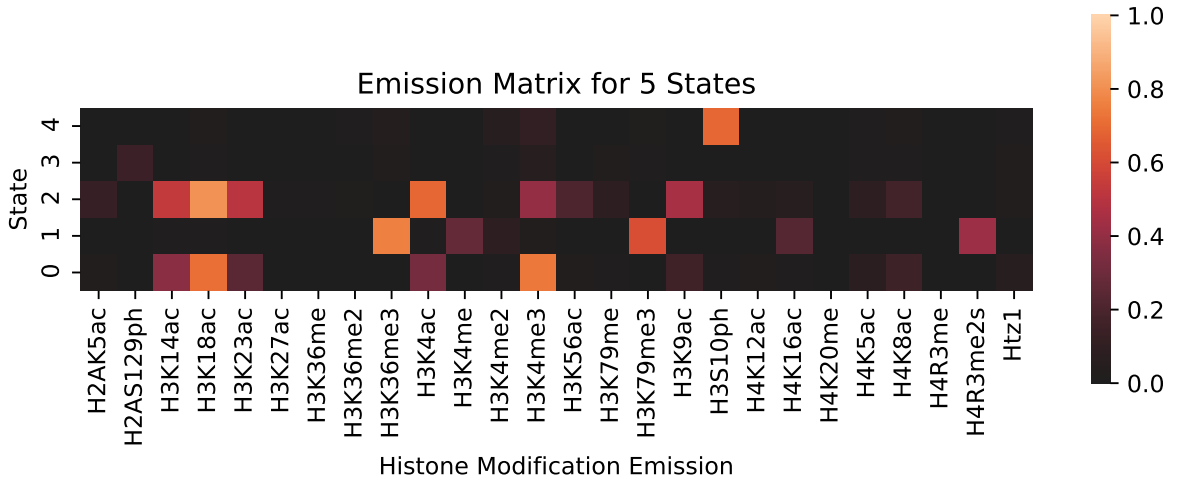


Figure 6.7 The emission matrix for an HMM with 5 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3.

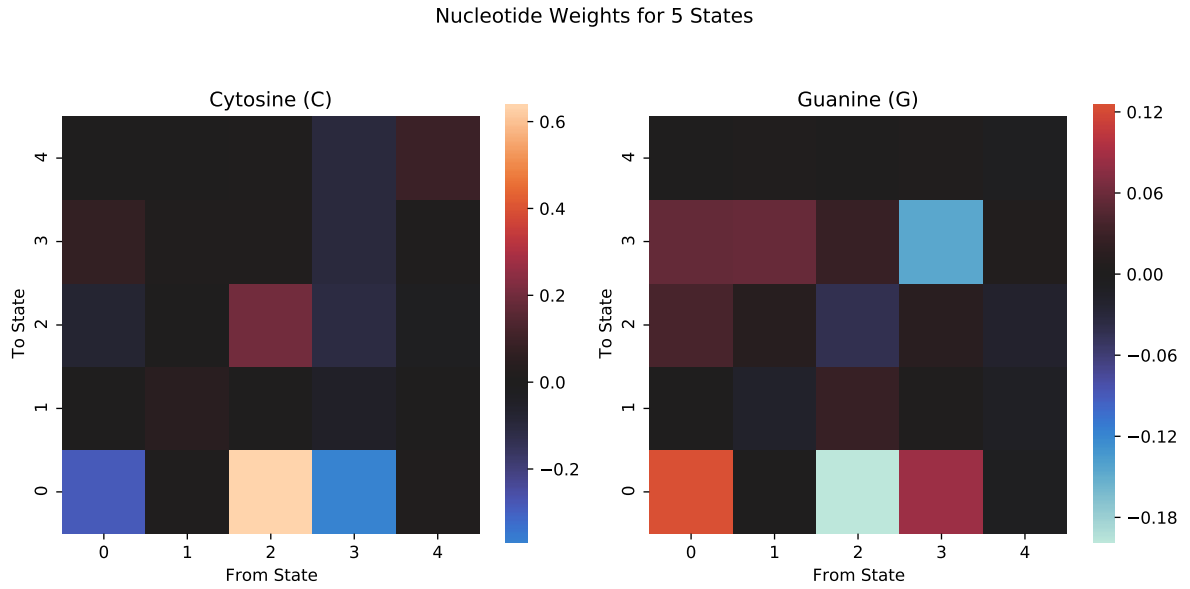
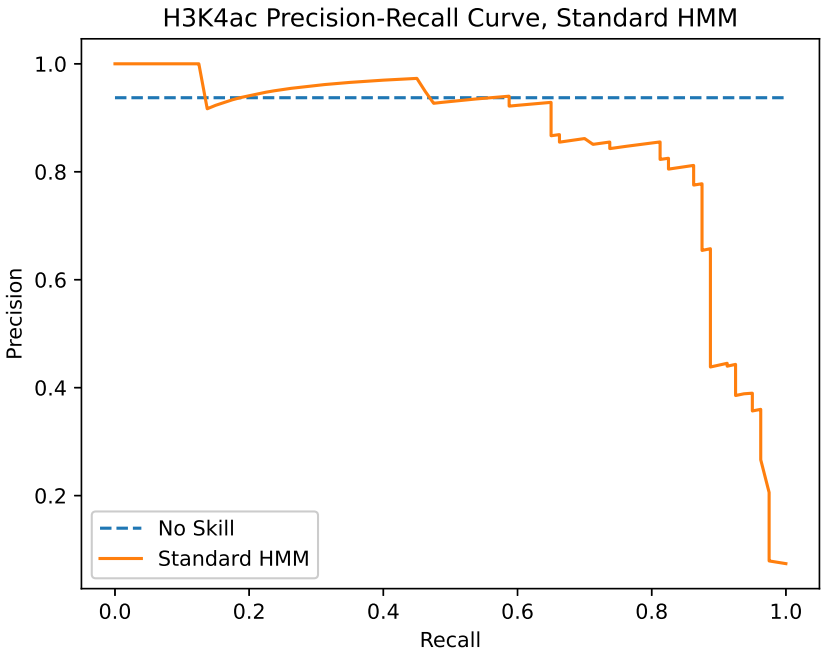
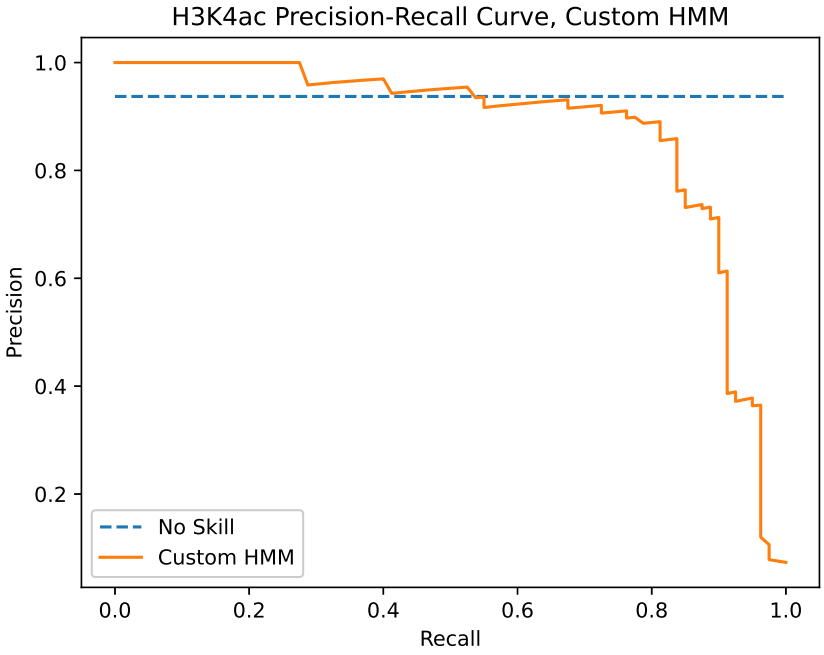


Figure 6.8 The genomic weights learned for a custom HMM with 5 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring.

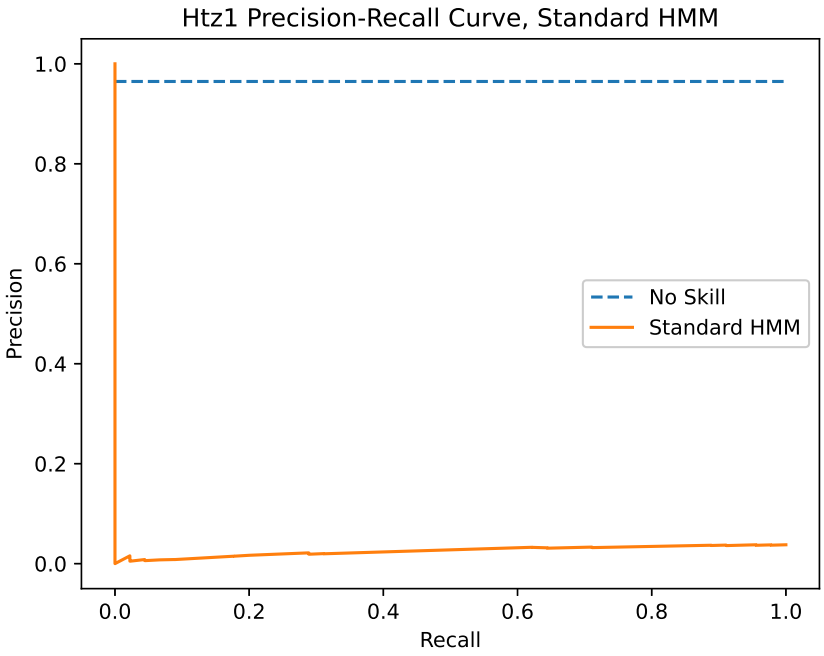


(a)

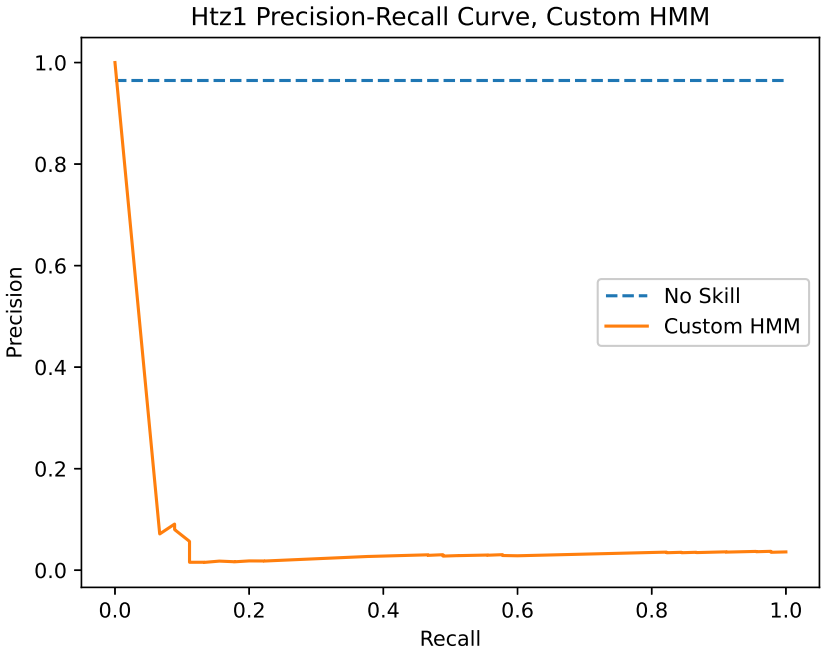


(b)

Figure 6.9 The precision-recall curves for 5-state standard (a) and custom (b) Hidden Markov models, when predicting H3K4ac occupancy. The no-skill lines represent models which always predict 0. The curves are similar to Figure 6.2. The custom HMM demonstrates slightly better performance than the standard. The precision-recall AUC for the standard model is 0.855, while the custom model had a slightly higher AUC of 0.875.



(a)



(b)

Figure 6.10 The precision-recall curves for 5-state standard (a) and custom (b) Hidden Markov models, when predicting Htz1 occupancy. The no-skill lines represent models which always predict 0. The curves are similar to Figure 6.3. The custom HMM demonstrates slightly better performance than the standard. The precision-recall AUC for the standard model is 0.025, while the custom model had a slightly higher AUC of 0.064.

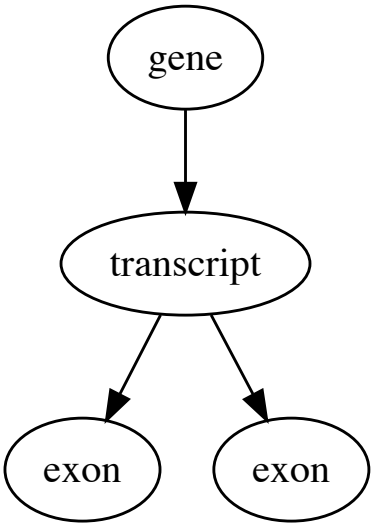


Figure 6.11 An example of one of the simplest annotation trees generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors.

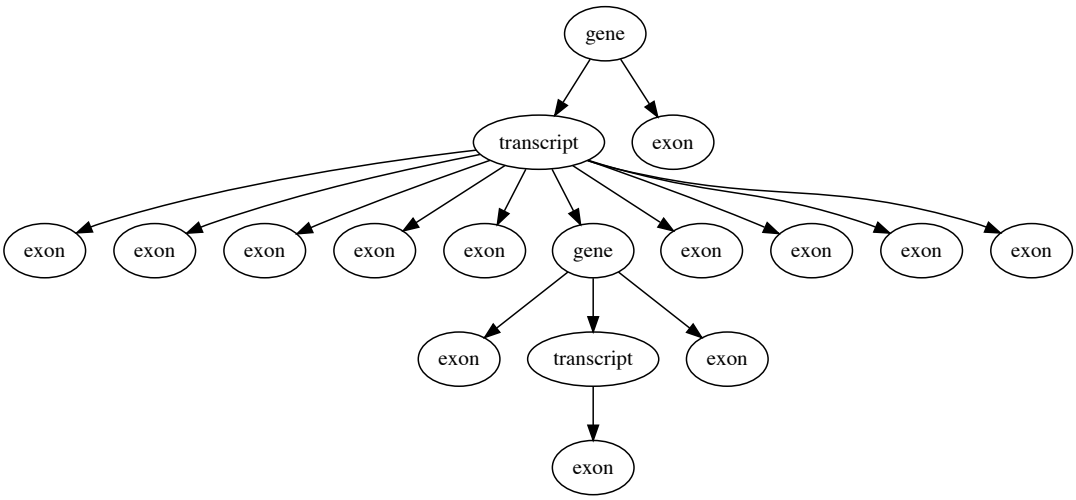


Figure 6.12 An example of an average-sized tree generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors.

