

# **Bioinformatic Investigations Into the Genetic Architecture of Renal Disorders**

Christopher Cheshire

A Thesis Submitted for the Degree of  
Doctor of Philosophy

**University College London**

July 2019

# Declaration

I, Christopher Cheshire confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

# Abstract

Modern genomic analysis has a significant bioinformatic component due to the high volume of complex data that is involved. During investigations into the genetic components of two renal diseases, we developed two software tools.

Genome-Wide Association Studies (GWAS) datasets may be genotyped on different microarrays and subject to different annotation, leading to a mosaic case-control cohort that has inherent errors, primarily due to strand mismatching. Our software REMEDY seeks to detect and correct strand designation of input datasets, as well as filtering for common sources of noise such as structural and multi-allelic variants. We performed a GWAS on a large cohort of Steroid-sensitive nephrotic syndrome samples; the mosaic input datasets were pre-processed with REMEDY prior to merging and analysis. Our results show that REMEDY significantly reduced noise in GWAS output results. REMEDY outperforms existing software as it has significantly more features available such as auto-strand designation detection, comprehensive variant filtering and high-speed variant matching to dbSNP.

The second tool supported the analysis of a newly characterised rare renal disorder: Polycystic kidney disease with hyperinsulinemic hypoglycemia (HIPKD). Identification of the underlying genetic cause led to the hypothesis that a change in chromatin looping at a specific locus affected the aetiology of the disease. We developed LOOPER, a software suite capable of predicting chromatin loops from ChIP-Seq data to explore the possible conformations of chromatin architecture in the HIPKD genomic region. LOOPER predicted several interesting functional and structural loops that supported our hypothesis. We then extended LOOPER to visualise ChIA-PET and ChIP-Seq data as a force-directed graph to show experimental structural and functional chromatin interactions. Next, we re-analysed the HIPKD region with LOOPER to show experimentally validated chromatin interactions. We first confirmed our original predicted loops and subsequently discovered that the local genomic region has many more chromatin features than first thought.

# Impact Statement

Many of the challenges presented by the vast amount of experimental data involved in modern genomic analysis are overcome with software tools. By developing two different pieces of software that specifically address two real problems which were encountered by our research group, our work has contributed to at least three papers at different stages of publication. Furthermore, our software has the potential to impact the broader research community and has contributed to a better understanding of rare renal genetic diseases.

Our first tool REMEDY sought to aid in the pre-processing stages of preparing genotyping datasets for genome-wide association studies (GWAS). We used the software to successfully merge data in two different GWAS projects, which would have been impossible without the technical research that is presented in this thesis. The first project, a study on steroid-sensitive nephrotic syndrome (SSNS) [1], has contributed to our understanding of the aetiology of this rare autoimmune kidney disease in children and may lead to better diagnosis and treatment in the future. The second GWAS project that REMEDY has contributed to is a rare blood disorder; our results again point to genomic regions which we are currently investigating to better understand the disease mechanism.

REMEDY was developed to be a deployable package, usable by non-technical clinicians to aid the wider research community in processing large genotyping based datasets. We believe our software can be useful to any genotyping data-based GWAS, especially where large control datasets are required; hence, REMEDY the potential to have a significant impact in the study of the genetic architecture of many different diseases. Standardisation of study protocol is an important aspect of study design; we think REMEDY contributes to the standardisation of GWAS practice and therefore, the reliability and repeatability of this class of experiments.

Our second tool, LOOPER, was designed to aid in the analysis and visualisation of chromatin interactions in the genome. We successfully used the tool to explore a specific genomic region where we previously found a mutation which we showed to be important in a newly characterised disease, Hyperinsulinism with polycystic kidney disease (HIPKD).



The loop predictions and later visualisation helped our team to build a hypothesis around the mechanism of change in the chromatin architecture which was contributing to the genetic component of the disease, and was later published in a high-impact journal [2].

The 3D regulatory architecture of the genome affects crucial cellular processes and is implicated in a wide range of diseases. 3D regulatory networks have been shown to be extremely complex, even in a small area; consequently, tools to process and visualise the raw data that allows researchers to explore this information in a concise and understandable manner are a critical component to understanding how the regulatory architecture of the genome functions. LOOPER provides both a prediction framework for the analysis of un-observed chromatin loops and provides a novel way of viewing existing chromatin confirmation data as a force-directed graph, which we believe is novel in the field.

# Acknowledgements

To my darling Emma, without your boundless love, tolerance and literal back support, I would have not made it through the last year let alone been able to complete my PhD. I love you always and forever.

To Tim and Maria, I cannot even begin to thank you for all the kindness you have shown me over the last four years. Without your financial support, understanding and belief in me, I would have not felt able to start this journey, and I certainly would have faltered long before the end. I will make it up to you one day I promise!

To my mother and Steve, your limitless enthusiasm about my course of action has kept me going throughout, even when I felt lost at sea. For both being the constant in my life that I always need, that's what truly great parents are, I think.

To my sisters and brothers: Holly, Jenny, Sophie, Bradley, Matt and Sam, for your love and support over the last year when I really needed it. Also for reminding me that it's not all about me and that life exists outside of a PhD! Without our time together playing games and relaxing over holidays, I may have lost sight of the big picture.

A very special thank you to Horia, my supervisor and friend, for your endless optimism, support and your tolerance of my many mood swings over the last few months, and for giving me a chance and believing in me in the first place; and for your encyclopedic knowledge of basically everything!

To Robert, our glorious leader, who gave me the opportunity to study in his group in the first place. Without your ability to see through everything and play 'devil's advocate', nothing I have achieved in this group would have been to nearly as high a standard.

Finally, but by no means least, to all my colleagues in our little corner of the research world, I love you all, I will never forget our time together: Matt, Sanj, Steph, Mallory, Anne, Vaksha, Detlef, Anselm, Monika, Dana, Mel and Mehmet.

# Viewing Recommendations

When viewing this in a PDF document, all citation, table and figure references can be followed by clicking the hyperlink to the target by hovering over the number.

In Adobe Acrobat, once the user wishes to resume reading of the original section, the keyboard shortcut [Alt]+[leftArrow] can be used to jump back to the previous section. To jump back to the reference, use the keyboard shortcut [Alt]+[rightArrow]. Similar shortcuts most likely exist in other programs but are not shown here.

# Contents

<b>1</b>	<b>Introduction</b>	<b>38</b>
<b>1.1</b>	<b>Genomic Analysis</b>	<b>38</b>
1.1.1	Genomic Analysis Methodology	39
1.1.2	Computing in Genomic Analysis	40
1.1.2.1	Databases and Interfaces	40
1.1.2.2	Software Tools	41
1.1.3	Thesis Justification	41
<b>1.2</b>	<b>Basic Genetics</b>	<b>42</b>
1.2.1	Deoxyribonucleic Acid	42
1.2.2	Proteins	46
1.2.3	Ribonucleic Acid	46
1.2.4	The Central Dogma of Molecular Biology	46
1.2.5	DNA as a Sequence	47
1.2.6	Genes	47
1.2.7	Gene Expression	48
1.2.8	Chromatin	49
1.2.9	Regulation of Gene Expression	51
1.2.9.1	Enhancers and Silencers	52
1.2.9.2	Insulators	53
1.2.10	The Human Genome	54

1.2.11	3D Genome Structure and Organisation . . . . .	55
1.2.11.1	3D Gene Regulation . . . . .	55
1.2.11.2	CTCF/Cohesin Loop Formation . . . . .	56
<b>1.3</b>	<b>Genetic Variation and Inheritance . . . . .</b>	<b>60</b>
1.3.1	Mutation . . . . .	61
1.3.2	Recombination . . . . .	61
1.3.3	Human Genetic Inheritance . . . . .	62
1.3.4	Genetic Variation . . . . .	64
1.3.4.1	Single Nucleotide Variants . . . . .	65
1.3.4.2	Insertions and Deletions . . . . .	67
1.3.4.3	Duplicates and Copy Number Variation . . . . .	67
1.3.4.4	Other Structural Variants . . . . .	68
1.3.5	Variation and Disease . . . . .	68
<b>1.4</b>	<b>Genotyping Data Analysis . . . . .</b>	<b>69</b>
1.4.1	DNA Sequencing . . . . .	69
1.4.2	Defining Genomic Position and Strand . . . . .	71
1.4.3	Defining Variance Using a Reference Genome . . . . .	73
1.4.3.1	Sequence Ambiguity . . . . .	74
1.4.4	Strand Designation Schemes . . . . .	74
1.4.4.1	POS/NEG (+/-) . . . . .	74
1.4.4.2	FWD/REV . . . . .	75
1.4.4.3	TOP/BOT . . . . .	75
1.4.4.4	DESIGN . . . . .	77
1.4.4.5	Converting Between Schemes . . . . .	77
1.4.5	Genotyping . . . . .	78
1.4.5.1	Microarray Technology . . . . .	78

1.4.5.2	Probe Design . . . . .	79
1.4.6	Illumina Genotyping . . . . .	80
1.4.6.1	Bead Chip Technology . . . . .	80
1.4.6.2	Manifest Files . . . . .	81
1.4.6.3	Genome Studio . . . . .	81
1.4.7	Multi-allelic and Structural Variation . . . . .	83
1.4.8	Data Consistency and Merging . . . . .	83
1.4.9	Positional Cloning . . . . .	84
1.4.9.1	Linkage Analysis . . . . .	84
1.4.9.2	Association Studies . . . . .	85
1.4.9.2.1	Basic Allele Testing . . . . .	86
1.4.9.2.2	Linkage Disequilibrium . . . . .	87
1.4.9.2.3	Quality Control . . . . .	87
1.4.9.2.3.1	Call Rate . . . . .	88
1.4.9.2.3.2	Minor Allele Frequency . . . . .	88
1.4.9.2.3.3	Hardy-Weinberg Equilibrium . . . . .	88
1.4.9.2.4	Principle Component Analysis . . . . .	88
1.4.9.2.5	Imputation . . . . .	89
1.4.9.2.6	Q-Q Plots . . . . .	89
1.5	Graph Visualisation . . . . .	90
<b>2</b>	<b>Methods</b>	<b>92</b>
2.1	Genomic Analysis Methodology . . . . .	92
2.1.1	Golden Rules . . . . .	92
2.1.1.1	Version Control . . . . .	92
2.1.1.2	Naming Conventions . . . . .	93

2.1.1.3	Documentation . . . . .	93
2.1.1.4	Security and Redundancy . . . . .	94
2.1.1.5	Repeatability . . . . .	94
2.1.2	Project Planning . . . . .	95
2.1.3	Pipeline Design . . . . .	96
2.1.4	Pipeline Repeatability . . . . .	100
<b>2.2</b>	<b>GWAS Methodology . . . . .</b>	<b>100</b>
2.2.1	Source Data Processing . . . . .	101
2.2.2	Microarray Selection . . . . .	101
2.2.3	Sample Preparation and Genotyping . . . . .	102
2.2.4	Illumina Base Calling . . . . .	102
2.2.5	Data Collection and Merging . . . . .	103
2.2.6	Data Integration . . . . .	104
2.2.7	Data Quality Control and Filtering . . . . .	104
2.2.8	Variance Control . . . . .	105
2.2.9	Further Analysis . . . . .	105
<b>2.3</b>	<b>REMEDY . . . . .</b>	<b>106</b>
2.3.1	Initial Research and Challenges . . . . .	107
2.3.1.1	Control Set Identification . . . . .	107
2.3.1.2	Association Test Noise . . . . .	108
2.3.1.3	Strand Designation Scheme Detection . . . . .	110
2.3.1.4	Strand Designation Transcoding . . . . .	111
2.3.1.5	Additional Noise Sources . . . . .	117
2.3.1.6	Final Algorithm . . . . .	118
2.3.2	Software Requirement Justification . . . . .	118
2.3.3	Programming Language Selection . . . . .	119

2.3.4	Development Environment . . . . .	120
2.3.5	Unit Testing . . . . .	120
2.3.6	Version Control . . . . .	121
2.3.7	Program Packing and Deployment . . . . .	121
<b>2.4</b>	<b>LOOPER . . . . .</b>	<b>121</b>
2.4.1	Software Requirement Justification . . . . .	121
2.4.2	Programming Language Selection . . . . .	122
2.4.3	Development Environment . . . . .	122
2.4.4	Unit Testing . . . . .	122
2.4.5	Version Control . . . . .	123
2.4.6	Program Packing and Deployment . . . . .	123
2.4.7	Common Tools and Methods . . . . .	123
2.4.7.1	Chromatin Immunoprecipitation . . . . .	123
2.4.7.2	UCSC Genome Segmentation Track . . . . .	124
2.4.8	Chromatin Loop Prediction Algorithm Methodology . . . . .	124
2.4.8.1	Motif Discovery . . . . .	124
2.4.8.2	Motif Scanning and Site Orientation . . . . .	125
2.4.8.3	Predicted Loop Visualisation and Interpretation . . . . .	126
2.4.9	Force Directed Graph Loop Visualisation . . . . .	129
2.4.9.1	Source Data . . . . .	129
2.4.9.2	Mango pre-processing . . . . .	130
2.4.9.3	Graph Loading and Visualisation . . . . .	130
<b>3</b>	<b>Results</b>	<b>131</b>
<b>3.1</b>	<b>REMEDY . . . . .</b>	<b>131</b>
3.1.1	Design Specification . . . . .	131



3.1.2	Software Overview . . . . .	133
3.1.3	Data Workflow . . . . .	134
3.1.4	Custom Variant Database . . . . .	134
3.1.5	Command Line Inputs . . . . .	138
3.1.6	Input File Formats . . . . .	140
3.1.7	Primary Modules . . . . .	140
3.1.7.1	Illumina Manifest Scanning . . . . .	140
3.1.7.1.1	Probe Screening . . . . .	142
3.1.7.1.2	rsID Mapping . . . . .	142
3.1.7.1.3	Variant Matching . . . . .	143
3.1.7.1.3.1	Unmatched Variants . . . . .	143
3.1.7.1.3.2	Structural Variants . . . . .	143
3.1.7.1.3.3	Multiallelic Variants . . . . .	143
3.1.7.1.3.4	Strand Mismatches . . . . .	144
3.1.7.1.3.5	Invalid Positions . . . . .	144
3.1.7.1.4	Manifest Comparison . . . . .	144
3.1.7.1.5	Logging Output Interpretation . . . . .	145
3.1.7.2	Genotype Data Scanning . . . . .	146
3.1.7.2.1	Genotype Scanning . . . . .	147
3.1.7.2.2	Strand Designation Detection . . . . .	148
3.1.7.2.3	Logging Output Interpretation . . . . .	148
3.1.7.3	Data Re-coding and Conversion . . . . .	149
3.1.7.3.1	Manifest Scan Filtering . . . . .	151
3.1.7.3.2	Re-coding . . . . .	151
3.1.7.3.3	Genome Build Differences . . . . .	152
3.1.7.3.4	VCF File Output . . . . .	152
3.1.7.3.5	Logging Output Interpretation . . . . .	152

3.1.8	Data Merging . . . . .	153
3.1.9	Installation and Setup . . . . .	154
3.1.10	Software Performance . . . . .	155
3.1.11	Steroid-Sensitive Nephrotic Syndrome Case Study . . . . .	156
3.1.11.1	Introduction . . . . .	156
3.1.11.2	Project Planning . . . . .	157
3.1.11.3	Sample Preparation and Genotyping . . . . .	158
3.1.11.3.1	Case Samples . . . . .	159
3.1.11.3.2	Control Samples . . . . .	159
3.1.11.4	Pipeline Design . . . . .	160
3.1.11.4.1	Source Data . . . . .	164
3.1.11.4.2	Microarray Supporting Data . . . . .	164
3.1.11.4.3	REMEDY-DB . . . . .	164
3.1.11.4.4	Redundant Data Storage . . . . .	164
3.1.11.4.5	Manifest Scan . . . . .	165
3.1.11.4.6	Genotype Scan . . . . .	165
3.1.11.4.7	Re-coding and Conversion . . . . .	165
3.1.11.4.8	Per Sample QC . . . . .	165
3.1.11.4.9	Per SNV QC . . . . .	165
3.1.11.4.10	Data Merging . . . . .	166
3.1.11.4.11	PCA Ethnicity Control . . . . .	166
3.1.11.4.12	Imputation . . . . .	166
3.1.11.4.13	Association Testing . . . . .	166
3.1.11.5	Node Resolution . . . . .	166
3.1.11.6	Edge Analysis . . . . .	167
3.1.11.7	Pipeline Implementation . . . . .	168
3.1.11.8	REMEDY Processing . . . . .	168

3.1.11.8.1	Manifest Scanning . . . . .	169
3.1.11.8.2	Genotype Scanning . . . . .	173
3.1.11.8.3	Re-coding and Conversion . . . . .	176
3.1.11.9	Quality Control, Filtering and merging . . . . .	176
3.1.11.10	Principal Component Analysis . . . . .	177
3.1.11.11	Imputation . . . . .	177
3.1.11.12	Association Testing . . . . .	178
<b>3.2</b>	<b>LOOPER . . . . .</b>	<b>182</b>
3.2.1	Software Overview . . . . .	182
3.2.2	Loop Prediction Algorithm . . . . .	183
3.2.2.1	ChIP-Seq Data . . . . .	185
3.2.2.2	Motif Discovery . . . . .	185
3.2.2.3	Motif Scanning and Site Orientation . . . . .	185
3.2.2.4	Structural Loop Prediction . . . . .	186
3.2.2.5	Functional Loop Prediction . . . . .	187
3.2.2.6	Predicted Loop Visualisation . . . . .	187
3.2.3	Loop Visualisation Algorithm . . . . .	188
3.2.3.1	Source Data . . . . .	188
3.2.3.2	Pre-processing . . . . .	188
3.2.3.3	Graph Generation Algorithm . . . . .	189
3.2.3.3.1	ChIA-Pet Interaction Strength Calculation . . . . .	192
3.2.4	Case Study Introduction . . . . .	194
3.2.5	Initial Research . . . . .	195
3.2.5.1	Clinical Presentation . . . . .	195
3.2.5.2	Genetic Analysis . . . . .	195
3.2.5.3	Functional Analysis . . . . .	196

3.2.6	Bioinformatic Analysis . . . . .	197
3.2.6.1	Binding Motif Discovery . . . . .	197
3.2.6.1.1	CTCF . . . . .	197
3.2.6.1.2	Cohesin . . . . .	199
3.2.6.1.3	ZNF143 . . . . .	199
3.2.6.2	Region of Interest Initial Analysis . . . . .	200
3.2.6.2.1	PMM2 . . . . .	202
3.2.6.2.2	TMEM186 . . . . .	202
3.2.6.2.3	CARHSP1 . . . . .	202
3.2.6.2.4	ABAT . . . . .	202
3.2.6.3	ZNF143 Binding Analysis . . . . .	202
3.2.6.4	Binding Site Prediction and Analysis . . . . .	203
3.2.6.4.1	CTCF . . . . .	203
3.2.6.4.2	ZNF143 . . . . .	205
3.2.6.4.3	Cohesin . . . . .	206
3.2.6.5	Structural Loop Prediction . . . . .	207
3.2.6.6	Functional Loop Prediction . . . . .	209
3.2.7	LOOPER Graph Analysis . . . . .	209
3.2.7.1	Source Data and Pre-Processing . . . . .	209
3.2.7.2	Graph Analysis . . . . .	210
3.2.7.3	Additional Tool Assessment . . . . .	221
3.2.7.4	Further Analysis . . . . .	222
3.2.7.5	Summary . . . . .	227

## 4 Discussion 228

### 4.1 REMEDY . . . . . 229

#### 4.1.1 Manifest Scanning Evaluation . . . . . 230

4.1.1.1	Structural Variation . . . . .	230
4.1.1.2	Comments File . . . . .	232
4.1.1.3	dbSNP Variant Assessment . . . . .	232
4.1.2	Strand Designation Evaluation . . . . .	234
4.1.3	Re-coding Evaluation . . . . .	236
4.1.4	REMEDY-DB Evaluation . . . . .	237
4.1.5	Software Performance . . . . .	240
4.1.6	Oxford Pipeline Comparison . . . . .	240
4.1.7	Future Work . . . . .	243
<b>4.2</b>	<b>LOOPER . . . . .</b>	<b>245</b>
4.2.1	Loop Prediction Evaluation . . . . .	245
4.2.1.1	CTCF/Cohesin Loop Prediction . . . . .	245
4.2.1.2	ZNF143 Loop Prediction . . . . .	247
4.2.2	Loop Visualisation Evaluation . . . . .	248
4.2.2.1	Chromatin Interaction Experiment Selection . . . . .	248
4.2.2.2	Source Data . . . . .	249
4.2.2.3	Graph Generation Algorithm Evaluation . . . . .	249
4.2.2.4	Case Study Graph Evaluation . . . . .	251
4.2.2.5	Comparison with software ecosystem . . . . .	252
4.2.2.6	Future Work . . . . .	254
4.2.2.7	Combined REMEDY and LOOPER Opportunities . . . . .	256
4.2.2.8	Research Impact Summary . . . . .	257
	<b>References</b>	<b>259</b>
	<b>Appendix A Hardware Specification</b>	<b>280</b>
	<b>Appendix B Software and Source Code</b>	<b>281</b>



# List of Figures

- 1.1 Simplified structure of Deoxyribonucleic acid (DNA). Pairs of nucleotide units bound by hydrogen bonds are assembled in a polymer using sugar-phosphate bonds. The hydrogen bonds are often visualised as the rungs of a DNA ladder with the sides corresponding to the backbone. The ladder configuration is twisted to form a double helix. . . . . 43
  
- 1.2 The chemical structure of Adenosine and Thymine showing the hydrogen bonding that forms between them when they are base paired in a molecule of DNA. The 'R' groups show the attachment point of the DNA backbone. . . 44
  
- 1.3 The chemical formulas of Guanine and Cytosine showing the hydrogen bonding that forms between them when they are base paired in a molecule of DNA. The 'R' groups show the attachment point of the DNA backbone. . . 44
  
- 1.4 The chemical structure of the DNA backbone. Alternating deoxyribose and phosphoric acid molecules are linked by phosphodiester bonds in a chain. The bonds are made on the 3rd and 5th carbon atoms of the sugar (numbers in red). The chain has directionality; the 5' end of the chain has a free phosphate group and the 3' end has a free -OH group. Reading from 5' to 3' would travel from top to bottom, and 3' to 5' from bottom to top. The 'R' groups show the attachment point of the nitrogen bases to the backbone. . . 45
  
- 1.5 The central dogma of molecular biology. DNA is transcribed to Messenger RNA (mRNA) which is then translated to protein. . . . . 47
  
- 1.6 The simplified structure of a gene showing a typical layout of the major parts. 48
  
- 1.7 DNA packing into chromatin. DNA wraps around a histone core to form a nucleosome. Nucleosomes have a 'beads on a string' configuration where DNA is accessible to proteins between the nucleosomes. DNA can be further packed into 30nm chromatin fibre where the nucleosomes are compressed in a zig-zag formation and held by further scaffolding proteins . . . 51

1.8	A genomic region shown before CCCTC-binding factor (CTCF)/Cohesin loop extrusion begins. For simplification, we only show a promoter (red) and enhancer (orange). The enhancer is too far away from the promoter to have any effect. Convergent CTCF sites with bound CTCF protein (brown) flank the region. . . . .	57
1.9	A pinched loop of DNA bound by a CTCF/Cohesin complex. Cohesin is made from the sub-components SMC1, SMC3, SA 1/2 and RAD21. The core components bind CTCF while the SMC1 and SMC3 form handcuffs around DNA that can slide along the molecule freely. The Loop extrusion theory hypothesises that cohesin binds around DNA in its linear configuration before extruding a loop through the handcuffs until it encounters a pair of convergent CTCF proteins bound to DNA where the loop formation then fixes. Inside the loop, genes and regulatory regions are in closer contact to each other and thus can form contacts. . . . .	58
1.10	A CTCF/Cohesin loop with a gene shown at the top of the loop and a regulatory region at the base of the loop. The gene region although in the loop is still not proximal to the regulatory region at the base of the loop and thus the enhancers cannot act on the promoter. . . . .	59
1.11	In a functional chromatin loop, Zinc-finger protein 143 (ZNF143) binds to the gene promoter and the regulatory region at the base of the loop; consequently top of the loop is 'pulled' down towards the loop anchor. The gene promoter is now in close proximity to the regulatory region at the base of the loop. . . . .	59
1.12	Transitions are interchanges purines or of pyrimidines: they therefore involve bases of similar structure. Transversions are interchanges of purine for pyrimidine bases, which involve the change of differently structured molecules. . . . .	66
1.13	Methods of linearly orientating DNA. DNA can be untwisted and shown as a ladder-like, flat surface; the base pairs then correspond to the rungs on the ladder. Displaying DNA in this way does not solve the orientation problem as the upper and lower strands are mirrors of each other with respect to the 5' and 3' terminators (they are not mirrors in terms of sequence). Chromosomes can be orientated as they have a short arm and a long arm that switch at the centromere, allowing for determinate orientation. . . . .	72



1.14	Sequence walking method for determining strand when given an ambiguous SNP in the TOP/BOT strand designation scheme. Sequence pairs are considered moving outwards until an unambiguous pair is found. . . . .	77
1.15	The Illumina BeadChip genotyping process. Oligonucleotide probes are designed so that they match the immediate preceding sequence of a variant, and are attached to beads mounted on silicon. DNA fragments hybridise to the probes as they are passed over in solution. The probability of hybridisation is related to the size of the matching sequence overlap between the probe and the fragment. The diagram shows an overhang of just one nucleotide but, in reality, the overhang is variable in length as the fragmentation process produces random sized sequences. The probes are then extended using fluorescent tagged nucleotide that are the complement to the next letter in the fragment sequence; this effectively reads the target variant due to the design of the probe. The tags can then be read by an imaging machine to determine the genotype of the variant. . . . .	79
1.16	Example of a Illumina matrix report. . . . .	82
1.17	Example of an Illumina final report. . . . .	82
1.18	Example of a graph of force-directed nodes. The nodes are placed so that none overlap and the edge lengths are equal as much as possible taking into account all other graph generation parameters. . . . .	91
2.1	Example data pipeline for a simple Next generation sequencing (NGS) workflow. The final result required is defined as a variant report for the input raw sequencing data. A reference genome and a local copy of The single nucleotide polymorphism database (dbSNP) were also defined as a requirement to satisfy the results. An interim alignment report was also required to check sequencing quality. The three primary steps of alignment, compression and variant calling were next defined along with the programs that would be run on each node. The edges were then filled in to link the nodes and estimations of redundant and temporary storage were made for each edge. The resultant pipeline shows that a single high performance Linux server is required that is attached to a Redundant array of inexpensive disks configuration (RAID) array with at least 60TB of space and a non-RAID capacity of at least 10TB. . . . .	99

2.2	<b>Top:</b> Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS with no initial processing. The plot shows a significant deviation from the expected at all levels of significance, indicating a systemic issue with the data.	
	<b>Bottom:</b> Manhattan plot showing association scores for our Steroid-sensitive nephrotic syndrome (SSNS) Genome-wide association study (GWAS). The red line indicates a rough calculation of genome-side significance. The plot shows high levels of noise present in the plot as large numbers of markers are above the genome-side significance level, indicating the something was wrong with the data merging and filtering stages of the GWAS pipeline. . . .	109
2.3	Screenshot of an Illumina manifest file showing the IllumStrand and Allele columns (red box) that we used to construct the first version of the transcoding algorithm. . . . .	112
2.4	<b>Top:</b> Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS after application of the first implementation of REMEDY. The plot shows an improved signal to noise with less low-level p-values at the extreme top right of the plot. There is still a significant deviation from the expected at higher p-value levels however.	
	<b>Bottom:</b> Manhattan plot showing the results of the first implementation of our transcoding algorithm. Although the noise levels are reduced when compared to Figure 2.2, p.109, it is still significant. . . . .	113
2.5	Screen shot from an online web-page [190] detailing critical information regarding the format of the probe ID in an Illumina manifest file. The text highlighted in green shows a calculated strand encoding for FWD/REV and TOP/BOT, enabling transcoding between the two using the probe DESIGN designation as a 'middle man'. . . . .	114

- 2.6 **Top:** Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS after application of the final implementation of REMEDY. The plot shows that number of observed p-values over the genome-wide threshold has dropped significantly to the point where the trend follows the expected line for much longer.
- Bottom:** Manhattan plot showing the results of the final implementation of our transcoding algorithm. The signal to noise ratio has improved to the point where all noise is pushed below genome-wide significance, leaving the real association signal intact. . . . . 116
- 2.7 The ChIP-Seq process. Protein bound DNA is cross-linked and then sonicated to create fragments. Antibodies which target the protein of interest (shown in green) are then used to precipitate the fragments out of solution. The proteins and antibodies are then washed away before the fragments are sequenced. After sequencing, the fragments are aligned on the genome and a statistical curve is drawn over the fragment overlap to give an area of predicted binding. The depth of fragments collected at a site is proportional to the final confidence value of the binding site. . . . . 127
- 2.8 Web logo for the ELK4 transcription factor in HEK293 cells reproduced from Factorbook data [212]. The positions visualise the relative nucleotide probabilities shown in the Position weight matrices (PWM) in (Table 2.4, p.126). . 128
- 3.1 Flow diagram of the Re-coding For Merging of Genotyping Data (REMEDY) program design. Sub-programs are shown in red, reports and data in blue and the REMEDY-DB database in yellow. . . . . 135
- 3.2 **Top:** Graphic representing how REMEDY-DB stores variant data. A 4-byte integer defines the length of the payload in bytes immediately after. The payload contains all the data for a variant encoded in JavaScript object notation (JSON) and are variable length. **Bottom:** Variant payloads are stored sequentially in the data file, the start position of the payload length integers are stored in a separate index file by 4-byte integers and are fixed length. The index file can be used to read variants from the data file. . . . . 137

3.3	Graphic representing the method by which rsID indexes are arranged into a binary tree. For each set of indexes, the median value is found and set as the next level in the tree. The remaining items are split in half and the process is repeated for each subset until none are left. . . . .	138
3.4	Logging output for the manifest scan sub-program of REMEDY. . . . .	146
3.5	Logging output for the genotype scan sub-program of REMEDY. . . . .	149
3.6	Logging output for the re-coding sub-program of REMEDY. . . . .	153
3.7	Internal Sample ID scheme for our SSNS GWAS. . . . .	159
3.8	A generalised pipeline design for a GWAS using REMEDY with genotyping input data. . . . .	161
3.9	Full SSNS GWAS pipeline data graph. . . . .	163
3.10	Principal component analysis (PCA) of our SSNS combined cases and controls after filtering outliers of more than three standard deviations. We plot the first and second principle components against each other. We include the full Illumina ethnicity controls in the input data for the PCA calculation so the ethnicity clustering is more clearly shown. The Asian cluster in the top right in light green and the African cluster in the bottom right have no additional sample dots. The European cluster in black is hidden by the sample dots from our case control set. The plot shows our case/control samples are clustered in the European sector indicating that they are all European in genetic origin. . . . .	178
3.11	<b>Top:</b> Manhattan plot showing the original noisy plot with many artefact's. REMEDY was primarily responsible for cleaning and merging the data so that a clear plot could be generated (shown in the bottom diagram). <b>Bottom:</b> Manhattan plot of a basic allele test (BAT) performed on our case/control SSNS dataset. Each plot point shows the BAT calculated for one variant. The X axis shows the genomic position of the SNP and the Y axis shows the p-value of the BAT. The smaller p-values show higher on the graph. The plot shows several large peaks in the Human leukocyte antigen (HLA) region on chromosome 6 and some smaller significant peaks in other regions. The graph is clean with a high signal to noise ratio indicating that the cleaning and Quality control (QC) was done correctly. . . . .	180

3.12	Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS. The plot shows an improved Q-Q plot that only deviates away from the expected at lower p-values. There is still a significant deviation away from the expected at high p-values as the signal on chromosome 6 is so strong. . . . .	181
3.13	LOOPER loop prediction program workflow. . . . .	184
3.14	LOOPER loop visualisation program workflow. . . . .	190
3.15	Consensus binding motif for CTCF from [240]. The sequence is 20 base pairs long and contains a GCCCCCT rich central motif combined with a secondary GTGT motif with a one base pair gap. The overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position. . . . .	197
3.16	MEME-ChIP web result output for CTCF run on 2012 ENCODE data release from the UCSC genome browser. The output shows three motifs with their calculated web-logos and p-values (third column). The bottom motif is a subset of the larger motifs and so was discounted. We selected the central motif as our final result as it had the highest p-value. We compared the 2012 motif with the previously calculated consensus motif from 2011 and found there to be no difference. . . . .	198
3.17	CTCF binding motif calculated from 2011 ENCODE release data used in [240]. Chromatin immunoprecipitation with next generation sequencing (ChIP-Seq) data was run through a custom pipeline that selected and transformed data for input into MEME-ChIP. The sequence above was the consensus motif from the MEME-ChIP output. Overall letter stack height indicates the sequence conservation at that position, while the height of the symbol within the stack shows the relative frequency of each base at that position. The motif contains the same two-part binding motif as the consensus motif and has no other extra motifs; therefore, they can be considered to be equivalent. When using the consensus motif as a positive control, this indicates that the custom pipeline was functioning correctly. . . . .	198

- 3.18 **Top:** Calculated CTCF binding motif based on 2012 ENCODE data selected for use in our loop prediction pipeline. The sequence is 18 base pairs long and contains a two-part binding motif. **Bottom:** Reverse complement of the original sequence which was used to detect binding orientation for CTCF.
- Overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position. . . . . 199
- 3.19 ZNF143 binding motif calculated from 2012 ENCODE release data. Overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position. The motif is 22 base pairs long and contains several islands. . . . . 200
- 3.20 A 3.2Mb region on chromosome 16 displayed in the UCSC genome browser, with the targeted region of interest centred inside the red box. The genomic position is shown at the top, genes are displayed in the centre. The three tracks shown at the bottom are visual representations of the genome segmentation ChromHMM project data; shown are predicted areas of euchromatin (green), heterochromatin (grey), promotor regions (red), insulator regions (blue) and enhancer/regulatory regions (yellow). The three lines represent data from three different cell lines for which this track has been calculated for (GM12878, H1-hESC and K562) [205]. The gene rich area in the centre is clearly flanked by regions of gene depleted, closed chromatin. The red box represents the zoomed in area shown in (Figure 3.21, p.201) . 201
- 3.21 A 600Kb region centred on the Hyperinsulinism with polycystic kidney disease (HIPKD) Region of interest (ROI). *Phosphomannomutase 2 (PMM2)* is shown in the centre with *Transmembrane Protein 186 (TMEM186)* (the other half of the bidirectional promoter for *PMM2*) shown to the left. There are finer grained areas of open and closed chromatin located within some of the genes. The red box represents the zoomed in area shown in (Figure 3.22, p.201). . . . . 201

- 3.22 A 200Kb genomic region showing a four-gene local cluster with *PMM2* and *TMEM186* flanking a bidirectional promotor whilst themselves being flanked by 4-Aminobutyrate Aminotransferase (*ABAT*) and Calcium Regulated Heat Stable Protein 1 (*CARHSP1*). Promoter regions (red bands) are clearly shown for all genes with some insulator regions (blue) flanking the whole region. Some enhancer activity (yellow) can also be seen towards the periphery of the region . . . . . 201
- 3.23 UCSC genome browser view of the bidirectional promotor containing the HIPKD mutation. The top row shows the location of the one base mutation while the second row shows the predicted ZNF143 sites from Motif alignment search tool (MAST). The third row shows the transcription start regions of both *PMM2* (left) and *TMEM186* (right). The fourth row shows binding site prediction for ZNF143; the black region representing strong ZNF143 ChIP-Seq binding while the green area shows the predicted strongest binding site. 203
- 3.24 UCSC genome browser view of three CTCF sites located in an inter-geneic region to the left of the *ABAT* promoter. All three sites have a strong ChIP-Seq signature (black boxes) with predicted binding sites that lie within the ChIP-Seq block. The ChromHMM track shows the Transcription factor binding site (TFBS) as predicted insulators. The sites are orientated from left to right as 3', 5' and 3' respectively. . . . . 204
- 3.25 UCSC genome browser view of three CTCF sites located near the *TMEM186* gene. The left most CTCF site was selected as a target loop anchor while the other two were discarded. The centre motif was only partly well-conserved and had no predicted motif while the right most was cell group specific and relatively weak. . . . . 204
- 3.26 UCSC genome browser view of four CTCF sites located to the right of the *CARHSP1* promoter region. The left two sites both orientated to the 3' strand, the third site has no predicted binding motif and the right most site is 5' orientated. . . . . 204
- 3.27 UCSC genome browser view of ZNF143 sites to the left of the *ABAT* promoter region. We additionally show the CTCF sites identified in the previous section. We show several weak ZNF143 sites with no binding motif, two of which overlap with previously identified CTCF sites. . . . . 205

3.28 UCSC genome browser view of ZNF143 sites to the right of <i>CARHSP1</i> . We show the CTCF sites identified in the previous section and a single weak ZNF143 site overlapping a CTCF insulator with an off-centre predicted binding motif. . . . .	205
3.29 UCSC genome browser view of ZNF143 sites located near the <i>TMEM186</i> gene. We show the CTCF sites identified in the previous section and two 'residue' style sites, one overlapping a CTCF insulator site and one with in the <i>PMM2</i> promoter region. The third site shows a strong ChIP-Seq signature with a binding motif; this was confirmed to be the TFBS identified in our functional analysis overlapping with the putative mutation. . . . .	206
3.30 UCSC genome browser view of cohesin sites to the left of the <i>ABAT</i> promoter region. We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that the left most CTCF had only weak RAD21 binding when compared to the other two sites which have strong SMC3 and RAD21 cohesin ChIP-Seq marks. . . . .	206
3.31 UCSC genome browser view of cohesin sites located near the <i>TMEM186</i> gene. We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that our target CTCF site in the region has strong SMC3 and RAD21 ChIP-Seq marks indicating the presence of cohesin.	207
3.32 UCSC genome browser view of cohesin sites sites to the right of <i>CARHSP1</i> . We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that the left most site has a medium CTCF and cohesin signature while the centre two CTCF sites have a weak cohesin mark. The right most site has strong SMC3 and RAD21 ChIP-Seq marks indicating the presence of cohesin. . . . .	207
3.33 LOOPER chromatin loop prediction results visualised as a bed track in UCSC browser. The potential loops are shown with IDs at the top as black blocks showing the loop domain; the loop anchors for each loop are located at the ends of each block. Below we show the gene and genome segmentation tracks for localisation. . . . .	208
3.34 UCSC browser diagram showing the LOOPER predicted loop number 14 (out best loop candidate. We overlay the Hi-C interaction experimentally detected from Belton <i>et al</i> , lending confirmation that LOOPER does indeed predict valid chromatin loops. . . . .	208



- 3.35 Diagram of the predicted Topologically-associated domain (TAD) around *PMM2* in the K562 cell line. The blue lines demark the TAD boundaries. Each red square has a red scale in proportion to the number of Hi-C contacts present at the intersection between the two sites on the x-axis. We propose that the actual TAD surrounding *PMM2* comprises both the primary TAD and the small TAD adjacent to the left. . . . . 211
- 3.36 LOOPER Graph output for the *PMM2* TAD region. The nodes represent points of contact from Chromatin interaction analysis by paired-end tag sequencing (ChIA-PET) data, the edges show which contact points were in contact with each other. The edge colour shows blue for CTCF ChIA-PET and red for POLR2A ChIA-PET. The edge thickness shows the relative interaction strength of the contact. The node colour shows blue for insulators, red for promoters, yellow for enhancers and orange for regions which contain both promoters and enhancers. The green dotted arcs represent nodes which are physically close to each other on the genome. The node labels show the contact site type, the ID and the gene promoters which are in proximity to the contact point. . . . . 212
- 3.37 Target region for LOOPER graph analysis shown in the UCSC browser. . . 212
- 3.38 A Highlighted section of the LOOPER analysis graph. **(a)**, *PMM2* promoter node - has proximal connection to another contact site (green dotted line), one CTCF ChIA-PET contact (blue line) and one POLR2A ChIA-PET contact (red line). UCSC browser of **(a)** shows the *PMM2/TMEM186* bidirectional promoter that overlaps with the ZNF143 ChIP-Seq block which contains the HIPKD mutation. The genome segmentation track shows red for promoter. There is a strong, well conserved POLR2A signature, indicating that the region is transcriptionally active. Refer to Figure 3.37, p.212 for positional context of **(a)** within the TAD. . . . . 213
- 3.39 A Highlighted section of the LOOPER analysis graph. **(a)**, highly connected loop anchor to the left of *Transmembrane Protein 114* (*TMEM114*) (CTCF\_11263 highlighted in blue). Strong evidence for contact with the *PMM2* promoter through the graph connection and indirect binding of ZNF143 and POLR2A. The site has several enhancers downstream in the intronic regions of *TMEM114*. Refer to Figure 3.37, p.212 for positional context of **(a)** within the TAD. . . . 215

- 3.40 A Highlighted section of the LOOPER analysis graph. **(a)**, a loop anchor situated in an intron of *Methyltransferase Like 22 (METTL22)* with a strong connection to the enhancer cluster at the left TAD boundary (highlighted in blue). Evidence for indirect ZNF143 and POLR2A binding is present. **(b)**, a loop anchor situated on the right TAD boundary (highlighted in blue). Evidence for indirect ZNF143 and POLR2A is present. Refer to Figure 3.37, p.212 for positional context of **(a)** and **(b)** within the TAD. . . . . 216
- 3.41 UCSC browser capture of the CTCF\_11264\_[TMEM114]. The node region is highlighted in blue. We show that the site is a loop anchor with several predicted enhancers clustered around the *TMEM114* region. The site has evidence for indirect binding of ZNF143 and is located near the promoter for *TMEM114*. . . . . 217
- 3.42 A Highlighted section of the LOOPER analysis graph. **(a)**, a loop anchor situated to the left of *TMEM186* (highlighted in blue), it is the closest anchor to the *PMM2* promoter. Evidence for indirect ZNF143 binding is present. **(b)**, a loop anchor situated to the left of *ABAT* (highlighted in blue). Evidence for indirect ZNF143 binding is present. Refer to Figure 3.37, p.212 for positional context of **(a)** and **(b)** within the TAD. . . . . 218
- 3.43 A Highlighted section of the LOOPER analysis graph. Four nodes are physically located in the same region while CTCF\_11272 and CTCF\_11270 have connectivity to the regulatory hub at CTCF\_11263. *CARHSP1* is shown with all splice variants in the UCSC browser. Two connected loop anchors: CTCF\_11272 (blue highlight) and CTCF\_11270 (purple highlight) are located at either end of the diagram. POLR2A\_3701\_[*CARHSP1*] is located in the promoter region for *CARHSP1* (red highlight) with the enhancer CTCF\_11271[*CARHSP1*] located to the right (yellow highlight). Refer to Figure 3.37, p.212 for positional context of *CARHSP1* within the TAD. . . . . 219
- 3.44 UCSC browser capture of the POLR2A\_3698 node. The site is located at the extreme end of the LHS TAD boundary and shows a predicted enhancer element with several strong POLR2A signatures suggesting promoter contact.220
- 3.45 UCSC browser capture of the POLR2A\_3702\_[*USP7*]. The site is located at the *Ubiquitin Specific Peptidase 7 (USP7)* promoter which shows a cluster of regulatory elements including CTCF signatures and several enhancers. . 220

3.46 3DIV capture of the <i>PMM2</i> TAD in a liver tissue sample. We show that the TAD structure is identical to one generate for K562 presented earlier (Figure 3.35, p.211). The TAD shows one liver specific super-enhancer: 'LI_superEnhancer_191'.	223
3.47 3DIV capture of the <i>PMM2</i> TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. We show strong contacts with the TAD boundaries with previously identified regulatory regions.	224
3.48 3DIV capture of the <i>PMM2</i> TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. We show strong contacts with the TAD boundaries with previously identified regulatory regions and with loop anchors in intergenic reigons.	225
3.49 3DIV capture of the <i>PMM2</i> TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. Show contacts with many of the regulatory elements show using ChIA-PET data in LOOPER including a liver specific super enhancer.	226

# List of Tables

1.1	Strand and allele designations for unambiguous SNPs in the TOP/BOT strand designation scheme. . . . .	76
1.2	Contingency table for case/control status versus major/minor allele . . . . .	86
2.1	A Summary of the case-control sets identified for the SSNS GWAS project.	108
2.2	Conversion table for initial strand designation transcoding algorithm shown for converting from TOP/BOT to FWD/REV. The reverse column refers to reversing the alleles in the Allele column shown in Figure 2.3, p.112. . . . .	112
2.3	Conversion table for final strand designation transcoding algorithm. The table shows enumerations of source and destination encoding for different probe strand designations. The last columns show the route of conversion from source to design and then from design to destination. In this table, we use the word 'Flip' as a short version of complement. To reverse from TOP/BOT back to FWD/REV the conversion schemes should simply be reversed. . . . .	115
2.4	Probability weight matrix for the ELK4 transcription factor in HEK293 cells reproduced from Factorbook data [212]. Rows represent the sequence position and the columns show the probability of each nucleotide at a given position. . . . .	126
3.1	Control data subsets for the SSNS GWAS. The table shows the range of formats and encoding schemes encountered. . . . .	160

3.2	Storage predictions for key parts of the SSNS pipeline. these values were calculated as part of the edge analysis stage of our data pipeline methodology. The predictions were based on approximate known ratios between different file formats and on test data. We estimated we needed at least 7.6 TB of working storage to store all data in a pipeline run, but only 1.1 TB of redundant storage for storing the final results. . . . .	168
3.3	List of Illumina microarrays assessed using REMEDY. . . . .	169
3.4	Summary of the core output from the manifest scan stage of the REMEDY toolset when run inside the SSNS GWAS pipeline. The <i>Strict Loci Rem.</i> rows show how many valid variants are left in the dataset after each filter pass. . . . .	172
3.5	Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the Illumina controls dataset. . . . .	173
3.6	Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the Oxford controls dataset. . . . .	173
3.7	Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the Wellcome trust case control consortium (WTCCC) birth controls dataset. . . . .	174
3.8	Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the WTCCC blood controls dataset. . . . .	174
3.9	Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the SSNS cases dataset. . . . .	175
3.10	List of annotation priorities when assessing genome segments which overlap a node. . . . .	192
3.11	List of Hi-C and ChIA-PET visualisation tools with comparison comments from the 4DN software portal. . . . .	222
4.1	REMEDY and Oxford pipeline feature comparison table. . . . .	242
A.1	Test and development hardware specification. . . . .	280

C.1	Full details of the graph nodes generated from the LOOPER visualisation algorithm. . . . .	283
C.2	Full details of the graph links generated from the LOOPER visualisation algorithm. . . . .	284

# Acronyms

**ABAT** 4-Aminobutyrate Aminotransferase. 26, 27, 29, 200–206, 208, 218, 223

**C16orf72** Chromosome 16 Open Reading Frame 72. 210, 223, 224

**CALHM6** Calcium Homeostasis Modulator Family Member 6. 179

**CARHSP1** Calcium Regulated Heat Stable Protein 1. 26, 27, 29, 200–207, 217, 219, 223, 227

**METTL22** Methyltransferase Like 22. 29, 214, 216, 217, 223

**PARM1** Prostate Androgen-Regulated Mucin-Like Protein 1. 179

**PLAR2** Anti-Phospholipase-A2-Receptor. 157

**PMM2** Phosphomannomutase 2. 25–30, 196, 197, 200–203, 205, 206, 209–215, 218, 220, 223–227, 229, 251, 252, 255

**TMEM114** Transmembrane Protein 114. 28, 29, 210, 214, 215, 217, 223, 227

**TMEM186** Transmembrane Protein 186. 25–29, 200–207, 209, 210, 213, 218

**USP7** Ubiquitin Specific Peptidase 7. 29, 210, 220, 224

**API** Application programming interface. 98, 237

**ATP** Adenosine triphosphate. 50

**BCF** Binary calling format. 152–154, 237

**BED** Browser extensible data. 126, 187, 188, 207

**BLAST** Basic local alignment search tool. 74, 75

**BLAT** Blast-like alignment tool. 144, 240, 241, 243

**ChIA-PET** Chromatin interaction analysis by paired-end tag sequencing. 28, 30, 129, 130, 182, 188, 189, 192, 209, 210, 212, 213, 217, 223, 226, 227, 229, 248, 249, 251, 255

**ChIP** Chromatin immunoprecipitation. 123, 124

**ChIP-Seq** Chromatin immunoprecipitation with next generation sequencing. 24, 26–28, 124, 125, 129, 182, 183, 185–187, 191, 197–199, 202–207, 209, 213, 220, 227, 229, 245–249

**CNV** Copy number variation. 67, 231

**CRC** Chromatin remodelling complexes. 50

**CTCF** CCCTC-binding factor. 19, 24–29, 53–59, 182, 183, 185–187, 189, 197–199, 202–207, 209, 210, 212–214, 217, 220, 223, 227, 229, 245–249, 251

**dbGAP** Database of genotypes and phenotypes. 106

**dbSNP** The single nucleotide polymorphism database. 20, 73–78, 81, 83, 99, 103, 104, 110, 111, 117, 132–134, 136, 142–145, 152, 154, 164, 230, 233, 234, 237–239, 241–244

**DDBJ** DNA data bank of Japan. 71

**DNA** Deoxyribonucleic acid. 18–20, 22, 39, 42–58, 60–62, 65, 67–74, 78–80, 83, 89, 94–96, 101, 103, 111, 123–125, 127, 129, 157, 159, 183, 185–187, 199, 205, 234, 245, 246

**EGA** European genome archive. 106, 108, 229

**EMBL** European molecular biology laboratory. 71

**EMSA** Electrophoretic mobility shift assay. 196

**GDPR** General data protection regulation. 94

**GUI** Graphical user interface. 93

**GWAS** Genome-wide association study. 21, 23, 31, 32, 85–87, 89, 100–110, 114, 117–119, 132, 133, 142–145, 149, 155–161, 163, 165, 166, 171–177, 179, 228–234, 236, 240, 242–244

**HIPKD** Hyperinsulinism with polycystic kidney disease. 25, 26, 28, 30, 122, 182, 187, 194, 196, 198, 201, 203, 205, 213, 217, 222–227, 229, 245–251, 254, 255

**HLA** Human leukocyte antigen. 23, 157, 179, 180

**HMM** Hidden markov model. 124

**HWE** Hardy-Weinberg equilibrium. 88, 105, 165, 167, 176, 177



**IBD** Identity by descent. 165, 167, 176

**JIT** Just in time. 119

**JSON** JavaScript object notation. 22, 136, 137, 238, 239

**LAD** Lamina-associated domain. 55

**LD** Linkage disequilibrium. 87, 89

**MAF** Minor allele frequency. 88, 167, 176, 177

**MAST** Motif alignment search tool. 26, 125, 126, 185, 187, 202, 203, 207

**MN** Membranous nephropathy. 157

**mRNA** Messenger RNA. 18, 46–49, 52, 125

**NCBI** National centre for biotechnology information. 71, 72, 237, 239, 244

**NGS** Next generation sequencing. 20, 99, 124

**OOD** Object-oriented design. 119, 122

**PAR** Pseudoautosomal region. 142, 232

**PCA** Principal component analysis. 23, 88, 89, 105, 158, 166, 177, 178

**POL2** DNA Polymerase II. 49, 249

**PWM** Position weight matrices. 22, 124–126, 128, 183, 185, 186, 197, 200, 202, 203, 246, 247

**QC** Quality control. 23, 166, 167, 176, 177, 180, 230

**RAID** Redundant array of inexpensive disks configuration. 20, 94, 99

**REMEDY** Re-coding For Merging of Genotyping Data. 22, 23, 32, 100, 101, 103, 107, 118, 119, 121–123, 131, 133–144, 146, 148, 149, 151–156, 161, 164–166, 168, 169, 171–176, 179, 182, 189, 228–244

**RNA** Ribonucleic acid. 46, 49, 69, 125

**ROI** Region of interest. 25, 84, 157, 186–189, 194, 195, 201, 205, 209, 210, 222, 229, 246–251, 255

**SNV** Single nucleotide variant. 65–67, 73, 74, 76, 83, 86–89, 107, 110, 112, 117, 118, 142, 143, 165–167, 171, 176, 230

**SSNS** Steroid-sensitive nephrotic syndrome. 21, 23, 31, 32, 100, 107–109, 155–157, 159, 160, 163, 164, 168, 172–176, 178, 180, 228–230, 238, 240

**SVS** SNP & Variation Suite. 108, 166–168, 176–178, 236, 237

**TAD** Topologically-associated domain. 28–30, 55–57, 191, 200, 210–220, 222–227, 245, 251, 255

**TF** Transcription factor. 49, 50, 52, 53, 55, 123–126, 129, 183, 185–189, 192, 197, 200, 209, 220, 246

**TFBS** Transcription factor binding site. 26, 27, 52, 56, 124–126, 129, 185, 186, 199, 200, 204, 206, 207, 209, 214, 227, 245, 246

**tRNA** Transfer RNA. 69

**TSS** Transcription start site. 49

**UTR** Untranslated region. 48, 49, 52

**VCF** Variant calling format. 133, 134, 149, 152–154, 165, 167, 176, 236, 237, 243

**WGS** Whole genome sequencing. 78, 89, 231, 244

**WTCCC** Wellcome trust case control consortium. 32, 159, 160, 164, 174, 176

**YY1** Yin-yang 1. 57, 248, 255

**ZNF143** Zinc-finger protein 143. 19, 25–29, 57, 59, 182, 187, 196, 197, 199, 200, 202, 203, 205–207, 209, 213–218, 223, 227, 229, 245, 247, 249, 251, 255

# Chapter 1

## Introduction

In this thesis, the development of two software tools is presented that were created in response to problems that were encountered while performing genomic analysis during the course of several projects. In this section, the concept of genomic analysis is introduced and why computer science and technology are critical to this field. I then justify how our developed software tools fit into the existing software ecosystem for genomic analysis. Next, some fundamental theory and approaches to genetic analysis are discussed before describing further in-depth theory regarding those concepts that are key to understanding the operation of both software solutions.

### 1.1 Genomic Analysis

The study of genetics is concerned with one overarching goal: to map an organism's genotype to its phenotype. The genotype of an organism could be defined as a set of genetic measurements which alter an observable set of characteristics or traits otherwise known as a phenotype. The problem of genetics could be said to be solved when the phenotype that will arise from any genotype can be determined. The field of genetics exists because there is no exact mapping of genotype to phenotype. One must, therefore, infer genotype to phenotype relationships while simultaneously attempting to improve the model of the mappings to make future inferences more accurate.

The foundations of modern genetic analysis have been practised since prehistoric times. Early humans used selective breeding (and continue to do so to this day) to select for desirable traits in both crops and animals [3] [4]. It is also highly probable that early humans understood that certain visible traits such as hair and eye colour were transmitted

to their children [5].

The foundation of classical genetics began in the mid-1800s with "the father of genetic analysis", Gregor Mendel. Mendel's experiments on pea plants led to the development of Mendel's laws of inheritance which set the foundation of our current understanding of genetic inheritance and provided a framework for genetic analysis based within the scientific method [6]. Further experiments on inheritance by scientists such as Thomas Morgan [7] [8] and others refined the practice of quantitative genetic analysis using breeding as the primary study mechanism.

The advent of biochemistry enabled a more fine-grained study of the biological mechanisms involved in genetics. As the volume of biological data grew exponentially, computing methods and data storage methods such as databases were applied to biological data, thus creating the field of bioinformatics. Today the term genetic analysis applies both to the methods used for genetic research and to processes which use existing models such as in genetic diagnostic testing for diseases.

Bioinformatics is loosely defined as the application of software tools, databases and data interfaces to study biological data. Genomics is defined as a field which looks at the genome as a whole rather than the gene-centric approach of genetics. Modern genomics often has a significant bioinformatic component given the large volume of data created by genomic experiments such as whole genome sequencing. The bioinformatic portion of genomic analysis utilises software tools, data interfaces, algorithms and sources of genetic data to build *in silico* models of biological systems which can be used to generate predictions.

### **1.1.1 Genomic Analysis Methodology**

Bioinformatic genomic analysis ultimately starts with a biological sample set, which includes genetic information in the form of the molecule Deoxyribonucleic acid (DNA). The samples must first be converted into digital data so that they can be acted upon using software tools. Given that the source samples are biological, bioinformatic source data is inherently noisy and must undergo several stages of quality control and filtering depending on the project specification before it is ready for further analysis. The core genomic data-set may also be augmented and annotated with other existing sources of data before the principal experiments are performed that generate results.

Genomic analysis projects ultimately infer links between genotype and phenotype; this is usually achieved by some form of comparative analysis. A comparative analysis takes

data which is split on a dependent variable (e.g. disease status) and conditioned on another set of variables (e.g. ethnicity) to infer some new information about the relationship between the samples. New *in silico* inferences can then be tested by *in vitro* functional analysis experiments that will support the computer model or not. In turn, this may drive further changes to the bioinformatic model, which generate new functional wet-lab experiments until a hypothesis is properly cemented or disproved.

### **1.1.2 Computing in Genomic Analysis**

Early in modern genetic analysis, there was a drought of data; many problems were challenging to solve because of the lack of available experimental information. As experimental science has progressed, the amount of data generated has increased exponentially, requiring the use of computers to process the information in a timely and accurate manner. The volume of data generated by some classes of experiments in genomics has made computing critical to these analyses.

Algorithms, software and databases provide the solutions, storage and processing required for the challenges of genomic analysis. Today, there are two main bottlenecks which drive innovation in the bioinformatic part of genomics. The problem of managing, processing, and analysing such large amounts of data requires evermore innovative solutions to overcome rising storage limitations, long processing times and implementation of complex algorithms. The second driver comes from the requirement for novel software tools to analyse data from new experiments; genetics is a fast-moving field where hardware, techniques to use existing hardware and experimental processes are continually updated.

Reliance on software solutions for genomic analysis has meant that computer scientists are increasingly recruited into a role that could best be described as a genetic bioinformatician. This role is part computer scientist, software engineer, data scientist and geneticist.

#### **1.1.2.1 Databases and Interfaces**

In modern operating systems, data is stored in discrete packages known as files. The term 'file' harks back to the hierarchical ways in which files are stored in folders in a filing cabinet, but in a general sense, files are an encapsulation of data stored in a specific structure; the simplest example being stored values in a sequence. Without any other logic, a file is read in sequence from one value to the next. In order to store structure within data, some logic must be applied to define access to the underlying information. Silberschatz *et al.* (1997) [9] defines a database as an organised collection of data. The logical interface

that sits between a program and the underlying data is known as a data interface. The encapsulation of both data and data interface is known as a database.

Databases define the format in which data is stored, the protocols for access to the data and the interface which connects the query engine with the requesting program or person. As the volume and complexity of data increases, it becomes more challenging to store, write and retrieve information in a consistent time frame. This problem is visible in many different fields but is no more apparent than in genomics, where the explosion of sequencing data has driven innovation in database design, data compression and efficient data access. Database technology, therefore, constitutes a central tenet of genomic analysis.

#### **1.1.2.2 Software Tools**

Bioinformatic data generated by experiments is generally stored in databases or large file repositories. To turn the raw data into results which can be used to test a hypothesis, a significant amount of processing must be performed using algorithms implemented in software and tools designed to streamline the flow and management of data during processing.

The number of software tools available for genetic analysis has increased exponentially in the last 20 years. Initially, many pieces of software were implemented on hardware and were directly connected to a wet-lab experiment. As genomic analysis has become more complicated, it is now necessary to perform layers of meta-analysis on data which originated directly from experimental hardware; this has resulted in the development of many more software tools than hardware systems. The burden of genomic analysis, therefore, has mostly shifted away from the raw data and onto the many layers of post-processing required to obtain a result.

Software is critical to all aspects of genomic analysis, and thus much of the change required when facing one of the key drivers of innovation discussed in the previous section results in a new or updated software tool.

#### **1.1.3 Thesis Justification**

Through several genomic analysis projects conducted by our team, we encountered two situations where the existing methodology and software tools were found to be lacking. We faced extensive difficulties when attempting to merge large, disparate genotyping datasets where tools to deal with the encountered errors did not exist. Arguably, this highlights the first key driver of genomic analysis innovation: volume and complexity of data.

Secondly, we encountered challenges in a project where we lacked the software tools to predict and visualise the mechanism by which a point mutation changed the three-dimensional structure of chromatin. We developed a new software suite which predicts and visualises chromatin loops, which allowed exploration of potential hypothesis for the disease mechanism with greater clarity. We argue this highlights the second key driver of innovation for genomic analysis, where new software tools are required to understand data from new experiments.

This thesis presents our work in both genomic analysis methodology and software technology that we believe furthers the field of genomic analysis. Our work has enabled the delivery of three published research projects to date, that would have been much more difficult without these innovations.

## **1.2 Basic Genetics**

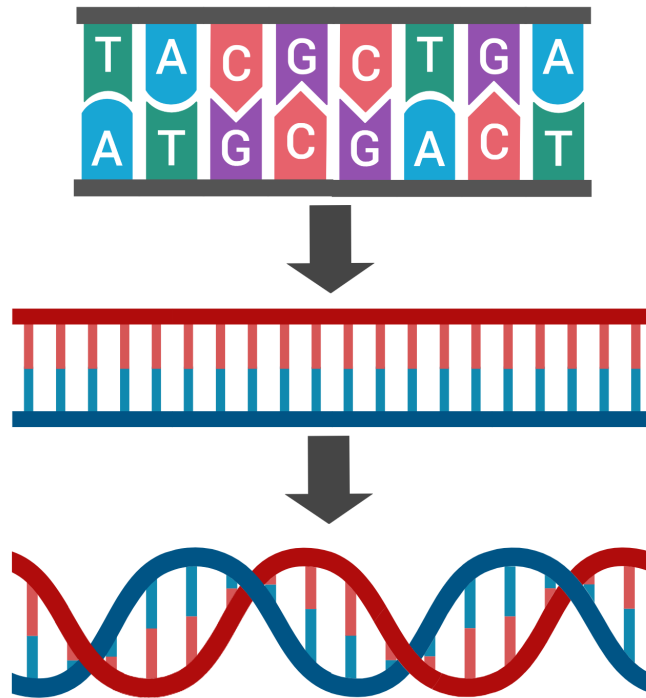
### **1.2.1 Deoxyribonucleic Acid**

DNA is an essential molecule in life that stores all the information needed to create and maintain an entire organism. The human genome is approximately 3.2 billion base pairs long [10] and is composed of 4 bases: Adenosine (A), Cytosine (C), Thymine (T) and Guanine (G); they combine into long polymers in a double helix configuration to form DNA [11] [12] [13].

DNA consists of two strands of polynucleotides bound together with hydrogen bonds in a double helical structure. A single nucleotide consists of a sugar-phosphate unit plus a nitrogen base; the units of a nucleotide bind to the units of another nucleotide to form a polymer chain. The sugar-phosphate chain is known as the backbone; bases bind to the backbone and then form hydrogen bonds with each other that hold the two strands of DNA together in a double helix configuration (Figure 1.1, p.43). The hydrogen bonds between bases exist in a pairwise fashion between specific combinations of letters; A always pairs to T (Figure 1.2, p.44) and C always pairs to G (Figure 1.3, p.44), each pair of letters is known as a base pair [11] [12] [13].

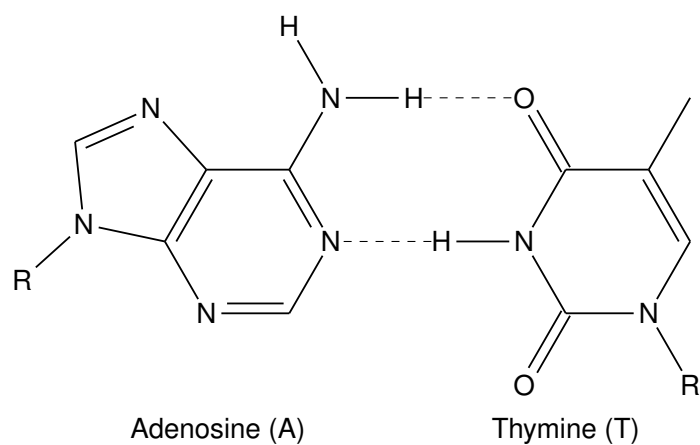
The DNA backbone is formed from alternating deoxyribose and phosphate molecules (Figure 1.4, p.45); the phosphate groups attach to deoxyribose at the third (3') and fifth (5') carbon of the sugar ring [13]. The ends of the backbone will always have an exposed phosphate group attached to the 5' ribose carbon at one end and an -OH substituent at the other end attached to the 3' ribose carbon; thus, one end of a DNA strand is termed the

5' end and the other the 3' (pronounced three-prime and five-prime). New nucleotides can join to either end by forming a phosphodiester bond with the 5' phosphate or the 3' hydroxyl group. DNA is structured such that the two strands are opposite in orientation with respect to the 5' and 3' ends. The direction with regards to DNA can be defined as either 5' to 3' or 3' to 5' and is important in a number of genetic mechanisms discussed in the following sections.

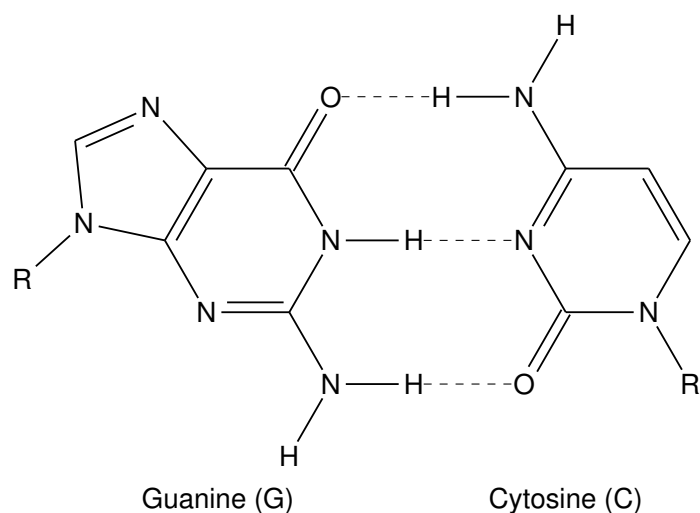


**Figure 1.1:** Simplified structure of DNA. Pairs of nucleotide units bound by hydrogen bonds are assembled in a polymer using sugar-phosphate bonds. The hydrogen bonds are often visualised as the rungs of a DNA ladder with the sides corresponding to the backbone. The ladder configuration is twisted to form a double helix.

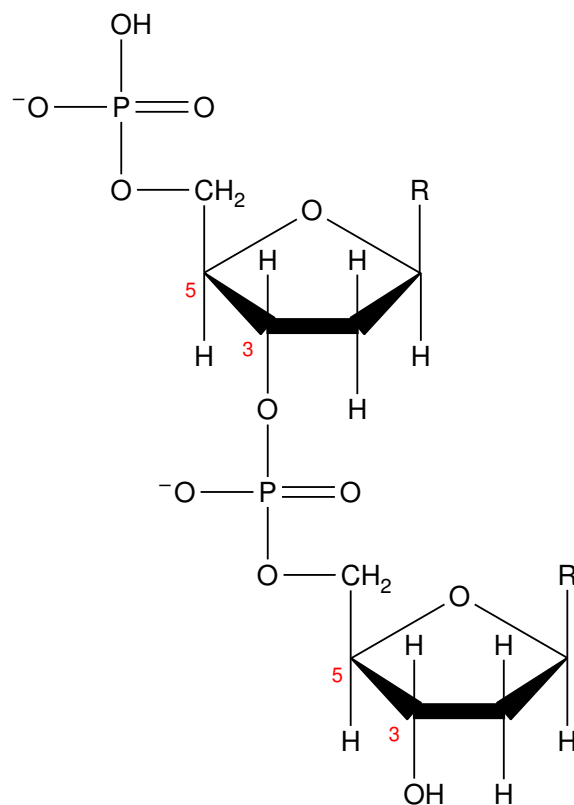




**Figure 1.2:** The chemical structure of Adenosine and Thymine showing the hydrogen bonding that forms between them when they are base paired in a molecule of DNA. The 'R' groups show the attachment point of the DNA backbone.