Figure 6.13 An example of a more complicated tree generated from the human genome. Each leaf node is additionally associated with a sequence of histone modification vectors.

Examples of these trees are shown in Figures 6.11 through 6.13. Figure 6.11 shows an especially simple tree structure, while Figure 6.13 shows a more complex one. However, while a very few trees in the data set are even more complex than Figure 6.13, the majority most closely resemble Figure 6.12.

Using these trees as training data, we attempted to learn the grammatical structure of genomic features based on sequences of histone modifications. Our approach involved building three different models. The first and simplest was a supervised PCFG, where the grammar is taken directly from the training data, and the probabilities of the various rules learned based on frequencies of occurrence in the training data. We then used a C implementation of the CKY algorithm to parse sentences based on these learned rule probabilities (Johnson, 1998).

The other two models were both built using unsupervised PCFG implementations, which rely upon an adaptation of the Inside-Outside algorithm to learn the production rules governing the relationships between non-terminals, as well as the probability distributions of these rules. The first of these unsupervised models was a "Neural PCFG" as implemented by Kim et al. (2019). While most PCFGs are parameterised by determining how often each rule occurs (as is done in the previous PCFG), these neural PCFGs' rule probabilities are represented in a distributed (neural) fashion.

As discussed in Chapter 3, in a standard PCFG which uses the Inside-Outside algorithm for training, the production rules must be in Chomsky Normal Form. If S is the start symbol, N is the set of non-terminals, and Σ is the set of terminals, then R is a set of rules which take the following form:

$$\begin{aligned} S \rightarrow A, \quad & A \epsilon N \text{ or} \\ A \rightarrow BC, \quad & A \epsilon N,\ B, C \epsilon N \text{ or} \\ T \rightarrow w, \quad & w \epsilon \Sigma. \end{aligned} \tag{6.4}$$

A PCFG is defined by this ruleset, G, as well as a set of probabilities over said rules, π, where π_r is the probability of the rule r.

In neural networks, an embedding is a mapping between continuous values and discrete, categorical values. In the neural PCFG, input embeddings w_N are introduced on the left side of each rule, for each symbol in N. For each rule of type r, the parameterisation of π_r is as follows:

$$\pi_{S\to A} = \frac{\exp\left(u_A^\intercal f_1(w_S)\right)}{\Sigma_{A'\epsilon N}\exp\left(u_{A'}^T f_1(w_S)\right)},$$

$$\pi_{A\to BC} = \frac{\exp\left(u_{BC}^\intercal(w_A)\right)}{\Sigma_{B'C'\epsilon M}\exp\left(u_{B'C'}^T(w_A)\right)}, \tag{6.5}$$

$$\pi_{T\to w} = \frac{\exp\left(u_w^\intercal f_2(w_T)\right)}{\Sigma_{w'\epsilon\Sigma}\exp\left(u_{w'}^T f_2(w_T)\right)},$$

where M is the product space $N \times N$, and both f_1 and f_2 are multilayer perceptrons with two residual layers.

While this neural parameterisation does not alter the underlying probabilistic assumptions required for PCFGs, it allows rule types to share distributed representations, improving the model's ability to learn meaningful grammars.

The last PCFG built was a "compound PCFG," the implementation of which was also taken from Kim et al. (2019). Compound probability distributions generalise mixture models for continuous use cases. This implementation of compound PCFGs relies upon rule probabilities being generated by the fucntion:

$$z \sim p_\gamma(z), \quad \pi_z = f_\gamma(z, E_G), \tag{6.6}$$

where $p_\gamma(z)$ represents the prior (with parameters γ, a spherical Gaussian for this implementation), and f_γ is a neural network that adds the input symbol embeddings to z and yields the rule probabilities for each sentence. π_z, the sentence level rule probabilities, are given by:

$$\pi_{z,S\to A} \propto \exp\left(u_A^\intercal f_1([w_S; z])\right),$$

$$\pi_{z,A\to BC} \propto \exp\left(u_{BC}^\intercal([w_A; z])\right), \tag{6.7}$$

$$\pi_{z,T\to w} \propto \exp\left(u_w^\intercal f_2([w_T; x])\right),$$

where $[w; z]$ represents the concatenation of vectors done by the neural network described above. It is then possible to sample a tree from a PCFG with rule probabilities π_z with

$$t \sim PCFG(\pi_z), \quad x = \text{yield}(t). \tag{6.8}$$

This implementation is akin to a continuous mixture of PCFGs, joined by context-free assumptions conditioned on z. This compound PCFG implementation is able to represent more intricate patterns than a normal PCFG, as each tree has its own rule probabilities. That being said, the above generative model is only context-free when conditioned on z.

This compound PCFG model also allows for nodes to depend on one another through this shared latent variable. Most PCFGs cannot represent, for example, dependancies between a child and an ancestor further back than its parent, or dependancies between siblings, as forgoing context free assumptions makes training intractable. By using the above generative model, compound PCFGs are able to capture such relationships without sacrificing the benefits of being context-free (Kim et al., 2019).

Using the implementations from Kim et al. (2019), an unsupervised neural PCFG was trained using the trees described above. Additionally, an unsupervised compound PCFG was trained using the trees described above, with GC content used as the latent variable, making this model in some ways analogous to the custom HMMs described in the previous section. Both of these models were trained using approximately 8,000 gene trees generated from chromosomes one through three of the human genome.

Table 6.4 shows statistics for these tests. Entirely unsurprisingly, the supervised model was the clear winner both in terms of log likelihood and description accuracy. However, what was surprising was how poorly the neural and compound PCFGs performed in terms of description accuracy. With average values of -0.061 and -0.011, they performed worse than random with regards to this metric. The compound PCFG did perform significantly better than the neural PCFG, which indicates that GC content may well be a latent variable which links genomic annotations, but the models clearly do a poor job of capturing the grammatical structure of the data.

The poor performance of the compound and neural PCFGs potentially indicates they were built with insufficient training data. Given the complexity of these models, and the relatively small corpus of only 8,000 trees, this is again unsurprising. Figures 6.14 through 6.19 show the log likelihoods and F1 scores of the compound and neural PCFGs throughout the training epochs. It is evident from these graphs that the models only were able to do a minimal amount of learning. However, there are only approximately 25,000 genes in the human genome, so even similar testing done with all of the data available to us in this problem space may well not yield useful results given these complex models.

Table 6.4 This table shows the log likelihoods and description accuracies of the neural, compound, and supervised PCFGs described in Section 6.3.4. The supervised model was trained by taking the rules and their frequencies directly from the input data. The unsupervised trees attempted to learn the rules and their frequencies from unlabelled parse trees. All three of these models were trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. Unsurprisingly, the supervised PCFG yielded both the highest log likelihoods and the highest description accuracies.

	Neural		Compound		Supervised	
Cross Fold	Avg LL	Avg DA	Avg LL	Avg DA	Avg LL	Avg DA
0	-323.895	-0.039	-292.600	-0.003	-63.596	0.514
1	-332.632	-0.044	-299.549	0.002	-61.600	0.536
2	-272.111	0.006	-254.531	0.043	-51.564	0.432
3	-278.366	-0.001	-303.692	-0.031	-77.538	0.570
4	-326.103	-0.135	-238.103	-0.002	-78.998	0.558
5	-295.253	-0.086	-263.692	-0.054	-28.836	0.510
6	-352.470	-0.152	-246.583	-0.032	-88.261	0.595
7	-364.672	-0.050	-335.194	-0.026	-41.795	0.521
8	-399.989	-0.055	-365.796	-0.024	-68.453	0.426
9	-306.730	-0.051	-257.012	0.014	-66.164	0.564
Averages	-325.222	-0.061	-285.675	-0.011	-62.681	0.523

Table 6.5 shows statistics generated using Evalb. Evalb implements the PARSEVAL protocol, in which predicted and gold-standard trees are represented as sets of spans, and the differences between these sets are used to produce similarity scores (Black et al., 1991; Emms, 2008; Sekine and Collins, 1997). This protocol is useful for evaluating how well tree-structures have been captured by the PCFG. By this metric, the supervised PCFG is the best performing by far, with the unsupervised models exhibiting surprisingly low scores.

That being said, by all metrics, the supervised PCFG performed well, exhibiting the highest description accuracy of all of the models trialed. While it may not currently be possible to usefully train complex unsupervised grammatical modela in this problem space, supervised PCFGs may well offer an interesting and effective way to model genetic and epigenetic relationships.

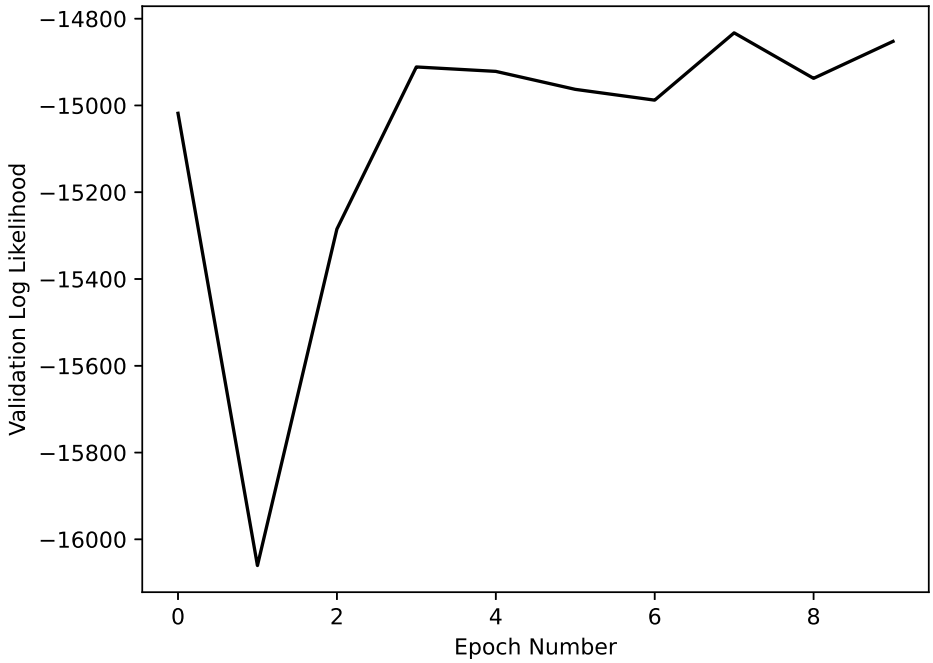


Figure 6.14 The log likelihood of the validation sequence for the Neural PCFG, averaged over 10 cross folds. This Neural PCFG was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. It is clear that this model was able to do only minimal learning, potentially indicating that not enough training data was provided.

Table 6.5 The PARSEVAL metrics of the various PCFGs generated using Evalb. 'Labelled' and 'unlabelled' refer to whether the annotation names were used when comparing the spans and generating the similarity scores. It was not possible to perform 'labelled' tests for the unsupervised PCFGs, as the parsed trees are inherently unlabelled. Each of these three models was trained using the same 8,000 annotation trees from chromosomes one through three of the human genome. However, the supervised model performed the best by far in regards to these metrics.

Model Type	Recall	Precision	F1
Neural PCFG	2.061	5.695	3.027
Compound PCFG	2.485	6.771	3.636
Supervised (Labelled)	32.89	36.68	34.68
Supervised (Unlabelled)	40.04	44.65	42.22

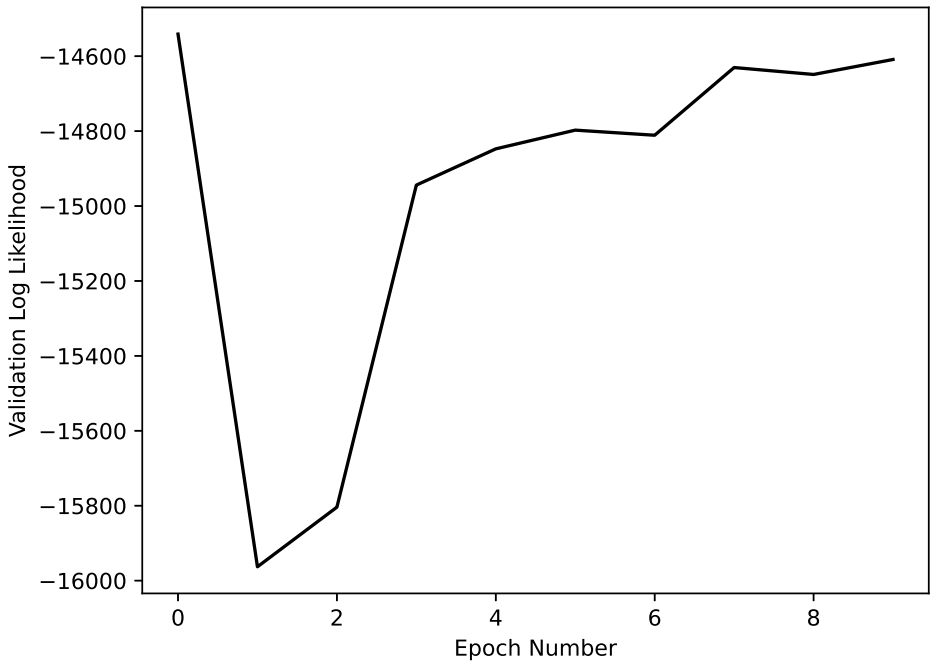


Figure 6.15 The log likelihood of the validation sequence for the Compound PCFG, averaged over 10 cross folds. This Compound PCFG was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the Neural PCFG, it is clear that this model was able to do only minimal learning, potentially indicating that not enough training data was provided.