**Figure 1.3:** The chemical formulas of Guanine and Cytosine showing the hydrogen bonding that forms between them when they are base paired in a molecule of DNA. The 'R' groups show the attachment point of the DNA backbone.



**Figure 1.4:** The chemical structure of the DNA backbone. Alternating deoxyribose and phosphoric acid molecules are linked by phosphodiester bonds in a chain. The bonds are made on the 3rd and 5th carbon atoms of the sugar (numbers in red). The chain has directionality; the 5' end of the chain has a free phosphate group and the 3' end has a free -OH group. Reading from 5' to 3' would travel from top to bottom, and 3' to 5' from bottom to top. The 'R' groups show the attachment point of the nitrogen bases to the backbone.

### **1.2.2 Proteins**

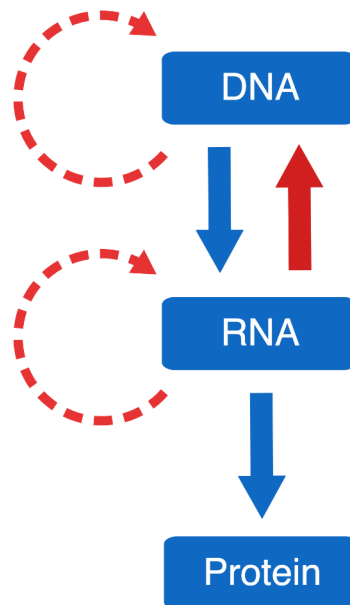
Proteins are large macro-molecules, consisting of one or more chains of amino acids, they perform a vast array of functions in the body and are essential to all life. Proteins were first described as a class of biological molecules in the eighteenth-century [14]. Mulder and Berzelius later characterised them and coined the term 'Protein' in 1838 [15] [16]. Mulder discovered that all proteins had the same empirical chemical formula that contained roughly the same elements in the same proportions. Hoffmeister and Fischer (1902) [17] both independently put forward the theory that proteins were made from chains of polypeptides. The first protein sequence to be identified was Insulin by Sanger in 1951 [18] while the first full structure to be solved was Myoglobin by Kendrew in 1958 [19]. The culmination of these key papers and many others showed that proteins are formed from polypeptides in specific amino acid sequences that fold into multiple levels of structure; therefore, the amino acid sequence defines the structure and subsequent function of a protein.

### **1.2.3 Ribonucleic Acid**

Ribonucleic acid (RNA) is another vital molecule in genetics that is essential for expression, regulation and maintenance of an organism's DNA sequence. It is a polynucleotide that is comprised of a sugar-phosphate backbone and three of the same bases as DNA, Adenosine, Cytosine and Guanine; the fourth base Thymine is substituted for Uracil [20]. RNA also differs from DNA in that it is almost always found in a single strand configuration. DNA is a very stable molecule and can be characterised as a long-term storer of information; RNA is less stable and performs a wide range of functions both in the nucleus and the cytosol. Perhaps RNAs most important function in genetics is its role as messenger RNA or Messenger RNA (mRNA), which is used as an information transfer medium in protein production [21] [22]. The discovery of mRNA crucially linked DNA to ribosomes in the cytosol, completing the route from DNA, to RNA to proteins.

### **1.2.4 The Central Dogma of Molecular Biology**

The central dogma of molecular biology first stated by Crick in 1958 [23], states that information is stored in DNA which is used to make RNA, which in turn is used to create proteins (Figure 1.5, p.47). DNA can be copied to DNA (DNA replication), DNA information can be copied to mRNA (transcription), RNA can be copied to RNA (RNA replication), DNA can be synthesised from RNA (reverse transcription), and proteins can be constructed from mRNA used as a template (translation).



**Figure 1.5:** The central dogma of molecular biology. DNA is transcribed to mRNA which is then translated to protein.

### 1.2.5 DNA as a Sequence

William Cumming discovered Threonine in 1935 [24], the last of the 20 amino acids universally present in proteins to be identified. After the discovery of the structure of DNA in 1953 by Watson and Crick [13], one of the most interesting questions was how the four bases of DNA encoded for the 20 amino acid sequences that constitute proteins.

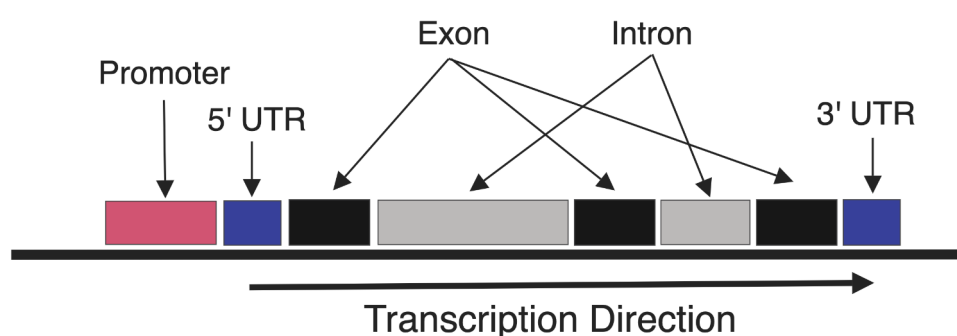
George Gamow first postulated that overlapping sets of three bases could be used to encode for 20 amino acids with a four base encoding; this was possible *via* the diamond shaped grooves formed from the bases on four sides inside the double helical structure of DNA [25]. Known as 'Gamows Diamonds', the overlapping triplet diamonds encoded for exactly 20 amino acids; however, this elegant theory was later proved wrong by Brenner (1957) [26] before Nirenberg and Matthaei (1961) [27] directly proved the specific sequence for the amino acid phenylalanine (TTT). Nirenberg and his team then went on to discover the code for several other amino acids, suggesting that codons were formed from sets of non-overlapping triplets.

### 1.2.6 Genes

A protein's function is determined by the linear sequence of amino acids from which it is composed [28]. Amino acids are encoded in DNA using three-letter sequences called

codons. All the codons for a single protein are encapsulated in a structure known as a gene, first described so by Johannsen in 1905 [29]. The initial structure of the gene was defined by Benzer [30] [31], which has then been refined to our current understanding in subsequent years by many different scientists.

At the codon level, groups of sequential triplets that code for amino acids are called exons. Exons are inter-spaced in a gene by untranslated units known as introns [32]. Sequences flank the alternating set of introns and exons called the 5' and 3' Untranslated region (UTR)s. The 5' UTR precedes a special 'start' codon (ATG), which denotes the start of the open reading frame for translation. The 3' UTR immediately follows a stop codon (TAG, TAA or TGA), another set of special codons which defines the end of the open reading frame. An area important for transcription initiation and regulating the production of mRNA from the gene called the promoter immediately precedes the 5' UTR. The complete set of the promoter, 5' UTR, exons, introns and 3' UTR are collectively known as a gene (Figure 1.6, p.48).



**Figure 1.6:** The simplified structure of a gene showing a typical layout of the major parts.

### 1.2.7 Gene Expression

In order to produce a protein, a blueprint for the sequence of amino acids is required; the blueprint is copied from the DNA sequence for the protein contained within a gene into mRNA in a process called transcription [21] [22]. The mRNA transcript then undergoes a complex process of transformations known as maturation until the final mRNA molecule is translated into a sequence of amino acids by ribosomes in the cytosol [33]. Finally, the string of amino acids produced by a ribosome undergoes a complex process of folding and alteration until the mature protein is released and allowed to perform its function in the cell [34]. A gene can be said to be 'expressed' when it is actively transcribed.

Transcription begins with the binding of specific DNA binding proteins known as Transcription factors (TFs) to the promoter region of a gene. Over 100 proteins bind to form a large complex which culminates in the binding of the enzyme DNA Polymerase II (POL2). POL2 slides along DNA uncoiling the double helix as it goes, moving in the 5' to 3' direction with respect to the gene sequence. The 3' strand (3' to 5' direction DNA strand) is used to create a molecule of mRNA using the corresponding complements of DNA to produce a sequence that directly matches the 5' strand (5' to 3' direction DNA strand) of DNA in RNA; the gene, therefore, is read in the 5' to 3' direction [35]. The 5' strand is known as the sense strand (or positive +) as an RNA version of the same sequence is translated or translatable into protein. The antisense or template strand (or negative -) corresponds to the 3' strand [36]. Sense and antisense strands are relative to the gene and not the underlying DNA strand as genes can be defined on the 5' or 3' strands of DNA and thus can be read in either direction with respect to DNA; however, a gene is always read from 5' to 3' in terms of the sense strand.

During transcription, a gene is read sequentially from the Transcription start site (TSS) in the promoter to the end of the 3' UTR to create an immature mRNA. After transcription, the introns are cut out from the mRNA transcript in a process called splicing. The mRNA may undergo further post-transcriptional modifications before being used in the translation process to create a protein. The 5' UTR contains the binding site for the ribosome to start translation and both the 5' and 3' UTRs are thought to have a strong influence on the post-transcriptional modification process and thus the regulation of protein production [35].

### **1.2.8 Chromatin**

The genome is defined as the total genetic material of an organism [37]. Genetic material in the cell resides in the nucleus and is organised into large single molecules of DNA in structures called chromosomes. If all the DNA in a cell were laid end to end, it would reach over 2 m in length, but the average diameter of the human cell nucleus is just 6 µm. Eukaryotic organisms have developed specific ways of packing DNA into smaller configurations that balance compactness versus accessibility [38].

The nucleosome is the basic unit of DNA packaging in the genome and consists of an octamer core made from proteins known as histones which act as a spool around which DNA wraps approximately 1.5 times. The histone octamer is composed of a dimer of two tetramer histones; one half is formed from the proteins H3 and H4, the other is made from H2A and H2B. The histone core proteins are highly conserved not only in the human genome but across many different species; this suggests that they perform an essential role in nucleosome structure. Each histone core protein has an additional N-terminal 'tail'

element that enables covalent bonding of other proteins, allowing the histone structure to be modified and its function changed [39].

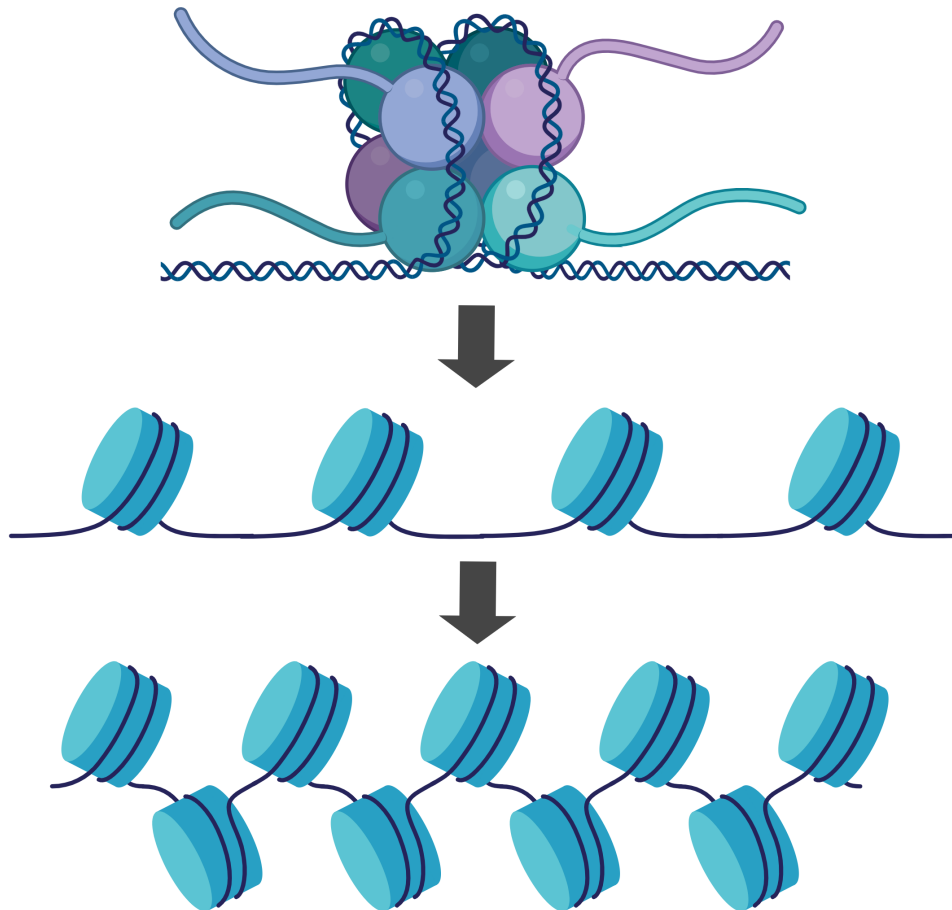
Nucleosomes have a dynamic structure that is targeted by many Adenosine triphosphate (ATP) dependent Chromatin remodelling complexess (CRCs) [40]. In a process known as nucleosome sliding, the DNA wrapped around the histone core can be shifted and moved to 'remodel' the local DNA structure using CRCs. With additional protein complexes known as histone chaperones, histone proteins can be removed from the DNA or replaced with another histone core made from rarer histone core types such as the H2AZ-H2B dimer. The positional change of nucleosomes, the modification of histones and the exchanging of histones types is highly dynamic and warps the local structure of DNA to make it less or more accessible to other protein types [41] [42].

Nucleosomes are linked by open stretches of DNA to form a model like 'beads on a string'. The nucleosomes, linker DNA and associated non-histone proteins that also bind to aid in packaging are collectively known as chromatin. Chromatin itself can be packaged into higher-order structures forming a layered structure similar to the way proteins are hierarchically folded. This multi-level packing, through histone modification and remodelling, is utilised in many areas of cell behaviour including division, differentiation and transcription regulation [43].

Linked nucleosomes can be packed into a more compact form by folding together in a zig-zag formation to make a 30 nm chromatin fibre (Figure 1.7, p.51). In metaphase cells, 30 nm chromatin is packed even tighter using scaffolding proteins into the classical X shaped chromosomes that are visible in light microscopy. This extremely tight configuration makes replication and segregation of genetic material easier [44].

During interphase, the default chromatin configuration is the 30 nm fibre, which is also known as heterochromatin or 'closed' chromatin. The tightly packed configuration of heterochromatin makes it difficult for proteins such as TFs and DNA polymerase II to bind to their target sites; heterochromatin is therefore thought to be transcriptionally inactive [45].

It is possible through histone modifications and other protein-DNA interactions to unpack chromatin back into the 'beads on a string' configuration, this is known as 'open' or euchromatin and allows for active gene transcription to take place. This does not necessarily mean genes in open chromatin are actively transcribed, but it does allow DNA access for TFs [45].



**Figure 1.7:** DNA packing into chromatin. DNA wraps around a histone core to form a nucleosome. Nucleosomes have a 'beads on a string' configuration where DNA is accessible to proteins between the nucleosomes. DNA can be further packed into 30nm chromatin fibre where the nucleosomes are compressed in a zig-zag formation and held by further scaffolding proteins

### 1.2.9 Regulation of Gene Expression

Proteins perform a vast array of functions in an organism. Some must exist at specific levels while others are only produced in distinct tissue types or at particular stages in development. Control of when, where and how much of a protein that is produced is primarily managed through the regulation of gene expression. Correct gene regulation requires the dynamic integration of genetic, cellular and extra-cellular environment information, which signal the correct combinations and levels of proteins that must exist in a cell at any given time.

For regulation to be possible, there must be signal generators, signal receivers and a medium of communication. The medium of communication in cells is primarily *via* protein signalling. Signal generators may be any biological construct, such as a cell membrane receptor or calcium sensor that is capable of triggering a protein 'signal', which is often



a cascade of protein signalling that eventually results in a final protein binding to a signal receiver. In the nucleus, the receiver is principally DNA, and the signal proteins which carry information to regulate transcription using DNA are TFs.

The first TF discovered was AP-1 [46], it was a DNA binding protein which bound to a gene promoter. Since this discovery, over 1,600 TFs have been characterised in humans [47]. Some TFs form the complex of proteins that immediately lead to transcription at the promoter; however, some are indirectly responsible for regulating the formation of the transcription complex and can actively accelerate, slow or block different stages of the transcription process. Some TFs can bind directly to DNA, but other TFs known as co-factors indirectly bind to DNA through other TFs in a chain-like structure [48].

TFs are the end of a signalling chain which may originate far from the DNA sequence; they are said to be *trans*-regulatory elements. The DNA sequences that bind TFs are close to DNA and therefore said to be *cis*-regulatory sequences [49]. Non-coding DNA sequences are defined as sequences of DNA that do not encode for protein sequences. Introns, UTRs, and promoters are all examples of non-coding regions.

Although many TF binding complexes have been identified at core promoters or proximal promoter regulatory regions, there are also many other *cis*-regulatory regions which affect gene transcription. Enhancers and silencers are sub-classes of *cis*-regulatory elements which have been shown to upregulate and downregulate target genes respectively [50]. Promoters, UTRs and introns can also all have *cis*-regulatory elements. Other classes such as insulators have a structural effect on DNA such that they have an indirect effect on gene expression [50]. TFs bind at specific sequences known as Transcription factor binding sites (TFBSs).

Ultimately, the signals from all affecting TFs are integrated into the binary decision of whether to produce a new mRNA transcript. The number of transcripts produced and the lifetime before their degradation controls the amount of protein that will be produced from the gene.

#### **1.2.9.1 Enhancers and Silencers**

Enhancers and Silencers are short stretches of DNA that serve as platforms to integrate cellular signals through TF binding. TFBS vary in specificity and form the basis from which further co-factors are recruited to the enhancer to form a protein complex that can regulate transcription of one or more genes [48] [51] [52].

The combinatorial binding, pattern of occupancy and timing of TFs at enhancers all contribute to the expression pattern of a target gene. An inactive enhancer can be defined as a stretch of DNA to which no TFs are bound. To activate an enhancer, TFs known as pioneer factors can bind to nucleosome-bound DNA at the ends of the enhancer domain. Pioneer TFs then recruit chromatin remodelling complexes which reposition the nucleosomes to allow access to the enhancer for other TFs. Once the section of DNA is open (nucleosome-free), a complex process of cascade TF binding occurs which ends with a complex of proteins which can actively affect gene transcription. Enhancers can react to various transcriptional signals by changing the state or timing of the TF population. For example, two competing TFs may bind to the same enhancer; a cellular signal causes a drop in one TF, and so the competing TF (which has an excitatory effect) is now more likely to bind to the enhancer that causes transcription to increase. Enhancers also interact with other *cis*-regulatory elements to further increase the number of regulatory possibilities for a gene [48] [51] [53] [52].

Enhancers and silencers are similar in function except that in general, enhancers have an excitatory effect on gene expression while silencers have an inhibitory effect [54].

#### **1.2.9.2 Insulators**

Regions of heterochromatin are thought to be formed from a complex process involving reader/writer protein complexes that modify the nucleosomes and bind them to H1 histones to condense the chromatin into a 30 nm filament. The reader/writer complex binds to the chromatin and spreads along the genome in a wave of condensation. A pattern of covalent histone modifications may be used by a code-reading complex to recognise where the spread start sites are before binding to the reader/writer complex to guide the machinery down onto the chromatin. Barrier DNA sequences can stop the spread of heterochromatin, thereby creating regions of open and closed chromatin [55].

The *cis*-regulatory elements which bind proteins that block the spread of heterochromatin are called insulators. Cell differentiation emerges from patterns of gene expression that changes the protein content of a cell to alter its structural and functional characteristics [56]. The interplay between open and closed chromatin is thought to be a primary mechanism in the transcriptional switching of genetic regions. The alternating pattern of open and closed chromatin on a chromosome, are characterised as A and B compartments respectively [57].

CCCTC-binding factor (CTCF) is a zinc-finger protein that was initially described as a transcriptional repressor of the *Myc* gene, it is highly conserved in humans and has

homologues in other species such as *Drosophila melanogaster* [58]. CTCF has a well-defined binding motif that allows it to bind to DNA as well as other proteins. Its properties as a transcriptional insulator were first reported when the placement of a CTCF binding motif between an enhancer and promotor blocked transcription [59] [60]; one of the functions of CTCF is thought to be to block the spread of heterochromatin, and consequently is termed an insulator.

### **1.2.10 The Human Genome**

The human genome is organised into 46 separate molecules of DNA in the cell nucleus called chromosomes. Depending on the sex of a person, chromosomes are organised into either 22 or 23 pairs; females have two X chromosomes whereas males have an X and a Y chromosome [61]. The non-sex chromosomes in the human genome are known as autosomes while the sex chromosomes are known as allosomes. Different chromosome pairs have different lengths and contain varying amounts of genetic material.

Chromosome pairs themselves are almost identical; broadly speaking, they have the same genes and the same arrangement and configuration of genetic material, hence, there are two copies of almost all autosomal genes. A pair of locations or loci, whether it is a base pair or a whole gene are termed alleles; the singular allele, referring to one of the loci in the pair [61]. In interphase cells, the chromosome pairs are treated independently, they occupy different locations in the nucleus, and can both contribute to gene transcription [62]. It is only during meiosis and mitosis that the chromosomes are brought together into pairs giving the characteristic X shape.

It is estimated that less than 2% of the human genome codes for proteins, the remaining 98% was once thought to be 'junk' DNA with no known function [10]. Some of this DNA has little or no predicted function such as areas which consist of simple repeats of letter sequences that have low information content; however, much of the non-coding genome has unique sequences and patterns suggesting that it contains valuable information. Recent research has shown that in fact, 80% of the human genome is either transcribed, binds to regulatory proteins, or is associated with some other biochemical activity [50] [63]; this has led to a paradigm change in the scientific consensus. It is now known that large portions of non-coding DNA are responsible for regulating gene transcription in complex regulatory networks that control everything from the complex nature of spatial and temporal gene transcription in development to responding to intra-cellular environmental changes [63].

### 1.2.11 3D Genome Structure and Organisation

Chromatin is packed into the nucleus in a hierarchical structure that occupies only 15% of the nuclear volume [64]. The position of a chromosome or gene is non-random in the nucleus and is probabilistic in terms of its radial distribution, and is cell-type- and tissue-specific [62]. The prototypical examples of radial chromosome positioning are human chromosomes 18 and 19, which localise preferentially to the centre (chromosome 19) and the periphery (chromosome 18) of the nucleus in human lymphocytes but are arranged differently in other cell-types [62] [65].

The nuclear lamina is a fibrous structure at the nuclear periphery that provides both mechanical support and has a role in chromatin organisation. Regions of chromatin called Lamina-associated domains (LADs) bind to the lamina and are usually flanked by CTCF binding sites [66]. LAD are typically 0.1–1 Mb in size and generally gene poor, and the contained genes are either silent or expressed at low levels [66]. LADs correlate with the A/B compartments which are demarked by insulators [57].

While LADs in chromosomes are anchored to the lamina and are associated with heterochromatin and transcriptional repression, there are other regions interspaced between LADs that occupy the nuclear centre and are associated with euchromatin and transcriptional activation. Spatially proximal regions of DNA outside of LADs are known as Topologically-associated domains (TADs) [67] [62] [68] [69] [70] [71] [72] [66] [73]. LADs and TADs alternate on the genome and broadly correspond to A/B compartments bounded by the insulator protein CTCF [66].

#### 1.2.11.1 3D Gene Regulation

TFs are available for binding to regulatory elements in response to developmental, cell-specific, environmental or housekeeping activities. TFs can directly influence transcription on promoters either by forming part of the transcription complex or by enhancing or blocking other TF binding activity. In both cases, to directly influence transcription, the TF must be present near the promoter site.

Enhancers have been found at proximal-promotor sites; however, many *cis*-regulatory enhancers have also been shown to affect gene transcription megabases away from the promoter, often skipping genes in-between [48] [51] [53] [52]. Given that an enhancer needs to be in physical proximity to a promoter to act on it, how is it that *cis*-regulatory elements such as enhancers can influence expression? It is clear that the genome cannot

be thought of as a simple linear sequence but that the three-dimensional structure must be taken into consideration when studying regulatory chromatin contacts [54].

In order to study 3D *cis*-regulatory contacts in the genome, a new class of experiments was developed known as chromatin conformation. Several classes of chromatin confirmation experiments exist that indicate which parts of the genome are in contact with each other at a given time snapshot; one such high-throughput experiment known as Hi-C can produce a genome-wide contact map for a given cell snapshot [74] [70]. Initial Hi-C contact maps generated by Rao *et al* [70] showed that the genome was separated into specific regions of contact in which there was a high frequency of interaction that was bounded by a sharp border; these interaction compartments overlapped exactly with the A/B transcription compartments defined earlier. The compartments were defined as TADs, the borders of which were found to be significantly enriched for several proteins including CTCF [59] [60] [67] [62] [68] [69] [71] [72] [73].

The sharp border pattern seen in the TADs was postulated to arise from chromatin loop configurations where the bases or anchors of the loop were located at the TAD border. CTCF and cohesin, a protein usually associated with metaphase chromatin compaction were found to co-localise at TAD borders. Given that these sites coincided with CTCF TFBS, it was hypothesised that CTCF provided the DNA binding anchor while cohesin bound two CTCF sites together to form a loop which increased contact frequency within the loop and decreased it outside [59] [60] [67] [62] [68] [69] [70] [71] [72] [73].

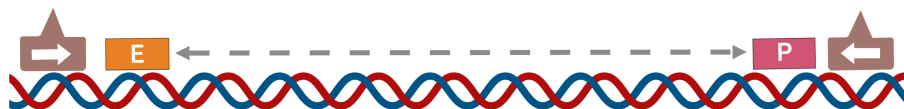
The chromatin loop has since been identified as a primary method of folding DNA so that *cis*-regulatory elements such as enhancers can be brought into proximity to gene promoters. To create a loop from a stretch of DNA, two sites must be 'pinched' together and held for as long as the regulatory elements need to influence transcription. Loops can be dynamic in that they are created in response to specific spatial-temporal or environmental conditions, or they can be more structural and contribute to the permanent regulatory landscape of a differentiated cell. Importantly, loops both increase the likelihood of contact for elements within the loop but also insulate elements external to the loop from elements within. There are many different types of chromatin loop that have been postulated; however, there is only one that has been studied in detail: the CTCF/Cohesin loop [54].

#### **1.2.11.2 CTCF/Cohesin Loop Formation**

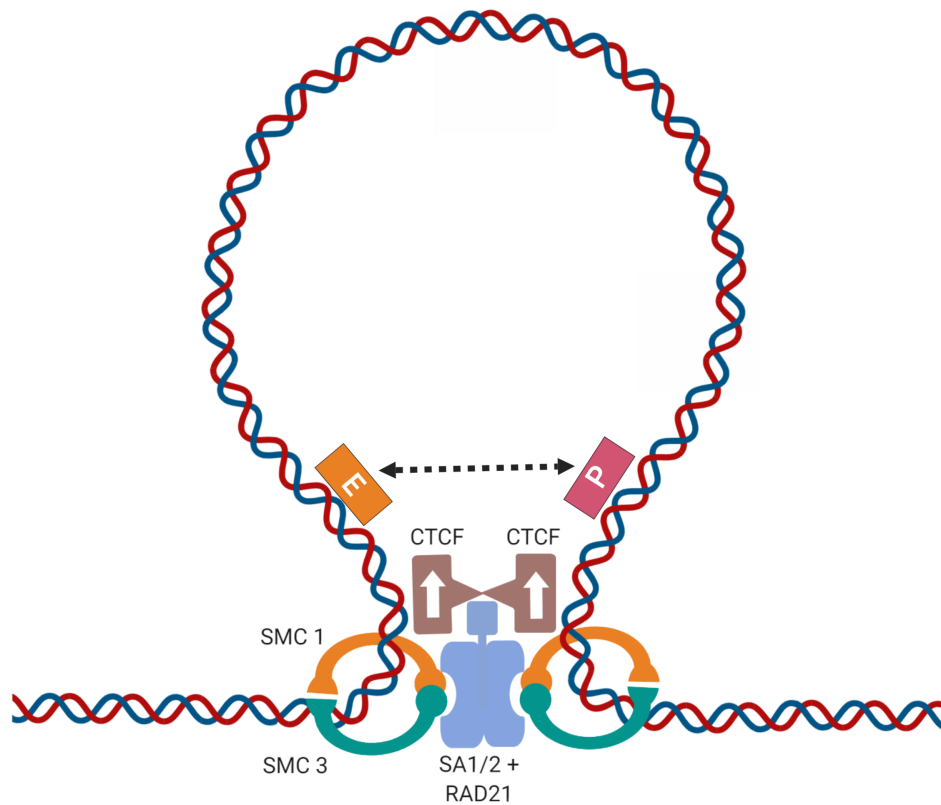
CTCF is a multi-function protein, but it was found that only when binding with cohesin and when a CTCF pair was orientated in the correct way, that a TAD boundary was detected. Functional experiments showed that inverting or deleting CTCF sites severely affects

TAD boundary formation and gene expression in the local region [67]. The most popular theory of CTCF/Cohesin loop formation is via loop extrusion whereby a loop is formed from cohesin forming a loop around two sections of chromatin in a handcuff configuration. Chromatin is then able to move freely through cohesin until an inward facing DNA-bound CTCF protein is encountered which halts extrusion on that side. Once both sides have encountered an inward facing CTCF motif, the loop is formed and stabilised [75]. Because the chromatin is pinched together, regions within the loop through fluid dynamic principles will statistically come into contact more often than areas outside the loop (Figure 1.8, p.57 and Figure 1.9, p.58). Through increased contact, *cis*-regulatory elements can interact with each other to regulate gene transcription [59] [60] [62] [68] [69] [70] [71] [72] [73].

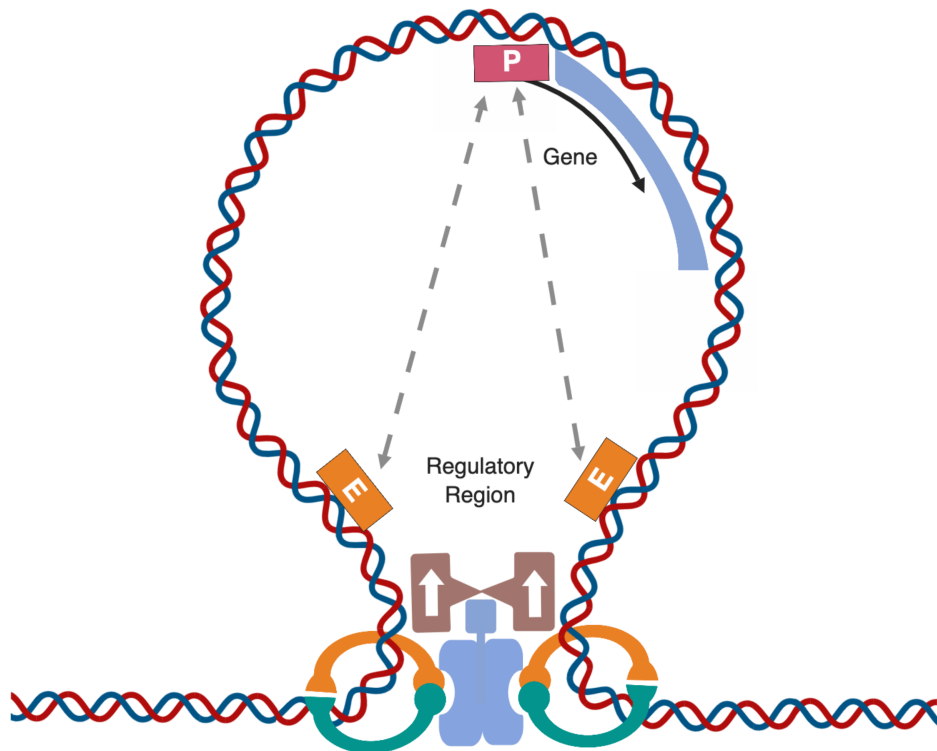
Although CTCF loop formation and maintenance is considered to be a dynamic process, TAD locations and overall stability are thought to be stable and are defined during cell differentiation. This gives rise to the postulation of more dynamic functional loop formation in response to non-developmental requirements; this area is less well understood, but it is thought that there are other proteins which can also form less defined loops such as the ZNF143 loop (Figure 1.10, p.59 and Figure 1.11, p.59). Here, it is thought another protein, known as Zinc-finger protein 143 (ZNF143), binds between a promoter and an enhancer directly to form a sub-TAD loop that pulls promoters down to the CTCF loop anchors that are enriched for *cis*-regulatory elements [76]. Other potential sub-loop formations have also been proposed such as Yin-yang 1 (YY1) and CTCF/Cohesin loops [77].



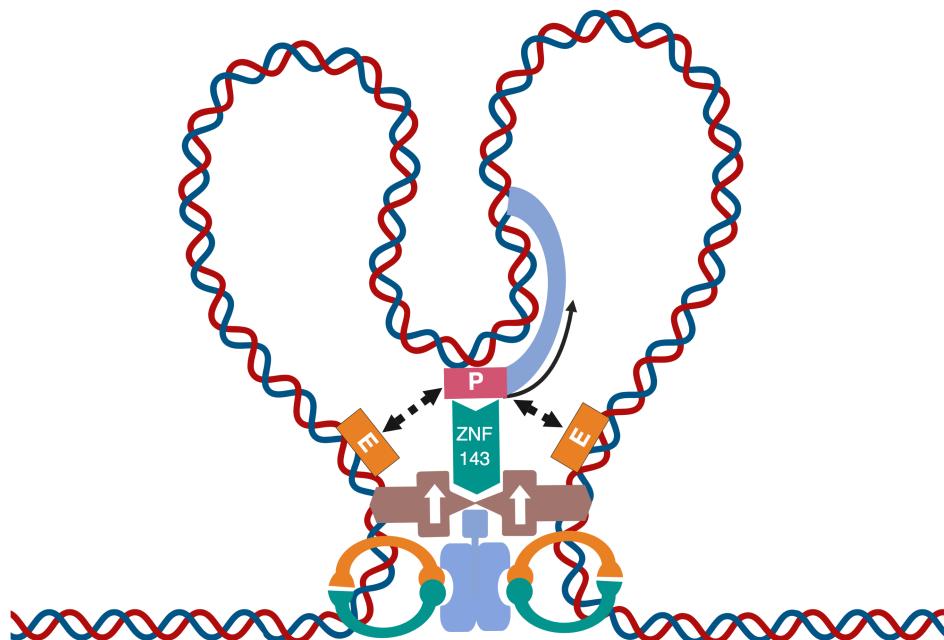
**Figure 1.8:** A genomic region shown before CTCF/Cohesin loop extrusion begins. For simplification, we only show a promoter (red) and enhancer (orange). The enhancer is too far away from the promoter to have any effect. Convergent CTCF sites with bound CTCF protein (brown) flank the region.



**Figure 1.9:** A pinched loop of DNA bound by a CTCF/Cohesin complex. Cohesin is made from the sub-components SMC1, SMC3, SA 1/2 and RAD21. The core components bind CTCF while the SMC1 and SMC3 form handcuffs around DNA that can slide along the molecule freely. The Loop extrusion theory hypothesises that cohesin binds around DNA in its linear configuration before extruding a loop through the handcuffs until it encounters a pair of convergent CTCF proteins bound to DNA where the loop formation then fixes. Inside the loop, genes and regulatory regions are in closer contact to each other and thus can form contacts.



**Figure 1.10:** A CTCF/Cohesin loop with a gene shown at the top of the loop and a regulatory region at the base of the loop. The gene region although in the loop is still not proximal to the regulatory region at the base of the loop and thus the enhancers cannot act on the promoter.



**Figure 1.11:** In a functional chromatin loop, ZNF143 binds to the gene promoter and the regulatory region at the base of the loop; consequently top of the loop is 'pulled' down towards the loop anchor. The gene promoter is now in close proximity to the regulatory region at the base of the loop.



## 1.3 Genetic Variation and Inheritance

One of the defining characteristics of life is the ability to reproduce and pass heritable information on to the next generation. The phenotype of an organism can change over time which, as discussed above is a consequence of an individual's genotype; thus, in eukaryotes, this implies DNA is passed to successive generations and changes over time. The change in heritable characteristics of a species over successive generations is called evolution [78]. Evolution allows a species to adapt to the changing conditions of its environment.

In eukaryotic cells, DNA information is passed *via* cell division. Asexual cell division occurs through a process called mitosis, where a cell divides into two genetically identical copies; it is an essential mechanism for the development and maintenance of a eukaryotic organism. In almost all eukaryotic organisms, new offspring are formed through the process of sexual reproduction, where DNA is passed onto children to form a blueprint for a new generation that uses a specialised form of cell division [79].

The changes to a species DNA sequence over successive generations is known as genetic variation. Variation constitutes changes to a DNA sequence which may or may not alter a phenotype; those that do alter a phenotype can be beneficial or detrimental to an organisms ability to perform in an environment. The concept of natural selection first defined by Darwin [80] states that over time, organisms which perform better in their environments as a result of variation can reproduce and pass on their genetic information at a higher rate than organisms that perform poorly. Fitness is a measure of the change in the abundance of a genotype or phenotype in successive generations. A phenotype that is better suited to an environment will be positively selected for over time through sexual selection (an organism must be alive first to be chosen as a mate), and consequently, its fitness will increase in the population [81].

Tracking genetic variation over generations is an essential tool for understanding the genotype/phenotype mapping as it can reveal which genetic changes are attributable to what traits. Genetic variation in eukaryotes arises from two processes: mutation and recombination. Mutation is defined as an alteration to a DNA sequence, whereas recombination is a shuffling process which occurs during cell meiosis. We discuss these concepts in more detail below.

### **1.3.1 Mutation**

Genetic mutations in organisms can arise from many different sources, both internal and external. Biological errors in cell division or cell repair processes can cause changes in a DNA sequence; external sources such as radiation and free radicals can also cause breaks and changes in cell DNA. There are two types of mutations: somatic and germline; germline cells are the reproductive cells of an organism, while somatic cells are all other cells. Mutations occur in the nucleus of a single cell, if the cell divides the change will be passed onto its progeny until the cell line dies out or the organism dies. Somatic mutations importantly result in no genetic information being passed to the next generation, and consequently cannot be defined as genetic variation at the species level. Some mutations affect germline cells, and so have a chance of being passed on to the next generation, these are known as germline mutations. When a mutation in the germline is successfully passed onto a subsequent generation, it becomes known as a genetic variant [79].

Mutations are defined as a change in the sequence of DNA, depending on where the change occurs and the type of change, it may or may not alter the phenotype. Mutation from external actors such as radiation generally produce random mutations in a DNA sequence; however, other processes such as cell division, DNA repair and somatic recombination are distributed non-randomly and result in mutation “hot spots” on the genome where changes are more frequent than in other areas [82]. Some areas of the genome have evolved to have deliberately high mutation frequencies in order to generate sufficient variation for phenotypes that must be highly adapted, such as the adaptive immune or olfactory systems.

It is important to note that mutation in cells happens much more frequently than the levels of somatic or germline mutation that become permanent in a cell line. Cells have a DNA repair method that can identify and correct mutation errors in the DNA sequence. This system is very efficient, resulting in very few mutations being transferred through cell division [79].

### **1.3.2 Recombination**

Genetic recombination is the process by which maternal and paternal chromosomal DNA is shuffled during the creation of sperm and egg cells known as gametes. During mitosis, the genetic information in a cell is copied so that the resulting child cells have identical sets of DNA. Human cells have two sets of 23 chromosomes, organisms with this configuration are known as diploid organisms. Gametes are created in a process known as meiosis that

results in haploid cells, meaning they only have one copy of each chromosome [81] [79].

Meiosis begins in the same way as mitosis; with DNA replication, which creates two identical sets of chromosome pairs known as homologs. Each homolog then undergoes a process of genetic recombination called homologous recombination. For each pair of chromosomes, the DNA is cut in random places and then swapped with the same place in the paired chromosome creating crossovers. For autosomes, humans receive one maternal and paternal chromosome; by shuffling DNA between the two chromosomes during gamete creation, a new genetically unique pair of chromosomes is generated that is a mix of both maternal and paternal DNA. After recombination, the cell divides to create two diploid cells and then immediately divides again to create four genetically unique haploid cells. Because the process of recombination happens at the point of gamete creation, every gamete produced in the human body is genetically different [81] [79].

Genetic recombination creates a vast amount of genetic variation as the space of all possible permutations of maternal and paternal DNA is randomly traversed, which ensures that every offspring that arises has a unique combination of maternal and paternal traits. Evolutionary speaking, if the offspring's parents have survived long enough to reach sexual maturity and reproduce, then they must have adapted to their environments at least to a reasonable degree. Accordingly, by randomly trying out different combinations of the two parents traits there is a chance that the new offspring will combine the best traits from both parents into an organism that is even better suited to its environment.

Recombination provides a vehicle for combining traits from parents, which over time results in better-adapted organisms for an environment; however, it is a poor vehicle for creating new adaptations as it only shuffles existing information. Although at the points of recombination, there is a probability that two halves of a functional genomic element will combine to create something new, mutations have a much more significant impact on genetic diversity. The powerful combination of the optimisation of existing information via recombination and the creation of new variation *via* mutation gives life the necessary tools to evolve.

### **1.3.3 Human Genetic Inheritance**

During human sexual reproduction, two randomly selected gametes come together to form a new embryo. The haploid DNA in both gametes is fused to create a new diploid organism before the single cell embryo divides *via* mitosis to eventually create fully formed offspring. Because humans are diploid organisms, they have two alleles per position in their genetic sequence. Genomic structures such as genes are active on both alleles (except in the

case of gene imprinting); they can have independent levels of activation and can interact with each other ways that are useful for genetic analysis. For a single genetic locus, we define two different alleles as being heterozygous and two identical alleles as homozygous [83].

Mendel's first law of segregation states that during gamete formation, the alleles for each gene segregate from each other so that each gamete carries only one allele for each gene. Mendel's second law of independent assortment states that Genes of different traits can segregate independently during the formation of gametes. Mendel's third law of dominance states that some alleles are dominant while others are recessive [6]. A dominant trait with at least one allele represented at the trait locus will mask the effect of the other allele; in a recessive model, both alleles are required for the trait to show in the phenotype. Penetrance in genetics is defined as the proportion of individuals which carry a variant or allele that then express the associated trait in their phenotype [84].

Using the definitions above, a model for transmission probability can be built for human autosomal and allosomal inheritance. We define the letter *A* to represent a dominant allele and the letter *a* to represent a recessive allele. Applying Mendel's laws and considering autosomal inheritance, for a parental combination of *Aa* and *Aa*, the possible combination of offspring would be *Aa*, *Aa*, *AA* and *aa*. In this example, 75% of offspring would show the phenotypic trait given complete penetrance. Extending from this example, for an autosomal dominant disease where one parent is affected, the chance that an offspring will exhibit the disease assuming complete penetrance is 50%.

Given one or two affected parents and assuming complete penetrance, the probabilities of an offspring exhibiting an autosomal recessive trait in their phenotype are 50% and 100% respectively. In recessive disease models, an individual is said to be a carrier if they have the alleles *Aa*; consequently, they are carriers of the trait but do not express the phenotype. With two carriers and applying the same model, the chance of expressing an autosomal recessive trait is 25%; however, the chance that offspring will be carriers is 75%. This can result in inheritance patterns where the phenotype only presents every few generations when two recessive alleles come together, such as the trait for red hair.

Allosomes and have different inheritance patterns when compared to the autosomes due to the combinations of chromosomes that different sexes have. In the human genome, males have one X and one Y chromosome, while females have two X chromosomes. In males, the Y chromosome always inherits from the paternal side as there is none to inherit from the maternal side; the X chromosome always comes from the maternal side. Females receive one X chromosome from each parent. Consider a recessive trait that resides on the X chromosome: the trait may be passed in a carrier pattern through females who

are carriers for a trait through several generations before being passed to a male who is hemizygous and so expresses the trait in their phenotype. Females will only express this same trait if they are homozygous, leading to an inheritance pattern where the trait appears to be expressed predominantly in men; this is known as an X-linked recessive trait. There are additional inheritance patterns for X-linked dominant and Y-linked dominant that are not discussed here.

From a genetic analysis point of view, these inheritance patterns give additional information about a trait being studied and give certain mathematical 'toe holds' when trying to decipher where on the genome a trait may be located. In reality, thinking of traits as purely qualitative is a simplification. A Mendelian trait is one that is controlled by a single locus in an inheritance pattern where mutations in such loci can cause disease in a model obeying Mendel's laws. While there are some diseases such as sickle cell anaemia [85] and cystic fibrosis [86] that obey Mendelian inheritance and other diseases that have complete penetrance such as type I neurofibromatosis [87], the majority of traits are non-Mendelian with incomplete penetrance. In addition, phenotypes are mostly continuous, as are the expression patterns of gene alleles; for complex diseases such as diabetes, there are hundreds of variants with different inheritance patterns that produce millions of different combinations of gene expression; some of which will give rise to the diabetes phenotype. Furthermore, Mendel's laws of independent assortment have been shown to only be true if the traits are on different chromosomes; traits on the same chromosome have a non-random probability to co-segregate during recombination [88] [7].

We can now define some more common genetic terms using inheritance as a background. A genotype is defined as a set of genetic loci that characterise an organism's genetic makeup. Genotypes are presented as pairs of sequences: one for each allele. Importantly, a genotype contains no information regarding which allele is paternal and which is maternal. A haplotype is defined the same as a genotype except that the maternal and paternal source orders the loci. Genetic data presented with inheritance information is said to 'be phased' [83].

### **1.3.4 Genetic Variation**

Genetic variants are mutations which have been inherited via an organism's germline and consequently have become part of the population of genetic variation. Given that the goal of genetic analysis is to explore the genotype-phenotype relationship then one could argue that genetic analysis must focus on the genetic variance in an organism in order to better understand this relationship; the identification, annotation and analysis of variants and their inheritance is therefore an essential part of genetic analysis.

Once a mutation becomes a variant, it becomes part of the local variance in a population. Traditionally, humans tend to breed with others that are geographically or culturally close to each other, which combined with migration patterns and other factors gives rise to pockets of variation that exist only in specific regions known as genetic ethnicity. The related phenotypic traits that arise from such groupings of variation can be broadly called an individual's race, although there are many problems and caveats to this definition [89]. Ethnicity in terms of genetics is a broad multi-tiered definition that can define an individual's genetic variance at a continent level but also the village level.

If a variant exists in a population, then it will have a prevalence known as the allele frequency. A population can be defined as any set of individuals; therefore, allele frequency must always be accompanied by a population definition. For example, a variant has an allele frequency of 5% in West European populations but 10% in South Asian and 1% in London. A variant which exists in less than 1% of a population is considered to be very rare or in some cases perhaps transient. Between 1% and 5% of the population is considered a rare variant while greater than 5% is considered a common variant that has a significant presence in a population [90].

The major variant at a given locus in a specific population would be considered to be the most common with the highest allele frequency, while other variants would be considered minor. We say allele frequency and not variant frequency because the human genome has two alleles at every position, therefore, the variant frequency lies at the allele level. Genetic drift is the process by which the allele frequency of an allele changes over time due to random sampling and inheritance. Genetic drift can cause variants to disappear entirely from a population over time or to become much more frequent [91].

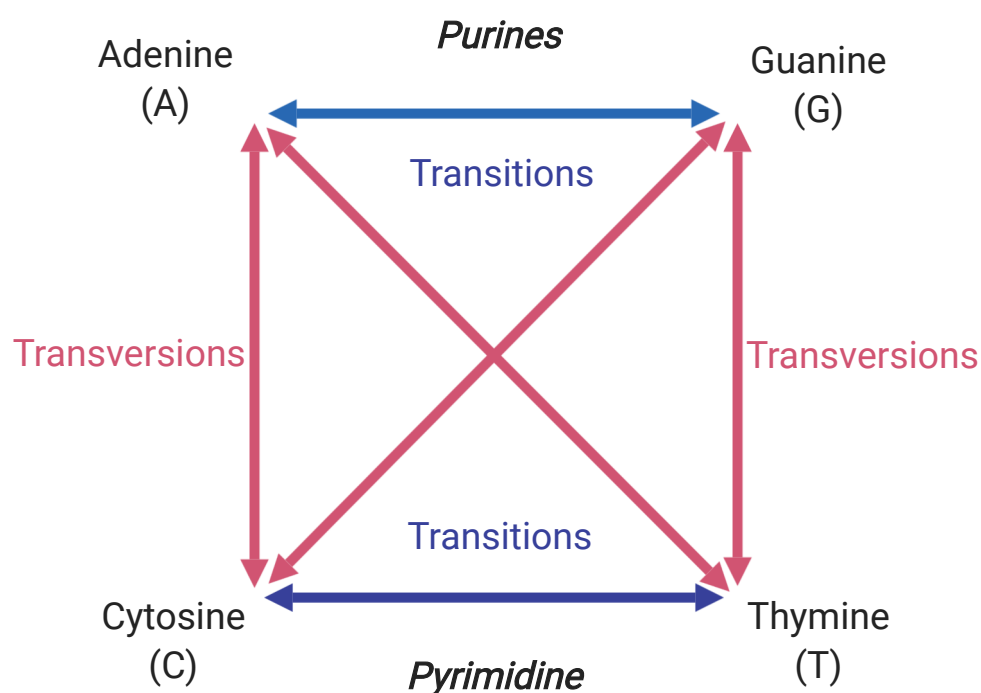
There are many different forms of genetic variation that range from single letter point mutations to duplications of whole chromosomes. Due to the importance of variation in genetic analysis, we discuss the different types of variation in detail below.

#### **1.3.4.1 Single Nucleotide Variants**

The simplest type of mutations are Single nucleotide variants (SNVs); they are single point mutations in the DNA sequence of a cell, for example, a change in sequence from an A to G. They are the most heavily used of all variants as they can be highly informative yet are the simplest to analyse.

There are two types of point mutation which are characterised by the chemical substitution taking place at the molecular level. Adenosine and Guanine are known as purines

and are chemically and structurally different from Cytosine and Thymine, which are known as pyrimidines (Figure 1.2, p.44 and Figure 1.3, p.44). A substitution between two purines or between two pyrimidines is known as a transition while a substitution from a purine to a pyrimidine or vice versa is known as a transversion. Transitions are twice as likely to occur than transversions as substituting a ring for a double ring, or *vice versa* requires more energy than substituting for the same structure [92] (Figure 1.12, p.66).



**Figure 1.12:** Transitions are interchanges purines or of pyrimidines: they therefore involve bases of similar structure. Transversions are interchanges of purine for pyrimidine bases, which involve the change of differently structured molecules.

The chemical mechanisms behind transitions and transversions are important as the same mechanisms are present in functional patterns and structures in the genome. For example, methylated cytosine is more prone to transition than unmethylated cytosine, which leads to rare unmethylated CpG islands (islands of genetic sequence which contain alternating C and G nucleotides) [93].

By focusing on SNVs in the genome instead of looking at the entire sequence, some parts of genetics analysis can be simplified while still capturing enough variation to obtain significant results.

#### **1.3.4.2 Insertions and Deletions**

An insertion or deletion is defined as either the adding or removing of information from a DNA sequence. For example, when comparing two genomes A and B, they are identical except that A has the four extra letters ATTG inserted into the middle of its sequence. This would be classified as an insertion if we took sequence B to be the reference while this would be identified as a deletion in sequence B if we took A to be the reference. Consider two more sequences C and D with C being the reference; it is identified that D not only has a deletion of the letters AGGGC but in the same position has an insertion of CCC. This swap of AGGGC for CCC is both an insertion and a deletion and is termed an indel [94].

Insertions, deletions and indels form part of a large group of variation known as structural variation. SNVs alter the sequence, but they do not change the overall length or structure of the DNA; a structural variant is defined as any variant that results in a change in length of a DNA sequence [94].

#### **1.3.4.3 Duplicates and Copy Number Variation**

Duplications are a particular form of insertion and cover a broad array of structural mutation events in the genome. They range from the repetition of small DNA sequences to gene duplication and up to chromosomal duplication. Gene duplication is thought to be one of the primary evolutionary processes behind genes with similar functions, where a random duplication event followed by a period of separate mutation in the two genes results in functionally similar but different genes [95]. We discuss several specific types of duplication below for broad information, but it is not exhaustive.

Copy number variation (CNV) is a broad type of structural variation that generally involves repetition of a large number of base pairs; they are generally vaguely categorised into short and long repeats. Short repeats are defined as a short sequence such as a tri-nucleotide pattern repeated as a number of insertions in the genome. Long repeats are large sections of the genome that are duplicated; examples may include whole genes or even whole chromosomes [96].

Short sequences that are repeated multiple times in tandem are known as tandem repeats. Tandem repeats between 10 and 60 nucleotides are known as minisatellites while shorter numbers of repeats are known as microsatellites or short tandem repeats. Variable tandem repeats are sections of repeats which have a repeat pattern rather than being identical such as a variable section of four letter repeats. This confusing terminology exists



in part because of the large number of different ways in which sequences can be duplicated and repeated mathematically; broadly speaking these terms are collectively known as segmental duplication [97].

#### **1.3.4.4 Other Structural Variants**

Translocations are defined by the abnormal transfer of DNA between two nonhomologous chromosomes [98]. Translocations may be non-reciprocated where a DNA sequence is transferred and spliced into another chromosome with no exchange of material; whereas reciprocated translocations involve the exchange material between chromosomes. Reciprocated translocations can be balanced where the exchange of material is equal, or unbalanced, which results in a disproportional material exchange. Translocations are implicated in several disease types, including cancer [99] and Down's syndrome [100].

A chromosomal inversion is a chromosomal rearrangement where a region of DNA is reversed *in-situ*. Interestingly, inversions appear to cause no abnormality unless the breakpoints cut a functional element in half such as a gene or regulatory element; or if the rearrangement disrupts a *cis*-regulatory network [101].

#### **1.3.5 Variation and Disease**

A disease can be defined as a phenotype that is undesirable or detrimental to an organism; hence, genetic variation and disease are inexorably linked. Linking genetic variants to disease is one of the primary applications of genetic analysis; by comparing and contrasting genomes from multiple individuals that have been grouped into distinct phenotypes, one can try to reveal which variants affect which disease phenotypes.

Once a link has been established between a variant and a disease, experiments can be designed to explore the relationship and reveal the biological mechanisms that yield the phenotype as the result of the genotype. The process of identifying variants in multiple individuals, comparing and contrasting them to obtain some association and then identifying the biological processes behind the link to form some causality defines one of the core project flows of a disease study.

## 1.4 Genotyping Data Analysis

The primary data source for any genomic analysis is genetic information. There are many experimental methods which enable the capture of this information from a biological sample into digital data. Genotyping is one such method and currently forms the backbone of the available data in research today for genomic analysis (although other methods such as whole genome sequencing are quickly overtaking).

In this section, we present the relevant theory needed to understand how genotyped data is handled and processed for genomic analysis, with a focus on the techniques relevant for generating input data for our proprietary software tool, REMEDY.

### 1.4.1 DNA Sequencing

In order to perform genomic analysis, there must be some available information about the DNA sequence of the individuals being studied. To obtain this information, the genetic material must be read from a biological sample and converted into a digital signal. DNA sequencing is defined as a process by which a sequence of nucleotides is read from a DNA molecule. The first sequencing of a nucleic acid molecule was performed on RNA as sequences were more readily available than DNA; RNA molecules are also single-stranded and tend to be considerably shorter than many DNA sequences [102]. The first whole RNA molecule was sequenced by Holley *et al* in 1965 [103], it was a short molecule of transfer RNA (Transfer RNA (tRNA)) approximately 90 nucleotides long.

After the purification of DNA based bacteriophages [102], there was an abundance of DNA to test new protocols with. In 1968 Ray Wu and Dale Kaiser discovered that at the ends of the molecules of some types of bacteriophage DNA, the 5'-terminated strands were 20 nucleotides longer than the 3'-terminated strands. They used DNA polymerase to fill the single-stranded ends with radioactive nucleotides while measuring which nucleotide filled each position in the sequence [104] [105]; thus, they succeeded in sequencing 186bp of DNA. Further improvements were made by priming DNA polymerase so that an oligonucleotide sequence could be read using radioactive nucleotides anywhere, not just at the ends of bacteriophage genomes [106] [107] [108]. However, these methods still only read small pieces of DNA and required considerable amounts of analytical chemistry.

Further advances in using electrophoresis through polyacrylamide gels led to separation and identification of polynucleotides by length, including Sanger's 'plus and minus' system [109] [110]. Sanger went on to use the system to sequence the first full genome, a

bacteriophage known as PhiX [111]. Sanger's plus and minus system and the Maxam and Gilbert technique developed in parallel can be considered the 'first-generation' of DNA sequencing [102]; however, the major breakthrough in sequencing technology came in 1977 with the development of Sanger's 'chain-termination' or dideoxy technique [112]. Sanger sequencing, also known as the "chain termination method", although similar to the plus and minus system, had a much higher accuracy, robustness and ease of use that led to it being widely adopted as the primary DNA sequencing method [102].

Additional improvements in the Sanger sequencing method led to the development of increasingly automated DNA sequencing machines until the first commercial machines were developed that produced reads of slightly less than one kilobase (kb) [113]. Many genomes were significantly longer than 1kb hence further techniques to merge multiple sequenced fragments, such as produced by 'shotgun' sequencing were developed in the 1980s [114] [115]. This work culminated in the successful completion of the human genome project in 2002, a colossal effort to completely sequence the human genome [10].

It is important to note that DNA sequencing does not infer the absolute position of the sequence fragment being read within the whole molecule. Indeed historically determining the position for a sequence fragment was a laborious task which resulted in an approximate position that, at its best, could be in a range of 500kb. The human genome project was an attempt to sequence and link many independent fragments together to form a single contiguous segment that described the whole human genome [10]. This first 'reference' genome was, in fact, an amalgamation of many different genomes; the final sequence was determined by the most common genotype at a given location. The first reference genome allowed for the first time a genome-wide view of allele frequency in the initial population of the samples that constituted the human genome project. Importantly, the reference also provided the first global position for a genotype on the human genome.

Acquiring complete information about a genome would require a complete genome sequence with full knowledge of every current variant that exists; this is an impossible task even without considering that genetic variation is a continuous process. The reference genome, therefore, provides a 'best guess' sequence representing the most common genotypes seen at any given position. As new DNA sequences for an organism are discovered, consensus on the most common reference sequences changes and thus new versions of the reference sequence must be released. This has resulted in reference genomes being given version numbers to identify the historical reference genome a given sequence maps to; the current human genome version is 38 [116].

Today, input nucleotides sequences that are constituted together to form a reference version for an organism are stored in sequence databases at key global locations by an

international consortium. The National centre for biotechnology information (NCBI) GenBank [117] database, the European molecular biology laboratory (EMBL) and the DNA data bank of Japan (DDBJ) all store public DNA sequences that are replicated between them.

### **1.4.2 Defining Genomic Position and Strand**

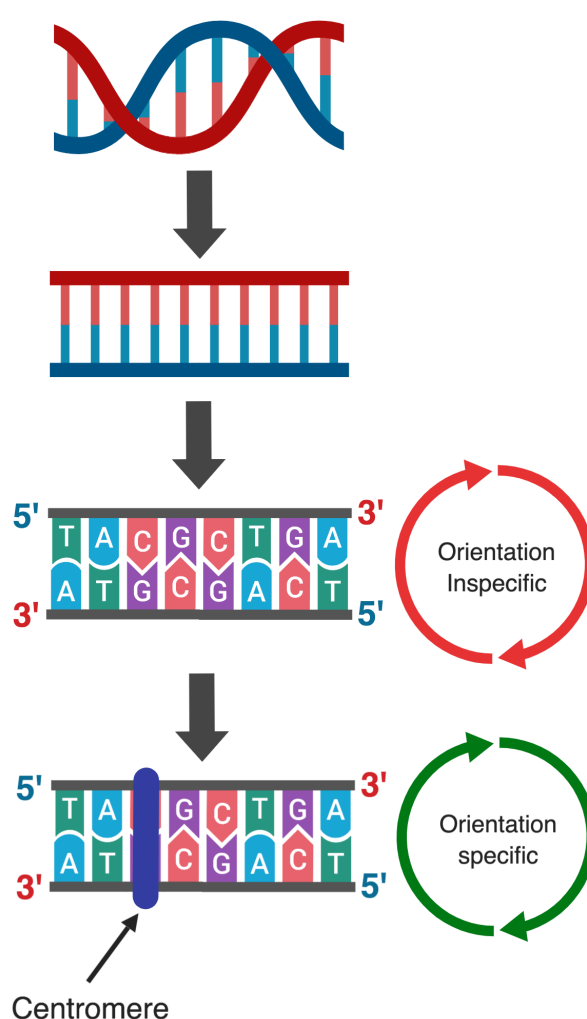
The concept of position on a genome is critical to all aspects of genomic analysis. The absolute or relative positioning of a sequence allows it to be compared to other sequences and enables mapping of the sequence to a model of a genome. Defining genomic position without complete information is a challenge which has been approached in a multitude of different ways since Watson and Crick first solved the structure of DNA [13]. In order to biologically define an absolute position on a molecule of DNA, three things must be known: which end of the molecule to count from, the number of bases to count in from the end and the strand of DNA if a single nucleotide rather than a base pair is required. When extracting a sequence, the end position or length of the sequence is also needed.

Orientation and strand when considering DNA is a confusing subject. A single strand of DNA has an orientation as one end has an exposed phosphate group attached to the 5th carbon atom of deoxyribose, and at the other, a hydroxyl group attached to the 3rd carbon atom; these are known as the 5' and 3' ends respectively. However, when in a double-stranded configuration, the DNA molecules are orientated in opposite ways so that from either end there is both a 5' and a 3' terminator. When studying DNA from a sequence perspective, it is useful to display it as an untwisted ladder where the 'sides' are the backbone, and the rungs are the base pairs; this then lends itself to displaying sequence on a page as two lines of letters. From this configuration, one could count from left to right along the bases to the desired position; however, the problem remains as to how to orientate the DNA molecule to the page: which strand is the upper line and which is the lower? We will hence use the term 'upper' and 'lower' strand when discussing a page representation of a DNA sequence as two rows of corresponding letters.

Early attempts at solving the strand problem named the different DNA strands Watson and Crick [118]. The earliest attempt at this used the cytosine-rich strand of the DNA molecule of a phage to denote Crick and the cytosine-poor strand as Watson [119] [120]. The assignments were later reversed when Watson was designated as the pyrimidine-rich strand, and the Crick the purine-rich [121]. The WC system was later applied to transcription where the Watson strand was defined as antisense and the Crick strand as sense [122]; however, this assignment is meaningless for orientation, as although transcription moves from 5' to 3', this could be read from the upper strand from left to right, or the lower

strand from right to left.

Arguably, the most popular way of assigning biological strand originated with the *Saccharomyces* Genome Database (SGD), which defines the Watson strand as the strand which has its 5'-end at the left telomere and the Crick strand as its complement [123]. The left telomere was defined based on pre-genomics linkage maps and was consistently chosen as the short arm. By adding an off-centre centromere point, a chromosome can be determinately orientated (this of course only works for organisms that have centromeres on their chromosomes) (Figure 1.13, p.72). Both the NCBI genome data viewer and the UCSC genome browser also support this view.



**Figure 1.13:** Methods of linearly orientating DNA. DNA can be untwisted and shown as a ladder-like, flat surface; the base pairs then correspond to the rungs on the ladder. Displaying DNA in this way does not solve the orientation problem as the upper and lower strands are mirrors of each other with respect to the 5' and 3' terminators (they are not mirrors in terms of sequence). Chromosomes can be orientated as they have a short arm and a long arm that switch at the centromere, allowing for determinate orientation.

The definition of strand shown above requires a full view of a chromosome to determine the correct orientation; in reality, sequencing is performed on short fragments. How then can small fragments be orientated to the correct strand? The human genome project employed various methods to build overlapping sequences of DNA into one contiguous sequence for each chromosome. This process was not sequential and was done in parallel so that multiple sections of contigs were built up at the same time. When building a reference genome for an organism, the gaps in the sequence are filled and the existing sequences are refined; hence the intervening sequences between the telomeres of a chromosome can change orientation as more data becomes available. The fluid nature of this causes any strand designation based on the centromeric 5'/3' method to be tied to the genome reference version of an organism.

Using the above definitions, we can finally define a genomic position bioinformatically using the reference genome, assembly version, position from left (short) arm telomere end and strand. Strand can be specified using the centromeric 5'/3' method that is tied to a reference and assembly; however, there are other also other methods to define strand that we discuss below. When comparing two sequences, it is critical in genomic analysis that the same system of positioning was used and that both sequences are defined on a common strand using the same designation scheme.

### **1.4.3 Defining Variance Using a Reference Genome**

A reference genome represents an agreed sequence for an organism that generally uses the most common sequences in the global population. Many variants may exist for a given reference position in multiple populations and sub-populations. The single nucleotide polymorphism database (dbSNP) is a database of variation that was designed to complement GenBank as a controlled method of submitting and recording observed variation in different populations [124]. dbSNP contains both SNVs and structural variation from many different organisms.

Both SNVs and structural variants can be identified on a reference sequence by a unique flanking sequence on either side of the variation. Using this method ensures that even a large deletion or insertion will match to the reference as it is essentially reduced to a single nucleotide to match the flanking sequence. A submitted observation of variance to dbSNP is required to be at least 25bp long in terms of the experimental assay; the actual submission must be at least 100bp in length. Any padding required must be obtained from the reference sequence. The submission process also requires geographical population information to be provided so that the variant can be assigned computed allele frequencies [124] [125] [126].

New submissions to dbSNP are given a unique SNP ID number (ss# or ssID). The flanking sequence is used to align each submitted variant to a reference genome using the Basic local alignment search tool (BLAST) and MegaBLAST [127]. If several ss numbers map to the same position, they are clustered together and given a unique RefSNP ID number (rs# or rsID). ss groups of only one are still assigned an rsID; new ssIDs can then be added to the cluster as they are submitted. For each build of dbSNP, all RefSNPs are recomputed and aligned against multiple versions of the reference genome. RefSNPs during this process may be merged as a result of co-localisation with the lower RefSNP number taking precedence [124] [125] [126]. Submitters to dbSNP can arbitrarily submit variations on either strand of a DNA sequence. When building a RefSNP cluster, the convention is that the ssID with the longest flanking sequence is used to set the orientation. Once the orientation is set, a comprehensive set of alleles for the locus is calculated [124] [125] [126].

The major or ref allele for an rsID is defined as the sequence reference allele, the minor or alternate alleles are the calculated variant alleles defined during the build process of dbSNP [124] [125] [126]. Both ref and alt alleles have allele frequencies in different populations; the Minor Allele Frequency is a commonly used term that relates to the allele frequency of an alt allele for a given population [128].

#### **1.4.3.1 Sequence Ambiguity**

Some variants have Ref and Alt nucleotides that are a complement to each other such as A/T and C/G. When comparing an observed genotype of T for example, to a variant A/T, it is impossible to know whether the observed genotype is the Ref or the Alt and consequently, it is also impossible to assign strand. These are known as Ambiguous SNVs; they can be problematic in genomic analysis if the flanking sequences are not used to orientate the variant to the correct strand.

### **1.4.4 Strand Designation Schemes**

#### **1.4.4.1 POS/NEG (+/-)**

The POS/NEG or +/- strand designation scheme reflects the 5'/3' centromeric biological strand designation of the subject sequence or variant. Because 5'/3' strand designation depends on the reference to which the sequence is mapped to, to adequately describe a sequence designated in POS/NEG requires the reference genome and assembly version.

+/- may also refer to the HapMap project [129] 5'/3' designations, which were an early attempt to give consistency to strand; care should be taken not to confuse the two. Sequence alignment tools such as BLAT [130] will provide the position and POS/NEG strand designation for a given sequence. The genotypes for POS/NEG designated sequences are generally encoded as GATC letters.