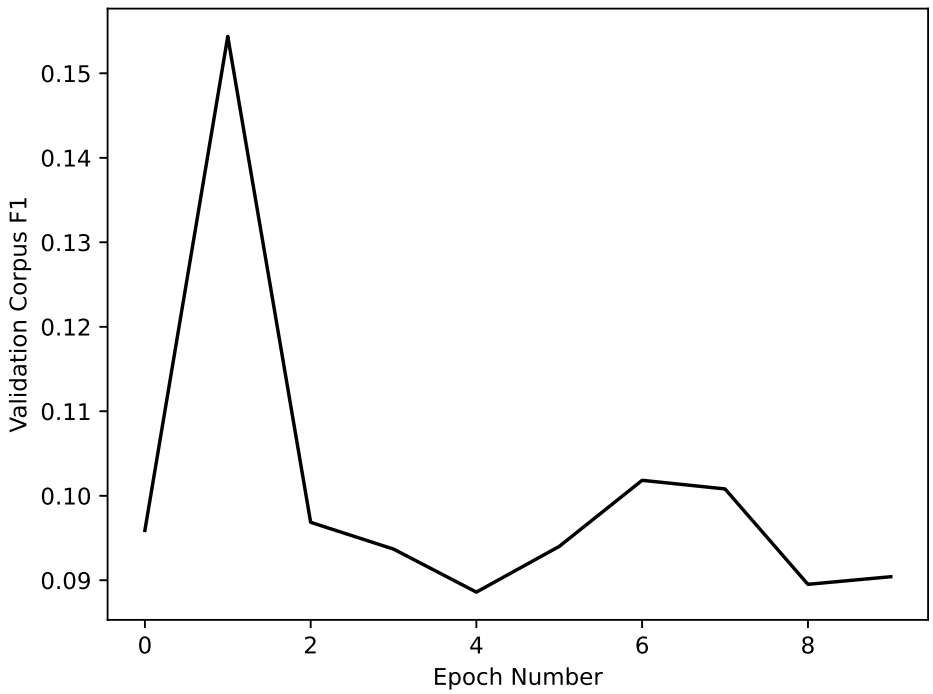


Figure 6.16 The corpus F1 score for the validation sequence with the Neural PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided.

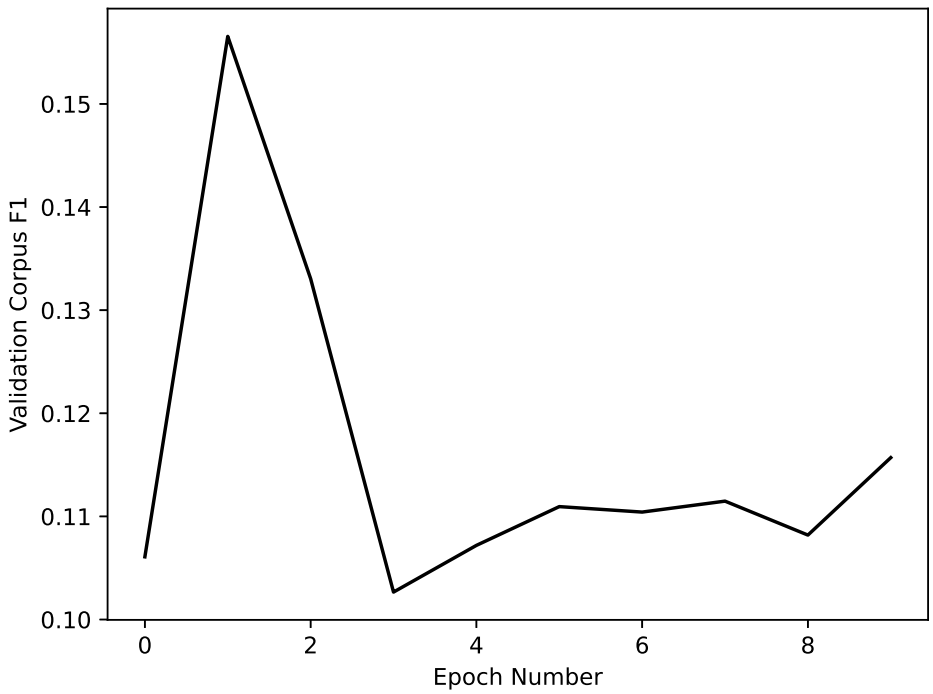


Figure 6.17 The corpus F1 score for the validation sequence with the Compound PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided.

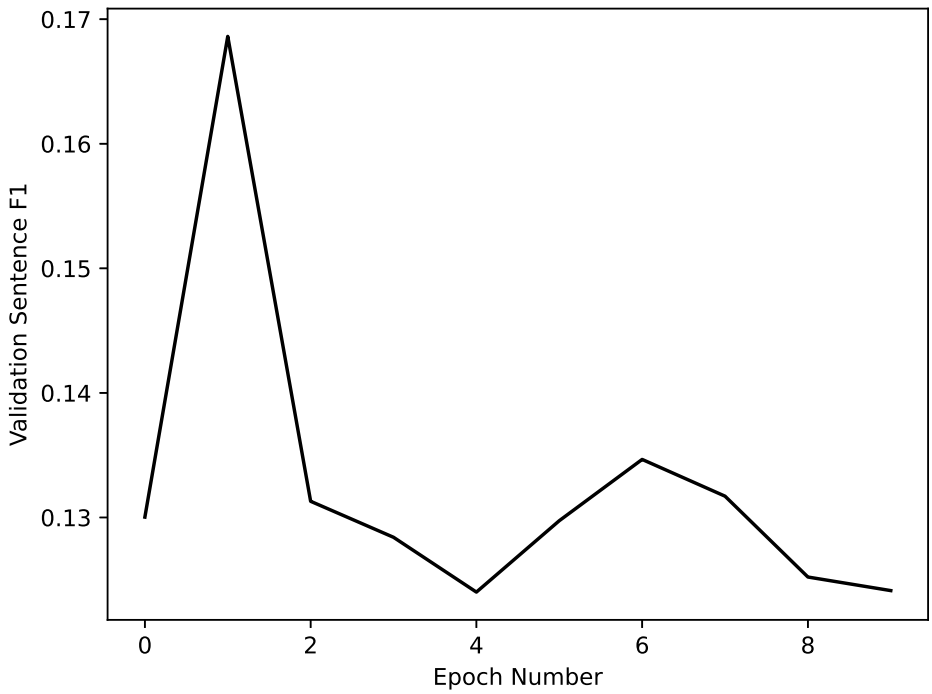


Figure 6.18 The sentence F1 score for the validation sequence with the Neural PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided.

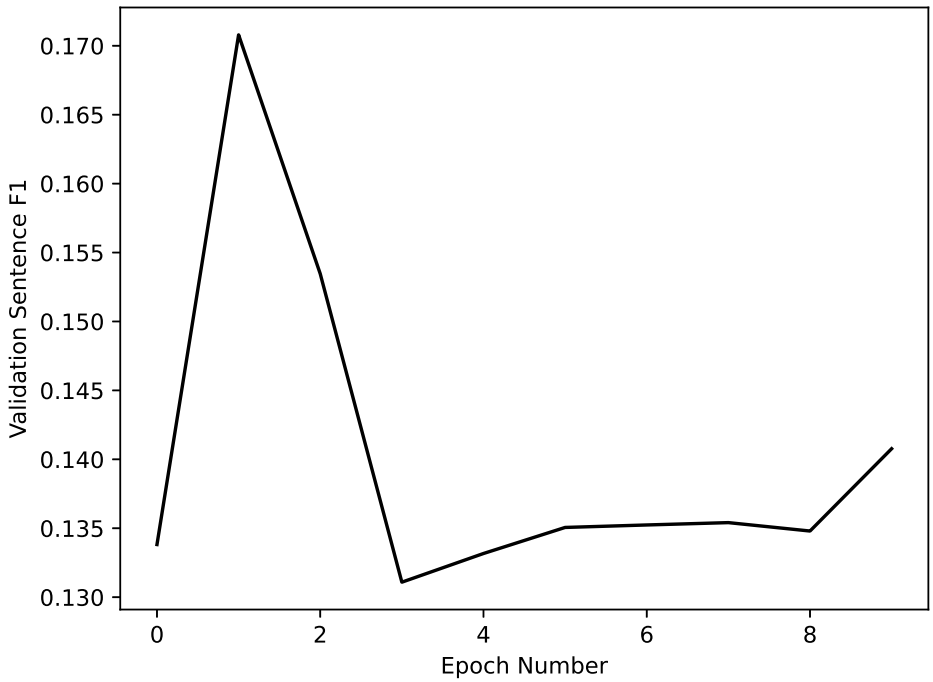


Figure 6.19 The sentence F1 score for the validation sequence with the Compound PCFG, averaged over 10 cross folds. This model was trained using approximately 8,000 annotation trees generated from chromosomes one through three of the human genome. As was the case with the log likelihood, the F1 score did not show substantial improvement over the training epochs, potentially indicating that not enough training data was provided.

6.4 Discussion

The work in this chapter has served as an exploratory proof of concept for attempting to represent genomic and epigenetic features using increasingly complex models, in order to characterise how these features might best be represented in Synthetic Biology design software. However, each of these models may well be useful for modelling histone modification sequences in different contexts.

While the n-gram models yielded the highest accuracies, the HMM models allow for chromatin state to be modelled through the hidden states, in a way which is intuitive and could potentially be useful in a more complex system incorporating representations of a variety of epigenetic factors.

The supervised PCFG allowed us to capture complex relationships between genomic features, and to harness these relationships to model histone modifications. This model also yielded the highest description accuracy, demonstrating how well we were able to represent the data with these models.

For this particular problem, the unsupervised PCFG models could not be trained with sufficient data. However, these models could potentially be useful for projects involving a large number of related genomes. These unsupervised models could, in these cases, potentially be used to discover yet unknown relationships between genetic and epigenetic features.

If, in the future, new DNA design tools are to be built which can help researchers predict where to insert novel sequences such that their epigenetic context can be most efficiently harnessed, many more such explorations must be undertaken. We have shown that several different language models can usefully be applied in this problem space to predict histone modification occupancy. However, the more contextual information that can be captured, the better. A future implementation of parts-based design for Synthetic Biology may well be influenced by language modelling, and the many different ways in which these techniques can capture complex patterns and long-range dependancies.

Chapter 7

Conclusion

Drew Endy's 2005 paper Foundations for engineering biology (Endy, 2005) begins by giving the reader an assignment:

> Please complete one of the following projects in the next hour: write down the DNA sequence that programmes a biofilm to take a photograph and perform distributed edge-detection on the light-encoded image; or, the DNA sequence that encodes a ring oscillator that works inside yeast; or, the DNA sequence that programmes any mammalian cell to count up to 256 in response to a generic input signal; or, the DNA sequence that programmes any prokaryote to produce 25gl-1 artemisinic acid.

Fifteen years later, some of these tasks are now possible for an expert with some time on their hands, while others remain well out of reach. However, the tasks proposed do not provide an accurate representation of what Synthetic Biology has become since 2005. While some experiments do still involve the production of genetic circuits for experiments heavily inspired by engineering problems, others involve searching for the simplest possible genome, designing lentiviral gene therapies, or creating robust, large-scale industrial production pipelines using microbial factories.

While the best way to achieve the production of a genetic circuit may well be to embrace abstraction, decoupling, and standardisation, the ways in which these engineering principles have been applied to molecular biology are not as helpful for more abstract problems, including those which aim to learn about the nature of biological systems.

A modern update to Drew Endy's paper might start by asking the reader to optimise the yeast genome, to use CRISPR-Cas9 to build a gene therapy which reliably and safely treats a monogenic disease, or to design an industrial process through which microbes can be used to produce a natural product as efficiently as possible at scale. For the originally

proposed projects, standardisation and decoupling are intuitively useful. For these more modern projects, it is unclear how such engineering princples could be applied, and if they were, how useful they would be.

Much work is yet to be done in the field in order to find the ideal design paradigms for the different modes of Synthetic Biology. Further work will then be required to build the computational tools which will take advantage of these paradigms. The projects presented in this thesis were all completed with the aim of improving these design paradigms, even in a small way.

The first of these projects offered a review of popular Synthetic Biology DNA design tools, an analysis of their underlying design paradigms, and a discussion of several potential alternatives.

The second project was the construction of a software tool, Part Crafter, which enables rational search over genomic features, such that users can find genomic parts for their DNA designs. Part Crafter offers a more flexible implementation of parts for Synthetic Biology. Rather than creating ultra-specific parts which can only serve the narrowest of functions, Part Crafter allows sequences to be adapted for any host and any manufacturing standard. Additionally, as much contextual information as possible is presented alongside genomic sequences, allowing for more deliberate and controlled design. It is impossible to design complex projects base pair by base pair, but abstraction in the face of high levels of uncertainty can also prevent the design of predictable experiments. Part Crafter uses as much data as possible to find a medium between these two outcomes. Until we learn more about the nature of genetics and sequence characterisation becomes cheaper and easier, this implementation of genetic parts may offer a useful paradigm for a variety of Synthetic Biology experiments.

The final project of this thesis presented a proof of concept of machine learning models for representing histone modification occupancy. This work harnessed an increasingly complex set of language models for representing sequences of these epigenetic proteins, in order to learn more about the complexity of the patterns which dictate their genomic locations. This work is an example of many future projects which will need to be undertaken to understand exactly what information we need to design reliable DNA sequences, and how that information should be represented by our conceptual paradigms and computational tools.

As the field of Synthetic Biology progresses, the tools researchers use to design DNA sequences must progress as well. Much work needs to be done for computational tools to sufficiently represent the complex nature of molecular genetics. This thesis has presented three projects which represent three small steps forward in building the conceptual

and computational tools required for the increasingly ambitious goals of the field to be accomplished.