#### **1.4.4.2 FWD/REV**

The forward/reverse (FWD/REV) strand designation scheme is an internal system used by dbSNP to denote variant orientation. The submissions that make up a RefSNP are resolved using BLAST to a contig sequence and finally, a position on the reference genome during the build process of dbSNP. Each ssID will, therefore, have an orientation with respect to the reference genome. When constructing a RefSNP, the ssID with the longest flanking sequence is chosen as the orientation for the cluster; FWD designates the orientation is on the + or forward strand, while a REV or reverse designation specifies that the orientation is on the - strand. All ssIDs are subsequently re-designated to represent their orientation with respect to the RefSNP [126].

To summarise, RefSNPs have a FWD/REV orientation with respect to a reference genome, and ssIDs have a FWD/REV orientation with respect to the RefSNP. RefSNP FWD/REV designations can change between builds of dbSNP; consequently, a variant described using FWD/REV should include the reference genome, assembly version and dbSNP build number. The genotypes for FWD/REV are generally encoded as GATC letters [126].

#### **1.4.4.3 TOP/BOT**

Both the POS/NEG and FWD/REV designation schemes ultimately rely on the biological strand assignment. The designation derives from a reference genome and so is permanently tied to the assembly version. While convenient for analysis, these schemes invite error as it requires sequence variant data to be always paired with meta-data describing the source reference, version and dbSNP version in the case of FWD/REV. In real-world data, this information can be lost or inaccurate. Furthermore, although the human genome reference is relatively stable and well characterised, other less developed genomes for species such as Zebrafish have a much higher rate of release and a significantly higher proportion of change within those releases [131] [132]. It is not uncommon for whole sections of chromosomes to switch strand, making correct strand designation more difficult and consistency of results challenging [133].



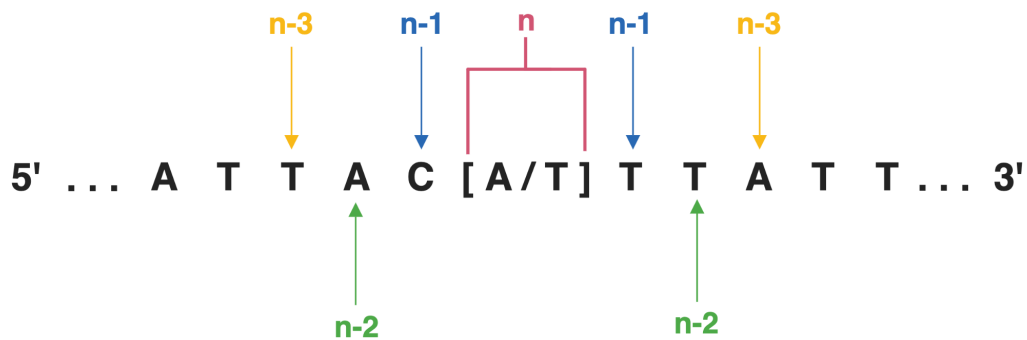
Illumina (The industry leader in sequencing technology) developed an internal strand designation known as TOP/BOT [131]. Rather than referencing evolving public databases such as dbSNP and reference genome versions, Illumina developed a method to consistently designate strand based on the target sequence or variant and its surrounding contextual sequence alone. Despite the designation terminology, TOP and BOT do not have any connection to the 5' or 3' strand; instead, the strand assignments are fixed for different combinations of variant. The scheme is primarily aimed at SNVs however, there are provisions in designation for structural variants, however, they will not be discussed here [131] [132].

A variant which has a ref of A or T and an alt of G or C respectively is known as an unambiguous variant, i.e. the alternate allele is different regardless of the strand designation. In the case where the reference is G or T and the variant is C or A respectively then it is unknown, which is the ref and which is the alt without first knowing the strand. The simplest case of determining strand in TOP/BOT is with unambiguous variants, where a fixed strand designation is called for each combination of unambiguous letters (Table 1.1, p.76). For ambiguous cases, the surrounding sequence is used to determine the strand designation; the sequence from positions n-1 and n+1 bases away from the target variant is read until an unambiguous pair is encountered, strand designation is then set from this pair instead (Figure 1.14, p.77). This process produces the same results as long as the local sequence remains the same; this makes the scheme ideal for highly variable genomes [132].

TOP/BOT strand designation is important to understand as it is the internal representation of all genotyping data in Illumina proprietary software (the primary input source for the REMEDY toolset). Genotypes designated as TOP/BOT can be encoded in GATC or AB. Illumina AB encoding refers to another internal representation scheme for genotypes that uses the Allele A and Allele B designations shown in Table 1.1, p.76. to encode a genotype as A or B rather than the source encoding GATC. This is done so there is no confusion regarding whether the source GATC is the original SNV or the 'walked' SNV pair in ambiguous variant cases.

SNP	Strand Designation	Allele A	Allele B
G/T	BOT	T	G
A/G	TOP	A	G
C/T	BOT	T	C
A/C	TOP	A	C

**Table 1.1:** Strand and allele designations for unambiguous SNPs in the TOP/BOT strand designation scheme.



**Figure 1.14:** Sequence walking method for determining strand when given an ambiguous SNP in the TOP/BOT strand designation scheme. Sequence pairs are considered moving outwards until an unambiguous pair is found.

#### 1.4.4.4 DESIGN

Genotyping microarrays have oligonucleotide probes which match a target sequence so that a target variant can be read. Each probe is designed to a biological strand based on its thermodynamic stability and hybridisation characteristics.

While not a designation scheme in of itself, Illumina stores probe variant information internally in what they call DESIGN designation. The DESIGN scheme indicates that each probe is designated to + or - depending on the strand to which it was designed; all alleles are then shown based on the designed designation. Each probe will also have an inherent TOP/BOT designation and a FWD/REV designation along with the genome version and dbSNP build number used for the FWD/REV and +/- designation [131].

#### 1.4.4.5 Converting Between Schemes

For most bioinformatic data flows, it is required to orientate genotyping data to a single strand before further processing. To convert data to a target strand in one of the encoding schemes described above, all variants called on the opposite strand must be complemented to the target strand when DESIGN designated. For example, three microarray probes in DESIGN are listed as TOP/FWD, TOP/REV and BOT/REV; the genotypes for the three probes are AC, AG and TC. We currently have a mix of strand designations as the genotypes are designated as DESIGN, to re-encode the data to TOP, we would need to complement the BOT genotype to flip it to TOP. In this case, the genotypes would become AC, AG and AG. To convert to REV encoding the first genotype would need to be flipped yielding TG, AG and TC and so on.

### **1.4.5 Genotyping**

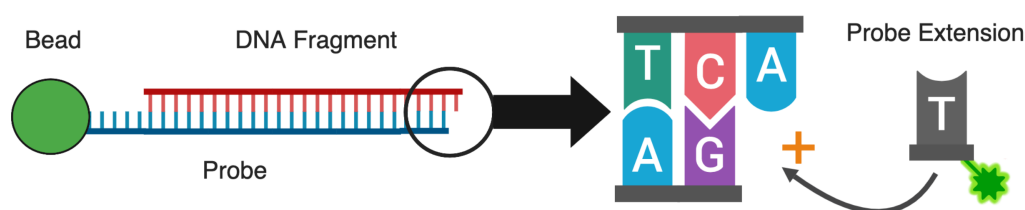
Since the development of Sanger sequencing machines, there have been several breakthroughs in sequencing technology culminating in Whole genome sequencing (WGS), a high-throughput method which can completely sequence a genome [134]. However, WGS is a relatively new technology and is still costly, time-consuming and often unnecessary for many forms of genomic analysis. When performing comparative genetic analysis, the most likely informative portions of the genome are locations where genetic variance is already known to exist. Using databases such as dbSNP, it is possible to identify a small subset of genotypes from a reference genome that are potentially informative for the analysis being performed. Many methods to obtain genotypes without direct sequencing were developed in the years between Sanger sequencing and WGS; these are collectively known as genotyping [102].

Many common problems in genomic analysis require a set of specific variants to be genotyped. For example, a researcher may want to compare the variance in specific areas of the genome to reveal potential causal variants in disease, or they may want to diagnose disorders in a patient by obtaining their variance in a specific gene and comparing it to previously identified causal variants. In these cases, the reference genome and previously identified variants can be used to create a set of target markers which, when read, will yield enough information to complete the project goals but at a reduced cost when compared to sequencing.

#### **1.4.5.1 Microarray Technology**

The most widely used genotyping technology in use today is DNA microarray technology. There are several different types of microarrays; however, they all follow the same basic principles. A single strand oligonucleotide probe is designed so that it maps to one area of the reference genome where the target genetic variant resides, and so that the variant to be read is the next letter in the sequence. The probes are attached to a processing substrate during manufacturing as a pre-prepared unit. A genotyping run begins with the preparation of the target DNA sample where it is denatured and fragmented before being flowed in solution over the processing substrate. The DNA fragments only hybridize to the probes where the sequence matches, hence theoretically, each probe only captures DNA fragments from the target marker region only. When the fragments hybridize, they will overlap by differing amounts depending on the size and relative position of the overlap to the probe. The probes are then extended with a single fluorescently tagged nucleotide that is the complement of the next nucleotide in the overlapping fragment. The next letter is

the target variant therefore the tag is effectively reading what letter the fragment has in the sequence for that position. The fluorescent signal is then read and interpreted to obtain a base call for each allele at the target variant position (Figure 1.15, p.79) [135].



**Figure 1.15:** The Illumina BeadChip genotyping process. Oligonucleotide probes are designed so that they match the immediate preceding sequence of a variant, and are attached to beads mounted on silicon. DNA fragments hybridise to the probes as they are passed over in solution. The probability of hybridisation is related to the size of the matching sequence overlap between the probe and the fragment. The diagram shows an overhang of just one nucleotide but, in reality, the overhang is variable in length as the fragmentation process produces random sized sequences. The probes are then extended using fluorescently tagged nucleotides that are the complement to the next letter in the fragment sequence; this effectively reads the target variant due to the design of the probe. The tags can then be read by an imaging machine to determine the genotype of the variant.

#### 1.4.5.2 Probe Design

In the context of microarrays, a probe can be defined as an engineered oligonucleotide that is designed to target a specific locus on a genome. Theoretically, a perfect probe will only hybridise with a DNA fragment that has a sufficient overlapping sequence that matches the complement of the probe exactly. If the rest of the read process is also perfect, the nucleotide which is next in the sequence after the probe end will be read with 100% accuracy [136] [137].

In reality, the hybridisation process results in multiple sequences binding to the same probes causing noise in the read signal. A probe must be designed with a length that is sufficient to bind to only one location on the genome, and so that other similar sequences do not bind by accident; however, long probe lengths also decrease the binding affinity of a sequence due to thermal considerations, so there exists a balance. Importantly, probes are also affected by variation in the target sequence; variation in the marker being read is desired as it is this data which is to be captured for further analysis, however, variation at other positions in the probe can affect binding affinity as well as causing similar sequences

to bind by accident. Structural variants; insertions, deletions and indels can also cause probe hybridisation to fail or bind to unexpected sequences in undetectable ways [136] [137].

New genetic variation is being discovered continuously therefore even though probes are designed with the latest variant and sequence databases in mind and are assessed for local variation which could affect the hybridisation quality, the nature of variation makes building a perfect probe impossible. For this reason, microarray manufacturers will often release updates to their arrays that remove faulty or poor-quality probes throughout the lifetime of the product.

#### **1.4.6 Illumina Genotyping**

Our software tool REMEDY is targeted to work with Illumina genotyping file outputs and encoding schemes. We also use Illumina Infinium bead chip technology in our case studies presented in this thesis; therefore we will discuss Illumina technology in more detail in this section.

##### **1.4.6.1 Bead Chip Technology**

Illumina bead chip technology is a proprietary method of manufacturing and reading DNA microarrays. To manufacture a bead chip, a silicon wafer is etched so that it contains millions of evenly distributed wells approximately 5.7 microns apart. 3-micron silica beads are next arranged on the chip so that they lie within the wells; the pre-manufactured oligonucleotide probes are then attached to the beads so that each bead has thousands of copies of a single probe. After hybridisation and enzymatic extension, each bead's light signature is read with a laser which can later be interpreted as a base call for each allele [138].

The raw light intensity signals that result from the laser reading the fluorescent tags attached to the microarray probes are stored in IDAT raw intensity files. To convert IDAT files into base calls, they must be processed by Genome Studio [139], Illumina's free proprietary software for managing genotyping data from their genotyping products. The IDAT files are read by assessing the colour and intensity of light on each bead. Illumina technology is designed to process two colours of fluorescent probe tags and hence, is capable only of reading two different signals and is therefore optimised for bi-allelic variants. One possible allele is tagged red and the other green, when reading together on the bead they produce different colour combinations dependant on how much of each allele is represented on the bead. For example, if the red represents T and green C, if the signal is red then this shows

a clear allele of TT; if the signal is green this shows CC, and a colour mixture would show mixed alleles of TC. When combined with a map of which beads map to which probes, the intensity calls can be associated with the variants, encoded and outputted to a tabular file format for further processing [138].

#### **1.4.6.2 Manifest Files**

Illumina manifest files are available from the manufacturer's website and contain essential probe information for each microarray product listed. Each manifest file contains information for every probe referenced on the microarray with its internal id, sequence, genome build, alleles and strand designations. The manifest file can be thought of as a bead chip design document; it is critical when analysing genotyping data for consistency, quality and for linking to meta-data sources such as dbSNP.

#### **1.4.6.3 Genome Studio**

Genome Studio [139] stores genotyping projects in the TOP/BOT AB format; however, there are several other available output formats, strand designation schemes and genotype encodings which a user can export data to. The export format and parameters are often at the discretion of the laboratory technician performing the analysis and can be a significant source of inconsistency when merging genotyping datasets.

There are two primary file formats, Final Report and Matrix. Matrix format is a simple format detailing one variant per row with an id column followed by one column per sample for genotype data; this format contains the minimum amount of information required to report genotyping results (Figure 1.16, p.82). Final report format details one variant/sample combination per row allowing for extra columns detailing additional probe or genotype meta-data such as quality score or probe information; this is at the cost of much larger file sizes (Figure 1.17, p.82).

Name	289	290	291	292	293	294	295	296
rs1000000		CC	TC	CC	CC	CC	TC	TC
rs1000002		AA	AA	AG	AA	AG	AA	AG
rs10000023		TG	TG	TG	GG	TG	TT	TT
rs1000003		CC	TT	TT	TT	CC	TC	TT
rs10000030		CC	CC	TC	CC	CC	CC	CC
rs10000037		CC	TC	CC	TC	CC	CC	TC
rs10000041		AA	AA	AA	AA	AC	AA	AA
rs10000042		GG	GG	GG	GG	GG	AG	GG
rs10000049		AA	AA	AC	AC	AA	AC	AA
rs1000007		AG	AG	AA	AA	AG	AG	AA
rs10000073		AA	AA	AA	AA	AA	AA	AA
rs10000081		TC	TT	TT	TT	TC	TT	TC
rs10000092		AA	AG	AG	GG	AA	AG	AG
rs10000105		GG	GG	GG	GG	GG	GG	GG
rs10000119		GG	GG	GG	GG	GG	GG	GG
rs10000124		GG	GG	GG	GG	TG	GG	GG
rs10000154		AG	AG	AA	AA	AG	AG	AG
rs1000016		TT	TT	TT	TT	TT	TC	TT
rs10000160		GG	GG	GG	GG	GG	GG	GG
rs10000169		AA	AG	AA	AG	AG	AA	AA
rs1000017		AC	AA	AC	AA	AA	AC	AA
rs10000174		AC	CC	AA	AC	AC	AC	AC
rs10000180		AG	AG	GG	AA	AG	GG	GG
rs10000185		TC	CC	CC	CC	TC	CC	TC
rs10000209		CC	CC	TC	CC	CC	CC	TC

**Figure 1.16:** Example of a Illumina matrix report.

Processing Date,10/14/2010 3:47 PM											
Content,,HumanOmniExpress-12v1_C.bpm											
Num SNPs,731442											
Total SNPs,731442											
Num Samples,89											
Total Samples,285											
[Data]											
SNP Name	Sample ID	Allele1 - Forward	Allele2 - Forward	GC Score	X	Y	X Raw	Y Raw	B Allele Freq	Log R Ratio	
rs1000000	NA18500	C	C	0.8534	0.013	1.669	831	20738	1	0.1613	
rs1000002	NA18500	G	G	0.8691	0.06	0.782	1182	9682	0.9771	-0.3282	
rs10000023	NA18500	T	G	0.8258	1.236	1.178	15159	15014	0.4944	0.1227	
rs1000003	NA18500	A	G	0.9012	0.85	0.649	10209	8266	0.4691	0.1484	
rs10000030	NA18500	G	G	0.6111	0.028	0.443	642	5235	1	-0.0033	
rs10000037	NA18500	A	G	0.9049	0.621	0.509	6955	6091	0.4719	0.0394	
rs10000041	NA18500	G	G	0.7978	0.042	1.203	981	13785	1	0.2001	
rs10000042	NA18500	T	C	0.8771	0.471	0.574	5947	7314	0.4637	0.2069	
rs10000049	NA18500	A	A	0.8986	1.151	0.043	15184	1113	0	-0.0439	
rs1000007	NA18500	A	G	0.9152	0.658	0.664	7377	7840	0.5083	-0.2044	
rs10000073	NA18500	T	C	0.878	0.617	0.705	8482	9868	0.4975	0.0569	
rs10000081	NA18500	T	C	0.8949	0.579	0.547	7223	7011	0.4818	0.0938	
rs10000092	NA18500	T	C	0.9066	0.437	0.409	5508	5267	0.5176	-0.047	
rs10000105	NA18500	G	G	0.8625	0.076	1.073	1580	14655	0.9905	-0.1036	
rs10000119	NA18500	C	C	0.8697	0.027	1.189	872	14803	1	-0.0687	
rs10000124	NA18500	C	C	0.8325	0.067	1.156	1335	14407	0.9853	-0.2245	
rs10000147	NA18500	A	A	0.4774	0.39	0.05	4737	894	0.0329	-0.2512	
rs10000154	NA18500	G	G	0.9059	0.044	0.677	936	8468	0.99	-0.1778	
rs1000016	NA18500	A	A	0.8689	0.657	0.051	7222	953	0.0098	-0.0006	
rs10000160	NA18500	G	G	0.4542	0.091	0.789	1524	9872	0.9723	0.2885	
rs10000169	NA18500	T	T	0.9332	1.097	0.051	13208	1023	0.0058	-0.0711	

**Figure 1.17:** Example of an Illumina final report.

### 1.4.7 Multi-allelic and Structural Variation

Illumina microarray probes are read with either red or green fluorescent tags and so are only able to detect the relative mix of two different bases in a sample at a given locus. An optimal probe design would incorporate only bi-allelic SNVs however, design considerations and error based on changing variant information results in probes that target tri-allelic, quad-allelic or structural variants.

Tri-allelic and quad-allelic SNVs are difficult to process with a two-way tagging system and are therefore generally unsuitable for accurate genotyping on Illumina microarrays [140]. Genetic variance data changes as new submissions are made to dbSNP resulting in some SNVs initially classified as bi-allelic before being re-classed as tri or quad in a later database version; they can be initially included on a microarray design as a bi-allelic SNV before being upgraded at a later date.

Structural variation can also be read on a microarray by interpretation of one or more probes that target different points of a structural variant. For example, a deletion results in a genomic locus changing from a G to a T, this can be used as evidence that the sample has a structural variant over the reference even though the exact variation has not been sequenced. This methodology is problematic as it is only circumstantial evidence and can be easily confounded by other variants in the same position. Furthermore, genomic regions where structural variant exists can be a factor in hybridisation performance and cause noise from non-specific DNA hybridisation [141] [142].

### 1.4.8 Data Consistency and Merging

Genotyping data is generated in batches, often by an external lab. The Genome Studio export process (unless strictly specified) can be different dependant on the lab or lab technician that is carrying out the analysis; therefore, batched data must be checked for consistency and merged as a critical first step in genotyping based genomic analysis. As a matter of standard practice, data must be checked for the correct genotyping chip (including version), genome build, encoding scheme, strand designation and output format. Ideally, all of these variables should match across batches.

It is becoming increasingly common for large-scale genotyping analysis projects to source data from multiple project sources, including both private and publicly available data. Inadequate consistency checking and incorrect merging processes can result in widespread errors in downstream analysis.



### 1.4.9 Positional Cloning

Positional cloning has been one of the most successful methods used to enrich the genotype to phenotype model for an organism. Positional cloning uses both *in vitro* and *in silico* methods to find associations genomic regions and phenotypes (primarily in disease).

There are two principal methods of detecting disease association. Direct sequencing attempts to identify a protein linked with a phenotype that can then be sequenced to find its amino acid code. The protein sequence is then mapped to a gene so that it can be characterised [143]. So-called reverse sequencing or positional cloning attempts the opposite approach whereby comparative analysis of phenotype carrying samples vs non-phenotype carrying samples is performed independently of any functional knowledge of the phenotype being studied. In this way, regions of genetic difference between groups associated with the phenotype can be identified and studied 'position first' with no previous knowledge about the genomic region needed.

The areas of interest identified via positional cloning are known as Region of interest (ROI). Identifying ROIs from positional cloning is one of the most common applications of genomic analysis. ROIs are vital as they identify genomic regions which have some causal relationship with the target phenotype and provide a focus for further analysis. There are two main methods of positional cloning: Association studies and Linkage analysis, both are discussed in more detail below.

#### 1.4.9.1 Linkage Analysis

Linkage analysis is a form of association that uses genotyping data from families combined with a family tree and trait inheritance model to reveal markers which tend to be inherited together. If an unknown locus associated with a disorder is included as part of the set of input markers, then a region of interest can be inferred from which markers tend to be inherited with the unknown trait locus. The term linkage refers to the relationship between the distance between two markers and the probability that they will be split by recombination and thus not be inherited together. The closer two markers are, the less chance of a recombination point between them; therefore, the larger the chance they will be inherited together. If this information is combined with a family tree and a model of the inheritance pattern of a trait including penetrance, associations can be drawn between markers and the target trait [88] [144].

Linkage analysis is used in the study of rare diseases where well-defined families exist

that show the explicit inheritance of a disease phenotype; it is a powerful technique that has been used extensively for positional cloning in genetic analysis. We do not discuss linkage analysis in more detail as the technique was not used in our case studies presented in the results.

#### **1.4.9.2 Association Studies**

Association studies are a form of genetic analysis where the genetic information of a set of samples is compared to discern if, at any position, one sample or group of samples are genetically significantly different from each other. In association analysis, the dependant variable can be qualitative or can be continuous (quantitative). For qualitative variables, the most common study design is to have a binary case/control dependent variable such that the target trait exists exclusively in the case group while the control group forms a cohort of 'normal' participants with respect to the target trait. For quantitative analysis, one or more variables can be used to perform classical association testing such as regression and T-testing. Importantly, non-disease traits (e.g. ethnicity) might also associate with a particular genetic locus but not directly cause disease if those non-disease traits are associated with disease through mechanisms unrelated to that genetic locus such as population stratification. These traits must be controlled for during analysis to obtain accurate results.

Genome-wide association studies (GWASs) are a significant category of association studies that search for association across the whole genome and have been instrumental in identifying disease-related genetic variants [145]. They are used for positional cloning when the input dataset contains unrelated samples and are therefore unsuitable for linkage analysis. GWAS studies rely heavily on the number of cases and controls in the study to yield significant results; the total number of cases and controls as well as the proportions of cases to controls, affect the study power, whose requirements can be calculated from the estimated effect size of a trait in a population [146]. With common diseases such as diabetes, this may require many thousands of samples to achieve sufficient sample power while rare diseases generally require fewer samples to reveal significant markers. With rare diseases, the rarity, correct diagnosis, identification and collection of samples can make gathering sufficient numbers for a study challenging. Both rare and common diseases often require international collaboration and pooling of samples to achieve genome-wide significance for potential causal variants [145].

For GWAS the primary source of biological input data is either whole-genome sequencing data or genotyping data. Other relevant biodata about the patient and phenotype data relating to disease meta-data such as progression or sub-typing may be collected and used to refine GWAS results further. GWAS have been traditionally performed with data

from relatively low-cost bead-chip genotyping and have been very successful in identifying associated loci in rare diseases.

The success of a genetic association study depends on good study design, marker selection and quality control protocols [147] [148] [149] [150]. Our REMEDY software tool was designed to prepare data for GWAS and we also present a GWAS as a case study in our results section; hence, we present some basic terminology and theory for standard GWAS protocol in the sections below.

**1.4.9.2.1 Basic Allele Testing** Tests for genetic association in GWAS are performed using SNVs; the data for each SNV can be encoded as minor allele  $a$  and major allele  $A$ . The ref allele and alt allele for  $n$  individuals for a single marker can be represented in a 2x2 contingency table against cases and controls (Table 1.2, p.86).

An allelic association test can be performed by applying a chi-squared  $\chi^2$  test for independence of rows and columns; the null hypothesis is that there is no association between the alleles and the target phenotype. The chi-squared test was developed in 1900 by Karl Pearson [151]; It is considered the standard statistical hypothesis test when comparing categorical data. When applied to an allele count contingency table, a test value can be obtained by the summed differences between the expected outcome of no association vs the actual association (Equation 1.1, p.86) [147].

Allele	$a$	$A$	Total
Cases	$m_{11}$	$m_{12}$	$m_{1\bullet}$
Controls	$m_{21}$	$m_{22}$	$m_{2\bullet}$
Total	$m_{\bullet 1}$	$m_{\bullet 2}$	$2n$

**Table 1.2:** Contingency table for case/control status versus major/minor allele

$$E[m_{ij}] = \frac{m_{i\bullet}m_{j\bullet}}{2n}$$

$$\chi^2 = \sum_{i=1}^2 \sum_{j=1}^2 \frac{(m_{ij} - E[m_{ij}])^2}{E[m_{ij}]} \quad (1.1)$$

P-values are a measure of the statistical significance of results in hypothesis testing. A p-value is the probability that the observed difference (or a quantitatively larger difference) between groups would have occurred if the null-hypothesis is in fact true (i.e. there is really no difference). A p-value of 1 represents certainty that the null hypothesis is correct; in

contrast, a p-value of 0 represents a complete rejection of the null hypothesis. By setting a threshold value  $\alpha$  we can define the point at which the null hypothesis is rejected. By convention, this is usually set to 0.05 or 0.01 (5% or 1% chance that the observations were due to random chance and not a true rejection of the null hypothesis respectively).

In a GWAS, the concept of Bonferroni correction [152] must be applied to calculate a  $\alpha$  threshold as multiple statistical tests are being performed simultaneously on the same dataset (per marker  $\chi^2$  testing). Consequently, given the large number of markers present in GWAS, the p-value thresholds used for significance are much lower than in other settings. The number of markers used in a GWAS varies on the study, therefore, one would assume that the  $\alpha$  threshold is study dependant; however, there has been a significant amount of research performed in this area that has resulted in a threshold of  $P < 5 \times 10^{-8}$  being selected as the standard for GWAS [153] [154] [155] [156]. The threshold is based on the theory that there is a finite number of loci in the genome; so if millions of markers are used, some of them will identify the same causal locus. The threshold of  $P < 5 \times 10^{-8}$  is therefore based on an estimate of the number of loci in the genome.

The  $\chi^2$  test values can be converted to a p-value by sampling from the  $\chi^2$  distribution for one degree of freedom (Our contingency table is 2x2 for a basic allele test). The p-values for all markers can be thresholded using a Bonferroni corrected  $\alpha$  threshold to find markers which are significantly associated with the case samples versus the control samples.

**1.4.9.2.2 Linkage Disequilibrium** Non-random associations between two or more genetic loci are referred to as Linkage disequilibrium (LD). LD exists as a consequence of several factors but results in the linked inheritance of the loci. Much in the same way as in linkage analysis, specific combinations of markers form haplotypes that tend to be inherited together. Association studies are often performed on genotyped data which is indirect association as the associated SNVs are unlikely to be directly causative for the phenotype; however, SNVs are present with such high density on microarrays that some are likely to be in LD with an underlying causative variant. Associated SNVs in a GWAS conducted on genotyped data are consequently indirect evidence that the SNV or another variant in LD with the SNV is causative for the target phenotype [147].

**1.4.9.2.3 Quality Control** The basic allele test is a simple statistical test that is sensitive to errors in input data. Erroneous data can produce noise in the form of spurious associations that can be misleading and hide any real associations in the data. Strict quality control methodology must be applied to GWAS data before any statistical association testing is performed so that signal to noise ratios and false positives are minimised [147]

[148] [149] [150].

**1.4.9.2.3.1 Call Rate** Call rate is a concept directly related to the experimental quality of genotyping data. When a genotyping platform scans a microarray, a miscall can occur for many reasons; these are represented by 'NC' or '-' in the output file. Call rate is defined as the proportion of called genotypes versus non-called genotypes. Call rate filtering can be applied by marker or by sample; samples with a low call rate may indicate that something went wrong with the sample (examples include contamination or bad pipetting), markers with low call rates may indicate a problem with the probe design. By filtering input data by call rate on both sample and marker, potential sources of experimental error are minimised.

**1.4.9.2.3.2 Minor Allele Frequency** Filtering of low Minor allele frequency (MAF) markers can also protect against sources of experimental error. Consider a marker which has GG called for 9,990 samples and GT for 10 samples. There is a probability that the GT calls are real, but there is also a probability that they are miscalled. The threshold of MAF at which calls are considered erroneous varies depending on the project; if the study phenotype is rare, aggressive filtering risks losing the markers which are causative but, with more common disorders, is it more likely that more common SNVs are causative for the phenotype, and therefore, low MAF can be excluded relatively safely.

**1.4.9.2.3.3 Hardy-Weinberg Equilibrium** In a large, randomly mating, homogenous population, a locus should have probabilistically stable combinations of major and minor alleles from generation to generation dependant on a minor allele frequency. Markers which deviate significantly from the model are said to be out of Hardy-Weinberg equilibrium (HWE). HWE is another tool for detecting potentially erroneous data in much the same way as MAF filtering, the same limitations and dangers of filtering out real associations also apply [157] [158].

**1.4.9.2.4 Principle Component Analysis** Principal component analysis (PCA) is a statistical procedure that converts sets of variables which have a possible correlation into sets of linear uncorrelated values called principal components. The transformation is defined so that the first principle component contains the most variation possible and so on for the next principle component until the maximum number of components specified for the transformation is reached [159]. PCA is a useful tool for viewing variance in high-dimensional data; by plotting the primary principle components against each other, a large

proportion of the variance in a dataset can be viewed in a small number of variance dimensions. This technique is known as dimensionality reduction.

PCA can be used to plot the genetic distance in data by reducing the variance present in a large number of markers down to two or three dimensions of variance. The Euclidean distance between two samples on a plot of the first two or three principal components shows how genetically different they are; this can be useful when grouping samples by their ethnicity [160]. Associations in cases versus controls can be produced by their natural ethnic separation reflected as genetic distance; this can hide true associations and must be controlled for in a GWAS.

**1.4.9.2.5 Imputation** Many GWAS are performed on genotyped DNA samples as WGS for large-scale studies is still prohibitively expensive for many research groups. Genotyping is still informative for GWAS as a significant association for one SNV implies that other unobserved variants on the same haplotype or those in LD may also be causative for the target trait. Indeed, historical studies of <10,000 markers have been very successful in identifying genes responsible for disease [161].

Imputation seeks to increase the resolution of genotyped data by imputing missing genotypes using the both the directly observed genotypes and an ethnically matched reference genome. A statistical prediction model is used to infer missing genotypes that can fill in gaps in unobserved genomic regions [162]. Modern genotyping microarrays have >1,700,000 markers defined on them [163]; imputation can increase the marker density to >10,000,000 [162].

**1.4.9.2.6 Q-Q Plots** Quantile-quantile (Q-Q) plots are 2-dimensional charts that plot values from two probability distributions [164]. The purpose of the Q-Q plot is to show graphically if the two datasets come from the same distribution. Two datasets on a Q-Q plot are said come from the same distribution if the data points fall along the line  $x = y$ . Q-Q plots can be used when analysing the quality of multiple statistical tests from which p-values have been obtained for expected and observed results; such as a basic allele test that has been applied to many SNVs.

An association study performed between cases and controls where there is no association between the target trait and the data will yield a situation where the expected and observed p-values are from the same distribution and thus, will cluster along the line  $x = y$ . In a typical positive association study, there will be a proportionally small number of SNVs that have an associated low p-value; therefore, the Q-Q plotline will follow  $x = y$  except at

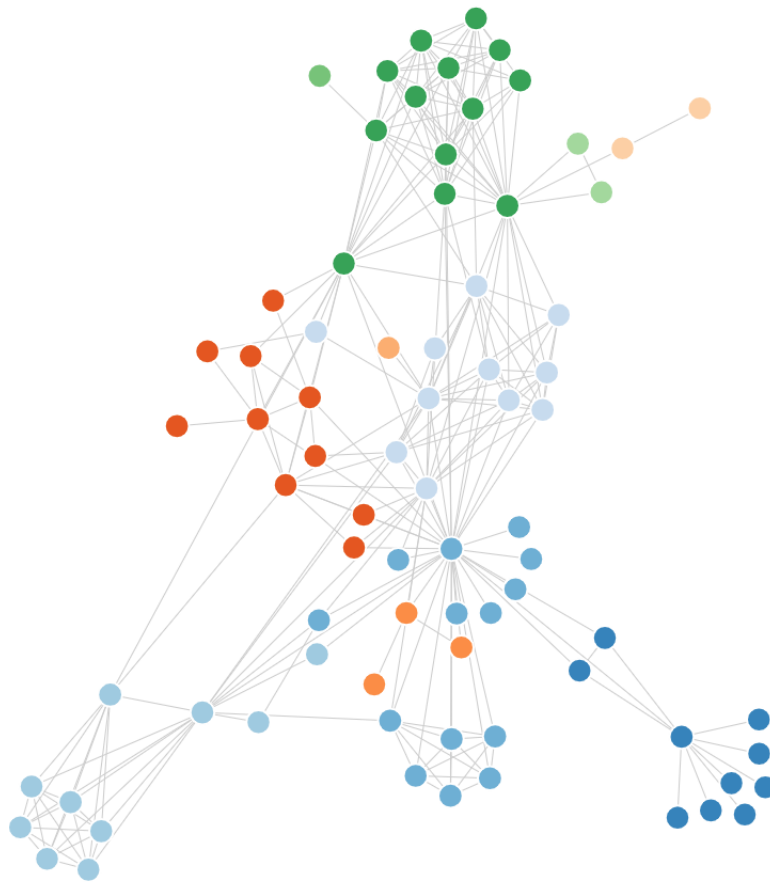
the extreme top right of the plot (lowest p-values if reversed) where there will be a skew to the observed values.

Q-Q plots can be used to identify association test results with large numbers of false positives. Datasets such as these will have large skews away from the  $x = y$  line, showing that there are many p-values with apparent significance. So many significant associations in a genetic association study are indicative of noisy data with a poor signal to noise ratio.

## 1.5 Graph Visualisation

A graph structure represents a set of related objects; nodes represent the objects themselves while vertices represent the relationships, the total set of nodes and vertices defines a graph. Graphs are useful ways of representing data as they can be traversed mathematically using graph theory concepts; they are also useful in the visualisation of connected data. A class of algorithms that can generate visually pleasing graphs are known as force directed graph algorithms; they aim to present nodes so that as many vertices are the same length as possible and there are as few crossovers as possible (Figure 1.18, p.91).

The primary function of regulatory chromatin looping in the genome is to bring distal genomic elements such as enhancers and promoters into proximity so that they can act on each other to regulate gene transcription. If the genomic elements are represented as nodes, we can say that two nodes are connected if they have some relationship in terms of physical proximity with each other, whether this is positional proximity on the genome or proximity as a result of chromatin looping. Force-directed graphs are an intuitive way of visualising chromatin confirmation experimental data.



**Figure 1.18:** Example of a graph of force-directed nodes. The nodes are placed so that none overlap and the edge lengths are equal as much as possible taking into account all other graph generation parameters.



## Chapter 2

# Methods

### 2.1 Genomic Analysis Methodology

We present here our general methodology for conducting the bioinformatic portion of genomic analysis projects.

#### 2.1.1 Golden Rules

Research projects have an inherently dynamic nature, and unless the concepts presented below are observed, a genomic analysis project can quickly become muddled and unstructured. Here we describe several core computer science principles and how they are especially important in genomic analysis.

##### 2.1.1.1 Version Control

Version control is a central tenant of professional programming. It not only allows tracking of changes in a project but provides a framework for multiple people to work concurrently and commit their changes to the source project without ambiguity about which changes take precedence. Version control frameworks also allow for multiple branches of a project to exist in parallel with methods to merge and split development branches.

The dynamic nature of genomic analysis makes correct version control across a project essential. It is advisable to apply version control not only to any source code developed for software tools but also to the data analysis pipeline, documentation, notes and results;

this ensures an absolute time-mapped record of all work undertaken in a project. Changes in data pipeline design occur regularly in a genomic analysis project either in response to problems or to changes in experimental design; it is essential to know which version of a pipeline a result set was generated from, version control provides a structured framework to achieve this.

Git is a distributed version control system that was developed by Linus Torvalds in 2005 for the development of the Linux kernel [165]. It is an open-source project and the most widely used of all source control systems; we therefore selected Git as our source control framework. Git can be managed from the command line, or from a program with a Graphical user interface (GUI); we selected GitKraken as our Git GUI as it is a popular, free program with a rich feature set [166].

#### **2.1.1.2 Naming Conventions**

Another subject core to professional programming practices is naming conventions. While this traditionally applies to program code, in genomic analysis, this can be applied across the workflow as a whole. The premise of naming conventions is that functions, variables and filenames follow a strict process of naming so that someone who reads another's work can immediately understand the type and function of the piece code. Names must balance being concise with providing maximum information as to the type and function of the data or code to which the name refers [167].

In genomic analysis, we apply naming conventions beyond code to every aspect of a project. Defining a standard nomenclature for specific data sets, processes, software tools as well as pipeline code is essential if proper communication about a project is to be possible. Care should especially be taken when naming datasets and sub-datasets; raw data will often go through several rounds of quality control and filtering before being used in multiple downstream workflows. By using good naming convention practice, a downstream user should be able to tell which raw dataset the data or sub-data he or she is working on came from and the general steps that were applied to it to reach the current dataset.

#### **2.1.1.3 Documentation**

Proper experimental process requires that both planning and methodology be rigorously recorded as well as any results or changes. Every data pipeline version or software tool version must be documented describing the inputs and outputs at each stage of processing. Version control must then be applied to this documentation so that it follows each change

in the underlying code [168] [169]. If for example, a colleague wished to view the version of the pipeline from which a specific set of results originated from, the user would only have the full set of information if they could see the documentation from that exact version.

#### **2.1.1.4 Security and Redundancy**

Rigorous data redundancy practices are ubiquitous in the data science world as it is often difficult or impossible to reacquire raw source data. In genomic analysis, the source data will often be patient data including highly sensitive information such as Deoxyribonucleic acid (DNA) sequences and medical history. Once information enters the project domain, it should be securely stored on servers with multiple redundancies both on and off-site. Redundancy should be provided against both hardware failure and human error.

We suggest protecting against local hardware failure by storing data on disks in a Redundant array of inexpensive disks configuration (RAID) [170]. RAID has several different modes of which RAID 5 is the most redundant - protecting against disk failures and reducing the amount of space required to implement it; however, RAID 5 has several performance issues when dealing with large amounts of data. RAID 1 is a simple mirroring of data that sacrifices space for simplicity whereby data is automatically duplicated to two or more disks. RAID 0 does not provide redundancy; instead, it specifies that files be split across multiple disks so that a single disk's speed does not limit read performance. RAID 0 can be combined with RAID 1 resulting in a mirrored and striped configuration known as RAID 10. We recommend using RAID 10 when storing genomic data locally as raw data is most likely to be written once, yet read many times.

To protect against location-based risk such as a building fire and to mitigate user errors such as accidental deletion, we recommend storing long-term backups of data in one or more other locations. Online cloud storage or backups stored in off-site safes are two possible ways of achieving this. Local security can be managed by controlling who has access to data on servers by implementing strict user control policies. Planning data security should also take into account local laws such as the recently introduced General data protection regulation (GDPR) in the EU to ensure that any storage solution meets in the minimum requirements for sensitive information [171].

#### **2.1.1.5 Repeatability**

Repeatability is a core principle of both software development and academic research. Experimental repeatability ensures that results can be independently verified either with

the same data or with different source samples to create an independent replicate. It also allows for internal verification of results and increases confidence in the accuracy of the results themselves. As well as verification, creating a repeatable experimental framework requires adherence to good experimental practice, therefore, acting as a test for the quality of a project design in general.

In genomic analysis, the initial stages of data collection and conversion from biological data to digital such as the genotyping of DNA samples are generally one-off processes due to time and cost constraints. The bioinformatic portion of the project, however, can and should be made with repeatability at its core; to achieve this, all golden rules specified above must be used to create a well-documented, version-controlled and secure analysis pipeline that produces good quality, repeatable results. Data pipelines should be portable and not tied to a single computer or operating system and made open-source where possible.

### **2.1.2 Project Planning**

Proper planning is crucial to the successful execution of projects in any field. Research projects, in general, are particularly difficult to plan as they tend to be open-ended, i.e. although the stages to achieve any initial results can be planned, it is not known where subsequent results will take a project. In the case of genomic analysis, there is always a data gathering phase where biological data is processed, and some initial results can be generated; the storage, organisation and pre-processing of this initial project data can be pre-planned as there are few unknowns. The type of analysis that is to be performed and thus the tools that will be used can also be pre-planned however, because the results generated are inherently unknown (a hypothesis is based on an unanswered question) it becomes more difficult to plan beyond the point of first results. Once this stage is reached, a research project becomes iterative and dependant on how the results support the research question. Are there errors in the results? If they are accurate, what do they say about the hypothesis?

Agile practices originated as a set of software design and project management methodologies. It is an iterative approach to software delivery that promotes incremental delivery of items rather than everything at the end. Work is broken down into iteration cycles where tasks are defined from goals which are then prioritised; the software run is then completed, versioned and documented before any complications, failures or new requirements are noted before the next iteration is planned [172]. Agile practices can be applied to genomic analysis to manage the inherently iterative nature of research; and, when applied alongside the golden rules detailed above, form a comprehensive framework for planning and

managing a genomic analysis project.

For planning of the bioinformatics portion of a genomic analysis project, it is useful to think in terms of pipelines. A data pipeline can be thought of as a directed graph of data which is transformed at specific points. Results can be taken at any point of the pipeline, but data only flows one way. The input to the pipeline is all of the points at which samples transition from biological to digital, such as patient history, genetic data or blood work analysis. The edges of the graph carry data to nodes which are responsible for processing the data; building out a data graph in a design program such as Microsoft Visio can help with visualisation and also provide documentation [173]. A complete pipeline design will be an optimised data flow graph that uses the available source data to produce all results required to test the specified aspects of a hypothesis. Using a data pipeline as a focus for a genomic analysis project encourages good project planning practice and ensures that the project data is always front and centre in the researcher's mind.

### **2.1.3 Pipeline Design**

Data pipeline design begins with all the points where source data enters a project space in digital form. It may be that some data must be converted from analogue to digital as is the case with DNA samples; however, this must be completed before entry into the analysis pipeline. Identification of all raw digital sources of data as the 'start' nodes of the pipeline is the first stage of pipeline design (Figure 2.1, p.99).

To continue the design of the pipeline from raw data, one must consider the set of results which are required to answer the questions set by the project. The challenge is then to fill in the analysis gap between the source data and the results i.e. what transformations need to be applied to the data and in what order to yield the desired results? Each transformation step can be defined as a node and the nodes are then linked with edges along the paths that the data will take.

Once all the edges and nodes have been created, the nodes can be scanned for simplification points. If the node requires multiple programs or it requires significant amounts of splitting or merging, then it may be eligible to be broken it down into smaller steps. This process is repeated until the pipeline is optimally simplified. Version control is applied at all points of design.

The transformation nodes can next be filled out by defining which software tools will be used, the licensing that must be obtained and the versions and operating systems required to run the software. Further constraints may also be placed on the node resolution

dependant on the type of genomic analysis performed. The software running within the node defines node input and output data formats, this usually takes the form of text or binary encoded data in files which follow a specific standard. Inputs and outputs of connected nodes must be matched so that only one format of data travels along an edge. Mismatched inputs and outputs are resolved by including additional processing nodes or by changing the parameters of the node to change the input or output format.

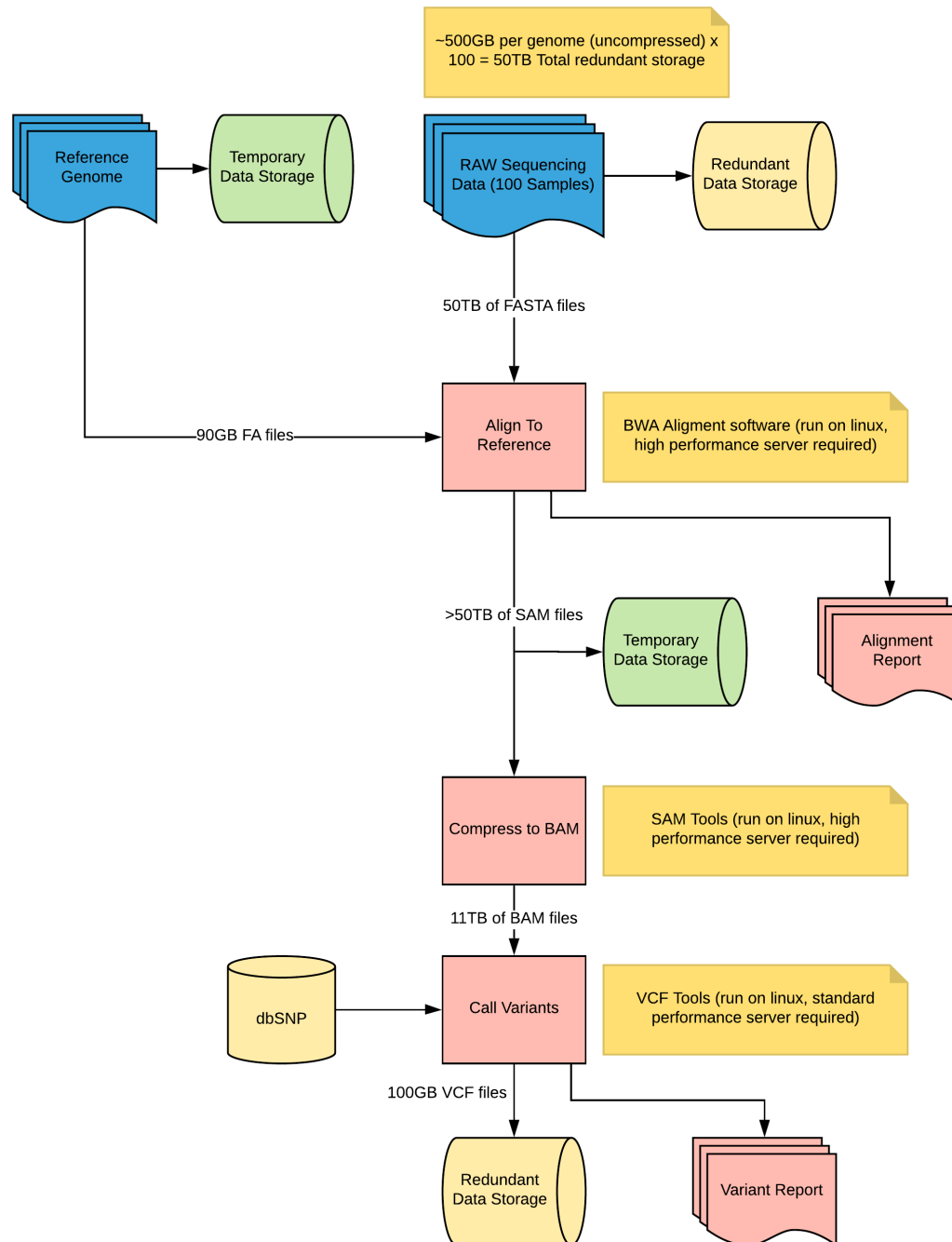
After input and output matching edges can be resolved to estimate the pipeline storage and performance requirements. All data inputs will already have an exact or estimated size in terms of data and the number of samples. Commencing with the start nodes, one can travel along each edge using the information from previous edges to estimate file sizes and storage requirements; this step enables proper planning of the redundancy and security golden rule. Each edge on a graph represents a point at which data will need to be stored either temporarily for the further processing or more permanently as part of a result set; in our methodology, we identify edges that have a large storage requirement and seek to reduce this via compression or by storing the data for less time.

After edge resolution, the estimated data volumes can be used to predict computing requirements at each node. We define at this point a reasonable processing time threshold for the whole pipeline; for a project to flow correctly, the pipeline cannot take an unreasonable amount of time to run. What is reasonable depends entirely on the project, resources and team. For example, In a large team project, it may be deemed that if the pipeline takes longer than an 8-hour over-night run, then this will cause researchers to be waiting for the pipeline to finish and thus be potentially wasting time. If the team is small, and the results take several days to analyse before deciding on the parameters for the next pipeline run, it may be deemed acceptable for the pipeline to be running in the background for several days. The computing power required for the project is a balance between cost and the acceptable run-time of the project.

Once the storage and computing requirements have been appropriately estimated, the servers and run locations of each node can be defined. Given that software may run on different platforms, it is possible that the pipeline is run on several different machines and operating systems. At this point parallelism and batching can be employed to speed up pipeline runs by running nodes in parallel on multiple machines thus maximising processing power utilisation and minimising runtime.

In the final design step, we define how data will be summarised into results. Results comprise two parts, the underlying data and the human-readable report. Often, the human readable portion of a result set is generated after the fact and is not subject to the same stringent quality control of a data pipeline. We propose that human-readable reports are

generated within the pipeline to the same standards, removing all manual processing possible. Results communication is an often-overlooked process but is critical to a project's success as results in binary data form although accurate are not publishable. Well communicated results help drive better feedback from a team and therefore help drive more focused project iterations. All major presentation programs such as Excel, PowerPoint or the Google equivalent have programming Application programming interfaces (APIs) that can be utilised to automated report generation from inside a pipeline.



**Figure 2.1:** Example data pipeline for a simple Next generation sequencing (NGS) workflow. The final result required is defined as a variant report for the input raw sequencing data. A reference genome and a local copy of The single nucleotide polymorphism database (dbSNP) were also defined as a requirement to satisfy the results. An interim alignment report was also required to check sequencing quality. The three primary steps of alignment, compression and variant calling were next defined along with the programs that would be run on each node. The edges were then filled in to link the nodes and estimations of redundant and temporary storage were made for each edge. The resultant pipeline shows that a single high performance Linux server is required that is attached to a RAID array with at least 60TB of space and a non-RAID capacity of at least 10TB.



### **2.1.4 Pipeline Repeatability**

Once the first iteration of the pipeline design is has been completed, it needs to be implemented. One of the principal keys to a successful genomic analysis project is that the results are repeatable from the raw data. Someone from another team must theoretically be able to take the pipeline, run it with the raw data and obtain the same results; this ensures maximum transparency while ensuring good design and implementation practice is used.

Repeatability of experimental results is a complex subject, as biological data is inherently noisy, therefore, it is unlikely the same results will be obtained twice. Instead, results must fall within a distribution defined by the expected model of noise for the experiment. In computing, for most non-stochastic applications and excluding processor rounding errors (which can be controlled for in code), running a pipeline should yield exactly the same results every time.

In practice, it is ensuring the code, source data and results can be read, processed and stored in a platform-independent manner, that is the challenge. If the tools and libraries used in the pipeline are closed source code or unobtainable, then it will not be repeatable by another team. If some of the tools used are custom in-house pieces but have not been made publicly available and user-friendly, then again it will not be repeatable. The same applies if the pipeline is not portable, if it is tied into a set of servers or a piece of computer architecture then it will be impossible to run it elsewhere. Repeatability also has significant implications within a research group; the pipeline will be run many times during a project, and it must be able to produce repeatable results that do not change when everything is kept constant. It must also be internally portable as computer equipment can be changed and upgraded over time. We recommend containerisation of source controlled code to achieve maximum portability [174] [175] [176].

## **2.2 GWAS Methodology**

We present here our methodology for Genome-wide association study (GWAS) that was developed during our Steroid-sensitive nephrotic syndrome (SSNS) project that we use as a case study in the results section. Association study methodology has been developed and applied extensively in literature; however, in our recent work we encountered several problems to which there was no viable solution available, this drove us to develop the proprietary tools presented in this thesis. The Re-coding For Merging of Genotyping Data (REMEDY) tool required specific alterations and improvements to the agreed standard for

GWAS methodology in literature to ensure that input data is correctly filtered and formatted so that maximum value is gained from GWAS datasets when run through the REMEDY pipeline.

The methodology is built around the rules and pipeline design guidelines set out in the previous section and includes our REMEDY software tool. We show here only the parts relevant for application to our reported case study, namely the methodology for projects utilising Illumina genotyping data only.

### **2.2.1 Source Data Processing**

Genomic analysis for GWAS begins with a sample cohort which will be cleaned, filtered and merged with other data sources before final association testing. We identify what samples exist and which data is available for each sample before moving on to any further planning so that any missing sources can be obtained and there is no unforeseen absent data later that affects any upstream analysis.

Sample data for the target phenotype will include a DNA sample as a minimum but may also contain a mix of other sources of data such as blood work, medical records or other potential co-variate information. Some of these sources will be in a biological form such as is the case with unprocessed DNA samples and some will already be in digital form. As we state in our pipeline design philosophy, we seek to convert all biological sources of information into digital before entry into the data workflow; we therefore at this stage set out a plan to achieve this.

In comparative studies, the source data owned by the project may not contain control samples for an association study. It is possible for example for patients' samples to be collected for a study and compared to pre-processed controls from another project to save on the considerable time and cost commitment of generating control data sets; hence, at this stage any additional required control sets and source data are identified.

### **2.2.2 Microarray Selection**

When planning genotyping, correct bead chip selection is paramount to project success; our first stage of planning is defining what markers may be informative for a project. For example, in a diagnostic scan, the variants that need to be tested to examine for a disease will already be known. In an experiment, the hypothesis may focus on a genomic region that has been identified in a previous study that must be tested further; in both situations

the region of interest has already been identified. For a GWAS, the whole genome must be considered; in this case, the optimal strategy is to define a broad set of markers across the genome that focus on areas where variation is most interesting such as in genes and regulatory regions. Study ethnicity must also be considered as many variants have different allele frequencies in different populations; therefore, if the study cohort is European in ethnicity, it makes little sense to include common variants from Asian populations. Marker density or total marker numbers are also key as they increase the resolution of the data.

In large studies, it may be possible to have a manufacturer design a custom chip that contains the specific markers required for a study, however, usually when designing genotyping pipelines the microarray must be selected from a set of pre-existing chip designs. When selecting the optimal chip design for a project, a balance must be found between coverage of the whole genome, specificity to genomic regions important for the study and ethnicity.

### **2.2.3 Sample Preparation and Genotyping**

After the samples have been collected and the genotyping chip has been selected, the sample preparation phase can begin. This process is mostly outside the scope of this thesis; however, it is important to note that we follow the golden rules of naming convention, documentation and version control at this stage. We assign every sample a sample ID before being prepared for genotyping which will stay with the sample for the duration of the project. The sample ID must contain no information that violates data protection (such as the subjects name), but it must also be as informative as possible. The growing practice of sending samples away for genotyping rather than analysing samples 'in house' makes this step especially important as the data will need to be sorted and matched to its original data on return.

We tightly manage sent and returned samples against our sample ID system by grouping samples into batches. Illumina genotyping returns data in the form of IDAT files; we collect these files and assign them to the correct sample ID and batch on return from the processing lab.

### **2.2.4 Illumina Base Calling**

Once the raw IDAT files from the genotyping process are generated, the results are loaded into Genome Studio [139] for base calling. The data can then be exported in several different formats and encoding schemes as described in the introduction. This process

will generally be undertaken by the lab which performed the genotyping and the raw IDAT files as well as the data export from Genome Studio will be returned to the research team. This introduces a source of human error into the project as every lab and lab technician may have their own method of data exporting. Specifying or requesting the exact export parameters used to export the data such as file format and strand encoding scheme is advisable if data consistency is to be maintained. In our methodology, we obtain the raw IDAT files, performing the load and base calling process internally within the project to eliminate any source of external human error. We then export all data as an Illumina matrix file in the DESIGN strand designation scheme. Although the primary strand scheme used by our pipeline is dbSNP FWD, The DESIGN scheme enables re-encoding to the latest version of dbSNP and represents the minimum amount of alteration by the Genome Studio program; conversion to other strand designations is performed by REMEDY.

### **2.2.5 Data Collection and Merging**

Once all data has been converted to digital form, this is defined as the core input for the data pipeline. We next collect, document and create automated redundancy routines for all the raw data to ensure that in the event of data loss or project corruption, the non-repeatable aspects of the project are safe. Data merging is an important aspect of early data pipeline processing and can be surprisingly complex. For example, genotyping samples are generally processed on 96 well plates in batches; the data may come back from the processing lab in pieces and must be checked for consistency before being merged back together again. Also, the sample genotyping output files will usually need to be reformatted to a common tabular form to merge additional data such as disease status or other non-DNA data that is needed for the analysis; this must all be automated, repeatable, reliable and well documented.

A typical genomic analysis project for GWAS will consist of genotyping data from samples that form the case-cohort and genotyping data from control samples which may or may not have been generated by the same project. Additional patient data may also be available such as medical records which must also be merged with the genotyping data. Downstream analysis may also require significant amounts of meta-data to be pulled from databases such as dbSNP to achieve project goals.

We first identify the format, data encoding, strand scheme and designations of all input genotyping data. Association testing requires that all genotyping data be designated to the same strand, to achieve this the format and data encoding must also match; we use REMEDY to perform any re-coding or strand designation changes required. After the genotyping data has been successfully merged, we then clean and format any additional

meta-data; meta-data handling is dependant on the project specification and will not be discussed in this section.

### **2.2.6 Data Integration**

The data integration stage is concerned with merging internal project data with required external meta-data sources such as variant databases. The data pipeline schematic must be consulted to ascertain if any additional meta-data is required; for example, a future analysis step may be to compare the allele frequency of cases and controls with that of different ethnicities, consequently, in the data integration step a requirement to merge allele frequency ethnicity meta-data from dbSNP is created and implemented into the pipeline.

### **2.2.7 Data Quality Control and Filtering**

Data quality control and filtering is perhaps the most critical step in a genomic analysis pipeline. After the initial collation, merging and integrating stages, a pipeline has one or more unprocessed datasets. Many experiments such as association studies are noise sensitive; therefore data must be appropriately controlled and filtered before being passed further down the pipeline. Under-filtering increases the signal to noise ratio and increases the likelihood of false positives while over-filtering may remove real signals leading to false negatives.

Quality control is dependent on the content of the raw dataset and what primary techniques exist to ensure the data is 'clean'. We define in our methodology that data quality control is concerned with cleaning data based only on the data itself and should be independent of the downstream analysis being performed. Call rate filtering is an example of quality control as it is dependant on the intrinsic quality of the source data. Data filtering occurs after quality control and is concerned with cleaning a dataset for a specific experiment. This may result in several filtering sub-pipelines that exist within a pipeline for different downstream experiments. The goal is to prepare the data so that the optimum signal to noise ratio is achieved for the target experiment.

We follow the best practice techniques detailed in several key papers for GWAS quality control and filtering. Zondervan and Cardon (2007) [148] and Clarke *et al* (2011) [147], discuss a general protocol for genetic case-control studies that detail use of sample sizing calculations,  $\chi^2$  tests, Bonferroni correction for genome-wide significance levels and control of population stratification to accurately test for association. Pettersson *et al* (2009) [149], review marker selection for genetic case-control studies on microarrays. Anderson

*et al* (2010) [150], suggest quality control filtering based on call rate, linkage disequilibrium, identity by state, population stratification, sample relatedness and Hardy-Weinberg equilibrium (HWE). We combined these techniques with methodology from previous GWAS conducted within the group [177] to produce our final list of standard quality controls; details of this can be found presented in the case study in our results section (3.1.11).

Genetic analysis pipelines will generally be run over a wide range of filtering parameters to help interpret and explore the results being generated. The filtering stage of the pipeline must be sufficiently parameterised so that it can be efficiently run with many different configurations. The pipeline and subsequent parameterisation are documented and versioned in a way that enables easy identification of the specific filtering configuration used for a result run.

### **2.2.8 Variance Control**

Variance in variables which are being directly measured for results is essential, otherwise, an analysis has no informative value. However, to properly measure a target variable, we must first control for the variance in other co-variants. Controlling for co-variants removes the dependency on any variable except the one being measured so that any signal obtained can then be attributed to the target variable.

GWAS analysis pipelines are comparative, and therefore co-variance minimisation is a common processing step. For example, we may want to control for a specific variable contained in the patients medical records so that this relationship is not shown in results. Another classical example in association studies, is variance control for ethnicity in samples to make sure that the results are not merely showing the differences between nationalities and obscuring any real disease association signal.

We as standard in our methodology, control for ethnicity using Principal component analysis (PCA) analysis [150] [177] and with numerical regression for association testing [147]. Other co-variant analysis may also be performed dependant on the source data available and the research question being answered.

### **2.2.9 Further Analysis**

After all our initial steps have been planned into the pipeline, the data can be fed into one or more experiments from which results can be generated. Further analysis is dependent on the research questions and the type of analysis being conducted, it is mentioned here

because planning for result generating experiments is an essential step in pipeline design and because our second tool described in this thesis, LOOPER, was developed as a result of further analysis experimental requirements.

## 2.3 REMEDY

GWAS are an essential tool for identifying trait-associated variants; in a qualitative GWAS they compare a case-cohort that have a trait to be considered and compare them against a control-cohort which does not have the trait. By statistically comparing the genotypes of the two cohorts, genetic loci can be attributed to an association score to describe how 'different' the case set is from the control set at that locus. To obtain the statistical power to detect significant associations, there must be sufficient numbers of cases and controls in the study (a 1:1 ratio being the most efficient). Increasing the number of cases in a study to improve power is preferable; however, often it is impractical to do so, therefore, it is often the number of controls that is increased as these can be obtained from other studies or public sources [148]. Large case and control sets can be difficult to generate in one batch or even in one project, resulting in the need to merge multiple genotyping datasets to achieve statistically significant findings [178] [179] [180].

Many genotyping data-based GWAS projects require the use of multiple genotyping datasets that are sourced from different studies, that have been processed on different microarrays or that must be merged from multiple processing batches. The problem of merging those datasets into a clean source for further analysis can be problematic and may be a considerable potential source of noise in the experimental portion of an association analysis pipeline [178] [179] [180].

Several methods for combining multiple datasets to increase the power of a GWAS have become popular. One approach is to combine complete GWAS results using meta-analysis [181]. There has also been a significant increase in the amount of online genetic data in repositories such as the European genome archive (EGA) [182] and the Database of genotypes and phenotypes (dbGAP) [183]; furthermore, many journals now require the datasets used to generate findings to be published onto one of these secure locations. The requirement for large GWAS control sets has led to some projects being funded purely to publish good quality control data to an online archive to be used by other teams [184].

Given the potentially large size requirements for GWAS control sets, it is often infeasible to generate both a case and control set within the time and budget constraints of a single project; this has given rise to a new strategy whereby all study resources are put into

collecting and preparing cases, which are then compared to one or more publicly available control sets. This strategy results in a situation whereby a mosaic of genotyping datasets are compared in an association test. GWAS results are susceptible to data quality and merging errors at the level of the genotype, if good quality control and screening are not observed when merging the mosaic datasets, the resulting GWAS may contain noise and false-positives.

Reliably merging different genotyping datasets requires that genotyping data must pass several quality control steps and be encoded to the same strand designation and data format. During the course of several genotyping analyses projects, we developed a strategy for merging multiple genotyping datasets which we subsequently wrapped into a software tool. REMEDY, is designed to assist users with the screening, re-coding, filtering and merging of disparate genotyping datasets; it is designed to perform individual and comparative genotyping data quality analysis, filtering and re-coding in a fast, reliable and repeatable way. REMEDY was applied successfully in several GWAS including the SSNS case study presented in the results section. It forms an integral part of our genomic analysis methodology in the data collection, merging, quality control and filtering stages.

In the following sections we present our justification for developing REMEDY grounded on the experiences of performing a GWAS based on merged genotyping data. We also show our software requirements specification and design methodology; we show the implementation of the REMEDY tool in the results section.

## **2.3.1 Initial Research and Challenges**

### **2.3.1.1 Control Set Identification**

Our team sought to perform a GWAS on SSNS in 2016. We collected a case-cohort of 1408 SSNS samples which were subsequently genotyped on an Illumina microarray with approximately 1,700,000 Single nucleotide variant (SNV)s [163]. We initially had a set of 423 European controls from a previous project which were genotyped on a different Illumina microarray with approximately 712,000 SNVs; they are referred to as the Oxford controls henceforth. Given our methodological guideline of a 2:1 controls to cases ratio, we estimated that we would need at least 3000 controls to perform a GWAS successfully; consequently, we searched for additional publicly available genotyped control sets to supplement the Oxford controls.

We first identified a set of controls available from the Illumina website for testing the



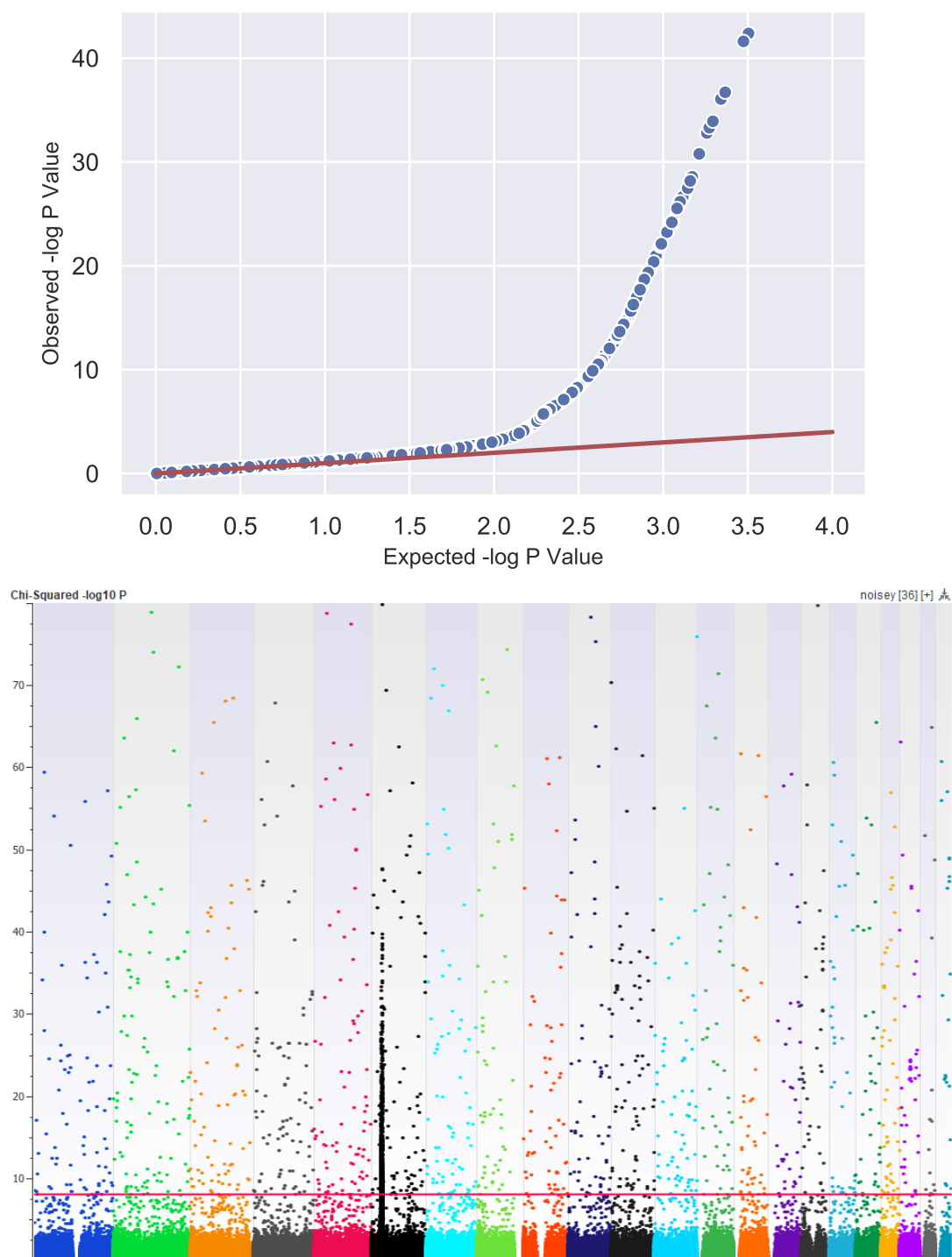
HumanOmniExpress-12 BeadChip [185] (the target microarray for the Oxford controls); these were based on the HapMap project [186] and were split into four files based on ethnicity: 95 CEU (Central European), 92 YRI (African), 47 JPT (Japanese) and 47 CHB (Han-Chinese). We refer to this dataset henceforth as the Illumina ethnicity controls. We next identified a large control set from the EGA [182] from a group known as the Wellcome Trust Case Control Consortium [187]. The control set was split into two subsets, one based on the 1958 UK birth cohort project and other on the UK blood service control group; the cohort was comprised of 5604 individuals of reported European descent and were genotyped on a different Illumina microarray, the Human1-2M-DuoCustom. The total number of control samples available was 6306 spread over three different microarray models (four counting the case samples) and two different Illumina file formats (Table 2.1, p.108). The addition of these datasets took us over our initial control estimate of 3000; therefore, we began to attempt to merge the datasets so that some initial association testing could be performed. At this point, we were unaware of the strand designation merging problem.