The founders of the field hoped that one day even a Biology novice could use powerful tools and design paradigms to build complex DNA circuits. While that dream is not yet a reality, a decade and a half of incremental progress has led to a flourishing scientific field in which researchers harness cells to do revolutionary research. Increasing the efficacy and efficiency of the tools and paradigms that researchers use to achieve these scientific discoveries can only lead to more scientists being even more innovative in the future.

Appendix A

PartCrafter Workshop Worksheet

A.1 PartCrafter Key Features

1. Genomic features are extracted from a wide range of genomes.
2. Data is aggregated about a large number and variety of genomic features.
3. Users can search for genomic features by their function.
4. Users can search for genomic features based on their similarity to a feature of interest.
5. Users can turn any genomic feature into a BioPart. This functionality includes the ability to:
   (a) Generate a promoter and terminator for a sequence.
   (b) Codon optimise a sequence for a new host and/or to remove forbidden sites.
   (c) Add the appropriate overhangs to a sequence based on any manufacturing standard.
   (d) Generate primers for a sequence.

PartCrafter can be accessed at www.partcrafter.com

A.2 Exercises

1. If while you're playing around with PartCrafter you find something that looks like a bug, please document it here.

2. Testing the search function:

   (a) Choose any Escherichia coli or Saccharomyces cerevisiae gene. Do a quick internet search to find out a little bit about its function. Please document the name of the gene you've chosen:

   (b) Use PartCrafter to search for this part using a description based on what you found. Please document that description here:

   (c) Does the gene you wanted appear in the top ten search results? If so, at which position?

   (d) For each of the top 10 search results, please indicate how relevant each item is to your search query:

      i. Name of feature:
      Not relevant Somewhat relevant Very relevant

      ii. Name of feature:
      Not relevant Somewhat relevant Very relevant

      iii. Name of feature:
      Not relevant Somewhat relevant Very relevant

      iv. Name of feature:
      Not relevant Somewhat relevant Very relevant

      v. Name of feature:
      Not relevant Somewhat relevant Very relevant

      vi. Name of feature:
      Not relevant Somewhat relevant Very relevant

      vii. Name of feature:
      Not relevant Somewhat relevant Very relevant

      viii. Name of feature:
      Not relevant Somewhat relevant Very relevant

      ix. Name of feature:
      Not relevant Somewhat relevant Very relevant

      x. Name of feature:
      Not relevant Somewhat relevant Very relevant

3. Part Generation:

   (a) Choose one of the features from your search results, and press "Make into a Part"

(b) Use the "Generate a Part" form to generate a part, using parameters of your choice

(c) Are there any features not currently included in this section of PartCrafter that you think would make part generation easier/more useful?

4. Overall Impressions:

(a) How would you improve PartCrafter?

(b) Can you see PartCrafter being useful for your research? Why/Why not?

(c) Is PartCrafter easy to use? If not, how could it be made more intuitive?

Appendix B

Additional N-Gram Model Statistics

Table B.1 This table shows additional performance statistics for the Bigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors.

Histone Modification	Accuracy	F1 Score	Precision	Recall	AUC (ROC)	AUC (Precision-Recall)
H2AK5ac	0.986	0.722	0.819	0.672	0.672	0.507
H2AS129ph	0.990	0.923	0.926	0.919	0.919	0.780
H3K14ac	0.975	0.887	0.926	0.856	0.856	0.876
H3K18ac	0.945	0.859	0.897	0.829	0.829	0.840
H3K23ac	0.983	0.880	0.887	0.873	0.873	0.861
H3K27ac	0.998	0.554	0.699	0.531	0.531	0.209
H3K36me	0.998	0.771	0.863	0.716	0.716	0.422
H3K36me2	0.993	0.663	0.785	0.615	0.615	0.313
H3K36me3	0.953	0.799	0.879	0.750	0.750	0.717
H3K4ac	0.983	0.914	0.924	0.904	0.904	0.925
H3K4me	0.984	0.741	0.849	0.686	0.686	0.509
H3K4me2	0.957	0.613	0.816	0.574	0.574	0.347
H3K4me3	0.889	0.827	0.837	0.818	0.818	0.781
H3K56ac	0.990	0.804	0.857	0.765	0.765	0.738
H3K79me	0.989	0.786	0.860	0.737	0.737	0.558
H3K79me3	0.976	0.796	0.855	0.754	0.754	0.687
H3K9ac	0.982	0.893	0.899	0.887	0.887	0.873

H3S10ph	0.992	0.908	0.908	0.909	0.909	0.694
H4K12ac	0.995	0.814	0.823	0.805	0.805	0.675
H4K16ac	0.998	0.794	0.896	0.735	0.735	0.615
H4K20me	1.000	0.700	0.833	0.643	0.643	0.452
H4K5ac	0.991	0.818	0.890	0.769	0.769	0.700
H4K8ac	0.983	0.790	0.907	0.727	0.727	0.643
H4R3me	0.999	0.676	0.800	0.625	0.625	0.519
H4R3me2s	0.996	0.832	0.910	0.780	0.780	0.750
Htz1	0.982	0.499	0.591	0.502	0.502	0.093

Table B.2 This table shows additional performance statistics for the Trigram models described in Section 6.3.2. Each histone modification was predicted individually, using n-gram models trained with a dataset containing all histone modifications present in a given nucleosome, plus the histone modifications present in the n previous nucleosomes. The dataset also contained base counts for the given nucleosome, as well as base counts for the n previous vectors.

Histone Modification	Accuracy	F1 Score	Precision	Recall	AUC (ROC)	AUC (Precision-Recall)
H2AK5ac	0.987	0.739	0.829	0.688	0.688	0.507
H2AS129ph	0.988	0.907	0.934	0.883	0.883	0.835
H3K14ac	0.976	0.890	0.929	0.858	0.858	0.878
H3K18ac	0.946	0.859	0.901	0.828	0.828	0.845
H3K23ac	0.983	0.882	0.891	0.873	0.873	0.862
H3K27ac	0.998	0.600	0.749	0.562	0.562	0.215
H3K36me	0.998	0.763	0.874	0.703	0.703	0.459
H3K36me2	0.993	0.655	0.788	0.606	0.606	0.308
H3K36me3	0.956	0.825	0.867	0.794	0.794	0.738
H3K4ac	0.983	0.915	0.927	0.904	0.904	0.925
H3K4me	0.984	0.740	0.848	0.684	0.684	0.537
H3K4me2	0.955	0.615	0.768	0.578	0.578	0.369
H3K4me3	0.893	0.829	0.850	0.813	0.813	0.797
H3K56ac	0.989	0.797	0.839	0.765	0.765	0.742
H3K79me	0.989	0.792	0.876	0.739	0.739	0.583
H3K79me3	0.977	0.806	0.871	0.762	0.762	0.704
H3K9ac	0.983	0.895	0.904	0.886	0.886	0.875
H3S10ph	0.992	0.901	0.935	0.872	0.872	0.786
H4K12ac	0.995	0.814	0.827	0.801	0.801	0.667

H4K16ac	0.998	0.785	0.892	0.724	0.724	0.601
H4K20me	1.000	0.682	0.750	0.643	0.643	0.351
H4K5ac	0.991	0.823	0.883	0.780	0.780	0.700
H4K8ac	0.984	0.801	0.902	0.742	0.742	0.659
H4R3me	0.999	0.722	0.833	0.667	0.667	0.429
H4R3me2s	0.996	0.827	0.913	0.771	0.771	0.750
Htz1	0.982	0.515	0.764	0.510	0.510	0.101

Appendix C

Results for Additional HMM Tests

Section 6.3.3 describes the construction of standard and custom Hidden Markov Models over sequences of histone modifications, where custom HMMs are reparameterized by DNA sequence information. The custom HMMs present are reparameterized by associated G and C levels. Additional tests were run in which the custom HMMs were reparameterized by A, T, C, and G counts.

The learned weight matrices for these custom HMMs take the form of a $5 \times S \times S$ matrix, in which the last $S \times S$ matrix represents the bias terms, which were initialised to the learned standard transition matrix values. The first four $S \times S$ matrices are dependant on the vector gv, which represents the absolute values of the differences between the A, T, C, and G counts in the current nucleosome as compared with the mean nucleotide counts.

The results for the models parameterised by GC content are very similar to those parameterised by ATCG counts, but in general the ATCG count model statistics are slightly worse. This may indicate some level of overfitting.

The tables below present additional statistics from the HMMs re-parameterised by GC content, as well as statistics from HMMs re-parameterised by ATCG counts.

C.1 HMMs Re-Parameterised by GC content

The following tables show a selection of statistics from the HMMs re-parameterised by GC content.

Table C.1 H2AS129ph, 137 Occurrences

This table shows the changes in accuracy and F1 in predicting H2AS129ph occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered an 8% improvement in accuracy, along with a 4% lower F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.624	0.715	0.090	0.070	-0.001	-0.071
4	0.681	0.759	0.078	0.069	0.062	-0.007
3	0.778	0.862	0.084	0.024	-0.015	-0.039
2	0.771	0.848	0.077	0.025	-0.022	-0.047

Table C.2 H3S10ph, 74 Occurrences

This table shows the changes in accuracy and F1 in predicting H3S10ph occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a 4% improvement in accuracy, along with a very minimal decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.801	0.797	-0.004	0.001	-0.002	-0.003
4	0.753	0.809	0.056	0.009	0.010	0.001
3	0.784	0.823	0.040	0.017	0.017	0.000
2	0.654	0.738	0.084	0.019	0.018	-0.001

Table C.3 H3K4ac, 80 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K4ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a modest improvement in accuracy, along with a 3% decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.926	0.931	0.005	0.239	0.190	-0.049
4	0.927	0.925	-0.003	0.207	0.179	-0.029
3	0.891	0.910	0.020	0.121	0.080	-0.041
2	0.921	0.932	0.011	0.123	0.107	-0.017

Table C.4 H3K14ac, 75 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K14ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered no improvement in accuracy, along with a very minimal decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.924	0.923	-0.001	0.071	0.045	-0.026
4	0.916	0.915	-0.001	0.013	0.013	0.000
3	0.918	0.918	0.000	0.140	0.138	-0.002
2	0.917	0.920	0.002	0.017	0.008	-0.008

Table C.5 H3K36me3, 91 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K36me3 occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered no improvement in accuracy, along with a very minimal increase in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.768	0.753	-0.015	0.094	0.104	0.010
4	0.730	0.751	0.021	0.049	0.057	0.009
3	0.818	0.810	-0.008	0.061	0.060	-0.001
2	0.776	0.764	-0.012	0.131	0.137	0.006

Table C.6 H3K18ac, 138 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K18ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered no improvement in accuracy, along with a 2% decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.835	0.819	-0.016	0.218	0.155	-0.063
4	0.796	0.790	-0.006	0.150	0.150	0.000
3	0.672	0.677	0.006	0.175	0.186	0.011
2	0.826	0.827	0.000	0.171	0.142	-0.028

Table C.7 H3K4me3, 170 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K4me3 occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a 2% decrease in accuracy, along with a 2% increase in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.677	0.678	0.001	0.315	0.365	0.050
4	0.671	0.639	-0.032	0.239	0.268	0.029
3	0.583	0.539	-0.044	0.222	0.251	0.029
2	0.763	0.739	-0.024	0.417	0.398	-0.020

C.2 HMMs Re-Parameterised by ATCG Counts

The following tables show a selection of statistics from the HMMs re-parameterised by ATCG counts.

Table C.8

This table shows the changes in test sequence log likelihood for each state number, and for the standard and custom HMMs. The final three columns show the computed description accuracies for each model. The custom HMMs log likelihoods offer a modest improvement over those of the standard HMMs. Similarly, the DA slightly improves for the custom HMMs.

# of States	Standard Test Seq LL	Custom Test Seq LL	Delta	Standard DA	Custom DA	Delta
5	-88.714	-86.374	2.341	0.473	0.472	-0.001
4	-124.587	-122.064	2.523	0.440	0.446	0.006
3	-386.132	-382.842	3.289	0.356	0.362	0.006
2	-241.635	-234.820	6.815	0.352	0.348	-0.004

Table C.9 H2AS129ph, 137 Occurrences

This table shows the changes in accuracy and F1 in predicting H2AS129ph occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered an 8% improvement in accuracy, along with a 2% lower F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.624	0.713	0.089	0.071	0.071	0.000
4	0.681	0.762	0.081	0.069	0.062	-0.007
3	0.778	0.856	0.078	0.024	-0.016	-0.039
2	0.771	0.851	0.080	0.030	-0.017	-0.047

Table C.10 H3S10ph, 74 Occurrences

This table shows the changes in accuracy and F1 in predicting H3S10ph occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a 4% improvement in accuracy, along with a very minimal decrease in F1 score.

States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.801	0.782	-0.019	0.001	-0.002	-0.003
4	0.753	0.812	0.059	0.009	0.010	0.001
3	0.784	0.824	0.040	0.017	0.017	-0.001
2	0.654	0.731	0.077	0.019	0.017	-0.002

Table C.11 H3K4ac, 80 Ocurrences

This table shows the changes in accuracy and F1 in predicting H3K4ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a modest improvement in accuracy, along with a 3% decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.9256	0.9269	0.0013	0.2389	0.1550	-0.0839
4	0.9272	0.9261	-0.0011	0.2075	0.1619	-0.0456
3	0.8908	0.9103	0.0195	0.1210	0.1224	0.0014
2	0.9214	0.9316	0.0102	0.1233	0.1174	-0.0059

Table C.12 H3K14ac, 75 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K14ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a small decrease in accuracy, along with a 1% decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.924	0.924	0.000	0.071	0.049	-0.022
4	0.916	0.912	-0.003	0.013	0.025	0.012
3	0.918	0.925	0.007	0.140	0.208	0.068
2	0.917	0.922	0.005	0.017	0.030	0.013

Table C.13 H3K18ac, 138 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K18ac occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a small decrease in accuracy, along with a 1% decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.8351	0.8258	-0.0093	0.2180	0.1645	-0.0535
4	0.7962	0.7881	-0.0081	0.1504	0.1520	0.0016
3	0.6718	0.6740	0.0022	0.1751	0.1667	-0.0084
2	0.8264	0.8270	0.0006	0.1705	0.1751	0.0046

Table C.14 H3K36me3, 91 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K36me3 occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a small decrease in accuracy, along with a minimal decrease in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.768	0.753	-0.015	0.094	0.104	0.010
4	0.730	0.748	0.018	0.049	0.069	0.020
3	0.818	0.817	-0.001	0.061	0.058	-0.003
2	0.776	0.757	-0.018	0.131	0.130	-0.001

Table C.15 H3K4me3, 170 Occurrences

This table shows the changes in accuracy and F1 in predicting H3K4me3 occupancy for each state number, and for the standard and custom HMMs. On average, the custom HMM offered a 2% decrease in accuracy, along with a 2% increase in F1 score.

# of States	Standard Accuracy	Custom Accuracy	Delta	Standard F1	Custom F1	Delta
5	0.677	0.678	0.001	0.315	0.372	0.057
4	0.671	0.642	-0.029	0.239	0.268	0.029
3	0.583	0.538	-0.045	0.222	0.248	0.026
2	0.763	0.745	-0.018	0.417	0.429	0.011

Appendix D

Additional HMM Matrix Heatmaps

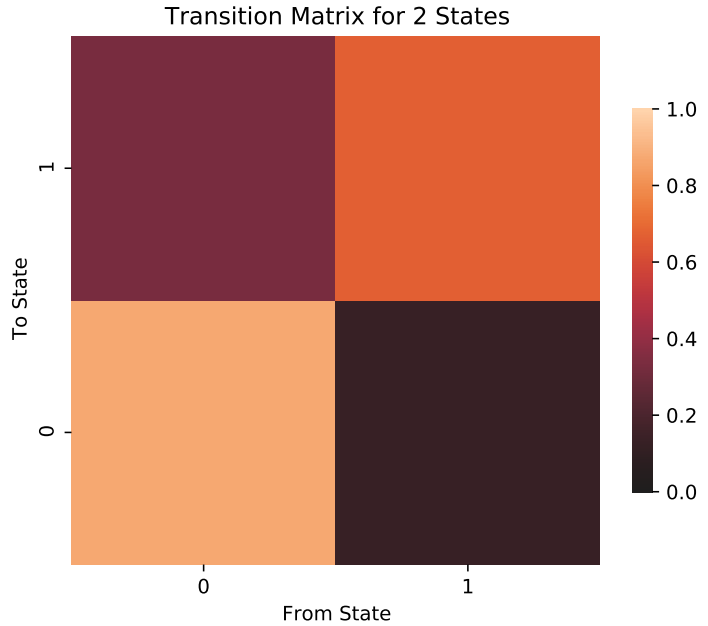


Figure D.1 The transition matrix for an HMM with 2 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices.

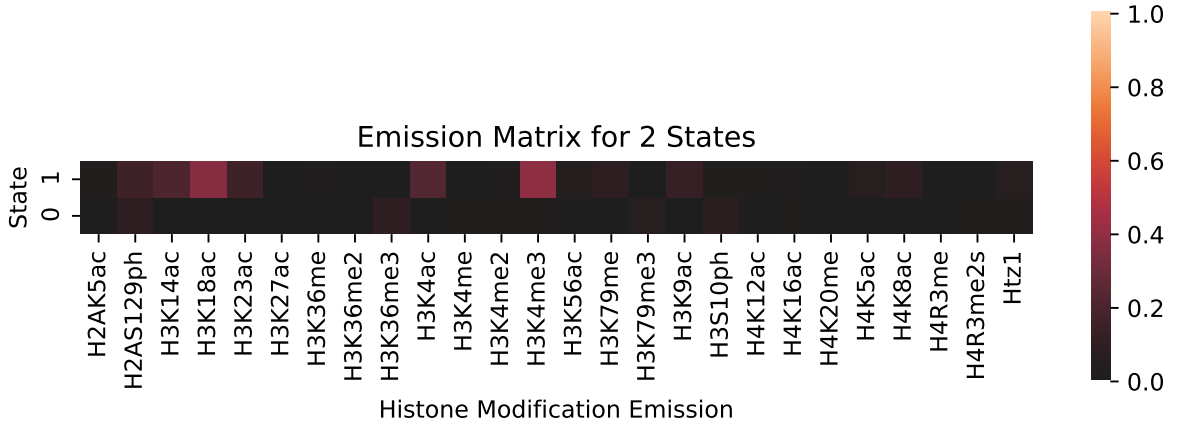


Figure D.2 The emission matrix for an HMM with 2 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications, though those associations appear to be much stronger for State 1. These associations are discussed further in Table 6.3.

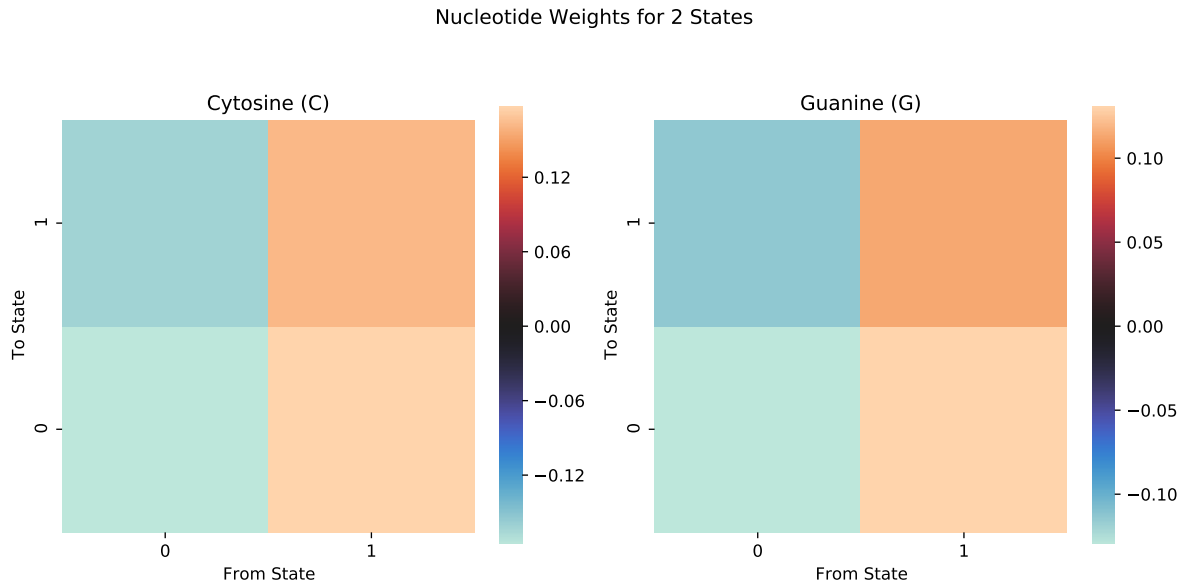


Figure D.3 The genomic weights learned for a custom HMM with 2 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring.

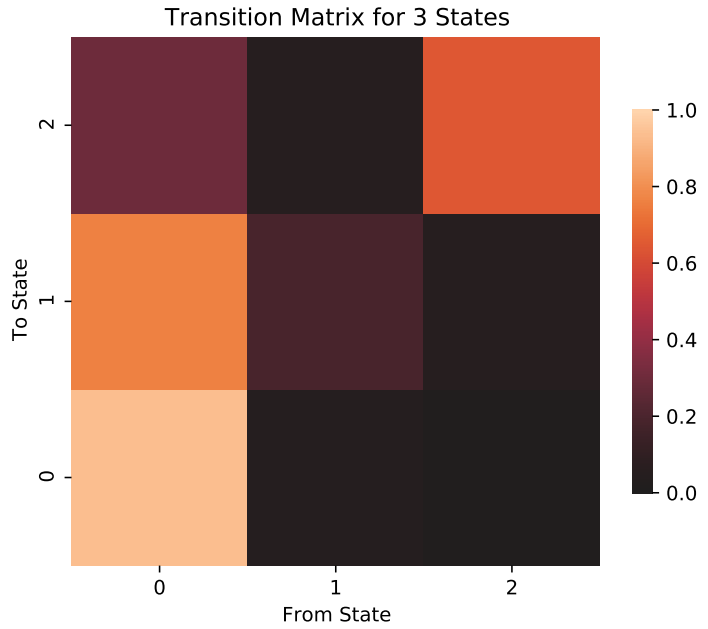


Figure D.4 The transition matrix for an HMM with 3 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices.

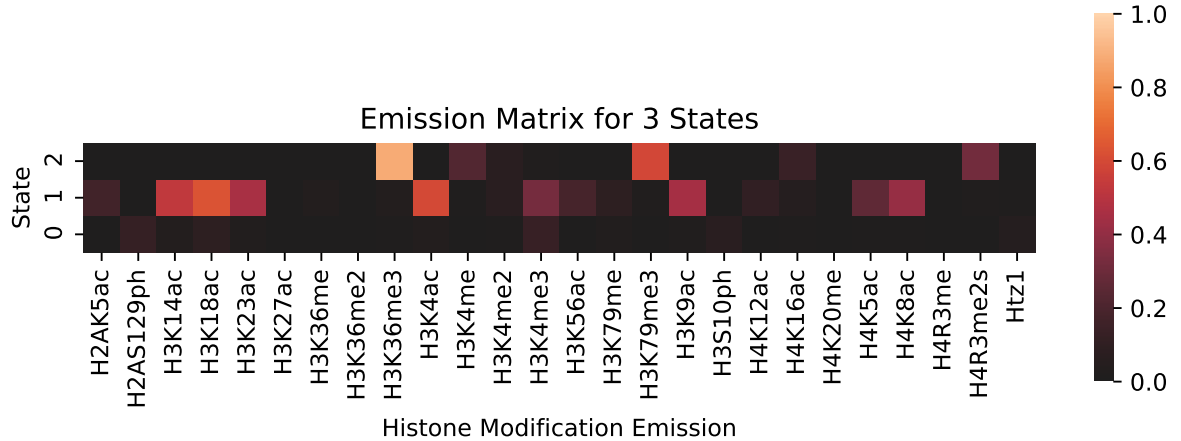


Figure D.5 The emission matrix for an HMM with 3 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3.

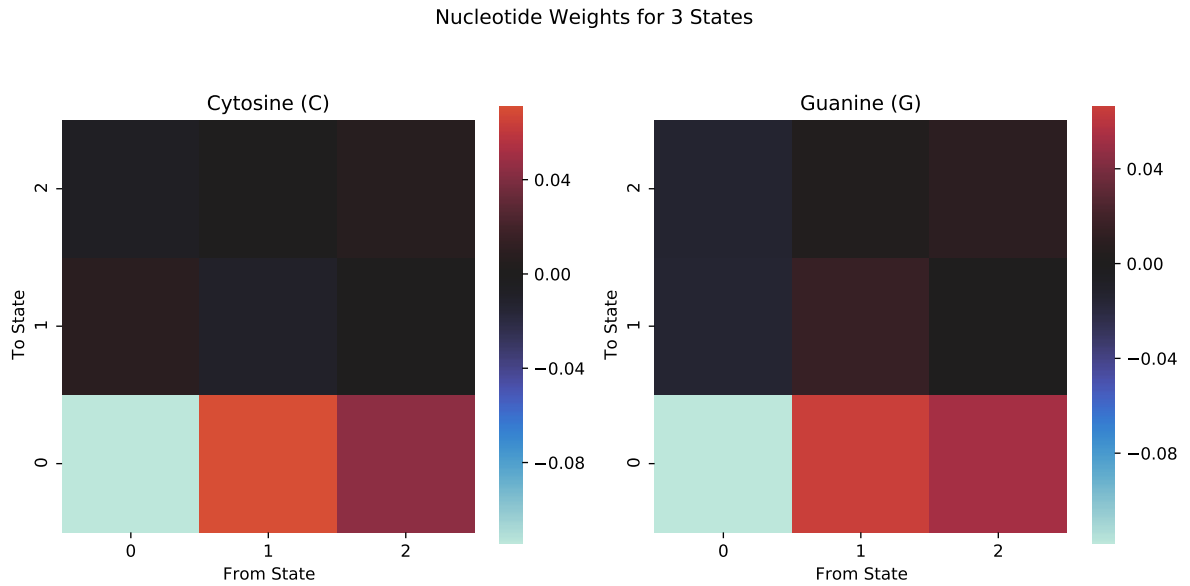


Figure D.6 The genomic weights learned for a custom HMM with 3 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring.

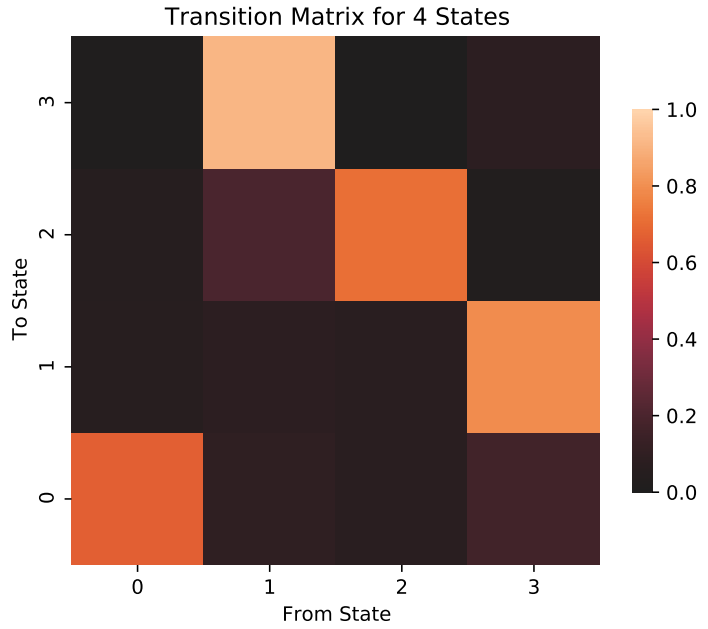


Figure D.7 The transition matrix for an HMM with 4 states. This matrix shows the likelihood of transitioning from one of the HMM's states to another. In the custom HMMs, this transition matrix is re-parameterised by base counts, using the genomic weight matrices.