Dataset	Source	Case/Ctrl.	Microarray	Sample No.	SNV Count	File Format
<b>SSNS</b>	Internal	Case	MEGA	1408	1,748,250	Matrix GATC
<b>Oxford</b>	Internal	Control	OmniExpress	432	712,893	Matrix GATC
<b>Illumina Eth</b>	Illumina	Control	OmniExpress	270	711,320	Final Report GATC
<b>WTCCC-1958</b>	EGA	Control	DuoCustom	2867	1,106,184	Matrix AB
<b>WTCCC-NBS</b>	EGA	Control	DuoCustom	2737	1,106,188	Matrix AB

**Table 2.1:** A Summary of the case-control sets identified for the SSNS GWAS project.

### 2.3.1.2 Association Test Noise

Each control set was imported into the SNP & Variation Suite (SVS) [188] and merged together. Initial plots of association testing showed systematic noise throughout the whole genome with many thousands of markers reaching genome-wide significance (Figure 2.2, p.109). We launched an extensive study into the causes of the noise which led early on in the investigation to the concept of strand designation. Subsequent literature research resulted in the realisation that there are several different types of strand designation scheme, we then designed some simple theoretical tests to ascertain what kind of noise would result from the merging of two datasets on different strands.



**Figure 2.2: Top:** Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS with no initial processing. The plot shows a significant deviation from the expected at all levels of significance, indicating a systemic issue with the data.

**Bottom:** Manhattan plot showing association scores for our SSNS GWAS. The red line indicates a rough calculation of genome-side significance. The plot shows high levels of noise present in the plot as large numbers of markers are above the genome-side significance level, indicating the something was wrong with the data merging and filtering stages of the GWAS pipeline.

In conclusion, given the simplest case of a single unambiguous SNV with a ref of G and an alt of A, that is uninformative for the disease being studied, we may see that all samples have the genotype G for both alleles; consequently, this would produce an association score of 1, as there are no differences between cases and controls. We then considered a situation where the case-cohort was designated dbSNP FWD and the control-cohort dbSNP REV for both alleles; this would result in all cases having G and all controls having A, a perfect maximum association of 0. If one were to scale this out to the whole genome, this would result in a single band of erroneous association scoring. In reality, if ambiguous SNVs, heterozygous alleles and genotyping error are added in, the false positives would be more randomly distributed as noise, which is what our results showed (Figure 2.2, p.109).

### **2.3.1.3 Strand Designation Scheme Detection**

To further test our hypothesis that the noise in our GWAS was being caused by strand designation mismatching, we needed to calculate the strand designation and strand encoding of our source data to ascertain if any of our datasets were incorrectly merged. To detect strand designation we designed a novel strategy using a point of known strand designation, dbSNP. We hypothesised that by taking the ref and alt alleles of a SNV known to be encoded as either FWD or REV in the FWD/REV designation scheme, one could then re-code the alleles in TOP, BOT, POS and NEG and then match an example genotype to all examples to check which one it most matched.

For example, consider a situation where only the FWD/REV strand designation scheme existed. A variant with a ref of G and an alt of A called to the FWD strand could have a corresponding REV encoding of C and T. Given a genotyping dataset with an unknown strand designation, by comparing all genotypes for the variant to G/A and C/T, a percentage match against FWD and REV can be calculated. If for example, all input genotypes were either C or T, then it is clear that the input dataset is encoded to the REV strand in the FWD/REV designation scheme.

The model can be extended to many designation schemes as, although a dataset encoded to TOP in the TOP/BOT scheme will have some matches with FWD and REV, there will always be a higher percentage match to the scheme which the data is encoded in. Consequently, by taking a variant from dbSNP, encoding it to all possible strand encodings, matching input genotypes and taking the highest percentage match - one can detect the source strand designation and encoding of a genotyping dataset.

#### 2.3.1.4 Strand Designation Transcoding

In order to implement our strand detection algorithm, we required two additional solutions: a source of dbSNP data to pull variant data from (we discuss our implementation of this in detail in the results section 3.1.4, p.134) and a methodology to transcode between the different strand designations and encodings. Given the surprising lack of information regarding this subject in literature and online, this part of the project proved to be very difficult to solve.

We first began with intra-scheme re-encoding. From our research, we understood that FWD/REV in the dbSNP scheme were on opposite biological DNA strands and so the algorithm to switch between them was to find the genetic complement. To test if TOP/BOT was the same, we consulted the documentation provided by Illumina and constructed a test platform using Illumina Genome Studio software. Genome Studio could already transcode between FWD/REV and TOP/BOT; however, the code to do so was closed source, and Illumina would not release their methodology. We, therefore, took a small amount of test data, loaded it into Genome Studio and exported the data in TOP and BOT. Next, we constructed a simple script to find the complement of the output data files and compare them to the opposite strand; the result was a 100% match, thus confirming that TOP/BOT re-coding is identical to FWD/REV re-encoding.

To find an algorithm to transcode between the encoding schemes we again used Genome Studio to export the test data into FWD, REV, TOP and BOT. We additionally downloaded the manifest file for the target microarray from the Illumina support website [189]. The manifest file specified for each SNP, a TOP or BOT designation called *IlmnStrand* along with an allele call, and another TOP/BOT designation known as the *SourceStrand* (Figure 2.3, p.112). By comparing these to the FWD/REV allele encoding in dbSNP, we developed a conversion table that utilised the *IlmnStrand* and *SourceStrand* columns to convert to the dbSNP strand designation scheme from TOP/BOT (Table 2.2, p.112). After successful manual testing, we implemented this into an algorithm and 'slotted' the solution into our overall program for strand designation detection. Our test harness for strand designation transcoding observed the following steps:

1. Implement a new version of the transcoding algorithm
2. Update scripts
3. Detect source designation and encoding of datasets
4. Convert all datasets to dbSNP FWD

## 5. Run GWAS

## 6. Analyse the results

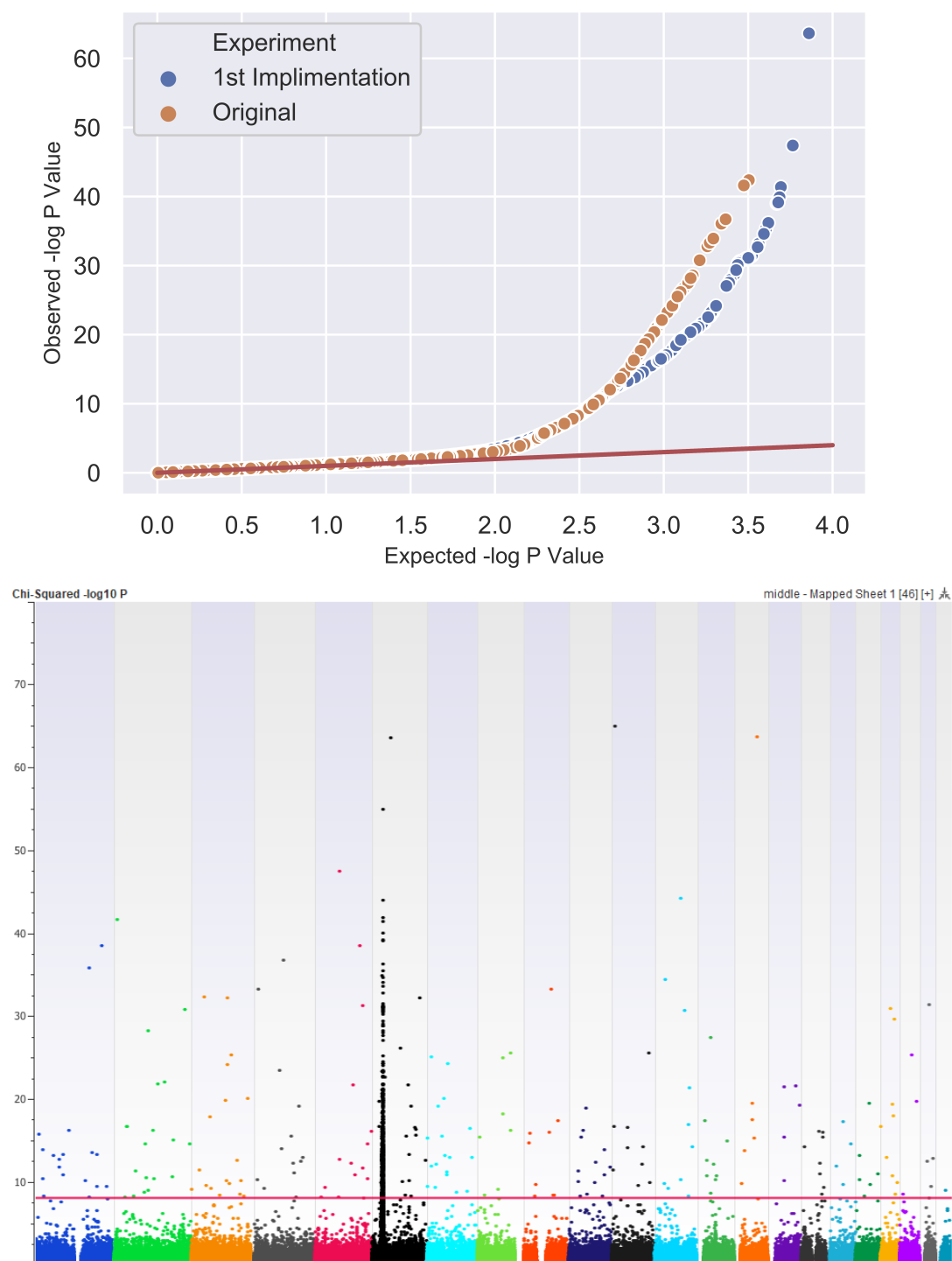
[Assay]					
IlmnID	Name	IlmnStrand	SNP	AddressA_ID	AlleleA_ProbeSeq
10:100012219-GT-0_B_F_2298934103	10:100012219-GT	BOT	[T/G]	22775276	GAGCCATTGAGAGTG
10:100013340-CT-0_T_R_2299260687	10:100013340-CT	TOP	[A/G]	44634128	CACACAGATCTACAA
10:100013459-TCTC-T-0_M_R_2301504613	10:100013459-TCTC-T	MINUS	[D/I]	5686381	CGGCCTACTTGGAGG
10:100013467-GA-0_T_F_2299260694	10:100013467-GA	TOP	[A/G]	69618863	GTAGGGCCAGTCCAT
10:100015474-GA-0_B_R_2299260701	10:100015474-GA	BOT	[T/C]	67800195	GACTGGGTGACGCTG
10:100016685-CT-0_B_F_2299260702	10:100016685-CT	BOT	[T/C]	67673224	GGACCCCGTTCACCT
10:100017801-CT-0_T_R_2299260721	10:100017801-CT	TOP	[A/G]	40782285	AGTGGGGCACGGTCT
10:100017854-CT-0_B_F_2299260722	10:100017854-CT	BOT	[T/C]	20630905	CACTGGCAGAGATGA
10:100020695-CT-0_B_F_2299260732	10:100020695-CT	BOT	[T/C]	25765197	CCTCCTGTGGGGCCTA

**Figure 2.3:** Screenshot of an Illumina manifest file showing the IllumStrand and Allele columns (red box) that we used to construct the first version of the transcoding algorithm.

IllumStrand	SourceStrand	Complement	Reverse
TOP	TOP	NO	NO
BOT	BOT	NO	YES
TOP	BOT	YES	YES
BOT	TOP	YES	NO

**Table 2.2:** Conversion table for initial strand designation transcoding algorithm shown for converting from TOP/BOT to FWD/REV. The reverse column refers to reversing the alleles in the Allele column shown in Figure 2.3, p.112.

We found the resultant Manhattan plots to have a significantly reduced noise profile (Figure 2.4, p.113); however the noise level was still considered to be too significant given the number of apparently random SNVs above the genome-wide significance level.

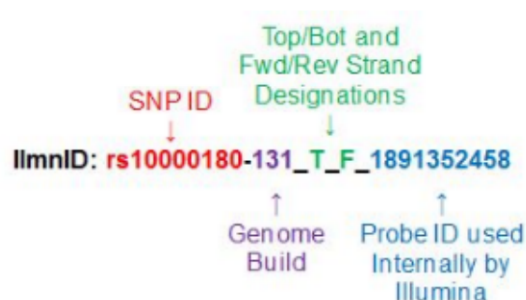


**Figure 2.4: Top:** Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS after application of the first implementation of REMEDY. The plot shows an improved signal to noise with less low-level p-values at the extreme top right of the plot. There is still a significant deviation from the expected at higher p-value levels however.

**Bottom:** Manhattan plot showing the results of the first implementation of our transcoding algorithm. Although the noise levels are reduced when compared to Figure 2.2, p.109, it is still significant.

After this point, many iterations of the algorithm were tried until an obscure Illumina document [190] was located which provided a 'rosetta stone' to the transcoding problem. The diagram showed that in every internal probe ID in the manifest, there exists a mark showing the calculated strand for the TOP/BOT and FWD/REV schemes; for example, T\_F indicates TOP and FWD, while B\_R would indicate BOT\_REV (Figure 2.5, p.114). We had already surmised that a file could be designated in DESIGN scheme where the strand encoding of each genotype was tied to the probe design strand rather than being all to encoded to a single strand. The diagram showed that the DESIGN scheme could act as a 'middle man' by using the internal ID to transcode between the different schemes. For example, to change from TOP to FWD, one would have to convert to the DESIGN encoding first using the ID and then encode to the destination scheme again using the ID.

Next, we constructed a new conversion table (Table 2.3, p.115) and implemented it into the GWAS pipeline as per the test harness steps shown above. The resulting Manhattan plot retained a strong signal (Figure 2.6, p.116), but with minimal noise, thus the strand designation problem was deemed to be solved.

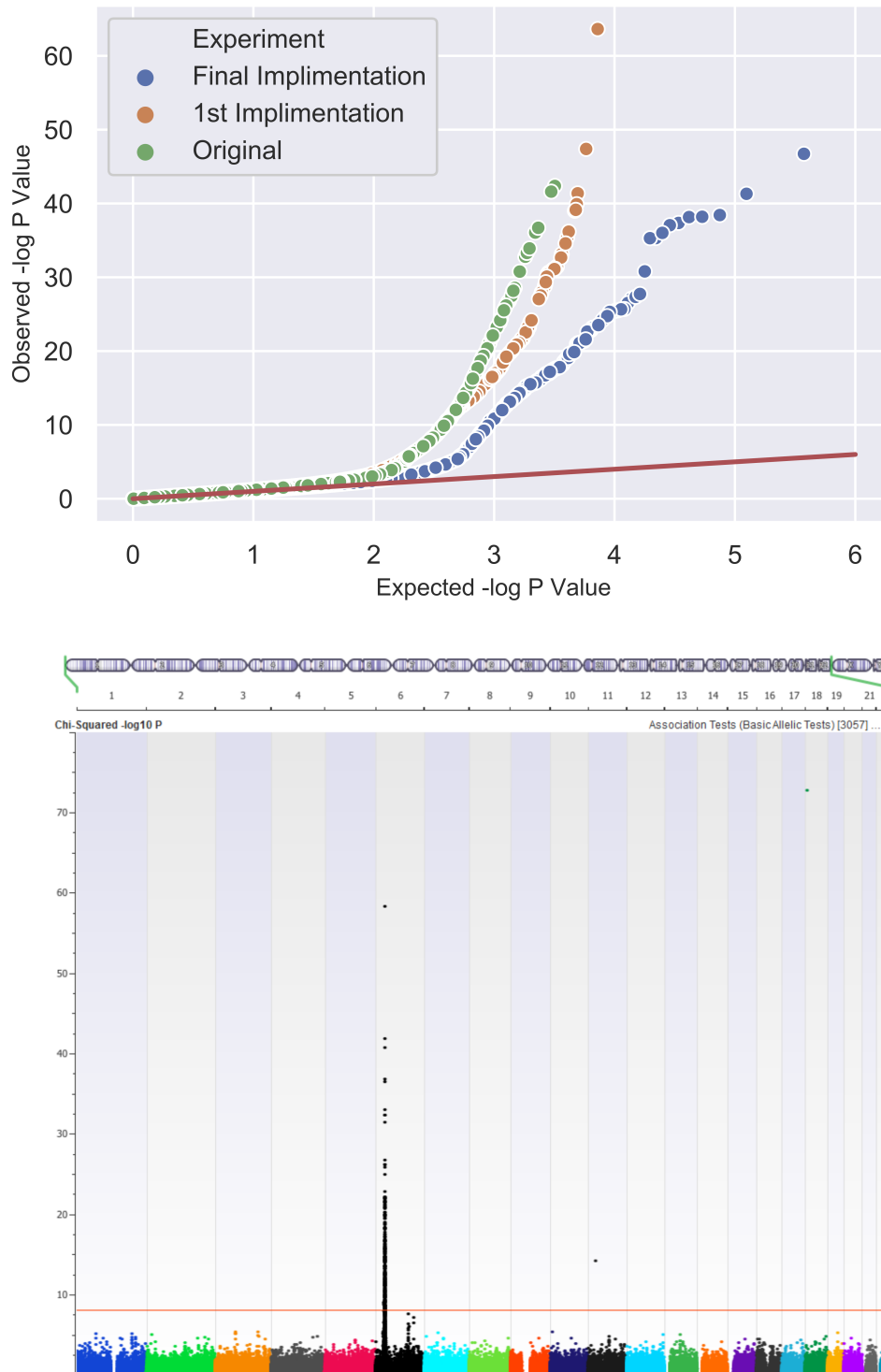


**Figure 2.5:** Screen shot from an online web-page [190] detailing critical information regarding the format of the probe ID in an Illumina manifest file. The text highlighted in green shows a calculated strand encoding for FWD/REV and TOP/BOT, enabling transcoding between the two using the probe DESIGN designation as a 'middle man'.

Illumina ID Key	Probe TB	Probe FR	Source Enc.	Dest. Enc.	Conv. DESIGN	Conv. Dest.
T_F	TOP	FWD	FWD	TOP	None	None
T_F	TOP	FWD	FWD	BOT	None	Flip
T_F	TOP	FWD	REV	TOP	Flip	None
T_F	TOP	FWD	REV	BOT	Flip	Flip
T_R	TOP	REV	FWD	TOP	Flip	None
T_R	TOP	REV	FWD	BOT	Flip	Flip
T_R	TOP	REV	REV	TOP	None	None
T_R	TOP	REV	REV	BOT	None	Flip
B_F	BOT	FWD	FWD	TOP	None	Flip
B_F	BOT	FWD	FWD	BOT	None	None
B_F	BOT	FWD	REV	TOP	Flip	Flip
B_F	BOT	FWD	REV	BOT	Flip	None
B_R	BOT	REV	FWD	TOP	Flip	Flip
B_R	BOT	REV	FWD	BOT	Flip	None
B_R	BOT	REV	REV	TOP	None	Flip
B_R	BOT	REV	REV	BOT	None	None

**Table 2.3:** Conversion table for final strand designation transcoding algorithm. The table shows enumerations of source and destination encoding for different probe strand designations. The last columns show the route of conversion from source to design and then from design to destination. In this table, we use the word 'Flip' as a short version of complement. To reverse from TOP/BOT back to FWD/REV the conversion schemes should simply be reversed.





**Figure 2.6: Top:** Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS after application of the final implementation of REMEDY. The plot shows that number of observed p-values over the genome-wide threshold has dropped significantly to the point where the trend follows the expected line for much longer.

**Bottom:** Manhattan plot showing the results of the final implementation of our transcoding algorithm. The signal to noise ratio has improved to the point where all noise is pushed below genome-wide significance, leaving the real association signal intact.

### 2.3.1.5 Additional Noise Sources

Further refining of our GWAS protocols led to an initial degradation in the signal to noise ratio of our Manhattan plot. The direct reason for this was because of increased numbers of markers being used to test for association, but the underlying reason for the noise was not well understood. Subsequent analysis showed that the GWAS false positives were due to four primary areas: differences in the encoding and strand designation of the genotyping datasets (this was already solved), differences in the variant databases used and quality of variant used for analysis, differences in genotyping chip designs and genome version changes.

Variant databases are dynamic sources of data and are updated continuously. Microarrays are relatively static (some version updates occur over the product lifespan, but these are infrequent) therefore over time, variants included on an array may become unsuitable for GWAS. During our investigation we identified many variants which were initially bi-allelic but had since been upgraded to tri-allelic or quad-allelic status; this resulted in different genotyping microarrays calling SNVs in different ways depending on the version of the variant database to which it was designed. These changes resulted in spurious allele frequency differences during association testing leading to noise. Furthermore, we identified updates in strand designation between versions of dbSNP, which caused unforeseen strand merging errors.

In addition to strand designation updates in dbSNP, we also encountered strand updates in the probe design between different microarrays. We found differences in the probe-target strand despite the fact the probe sequences were identical; this is most likely an artefact of updated internal probe reference mapping pipelines.

Finally, we identified differences in the target reference genome between different microarrays. While not an error *per se* (microarrays are designed to the latest available genome version at the time of manufacture), merging data targeted to different genome versions can result in serious error. Despite this, we found it difficult in some cases to ascertain the exact genome version of our input data and then to find the best methodology to lift all data to a standard reference version successfully.

Each one of the areas discussed above was investigated in detail, and a solution was developed to manually correct each error for a small sample set of data before being scaled into our larger algorithm of strand detection and re-coding.

### **2.3.1.6 Final Algorithm**

The initial scripts developed for correcting the various types of noise sources we encountered during our investigations were subsequently turned into a single piece of software and then scaled up into the software tool presented in the results, REMEDY.

In conclusion, we were required to merge locally generated source genotyping data with control data from several separate projects; initial merging, and subsequent GWAS results had severe data artefacts and spurious results. After an extensive investigation, we discovered that our source datasets had a range of different strand encoding schemes leading to erroneous allele frequencies in subsequent analysis. Within our case-cohort, we had received our genotyping data in batches from our processing lab, but there were inconsistencies in both file format and strand encoding scheme between batches; most likely this was due to human error at the export stage from Genome Studio.

Our control samples were sourced from multiple projects each with their own processing methodology; within these, we found a broad array of file formats, genotype and strand encoding schemes. With some datasets, the source encoding scheme was not indicated or was incorrect; this led to the improper merging of our genotyping data, which then led to many of the errors we observed in our results. To solve this problem, we developed our REMEDY software. REMEDY was primarily developed to scan source genotyping data and detect its source strand encoding scheme and be able to re-encode data to a strand scheme of choice. After further investigation, several layers of SNV cleaning were added to the core algorithm to remove as many potential sources of noise as possible before standard GWAS quality control and filtering was applied.

The initial REMEDY design was to correct for inter-project merging errors arising from differing encoding and strand designation schemes of genotyping data. It is important to note that errors produced by genotyping technology or intra-project batch effects cannot be detected by REMEDY. These effects can be corrected by other methods if the raw intensity file are available but this is not the focus of the REMEDY software suite.

### **2.3.2 Software Requirement Justification**

After the development of prototype scripts to correct the significant data inconsistencies we were encountering in our datasets, we then searched for literature describing the same problems to see how extensive this problem was and if there were any software tools which could support us. At the time of development, we found a single tool, an unpublished yet

publicly available tool known as the Oxford pipeline developed by the Wrayner group [191]. The tool was able to perform lift over functionality and strand correction functions as well as handle A/B genotyping calling; however, the tool creator was required to create a new set of support files for each microarray/strand encoding scheme combination. Given that some of the microarrays contained in our datasets were unsupported and no methods for detecting source encoding and designation were provided, we viewed this as insufficient for our needs.

We realised that the problem of merging genotyping datasets was a relatively common problem that would need to be solved again given the increasing size of both case and control GWAS cohorts. While the current tool available could tackle some areas of the merge process, there was no single tool able to correct for all of the errors which we encountered during our work; we, therefore, sought to develop REMEDY into a tool that could be used by other users and groups.

### **2.3.3 Programming Language Selection**

Given that our tool suite was designed to be modular with many common operations between different modules, we opted for an Object-oriented design (OOD) philosophy [192]. OOD promotes encapsulation of code into reusable modules called classes. Methods and variables within a class allow grouping of pieces of functionality together while enabling functionality inheritance and strict access to variables.

We predicted that our software may need to run on both Linux and Windows-based operating systems; therefore, our language choice also needed to have cross-platform compilation. Given this, we restricted our choice to either Python or C# as our project language. C# is a modern, powerful OOD language with cross-platform compilation options while Python is the current primary scientific computing language and has extensive scientific library support. Python code can be run as a script with Just in time (JIT) compilation and has some OOD functionality.

We selected C# as our main language over python primarily because of the performance requirements of processing large amounts of genotyping data combined with the requirement for a custom, high-speed variant database. C# is a compiled language which means that it must be compiled into an operating system specific executable before it will run. This adds some complexity to the build process as one executable must be made for each OS; however, compiled languages can take advantage of platform-specific operating system and processor directives making them significantly faster than script-based languages. Python has good run-time performance despite the fact it is not compiled, but

C# was selected as it still maintains the edge on native performance.

### **2.3.4 Development Environment**

Microsoft provides extensive development environments both free and subscription-based for C# with a full ecosystem of build and test tools. For this project we selected to use Visual Studio 2017 as it provides a comprehensive set of code writing, debugging and test development tools for C#.

### **2.3.5 Unit Testing**

Unit testing is an essential part of modern software development. During software design and development, computer code is split into many sub-modules called functions or methods. Each function should be designed so that it can be treated as a black box and tested by passing arguments into the function and testing the output; a set of inputs with expected outputs can be defined as one test; a method may have multiple tests to verify different aspects of functionality. Each function can be defined as a unit, each test a unit test; to verify a piece of code works a set of unit tests can be presented to the functions within it, and a pass/fail condition can be ascertained for each test.

Good unit test design will verify a broad set of possible inputs for the function, paying particular attention to the boundaries of a function without having to generate large numbers of tests. For example, for a function which adds two integers together, it would be impossible to test every possible combination of two numbers; instead, we can generate two random numbers for a test and verify the expected output while adding specific tests for boundary conditions such as unexpected values and negative numbers. In our example, we may design tests for negative numbers, infinite numbers and other mathematically strange numbers which may be encountered.

Unit testing has several benefits besides to be being able to verify that a piece of code works correctly. Designing functions which fit into a unit testing framework requires a certain level of design modularity; this maintains a code-base in reusable chunks and forces some level of good code design. Unit testing also increases confidence when changing code to create new features or fix errors; when adding a feature, we can change the function code, add some tests to cover the new feature but also make sure that the existing tests also pass. If all tests pass after a code change, assuming the tests were designed correctly, then the change did not break any previous functionality.

REMEDY is written in C#; therefore we used the Microsoft Unit testing framework provided as part of Visual Studio. For the code portion of Unit testing, we used XUnit [193] rather than Microsoft Test as we prefer to use open source options where convenient.

### **2.3.6 Version Control**

Version control is an essential part of any software project as discussed in the introduction. For REMEDY we used GitHub, a popular online GIT repository [165]. Visual Studio has basic built-in GIT functionality, for more advanced branching and merging we used GitKraken [166], an free GIT repository visualiser. We selected GitKraken for its rich feature set and its ethical charging structure.

### **2.3.7 Program Packing and Deployment**

We selected continuous delivery as our build and deployment method; we opted to use Visual Studio Online as the host for our build server. Continuous delivery is defined as a process by which an automatic build is triggered on a successful commit of code to a branch of a code repository. If the build passes all unit and Integration tests, then it continues to packaging and deployment; this ensures that the latest working build of a piece of software is always available via an automated process, removing the need for a version release cycle. On completion of successful build and test runs the software executables and other build files are copied into a compressed folder and uploaded to a file repository hosted on the Microsoft Azure cloud computing infrastructure [194].

## **2.4 LOOPER**

In this section we present our justification for the requirement of the LOOPER program along with our software development methodology. We also show the methods that underpin our chromatin loop prediction and visualisation algorithms which are presented in our LOOPER implementation in the results section.

### **2.4.1 Software Requirement Justification**

Chromatin looping in the nucleus has been shown to fine-tune gene expression by mediating contact between *cis*-regulatory elements that can lie distant from one another on

the linear genome [195]. As discussed in more detail in the introduction, many non-coding variants are being found located in regulatory elements which then act on promoters over large genomic distances. We discovered a promotor mutation during our study of a new disease, Hyperinsulinism with polycystic kidney disease (HIPKD), that we hypothesised to be affecting the expression of a gene through the alteration of the local chromatin architecture [2].

Subsequent investigation of the local chromatin landscape was challenging primarily due to a lack of software tools and available chromatin confirmation data. After many attempts to draw the loop configuration on paper or develop static models, we sought to develop a program that could predict and display potential chromatin loops to aid in our visualisation of the HIPKD region of interest. After the publication of our initial results for HIPKD, there was a rapid rise of new chromatin confirmation experiments and subsequent experimental data. With the experimental confirmation of chromatin loops, we revisited the LOOPER program to add functionality to visualise loops from existing experimental data as well as the original *de novo* loop prediction.

#### **2.4.2 Programming Language Selection**

As with REMEDY, we opted for the C# programming language with an OOD philosophy. OOD promotes encapsulation of code into reusable modules called classes. Methods and variables within a class allow grouping of pieces of functionality together while enabling functionality inheritance and strict access to variables.

#### **2.4.3 Development Environment**

Microsoft provides extensive development environments both free and subscription-based for C# with a full ecosystem of build and test tools. For this project we selected to use Visual Studio 2017 as it provides a comprehensive set of code writing, debugging and test development tools for C#.

#### **2.4.4 Unit Testing**

As with REMEDY, LOOPER is written in C#; therefore we used the Microsoft Unit testing framework provided as part of Visual Studio. For the code portion of Unit testing, we used XUnit [193] rather than Microsoft Test as we prefer to use open source options where

possible.

#### **2.4.5 Version Control**

Version control is an essential part of any software project as discussed in the introduction. For REMEDY we used GitHub, a popular online GIT repository [165]. Visual Studio has basic built-in GIT functionality, for more advanced branching and merging we used GitKraken [166], an free GIT repository visualiser. We selected GitKraken for its rich feature set and its ethical charging structure.

#### **2.4.6 Program Packing and Deployment**

LOOPER was intended as a collection of scripts rather than a full program; therefore, no program packing or continuous build delivery was required. The program was designed to be downloaded by a competent user as source code and run locally with maximum supervision. By hosting on GitHub, LOOPER is essentially already deployed as the source-code can be downloaded as a Zip file from the web browser.

#### **2.4.7 Common Tools and Methods**

We present here relevant methods for tools and techniques which were used in both parts of LOOPER.

##### **2.4.7.1 Chromatin Immunoprecipitation**

One of the leading experimental tools for investigating Transcription factor (TF) and DNA interactions is Chromatin immunoprecipitation (ChIP). First used to detect histone binding to the heat shock protein 70 gene in *D. melanogaster* [196], it can determine whether or not a protein binds to a specific DNA sequence *in vivo* by enriching DNA fragments associated with the protein. The ChIP process begins with cross-linking of chromatin *in vivo* using formaldehyde which is then sheared into 200-600bp fragments by sonication. An antibody that is specific to the TF of interest is then utilised to precipitate the DNA protein complex before the DNA cross-linking is reversed, leaving the DNA fragments the protein was bound to at the point of cross-linking [197] (Figure 2.7, p.127).



Various methods have evolved to analyse the DNA fragments produced by ChIP as DNA sequencing technology has improved. DNA was initially isolated in targeted stretches and electrophoresed in agarose gel before being blotted and hybridised on labelled probes, to indicate binding specificity [196]. The introduction of microarray technology allowed fragments to be hybridised to a microarray enabling a faster, more accurate assay which was genome-scale rather than localised to a specific region; this is known as ChIP-chip [198] [199]. With the advent of NGS the DNA fragments were able to be sequenced directly rather than being hybridised on an array; this is known as Chromatin immunoprecipitation with next generation sequencing (ChIP-Seq) [200] [201] [202] [197]. ChIP-Seq offers single base-pair resolution, fewer artefacts, greater coverage and a broader dynamic range which produces significantly better data compared to ChIP-chip [203].