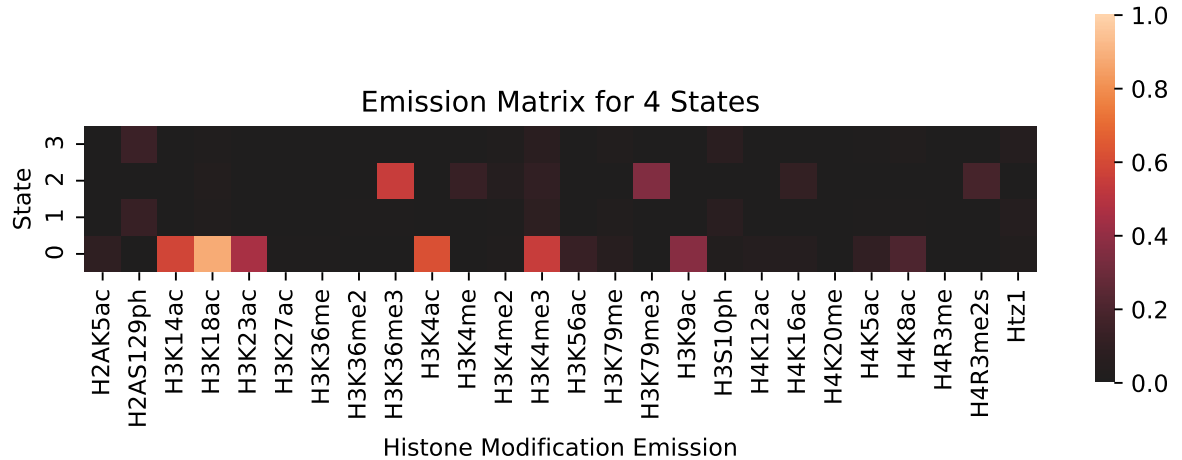


Figure D.8 The emission matrix for an HMM with 4 states, which dictates how likely it is that a particular histone modification will be emitted as part of a histone modification vector, while in a particular state. When using this matrix to make predictions, a threshold of 0.5 was used. Each state appears to be associated with a certain subset of histone modifications. These associations are discussed further in Table 6.3.

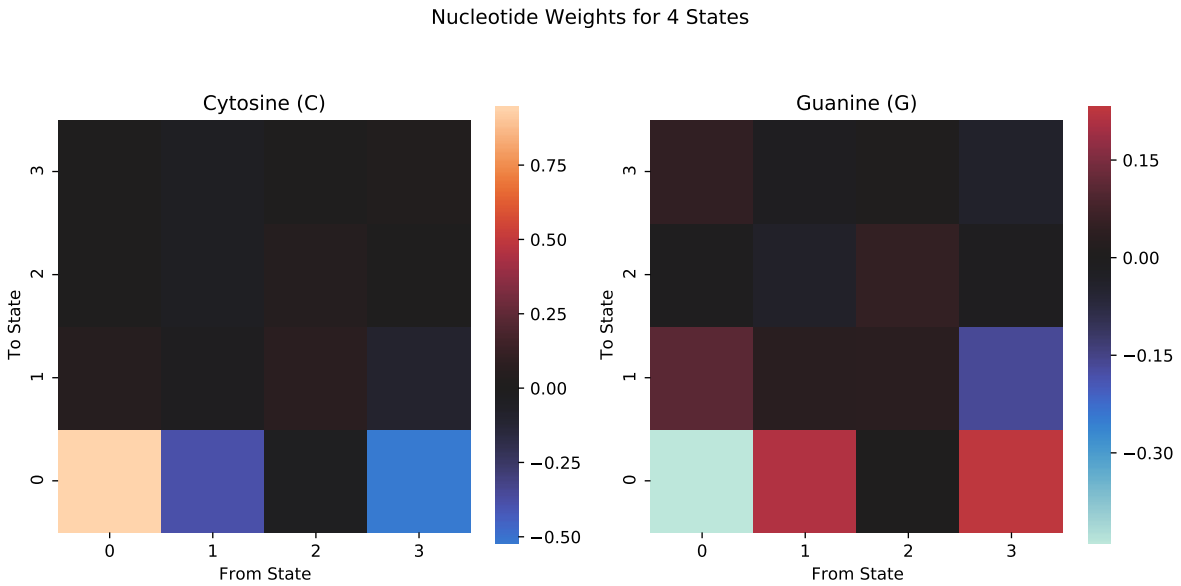


Figure D.9 The genomic weights learned for a custom HMM with 4 states. These weights were trained to re-parameterise the transition matrix with nucleotide counts. A positive value indicates that a larger difference in the amount of that nucleotide for a particular nucleosome increases the likelihood of a particular state transition occurring. A negative value indicates that a larger difference in the amount of that nucleotide in a particular nucleosome decreases the likelihood of a particular state transition occurring.

Bibliography

Maureen A. O'Malley, Alexander Powell, Jonathan F. Davies, and Jane Calvert. Knowledge-making distinctions in synthetic biology. *BioEssays*, 30(1):57–65, January 2008. ISSN 02659247, 15211878. doi: 10.1002/bies.20664. URL http://doi.wiley.com/10.1002/bies.20664.

Bryn L. Adams. The Next Generation of Synthetic Biology Chassis: Moving Synthetic Biology from the Laboratory to the Field. *ACS Synthetic Biology*, 5(12):1328–1330, December 2016. ISSN 2161-5063, 2161-5063. doi: 10.1021/acssynbio.6b00256. URL https://pubs.acs.org/doi/10.1021/acssynbio.6b00256.

Julius Adler, IR Lehman, Maurice J Bessman, ES Simms, and Arthur Kornberg. Enzymatic synthesis of deoxyribonucleic acid. IV. Linkage of single deoxynucleotides to the deoxynucleoside ends of deoxyribonucleic acid. *Proceedings of the National Academy of Sciences of the United States of America*, 44(7):641, 1958.

Bruce Alberts, editor. *Molecular Biology of the Cell*. Garland Science, New York, 5th ed edition, 2008. ISBN 978-0-8153-4105-5 978-0-8153-4106-2. OCLC: ocm82473851.

Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3): 403–410, October 1990. ISSN 00222836. doi: 10.1016/S0022-2836(05)80360-2. URL https://linkinghub.elsevier.com/retrieve/pii/S0022283605803602.

Narayana Annaluru, Sivaprakash Ramalingam, and Srinivasan Chandrasegaran. Rewriting the blueprint of life by synthetic genomics and genome engineering. *Genome Biology*, 16(1):125, December 2015. ISSN 1474-760X. doi: 10.1186/s13059-015-0689-y. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-015-0689-y.

Oswald T Avery, Colin M MacLeod, and Maclyn McCarty. Studies on the chemical nature of the substance inducing transformation of pneumococcal types. Induction of

transformation by a desoxyribonucleic acid fraction isolated from Pneumococcus type III. *The Journal of experimental medicine*, 79(2):137–158, 1944.

Taiwo Oladipupo Ayodele. Types of Machine Learning Algorithms. *New advances in machine learning*, pages 19–48, 2010.

Rama Balakrishnan, Julie Park, Kalpana Karra, Benjamin C Hitz, Gail Binkley, Eurie L Hong, Julie Sullivan, Gos Micklem, and J Michael Cherry. YeastMine—an integrated data warehouse for Saccharomyces cerevisiae data as a multipurpose tool-kit. *Database*, 2012, 2012.

Geoff Baldwin, Travis Bayer, Robert Dickinson, Tom Ellis, Paul S Freemont, Richard I Kitney, Karen Polizzi, and Guy-Bart Stan. *Basic Concepts in Engineering Biology*, chapter 2, pages 19–40. Imperial College Press, 2015.

Rodolphe Barrangou. Cas9 targeting and the CRISPR revolution. *Science*, 344(6185): 707–708, 2014.

Marcelo C Bassalo, Rongming Liu, and Ryan T Gill. Directed evolution and synthetic biology applications to microbial systems. *Current Opinion in Biotechnology*, 39: 126–133, June 2016. ISSN 09581669. doi: 10.1016/j.copbio.2016.03.016. URL https://linkinghub.elsevier.com/retrieve/pii/S0958166916300726.

Maxwell Bates, Joe Lachoff, Duncan Meech, Valentin Zulkower, Anaïs Moisy, Yisha Luo, Hille Tekotte, Cornelia Johanna Franziska Scheitz, Rupal Khilari, Florencio Mazzoldi, et al. Genetic Constructor: An Online DNA Design Platform. *ACS synthetic biology*, 6(12):2362–2365, 2017.

Leonard E Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3(1):1–8, 1972.

Jan Bednar, Rachel A Horowitz, Sergei A Grigoryev, Lenny M Carruthers, Jeffrey C Hansen, Abraham J Koster, and Christopher L Woodcock. Nucleosomes, linker DNA, and linker histone form a unique structural motif that directs the higher-order folding and compaction of chromatin. *Proceedings of the National Academy of Sciences*, 95 (24):14173–14178, 1998.

Sam Behjati and Patrick S Tarpey. What is next generation sequencing? *Archives of Disease in Childhood-Education and Practice*, 98(6):236–238, 2013.

Dan Benveniste, Hans-Joachim Sonntag, Guido Sanguinetti, and Duncan Sproul. Transcription factor binding predicts histone modifications in human cell lines. *Proceedings of the National Academy of Sciences*, 111(37):13367–13372, 2014.

Ezra Black, Steven Abney, Dan Flickinger, Claudia Gdaniec, Ralph Grishman, Phil Harrison, Donald Hindle, Robert Ingria, Frederick Jelinek, Judith L Klavans, et al. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Speech and Natural Language: Proceedings of a Workshop Held at Pacific Grove, California, February 19-22, 1991*, 1991.

Albert F. Blakeslee, A. Dorothy Bergner, and Amos G. Avery. Methods of Synthesizing Pure-breeding Types with Predicted Characters in the Jimson Weed, Datura stramonium. *Proceedings of the National Academy of Sciences of the United States of America*, 19(1):115–122, 1933. ISSN 00278424. URL http://www.jstor.org/stable/85781.

John Blazeck and Hal S Alper. Promoter engineering: recent advances in controlling transcription at the most fundamental level. *Biotechnology journal*, 8(1):46–58, 2013.

J. D. Boeke, G. Church, A. Hessel, N. J. Kelley, A. Arkin, Y. Cai, R. Carlson, A. Chakravarti, V. W. Cornish, L. Holt, F. J. Isaacs, T. Kuiken, M. Lajoie, T. Lessor, J. Lunshof, M. T. Maurano, L. A. Mitchell, J. Rine, S. Rosser, N. E. Sanjana, P. A. Silver, D. Valle, H. Wang, J. C. Way, and L. Yang. The Genome Project-Write. *Science*, 353(6295):126–127, July 2016. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aaf6850. URL http://www.sciencemag.org/cgi/doi/10.1126/science.aaf6850.

Maarten Boudry and Massimo Pigliucci. The mismeasure of machine: Synthetic biology and the trouble with engineering metaphors. *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences*, 44(4):660–668, 2013.

Yizhi Cai, Matthew W. Lux, Laura Adam, and Jean Peccoud. Modeling Structure-Function Relationships in Synthetic DNA Sequences using Attribute Grammars. *PLoS Computational Biology*, 5(10):e1000529, October 2009. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1000529. URL https://dx.plos.org/10.1371/journal.pcbi.1000529.

D. Ewen Cameron, Caleb J. Bashor, and James J. Collins. A brief history of synthetic biology. *Nature Reviews Microbiology*, 12(5):381–390, May 2014. ISSN 1740-1526, 1740-1534. doi: 10.1038/nrmicro3239. URL http://www.nature.com/articles/nrmicro3239.

Luis Campos. That Was the Synthetic Biology That Was. In Markus Schmidt, Alexander Kelle, Agomoni Ganguli-Mitra, and Huib Vriend, editors, *Synthetic Biology*, pages 5–21. Springer Netherlands, Dordrecht, 2009. ISBN 978-90-481-2677-4 978-90-481-2678-1. doi: 10.1007/978-90-481-2678-1_2. URL http://link.springer.com/10.1007/978-90-481-2678-1_2.

Daniel F Carlson, Mark W Walton, Scott C Fahrenkrug, Perry B Hackett, et al. Precision editing of large animal genomes. In *Advances in genetics*, volume 80, pages 37–97. Elsevier, 2012.

Dana Carroll. Genome engineering with zinc-finger nucleases. *Genetics*, 188(4):773–782, 2011.

J. Cello. Chemical Synthesis of Poliovirus cDNA: Generation of Infectious Virus in the Absence of Natural Template. *Science*, 297(5583):1016–1018, August 2002. ISSN 00368075, 10959203. doi: 10.1126/science.1072266. URL http://www.sciencemag.org/cgi/doi/10.1126/science.1072266.

Raj Chari and George M. Church. Beyond editing to writing large genomes. *Nature Reviews Genetics*, 18(12):749–760, December 2017. ISSN 1471-0056, 1471-0064. doi: 10.1038/nrg.2017.59. URL http://www.nature.com/articles/nrg.2017.59.

J Michael Cherry, Caroline Adler, Catherine Ball, Stephen A Chervitz, Selina S Dwight, Erich T Hester, Yankai Jia, Gail Juvik, TaiYun Roe, Mark Schroeder, and David Botstein. SGD: Saccharomyces genome database. *Nucleic acids research*, 26(1):73–79, 1998.

Ryan E. Cobb, Ran Chao, and Huimin Zhao. Directed evolution: Past, present, and future. *AIChE Journal*, 59(5):1432–1440, May 2013. ISSN 00011541. doi: 10.1002/aic.13995. URL http://doi.wiley.com/10.1002/aic.13995.

Celeste M. Condit, Benjamin R. Bates, Ryan Galloway, Sonja Brown Givens, Caroline K. Haynie, John W. Jordan, Gordon Stables, and Hollis Marshall West. Recipes or blueprints for our genes? How contexts selectively activate the multiple meanings of metaphors. *Quarterly Journal of Speech*, 88(3):303–325, August 2002. ISSN 0033-5630, 1479-5779. doi: 10.1080/00335630209384379. URL http://www.tandfonline.com/doi/abs/10.1080/00335630209384379.

UniProt Consortium. UniProt: a hub for protein information. *Nucleic acids research*, 43 (D1):D204–D212, 2014.

Francis HC Crick. On protein synthesis. In *Symp Soc Exp Biol*, volume 12, page 8, 1958.

Ralf Dahm. Discovering DNA: Friedrich Miescher and the early years of nucleic acid research. *Human genetics*, 122(6):565–581, 2008.

Noah Davidsohn, Jacob Beal, Samira Kiani, Aaron Adler, Fusun Yaman, Yinqing Li, Zhen Xie, and Ron Weiss. Accurate Predictions of Genetic Circuit Behavior from Part Characterization and Modular Composition. *ACS Synthetic Biology*, 4 (6):673–681, June 2015. ISSN 2161-5063, 2161-5063. doi: 10.1021/sb500263b. URL https://pubs.acs.org/doi/10.1021/sb500263b.

Carrie A Davis, Benjamin C Hitz, Cricket A Sloan, Esther T Chan, Jean M Davidson, Idan Gabdank, Jason A Hilton, Kriti Jain, Ulugbek K Baymuradov, Aditi K Narayanan, et al. The Encyclopedia of DNA elements (ENCODE): data portal update. *Nucleic acids research*, 46(D1):D794–D801, 2018.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Sven Dietz and Sven Panke. Microbial systems engineering: First successes and the way ahead. *BioEssays*, 32(4):356–362, March 2010. ISSN 02659247. doi: 10.1002/bies.200900174. URL http://doi.wiley.com/10.1002/bies.200900174.

Walter Doerfler. Beware of manipulations on the genome: epigenetic destabilization through (foreign) DNA insertions, 2016.

Jessica S Dymond, Sarah M Richardson, Candice E Coombes, Timothy Babatz, Héloïse Muller, Narayana Annaluru, William J Blake, Joy W Schwerzmann, Junbiao Dai, Derek L Lindstrom, et al. Synthetic chromosome arms function in yeast and generate phenotypic diversity by design. *Nature*, 477(7365):471–476, 2011.

EBRC. What is Synthetic/Engineering Biology? | EBRC. https://ebrc.org/what-is-synbio/. Accessed: 2020-02-19.

Michael B. Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, January 2000. ISSN 0028-0836, 1476-4687. doi: 10.1038/35002125. URL http://www.nature.com/articles/35002125.

Martin Emms. Tree Distance and Some Other Variants of Evalb. In *LREC*, 2008.

Drew Endy. Foundations for engineering biology. *Nature*, 438(7067):449–453, November 2005. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature04342. URL http://www.nature.com/articles/nature04342.

Daniel Falkner. Metaphors of Life: Reflections on Metaphors in the Debate on Synthetic Biology. In Kristin Hagen, Margret Engelhard, and Georg Toepfer, editors, *Ambivalences of Creating Life*, volume 45, pages 251–265. Springer International Publishing, Cham, 2016. ISBN 978-3-319-21087-2 978-3-319-21088-9. doi: 10.1007/978-3-319-21088-9_13. URL http://link.springer.com/10.1007/978-3-319-21088-9_13.

Stanley Eugene Fish. *Is there a text in this class? The authority of interpretive communities*. Harvard Univ. Press, Cambridge Mass., 11. print edition, 2000. ISBN 978-0-674-46726-2. OCLC: 175040115.

A. C. Forster and G. M. Church. Synthetic biology projects in vitro. *Genome Research*, 17(1):1–6, December 2006a. ISSN 1088-9051. doi: 10.1101/gr.5776007. URL http://www.genome.org/cgi/doi/10.1101/gr.5776007.

Anthony C Forster and George M Church. Towards synthesis of a minimal cell. *Molecular Systems Biology*, 2(1):45, January 2006b. ISSN 1744-4292, 1744-4292. doi: 10.1038/msb4100090. URL https://onlinelibrary.wiley.com/doi/abs/10.1038/msb4100090.

M Gardiner-Garden and M Frommer. CpG islands in vertebrate genomes. *Journal of molecular biology*, 196(2):261–282, 1987.

Timothy S Gardner, Charles R Cantor, and James J Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403(6767):339–342, 2000.

ER Gibney and CM Nolan. Epigenetics and gene expression. *Heredity*, 105(1):4–13, 2010.

D. G. Gibson, J. I. Glass, C. Lartigue, V. N. Noskov, R.-Y. Chuang, M. A. Algire, G. A. Benders, M. G. Montague, L. Ma, M. M. Moodie, C. Merryman, S. Vashee, R. Krishnakumar, N. Assad-Garcia, C. Andrews-Pfannkoch, E. A. Denisova, L. Young, Z.-Q. Qi, T. H. Segall-Shapiro, C. H. Calvey, P. P. Parmar, C. A. Hutchison, H. O. Smith, and J. C. Venter. Creation of a Bacterial Cell Controlled by a Chemically Synthesized Genome. *Science*, 329(5987):52–56, July 2010. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1190719. URL http://www.sciencemag.org/cgi/doi/10.1126/science.1190719.

Daniel G Gibson, Lei Young, Ray-Yuan Chuang, J Craig Venter, Clyde A Hutchison, and Hamilton O Smith. Enzymatic assembly of DNA molecules up to several hundred kilobases. *Nature Methods*, 6(5):343–345, May 2009. ISSN 1548-7091, 1548-7105. doi: 10.1038/nmeth.1318. URL http://www.nature.com/articles/nmeth.1318.

Sylvestre Grizot, Julianne Smith, Fayza Daboussi, Jesus Prieto, Pilar Redondo, Nekane Merino, Maider Villate, Severine Thomas, Laetitia Lemaire, Guillermo Montoya, et al. Efficient targeting of a SCID gene by an engineered single-chain homing endonuclease. *Nucleic acids research*, 37(16):5405–5419, 2009.

Călin C. Guet, Michael B. Elowitz, Weihong Hsing, and Stanislas Leibler. Combinatorial Synthesis of Genetic Networks. *Science*, 296(5572):1466–1470, 2002. ISSN 0036-8075. doi: 10.1126/science.1067407. URL https://science.sciencemag.org/content/296/5572/1466.

Jeff Hasty, David McMillen, Farren Isaacs, and James J. Collins. Computational studies of gene regulatory networks: in numero molecular biology. *Nature Reviews Genetics*, 2 (4):268–279, April 2001. ISSN 1471-0056, 1471-0064. doi: 10.1038/35066056. URL http://www.nature.com/articles/35066056.

Gary Hatfield. Rene Descartes. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2018 edition, 2018.

Nathaniel D Heintzman, Rhona K Stuart, Gary Hon, Yutao Fu, Christina W Ching, R David Hawkins, Leah O Barrera, Sara Van Calcar, Chunxu Qu, Keith A Ching, et al. Distinct and predictive chromatin signatures of transcriptional promoters and enhancers in the human genome. *Nature genetics*, 39(3):311–318, 2007.

Nathan J Hillson, Rafael D Rosengarten, and Jay D Keasling. j5 DNA assembly design automation software. *ACS synthetic biology*, 1(1):14–21, 2011.

Michael M Hoffman, Orion J Buske, Jie Wang, Zhiping Weng, Jeff A Bilmes, and William Stafford Noble. Unsupervised pattern discovery in human chromatin structure through genomic segmentation. *Nature Methods*, 9(5):473–476, May 2012. ISSN 1548-7091, 1548-7105. doi: 10.1038/nmeth.1937. URL http://www.nature.com/articles/nmeth.1937.

Robert Hoffmann. A wiki for the life sciences where authorship matters. *Nature genetics*, 40(9):1047, 2008.

Karen Hopkin. The Evolving Definition of a Gene: With the discovery that nearly all of the genome is transcribed, the definition of a 'gene' needs another revision. *Bioscience*, 59(11):928–931, 2009.

Robert Illingworth, Alastair Kerr, Dina DeSousa, Helle Jørgensen, Peter Ellis, Jim Stalker, David Jackson, Chris Clee, Robert Plumb, Jane Rogers, et al. A novel CpG island set identifies tissue-specific methylation at developmental gene loci. *PLoS biology*, 6(1), 2008.

International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, February 2001. ISSN 0028-0836, 1476-4687. doi: 10.1038/35057062. URL http://www.nature.com/articles/35057062.

David A Jackson, Robert H Symons, and Paul Berg. Biochemical method for inserting new genetic information into DNA of Simian Virus 40: circular SV40 DNA molecules containing lambda phage genes and the galactose operon of Escherichia coli. *Proceedings of the National Academy of Sciences*, 69(10):2904–2909, 1972.

Apache Jakarta. Apache Lucene — a high-performance, full-featured text search engine library. *Apache Lucene*, 2004.

Josep C Jiménez-Chillarón, Rubén Díaz, and Marta Ramón-Krauel. Omics tools for the genome-wide analysis of methylation and histone modifications. In *Comprehensive Analytical Chemistry*, volume 63, pages 81–110. Elsevier, 2014.

Mark Johnson. Pcfg models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632, 1998.

Mark Johnson. PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1157. Association for Computational Linguistics, 2010.

J Keith Joung and Jeffry D Sander. TALENs: a widely applicable technology for targeted genome editing. *Nature reviews Molecular cell biology*, 14(1):49–55, 2013.

Dan Jurafsky and James H. Martin. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall series in artificial intelligence. Pearson Prentice Hall, Upper Saddle River, N.J, 2nd ed edition, 2009. ISBN 978-0-13-187321-6. OCLC: 213375806.

Lily E Kay. *Who wrote the book of life?: A history of the genetic code*. Stanford University Press, 2000.

Evelyn Fox Keller. What Does Synthetic Biology Have to Do with Biology? *BioSocieties*, 4(2-3):291–302, September 2009. ISSN 1745-8552, 1745-8560. doi: 10.1017/S1745855209990123. URL http://link.springer.com/10.1017/S1745855209990123.

Evelyn Fox Keller. From gene action to reactive genomes. *The Journal of Physiology*, 592 (11):2423–2429, June 2014. ISSN 00223751. doi: 10.1113/jphysiol.2014.270991. URL http://doi.wiley.com/10.1113/jphysiol.2014.270991.

Theresa K Kelly, Yaping Liu, Fides D Lay, Gangning Liang, Benjamin P Berman, and Peter A Jones. Genome-wide mapping of nucleosome positioning and DNA methylation within individual DNA molecules. *Genome research*, 22(12):2497–2506, 2012.

Richard Kelwick, James T. MacDonald, Alexander J. Webb, and Paul Freemont. Developments in the Tools and Methodologies of Synthetic Biology. *Frontiers in Bioengineering and Biotechnology*, 2:60, 2014. ISSN 2296-4185. doi: doi:10.3389/fbioe.2014.00060. URL https://www.frontiersin.org/article/10.3389/fbioe.2014.00060.

Hadas Keren, Galit Lev-Maor, and Gil Ast. Alternative splicing and evolution: diversification, exon definition and function. *Nature Reviews Genetics*, 11(5):345–355, 2010.

Sehrish Khan, Muhammad Shahid Mahmood, Sajjad ur Rahman, Hassan Zafar, Sultan Habibullah, Aftab Ahmad, et al. CRISPR/Cas9: the Jedi against the dark empire of diseases. *Journal of Biomedical Science*, 25(1):29, 2018.

Hyo-Joong Kim and Steven A. Benner. Prebiotic stereoselective synthesis of purine and noncanonical pyrimidine nucleotide from nucleobases and phosphorylated carbohydrates. *Proceedings of the National Academy of Sciences*, 114(43):11315–11320, October 2017. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1710778114. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1710778114.

Yoon Kim, Chris Dyer, and Alexander Rush. Compound Probabilistic Context-Free Grammars for Grammar Induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2369–2385, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1228. URL https://www.aclweb.org/anthology/P19-1228.

Werner Kogge and Michael Richter. Synthetic biology and its alternatives. Descartes, Kant and the idea of engineering biological machines. *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences*, 44(2):181–189, 2013.

Veerendra Koppolu and Veneela KR Vasigala. Role of *Escherichia coli* in Biofuel Production. *Microbiology Insights*, 9:MBI.S10878, January 2016. ISSN 1178-6361, 1178-6361. doi: 10.4137/MBI.S10878. URL http://journals.sagepub.com/doi/10.4137/MBI.S10878.

Arthur Kornberg. Biologic synthesis of deoxyribonucleic acid. *Science*, 131(3412): 1503–1508, 1960.

Roger D Kornberg. Chromatin structure: a repeating unit of histones and DNA. *Science*, 184(4139):868–871, 1974.

Vivek Krishnakumar, Sergio Contrino, Chia-Yi Cheng, Irina Belyaeva, Erik S Ferlanti, Jason R Miller, Matthew W Vaughn, Gos Micklem, Christopher D Town, and Agnes P Chan. ThaleMine: a warehouse for Arabidopsis data integration and discovery. *Plant and Cell Physiology*, 58(1):e4–e4, 2016.

Grzegorz Kudla, Leszek Lipinski, Fanny Caffin, Aleksandra Helwak, and Maciej Zylicz. High guanine and cytosine content increases mRNA levels in mammalian cells. *PLoS biology*, 4(6), 2006.

Roberta Kwok. Five hard truths for synthetic biology. *Nature*, 463(7279):288–290, January 2010. ISSN 0028-0836, 1476-4687. doi: 10.1038/463288a. URL http://www.nature.com/articles/463288a.

George Lakoff and Mark Johnson. *Metaphors we live by*. University of Chicago Press, Chicago, 2003. ISBN 978-0-226-46801-3.

Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature methods*, 9(4):357, 2012.

Tuuli Lappalainen and John M. Greally. Associating cellular epigenetic models with human phenotypes. *Nature Reviews Genetics*, 18(7):441–451, July 2017. ISSN 1471-0056, 1471-0064. doi: 10.1038/nrg.2017.32. URL http://www.nature.com/articles/nrg.2017.32.

James Lennox. Aristotle's Biology. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2019 edition, 2019.

Ting Li, Sheng Huang, Wen Zhi Jiang, David Wright, Martin H Spalding, Donald P Weeks, and Bing Yang. TAL nucleases (TALNs): hybrid proteins composed of TAL effectors and FokI DNA-cleavage domain. *Nucleic acids research*, 39(1):359–372, 2011.

M. Susan Lindee and Evelyn Fox Keller. Refiguring Life: Metaphors of Twentieth-Century Biology. *Technology and Culture*, 38(2):497, April 1997. ISSN 0040165X. doi: 10.2307/3107136. URL https://www.jstor.org/stable/3107136?origin=crossref.

John Logan, Erik Falck-Pedersen, James E Darnell, and Thomas Shenk. A poly (A) addition site and a downstream termination region are required for efficient cessation of transcription by RNA polymerase II in the mouse beta maj-globin gene. *Proceedings of the National Academy of Sciences*, 84(23):8306–8310, 1987.

Richard Losick and Lucy Shapiro. Bringing the mountain to Mohammed. *Science*, 282 (5393):1430–1431, 1998.

Anke Lüdeling and Merja Kytö. *Corpus linguistics*, volume 1. Walter de Gruyter, 2008.

Lorena Magraner-Pardo, Vicent Pelechano, María Dolores Coloma, and Vicente Tordera. Dynamic remodeling of histone modifications in response to osmotic stress in Saccharomyces cerevisiae. *BMC genomics*, 15(1):247, 2014.

Colin D Malone and Gregory J Hannon. Small RNAs as guardians of the genome. *Cell*, 136(4):656–668, 2009.

Alessandro Mammana and Ho-Ryun Chung. Chromatin segmentation based on a probabilistic model for read counts explains a large portion of the epigenome. *Genome Biology*, 16(1):151, December 2015. ISSN 1474-760X. doi: 10.1186/s13059-015-0708-z. URL https://genomebiology.biomedcentral.com/articles/10.1186/s13059-015-0708-z.

Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

Xizeng Mao, Qin Ma, Chuan Zhou, Xin Chen, Hanyuan Zhang, Jincai Yang, Fenglou Mao, Wei Lai, and Ying Xu. DOOR 2.0: presenting operons and their functions through dynamic and integrated views. *Nucleic Acids Research*, 42(D1):D654–D659,

11 2013. ISSN 0305-1048. doi: 10.1093/nar/gkt1048. URL https://doi.org/10.1093/nar/gkt1048.

Mario A Marchisio and Jörg Stelling. Computational design tools for synthetic biology. *Current opinion in biotechnology*, 20(4):479–485, 2009.

Michael McCandless, Erik Hatcher, and Otis Gospodnetic. *Lucene in action: covers Apache Lucene 3.0*. Manning Publications Co., 2010.

Göksel Mısırlı, Curtis Madsen, Iñaki Sainz de Murieta, Matthieu Bultelle, Keith Flanagan, Matthew Pocock, Jennifer Hallinan, James Alastair McLaughlin, Justin Clark-Casey, Mike Lyne, and Anil Wipat. Constructing synthetic biology workflows in the cloud. *Engineering Biology*, 1(1):61–65, 2017.

TM Nafee, WE Farrell, WD Carroll, AA Fryer, and KMK Ismail. Epigenetic control of fetal gene expression. *BJOG: An International Journal of Obstetrics & Gynaecology*, 115(2):158–168, 2008.

Abhinav Nellore, Konstantin Bobkov, Elizabeth Howe, Aleksandr Pankov, Aaron Diaz, and Jun S Song. NSeq: a multithreaded Java application for finding positioned nucleosomes from sequencing data. *Frontiers in genetics*, 3:320, 2013.

Brigitte Nerlich and Iina Hellsten. Beyond the human genome: microbes, metaphors and what it means to be human in an interconnected post-genomic world. *New Genetics and Society*, 28(1):19–36, March 2009. ISSN 1463-6778, 1469-9915. doi: 10.1080/14636770802670233. URL http://www.tandfonline.com/doi/abs/10.1080/14636770802670233.

Antonine Nicoglou and Francesca Merlin. Epigenetics: A way to bridge the gap between biological fields. *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences*, 66:73–82, December 2017. ISSN 13698486. doi: 10.1016/j.shpsc.2017.10.002. URL https://linkinghub.elsevier.com/retrieve/pii/S1369848617300444.

A. A. K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, and C. A. Voigt. Genetic circuit design automation. *Science*, 352(6281):aac7341–aac7341, April 2016. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aac7341. URL http://www.sciencemag.org/cgi/doi/10.1126/science.aac7341.

Jens Nielsen and Jay D Keasling. Engineering cellular metabolism. *Cell*, 164(6):1185–1197, 2016.

Friedrich Wilhelm Nietzsche. *On truth and lies in a nonmoral sense*. Book Liberation Front, S.l., 1896. ISBN 978-1-5121-0939-9. OCLC: 1116154147.

Helen Pearson. What is a gene? *Nature*, 441(7092):398–401, May 2006. ISSN 0028-0836, 1476-4687. doi: 10.1038/441398a. URL http://www.nature.com/articles/441398a.

Elizabeth Pennisi. The CRISPR craze, 2013.

Benjamin A Pierce. *Genetics: A conceptual approach*. Macmillan, 2012.

Clémence Pinel, Barbara Prainsack, and Christopher McKevitt. Markers as mediators: A review and synthesis of epigenetics literature. *BioSocieties*, 13(1):276–303, 2018.

Leslie Pray. Discovery of DNA structure and function: Watson and crick. *Nature Education*, 1(1):100, 2008.

Sarah M Richardson, Sarah J Wheelan, Robert M Yarrington, and Jef D Boeke. Genedesign: rapid, automated design of multikilobase synthetic genes. *Genome research*, 16(4):550–556, 2006.

Sarah M. Richardson, Leslie A. Mitchell, Giovanni Stracquadanio, Kun Yang, Jessica S. Dymond, James E. DiCarlo, Dongwon Lee, Cheng Lai Victor Huang, Srinivasan Chandrasegaran, Yizhi Cai, Jef D. Boeke, and Joel S. Bader. Design of a synthetic yeast genome. *Science*, 355(6329):1040–1044, March 2017. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.aaf4557. URL http://www.sciencemag.org/lookup/doi/10.1126/science.aaf4557.

Richard J Roberts. How restriction enzymes became the workhorses of molecular biology. *Proceedings of the National Academy of Sciences*, 102(17):5905–5908, 2005.

Steve Rozen and Helen Skaletsky. Primer3 on the WWW for general users and for biologist programmers. In *Bioinformatics methods and protocols*, pages 365–386. Springer, 2000.

Frederick Sanger, Steven Nicklen, and Alan R Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences*, 74(12):5463–5467, 1977.

Sc2.0. Synthetic Yeast 2.0, Goals. URL http://syntheticyeast.org/sc2-0/goals/. Accessed: 2020-01-06.

Emily Scher, Yisha Luo, Aaron Berliner, Jacqueline Quinn, Carlos Olguin, and Yizhi Cai. GenomeCarver: harvesting genetic parts from genomes to support biological design automation. In *6th International Workshop on Bio-Design Automation, Seattle, WA*, 2014.

Emily Scher, Shay B Cohen, and Guido Sanguinetti. PartCrafter: find, generate and analyze BioParts. *Synthetic Biology*, 4(1):ysz014, 2019.

Dominic Schmidt, Michael D Wilson, Christiana Spyrou, Gordon D Brown, James Hadfield, and Duncan T Odom. ChIP-seq: using high-throughput sequencing to discover protein–DNA interactions. *Methods*, 48(3):240–248, 2009.

Markus Schmidt. Do I Understand What I Can Create? In Markus Schmidt, Alexander Kelle, Agomoni Ganguli-Mitra, and Huib Vriend, editors, *Synthetic Biology*, pages 81–100. Springer Netherlands, Dordrecht, 2009. ISBN 978-90-481-2677-4 978-90-481-2678-1. doi: 10.1007/978-90-481-2678-1_6. URL http://link.springer.com/10.1007/978-90-481-2678-1_6.

David B Searls. The language of genes. *Nature*, 420(6912):211–217, 2002.

Satoshi Sekine and Michael Collins. Evalb bracket scoring program. 1997.

Yue Shen, Giovanni Stracquadanio, Yun Wang, Kun Yang, Leslie A. Mitchell, Yaxin Xue, Yizhi Cai, Tai Chen, Jessica S. Dymond, Kang Kang, Jianhui Gong, Xiaofan Zeng, Yongfen Zhang, Yingrui Li, Qiang Feng, Xun Xu, Jun Wang, Jian Wang, Huanming Yang, Jef D. Boeke, and Joel S. Bader. SCRaMbLE generates designed combinatorial stochastic diversity in synthetic chromosomes. *Genome Research*, 26(1): 36–49, January 2016. ISSN 1088-9051, 1549-5469. doi: 10.1101/gr.193433.115. URL http://genome.cshlp.org/lookup/doi/10.1101/gr.193433.115.

Vipula K Shukla, Yannick Doyon, Jeffrey C Miller, Russell C DeKelver, Erica A Moehle, Sarah E Worden, Jon C Mitchell, Nicole L Arnold, Sunita Gopalan, Xiangdong Meng, et al. Precise genome modification in the crop species Zea mays using zinc-finger nucleases. *Nature*, 459(7245):437–441, 2009.

Tong Si and Huimin Zhao. A brief overview of synthetic biology research programs and roadmap studies in the United States. *Synthetic and Systems Biotechnology*, 1(4): 258–264, December 2016. ISSN 2405805X. doi: 10.1016/j.synbio.2016.08.003. URL https://linkinghub.elsevier.com/retrieve/pii/S2405805X1630031X.

Piro Siuti, John Yazbek, and Timothy K Lu. Synthetic circuits integrating logic and memory in living cells. *Nature biotechnology*, 31(5):448, 2013.

Hamilton O Smith and KW Welcox. A restriction enzyme from Hemophilus influenzae: I. Purification and general properties. *Journal of molecular biology*, 51(2):379–391, 1970.

Siddarth Srinivasan, Geoff Gordon, and Byron Boots. Learning hidden quantum Markov models. *arXiv preprint arXiv:1710.09016*, 2017.

Waclaw Szybalski. Nobel prizes and restriction enzymes. *Gene*, 4(3):181–182, November 1978. ISSN 03781119. doi: 10.1016/0378-1119(78)90016-1. URL https://linkinghub.elsevier.com/retrieve/pii/0378111978900161.

Erika Szymanski and Emily Scher. Models for DNA Design Tools: The Trouble with Metaphors Is That They Don't Go Away. *ACS Synthetic Biology*, 8(12):2635–2641, 2019.

Chris Tachibana. Beyond CRISPR: What's current and upcoming in genome editing. *Science*, 365(6460):1484–1484, Sep 2019. doi: 10.1126/science.365.6460.1484-b.

Alvin Tamsir, Jeffrey J Tabor, and Christopher A Voigt. Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature*, 469(7329):212, 2011.

Yang Tang, Xiang-Dong Gao, Yinsheng Wang, Bi-Feng Yuan, and Yu-Qi Feng. Widespread existence of cytosine methylation in yeast DNA measured by gas chromatography/mass spectrometry. *Analytical chemistry*, 84(16):7249–7255, 2012.

Volker Thiel. Synthetic viruses — Anything new? *PLOS Pathogens*, 14(10):e1007019, October 2018. ISSN 1553-7374. doi: 10.1371/journal.ppat.1007019. URL https://dx.plos.org/10.1371/journal.ppat.1007019.

CH Waddington. The Strategy of the Genes, 1957.

Conrad H Waddington. The epigenotype. *International journal of epidemiology*, 41(1):10–13, 2012.

J. D. Watson and F. H. C. Crick. Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid. *Nature*, 171(4356):737–738, April 1953. ISSN 0028-0836, 1476-4687. doi: 10.1038/171737a0. URL http://www.nature.com/articles/171737a0.

Bruce Weber. Life. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2018 edition, 2018.

Assaf Weiner, Tsung-Han S Hsieh, Alon Appleboim, Hsiuyi V Chen, Ayelet Rahat, Ido Amit, Oliver J Rando, and Nir Friedman. High-Resolution Chromatin Dynamics during a Yeast Stress Response. *Molecular Cell*, 58(2):371–386, April 2015. ISSN 10972765. doi: 10.1016/j.molcel.2015.02.002. URL https://linkinghub.elsevier.com/retrieve/pii/S1097276515000945.

Bernard Weiss and Charles C Richardson. Enzymatic breakage and joining of deoxyribonucleic acid, I. Repair of single-strand breaks in DNA by an enzyme system from Escherichia coli infected with T4 bacteriophage. *Proceedings of the National Academy of Sciences of the United States of America*, 57(4):1021, 1967.

Huihuang Yan, Shulan Tian, Susan L Slager, Zhifu Sun, and Tamas Ordog. Genome-Wide Epigenetic Studies in Human Disease: A Primer on -Omic Technologies. *American journal of epidemiology*, 183(2):96–109, 2016.

Benedikt Zacher, Margaux Michel, Bjoern Schwalb, Patrick Cramer, Achim Tresch, and Julien Gagneur. Accurate promoter and enhancer identification in 127 ENCODE and roadmap epigenomics cell types and tissues by GenoSTAN. *PloS one*, 12(1), 2017.

Yong Zhang, Tao Liu, Clifford A Meyer, Jérôme Eeckhoute, David S Johnson, Bradley E Bernstein, Chad Nusbaum, Richard M Myers, Myles Brown, Wei Li, et al. Model-based analysis of ChIP-Seq (MACS). *Genome biology*, 9(9):R137, 2008.

Yingming Zhao and Benjamin A Garcia. Comprehensive catalog of currently documented histone modifications. *Cold Spring Harbor perspectives in biology*, 7(9):a025064, 2015.

Valentin Zulkower. DNA Chisel, 2018. URL https://edinburgh-genome-foundry.github.io/DnaChisel/index.html. (Accessed on 24/05/2018).