We obtained ChIP-Seq summary data from the UCSC browser ENCODE TFBS track which contains ChIP-Seq data for a range of TFs across different cell lines. More information on ChIP-Seq post-processing leading to the summary data we obtained from UCSC can be found at [204]; whenever ChIP-Seq data is mentioned in LOOPER henceforth, we refer to this dataset.

#### **2.4.7.2 UCSC Genome Segmentation Track**

The genome segmentation track in UCSC represents a common set of states across six human cell types that were learned by computationally integrating ENCODE ChIP-seq, DNase-seq, and FAIRE-seq data using a Hidden markov model (HMM) [205]. The chromatin states seek to functionally annotate regions of DNA with keywords such as Promoters, Enhancers, Open chromatin and Closed chromatin. The segmentation track can be downloaded from UCSC browser as a table with data for all six cell-lines.

### **2.4.8 Chromatin Loop Prediction Algorithm Methodology**

We present here our methods for the tools, experiments and source data that we used to develop our loop prediction algorithms.

#### **2.4.8.1 Motif Discovery**

Motif discovery uses collections of sequences to build a statistical model that shows the similarities between them. Position weight matrices (PWMs) were developed to represent

statistical similarity sequence models in a compact form. PWM were first introduced in 1982 as a method of representing Ribonucleic acid (RNA) binding specificity within aligned Messenger RNA (mRNA) fragments [206]. By aligning training sequences together, a probability map can be calculated for the occurrence of a nucleotide at a given position in the aligned sequences. The likelihood then of a new mRNA fragment fitting the model is found by applying the probability matrix to the new sequence. For example, if a position is weighted 100% to A and we see a C in this position for a new sequence, this will score a 0 where as if the position was weighted evenly across the nucleotides, then any letter would score equally. The positions can be added up to give an overall gauge of likelihood of similarity for a new sequence, given the sequence library used to train the PWM.

PWMs can be represented as a matrix (Table 2.4, p.126) but also in a more visual form as a sequence logo where the proportion of the size of each letter in the sequence represents its probability in the PWM [207] (Figure 2.8, p.128). When applied to DNA-protein binding, the PWM can be used to quantitatively assess for binding specificity; they have limitations in that they incorporate no knowledge of molecular binding theory and are simply a statistical tool however, they are generally considered to give a good approximation of a binding site given enough data for most DNA-binding proteins.

TF binding is a major research application for PWMs; several tools and databases have arisen to create, compare and store PWMS for Transcription factor binding site (TFBS). The TRANSFAC [208] and JASPAR [209] databases store thousands of PWMs for TFs based on experimental data of varying quality and quantity.

The most prominent toolset for PWM discovery and analysis is the MEME suite [210]; it is a powerful suite of software for studying sequence motifs and contains a specialised tool for generating binding motifs from ChIP-Seq data known as MEME-ChIP, which generates binding motifs for a TF using randomly sampled, centred, DNA fragment data from a ChIP-Seq experiment [211]. As ChIP-Seq experiments are already TF specific given that an antibody for the TF must be selected for the experiment, the ChIP-Seq data can be thought of as a library of TFBS fragments. Once aligned, these can be fed as a direct input into the classical PWM formulae to give a consensus motif for the binding specificity of that TF.

#### **2.4.8.2 Motif Scanning and Site Orientation**

For motif scanning and orientation we used another tool in the MEME-Suite called Motif alignment search tool (MAST) [213]. MAST scans an input sequence using a list of PWMs using a sliding window to score a length of sequence for similarity. Each window is given a score and the statistically significant sites are outputted to results. If the input of

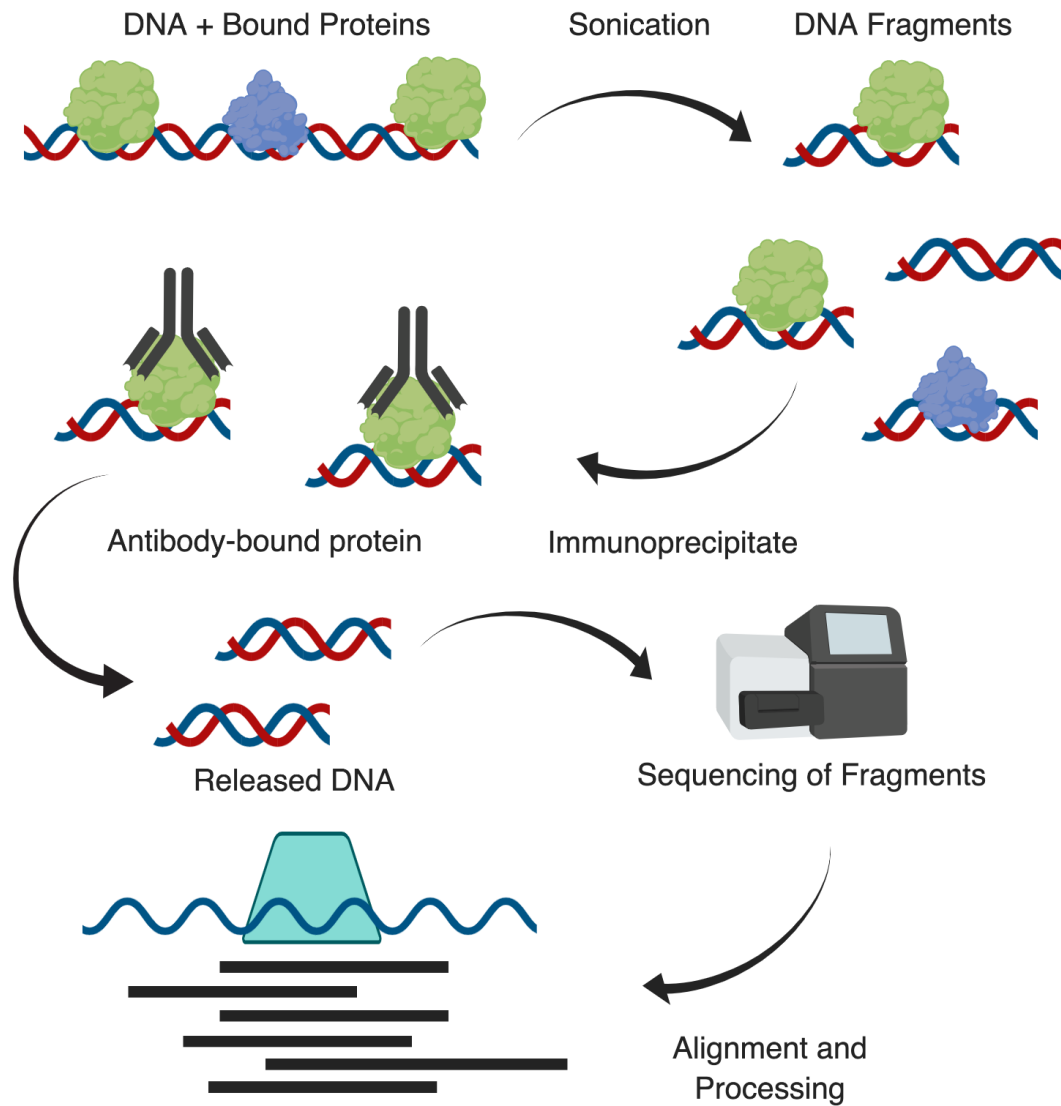
Position	A	C	G	T
1	0.212938	0.264151	0.439353	0.083558
2	0.301887	0.196765	0.380054	0.121294
3	0.029650	0.743935	0.199461	0.026954
4	0.132075	0.867925	0.000000	0.000000
5	0.000000	0.000000	1.000000	0.000000
6	0.000000	0.000000	1.000000	0.000000
7	1.000000	0.000000	0.000000	0.000000
8	0.973046	0.000000	0.000000	0.026954
9	0.185984	0.045822	0.768194	0.000000
10	0.072776	0.312668	0.070081	0.544474
11	0.161725	0.167116	0.536388	0.134771

**Table 2.4:** Probability weight matrix for the ELK4 transcription factor in HEK293 cells reproduced from Factorbook data [212]. Rows represent the sequence position and the columns show the probability of each nucleotide at a given position.

MAST is a PWM generated for a TF, the resultant output sites show probable TFBS with an associated p-value. MAST scans in both the 5' and 3' directions therefore, every site prediction also has an orientation associated with it; this is important for chromatin loop prediction.

#### 2.4.8.3 Predicted Loop Visualisation and Interpretation

The predicted chromatin loops from LOOPER were outputted in the Browser extensible data (BED) format [214] and coded so the loops would appear as black blocks on the screen to show the loop domain in UCSC browser [215]. BED format files can be uploaded to UCSC browser as custom tracks and shown on-screen alongside other tracks to aid further analysis of chromatin looping in a region of interest.



**Figure 2.7:** The ChIP-Seq process. Protein bound DNA is cross-linked and then sonicated to create fragments. Antibodies which target the protein of interest (shown in green) are then used to precipitate the fragments out of solution. The proteins and antibodies are then washed away before the fragments are sequenced. After sequencing, the fragments are aligned on the genome and a statistical curve is drawn over the fragment overlap to give an area of predicted binding. The depth of fragments collected at a site is proportional to the final confidence value of the binding site.



**Figure 2.8:** Web logo for the ELK4 transcription factor in HEK293 cells reproduced from Factorbook data [212]. The positions visualise the relative nucleotide probabilities shown in the PWM in (Table 2.4, p.126).

## 2.4.9 Force Directed Graph Loop Visualisation

We present here our methods for the tools, experiments and source data that we used to develop our loop visualisation algorithms.

### 2.4.9.1 Source Data

There are several different experiments currently that enable the study of chromatin interactions in the genome. The first experiments developed were known as Chromatin Confirmation Capture or 3C based methods. They begin with cross-linking of DNA in the cell using formaldehyde before various digestion and linking steps designed to produce paired-end, tags pieces of DNA which when sequenced, show instances the sequence at one paired-end was in contact with a sequence at another paired-end on the genome. If the paired ends are aggregated and statistics applied, a picture can be built of chromatin contacts. There are 3C, 4C, 5C and Hi-C experiments which build on each other and range in their targeting method and scope. 4C, for example, can target all interactions with respect to a target point whereas Hi-C is a high throughput, whole genome experiment [216].

Hi-C suffers from high levels of non-specific interactions which increase as the distance of the contacts increases; the signal to noise ratio limits the contact resolution which can be achieved with these types of experiments and makes it difficult to study individual interactions between transcription factors for resolving chromatin loops.

ChIP-Seq is a well-known technique for studying TFBS; however, the results are linear and do not infer any interaction between TFBS. Chromatin interaction analysis by paired-end tag sequencing (ChIA-PET) is a technique which incorporates both 3C and ChIP-Seq methodologies to reduce the background noise level prevalent in Hi-C and thus increase resolution. Contacts are effectively filtered by pulling down a target TF with immunoprecipitation, consequently removing non-specific interactions.

ChIA-PET is ideal for studying looping as different loop types can be targeted by selecting the TFs involved in that particular structure. The ENCODE project has a repository of ChIA-PET data which was generated from one protocol, making individual experiments comparable with each other. We selected ChIA-PET as our data source for visualising interactions in a force directed graph with LOOPER.

All ChIA-PET source data was obtained directly from the ENCODE project main data portal in the form of FASTA files.

#### **2.4.9.2 Mango pre-processing**

The ENCODE data repository contains the experimental outputs for ChIA-PET data in FASTA format, the standard output for sequencing data [217]. Paired-end sequencing data has two FASTA files to each sequencing run where each sequence in one file has the corresponding pair in the other file; in ChIA-PET the two sequences represent the points of contact for a single chromatin interaction. This data must be aggregated and pre-processed to reach a state where single pairs of contact are represented by one row so that a graph can be built with minimal processing.

Several ChIA-PET processing suites already exist, we selected Mango based on its ease of use and prevalence in the research community [218] [219]. The primary output from Mango provides all of the required information needed to build a graph; we use the following columns: chromosome1, start1, end1, chromosome2, start2, end2, PETs supporting the interaction and the adjusted P-value of the interaction.

The Mango pipeline uses statistical confidence estimates for interactions and corrects for significant sources of bias in ChIA-PET experiments including differential peak enrichment and genomic proximity. Long range intra-chromosomal interactions of greater than 1Mb and short-range interactions are filtered out by Mango as at both extremes experimental error noise becomes a significant factor. PETs are grouped into putative interactions and then anchored to the chromatin immunoprecipitation peaks calculated in previous processing steps. Chromatin immunoprecipitation peak calling is not discussed in detailed here, more information can be found at [220]. Mango next models the probability of observing a single PET linking two loci as a function of their genomic distance of separation and the product of their read depths; the final interactions are then thresholded to produce the final output file [218].

#### **2.4.9.3 Graph Loading and Visualisation**

Cytoscape is an open source program for visualising complex networks with tight integration of annotation data. It is used in many different domains including Bioinformatics and Proteomics. Given the scope of this program, we deemed it unnecessary to develop a proprietary graph visualisation suite, preferring to concentrate on enriching the annotation that we pass to Cytoscape. In order to view our generated graphs correctly, some manual configuration of Cytoscape is required.

## Chapter 3

# Results

In this section, we present the implementation of the Re-coding For Merging of Genotyping Data (REMEDY) and LOOPER software tools, where we also apply them to two different case studies that demonstrate their effectiveness.

### 3.1 REMEDY

#### 3.1.1 Design Specification

The design specification for REMEDY was split into several key areas: microarray screening and comparison, variant matching and filtering, strand re-coding and output formatting. These were based on the requirements identified during literature research and by our prototype software solution to our merging problem presented in the methods section (2.3, p.106).

We developed our microarray screening specifications with the dual premises that microarray probe design is an imperfect process and that microarrays are designed against snapshots of external sources of information that change over time. Both situations lead to inconsistencies when merging genotyping datasets from different microarrays; therefore, we created some requirements that would ensure these were minimised:



- Detect and filter poorly designed probes that for example, map to multiple genomic locations
- Detect and filter probes designed against structural variants
- Detect the target genome build of a microarray
- Match non-custom probes to variant databases so that meta-data can be used to update or filter out out-dated information
- Summarise to the user general statistics about the probes
- Include features to compare two microarray designs to pre-empt any potential data inconsistencies

Matching variants to The single nucleotide polymorphism database (dbSNP) to assess their viability for Genome-wide association study (GWAS) was considered a critical component to successful data cleaning and merging in our prototype solution. Variant data is limited in a microarray manifest where just the bi-allelic call and its rsID (if it is a non-custom probe) is provided. The rsID can be used to link a target variant into dbSNP where additional meta-data can be matched to enable more in-depth quality control and filtering. To identify the variant inconsistencies we presented in the previous section, we included the following design specifications:

- Match manifest rsIDs to dbSNP for meta-data retrieval
- Detect and filter non-matching variants
- Detect and filter tri-allelic and quad-allelic SNVs
- Detect and filter structural variants
- Detect and filter other variants based on additional criteria dependant on available meta-data

Strand designation was a significant source of data inconsistency when dealing with datasets generated by different groups. There are several competing standards for strand designation discussed in the introduction (1.4.4, p.74); we added requirements to detect and manipulate between all designation schemes available from within Genome Studio.

- Predict source designation based on input genotyping data
- Re-code genotyping data to/from any designation scheme

Given the different input text formats that we encountered, we decided to specify the capability to convert all input formats from Genome Studio into a standard format to ease the process of post-processing merging. We elected only to specify Illumina input file

formats for the initial version, to speed up development time but with the proviso that the software was designed to support more in the future. We chose the Variant calling format (VCF) file format [221] as our output format, as it is a well-supported file format that has extensive merging capabilities built into the formats companion software, VCF tools [221]:

- Read all Illumina Genome Studio export file outputs
- Capabilities to add additional input file formats later
- Convert all input data to VCF format after processing

We also added further general design specifications:

- Variant matching is optimised as much as possible
- The software should be command line and run on both Linux and Windows
- Be usable by non-computer scientists
- Have multiple logging output levels
- Be modular in design so that the analysis can be done in stages that are transparent and easy to record

### **3.1.2 Software Overview**

There is currently no tool which can address all the problems associated with merging and cleaning disparate genotyped datasets in preparation for a GWAS in one package. We developed REMEDY; a software toolset that aims to import, clean and match all input data against dbSNP while assuring strand and genome build consistency. As the source encoding and strand may not be known for a dataset, REMEDY can automatically detect the genotype encoding scheme and the strand designation that was applied to the data and can transcode between common strand formats including FWD/REV, TOP/BOT and DESIGN.

When merging disparate datasets, we required common point of comparison for variant position and meta-data for which we chose dbSNP [124], as it is the most well-known and comprehensive variant database. To perform the high-speed, low memory matching to dbSNP required for efficient processing of large input datasets we developed a proprietary, on-disk database representation of dbSNP called REMEDY-DB. Database files can be generated for any combination of dbSNP version and genome build required and can match by position or rsID.

### 3.1.3 Data Workflow

The REMEDY workflow is split into three different sub-programs (Figure 3.1, p.135). The first scans the target microarray manifest file for unmatched rsIDs to dbSNP, multiallelic and structural variants, invalid variant data and other fundamental errors. Functions are also available to compare manifests together in the case that multiple input datasets are from different microarrays. The result of this stage is the output of several files which report the assessment of the target microarray. The primary output file is a list of variants which pass all the scanning tests that is used by the other sub-programs as a filter.

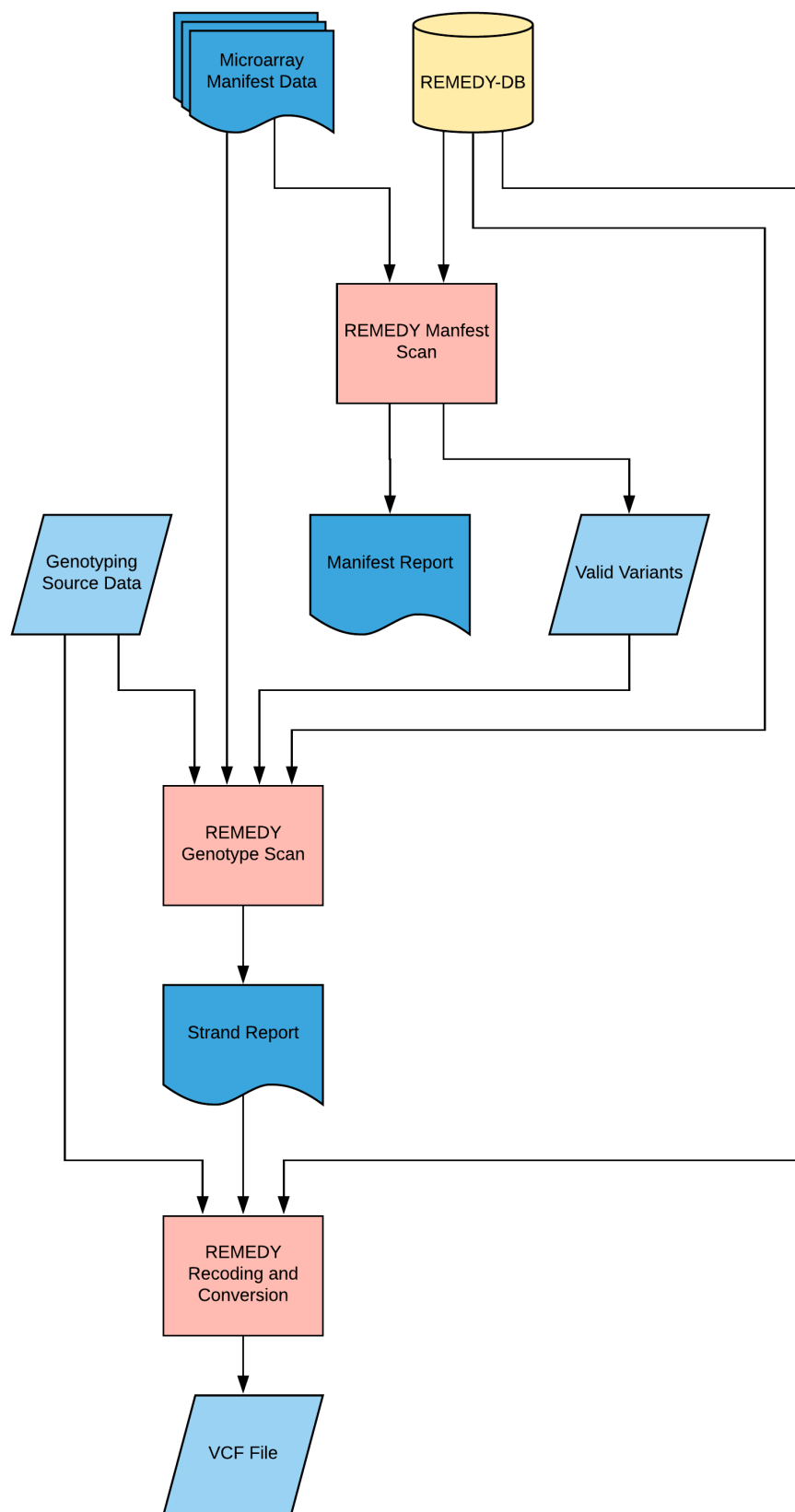
The second sub-program scans the genotyping data and makes a predicted strand designation recommendation. This is important, as to merge genotyping data correctly, all datasets must be designated to the same strand. Source encoding is not always reported accurately, if at all, therefore, this stage forms an essential stage of processing for REMEDY.

The third stage utilises a filtered list of valid variants from the first stage to filter out unwanted variants while giving the user options to re-code data to another strand or strand designation scheme based on the results of the second stage. The sub-program always outputs to VCF file format regardless of the input providing a stable platform to merge the cleaned genotyping data with publicly available tools such as VCF-Tools.

### 3.1.4 Custom Variant Database

Typical modern genotyping chips have millions of probes configured on the array surface; to assess each variant for quality it can be evaluated on its own or it can be matched to a variant database like dbSNP to pull additional meta-data, enabling more in-depth quality control. dbSNP has an online API for requesting variant data; however, matching millions of variants via the internet would take a prohibitive amount of time. The dbSNP SQL database can be downloaded and hosted locally to improve matching times; however, the relational design of SQL databases inherently slows down match times of simple queries for key-value data [222]. Relational databases are designed to structure data so that it can be queried with a structured query language such as SQL; data is modelled into relationships based on unique keys and then split into tables. When data is queried, tables are then linked together dynamically to produce a result set; this design enables flexible queries at the cost of performance [223].

NoSQL databases or non-relational databases store data in a more simplified way by



**Figure 3.1:** Flow diagram of the REMEDY program design. Sub-programs are shown in red, reports and data in blue and the REMEDY-DB database in yellow.

moving responsibility for querying the data correctly onto the user. NoSQL databases can exist both on file and in memory and offer large performance increases for accessing data with simple key-matching queries [222]. Key-value stores are a specialised form of non-relational database where data is stored as a value that is referenced with a simple key; the key can then be indexed in different ways to create a high-performance database [224]. Key-value stores are acknowledged as the fastest databases currently in existence for simple data.

The dbSNP SQL database is relational and contains many forms of additional data on top of the basic variant data required by REMEDY; we, therefore, opted to design our own high-performance key-value store for dbSNP variant data. We downloaded dbSNP variant data in simple tabular form from the UCSC Genome Browser [215]; we then created a custom file-based key-value store called REMEDY-DB.

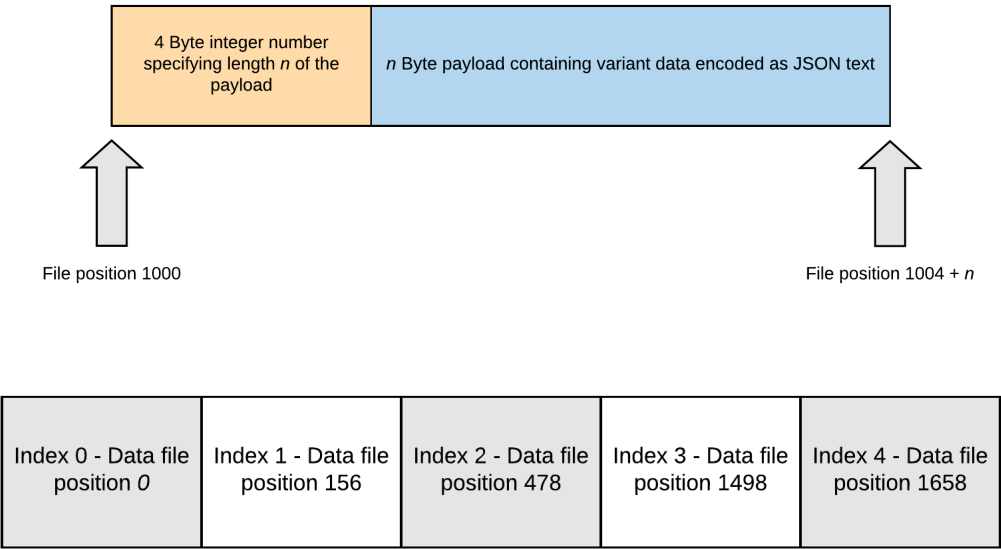
REMEDY-DB is a file-based database which uses a binary tree search system for fast variant matching using rsIDs that integrates directing into REMEDY via a companion C# interface. To build a new database, we start with the tabular representation of dbSNP provided by UCSC browser. We then read the file line-by-line into a C# variant class which contains all of the data required by the current version of REMEDY; by keeping only the exact data required by our software, we aim to keep database size and search times minimised. We next serialize our variant data into data in the JavaScript object notation (JSON) format before converting it into byte code.

JSON is a compact way of representing data that is easily interchangeable to and from C# data structures. We store the JSON variant data using a two-file system (Figure 3.2, p.137). We first create a payload for the variant which contains the length of the variant data in bytes, encoded as a 4-byte integer; we then write the variant data directly after before recording the position of the data object in a separate position file as an 8-byte number. Using a two-file system, a variant can be read using the index number of the variant in the position file while keeping the data as compressed as possible. In a single file system, variant data would need to be stored in fixed length chunks resulting in wasted file space.

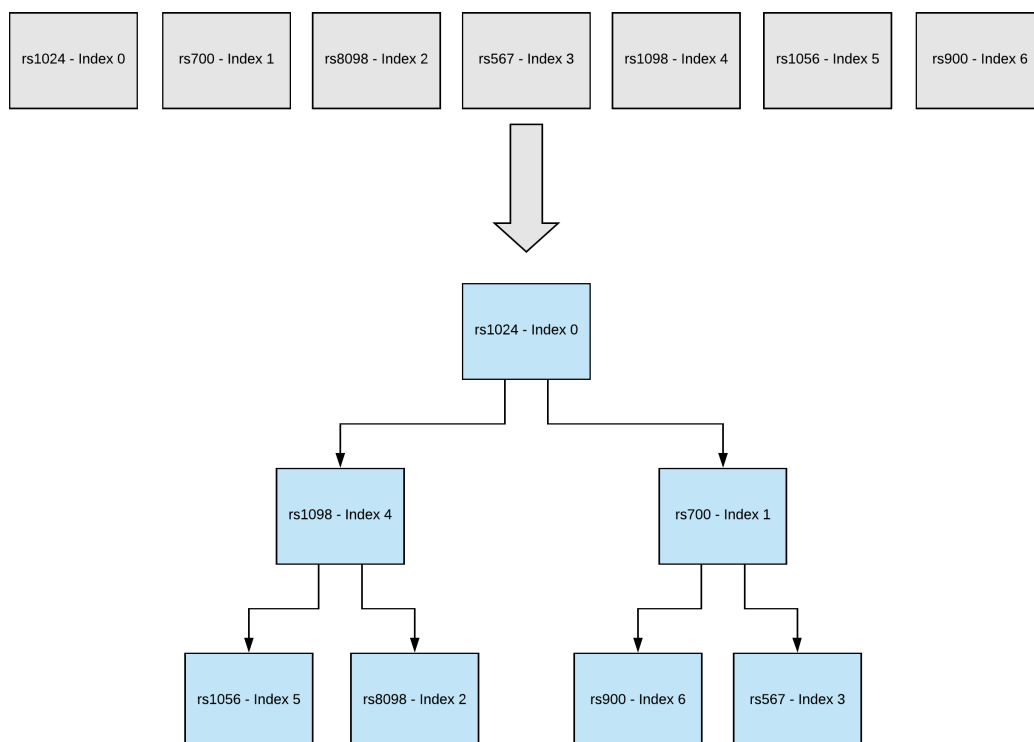
Next, we implement a binary tree search that maps an rsID to an index in the position file (Figure 3.3, p.138). We first arrange all the possible rsIDs in ascending order by stripping the 'rs' position of the identifier leaving a unique number. The median variant of the initial list is found and stored as the root tree item based on the rsID. The list is then split into two sub-lists and the process is recursively repeated, by finding the median value, setting this to the next item in the tree and then splitting the lists again. The process continues until all variants have been added to the tree. Using a binary tree cuts search times by 50% compared to sequential searching in theory. In practice, this is increased

further when processing multiple files by the caching system on a hard-disk; recently used index locations are stored by the hard-drive internal systems and are retrieved an order of magnitude quicker than random access.

When searching for a variant using the tree, the root node is assessed and compared to the search value to determine which branch to travel down; this process is recursively repeated until a leaf node is reached which will indicate the position in the position file to read. The position file can then be used to obtain a location from which to read the payload in the data file. Reading the data file location will first give a length in bytes of the rest of the payload which can then be read as JSON. The variant JSON can be turned back into a C# variant object before being passed on to the REMEDY program.



**Figure 3.2: Top:** Graphic representing how REMEDY-DB stores variant data. A 4-byte integer defines the length of the payload in bytes immediately after. The payload contains all the data for a variant encoded in JSON and are variable length. **Bottom:** Variant payloads are stored sequentially in the data file, the start position of the payload length integers are stored in a separate index file by 4-byte integers and are fixed length. The index file can be used to read variants from the data file.



**Figure 3.3:** Graphic representing the method by which rsID indexes are arranged into a binary tree. For each set of indexes, the median value is found and set as the next level in the tree. The remaining items are split in half and the process is repeated for each subset until none are left.

### 3.1.5 Command Line Inputs

REMEDY is called using a hierarchical function/option system. The primary system command `remedy` can be called with a set of options before a further sub-function can then be called with its own set of options.

```
:>remedy [OPTIONS] function-name [OPTIONS]
```

Command line options are supplied using industry standard command line syntax, with a single dash for a single character option short-code and a double dash for the full option identifier. The following paragraphs discuss the functions in detail.

The primary REMEDY function has several general options for controlling logging level and debug verbosity.

**(-? / -h / --help)** Show Help: Can be used at any function level and shows usage instructions and options available for the respective level.

**(-v)** Verbose Mode: Enables logging output. Logging output defaults to the screen unless otherwise specified. Even if disabled, error logs will still log to the screen as default or to a log file if the **-f** option has been specified.

**(-d)** Debug Mode: Enables debugging output. A more verbose logging output, recommended when program or data errors are encountered, and the user wishes to troubleshoot the program to ascertain where the error may have occurred.

**(-s)** Log to screen: Routes logging output to the screen. Recommended when manually running REMEDY so program output can be viewed.

**(-f)** Log to file: Routes logging output to the file specified as the option argument (can be used in conjunction with the **-s** option to log to screen and file simultaneously). The log file is overwritten on each program start of REMEDY unless otherwise specified.

**(-a)** Append log file: Appends logging output to the log file rather than overwriting.

**(--silent)** Silent run mode: Suppresses all logging, even errors. Only use if absolutely required, the preferred method when minimal logging is **required** is to write to file errors only.

REMEDY can be called with any combination of the options listed above to configure the program to the required specification. We recommend the following starting configuration for manual running:

```
:>remedy -v -s function-name [OPTIONS]
```



### 3.1.6 Input File Formats

REMEDY accepts both the Illumina final report and matrix formats for genotyping data that are available as export options in the Genome Studio software package. The Illumina final report output represents each line as a combination of sample and variant (Figure 1.17, p.82); a more compact representation is available via the matrix format where each line represents a variant and each column a sample (Figure 1.16, p.82). The matrix format is the preferred input file type for REMEDY as it provides all the required information in the most compact form. If the Illumina final report format is supplied at input, REMEDY will auto-convert the file to matrix format before continuing with processing.

REMEDY also requires manufacturer data files for the microarray the samples were genotyped on; these are available from the Illumina website from the support section. As a minimum, the manifest file that details all the probe information is required for microarray scanning; however, further files can be supplied such as rsID mapping and comment files which supplement the manifest and increase the accuracy of processing. Not all microarrays have all supporting files. Full details of all required input files are detailed below in the function descriptions.

### 3.1.7 Primary Modules

Here we detail the operation of the three primary sub-programs in REMEDY.

#### 3.1.7.1 Illumina Manifest Scanning

This sub-function can be called using the keyword `scan-illumina-manifest`. Its role is to scan, compare and evaluate Illumina manifests and supporting files to determine a set of valid variants that can be used for association studies. The minimum requirement for this function is the manifest CSV file for target microarray, but additional rsID mapping files and comment files can also be supplied to assist the scanning process. If mapping or comment files are available for the target microarray, we recommend that these are downloaded and included as inputs as default. The function outputs three files: *summary.txt*, *mismatches.txt* and *valid.txt*.

The summary file contains all the data used to make the calculations for the valid variants file. It can be used as a reference when troubleshooting or when answering questions regarding why a variant was included or not included in an analysis.

The mismatches file contains the strand mismatches which were detected during processing; this is discussed in more detail in the strand mismatches section below.

The valid file contains the same information as the summary file but with valid variants only. This file is primarily for use by other REMEDY sub-functions as a list of valid variants.

`scan-illumina-manifest` option details and example syntax are shown below:

**(--input)** Input file path - **required:** Specifies the file path to the manifest file that requires scanning.

**(--output)** Output file path: Specifies an output folder to write the report files to, if this option is not specified then the program will still run and log but will not write any reports.

**(--prefix)** Output file prefix: Prefixes filenames before writing to the output folder. Use if writing multiple report runs to the same folder so that the results are not overwritten.

**(--rsid-map)** rsID mapping file: Path to an rsID map file. Some manifest files have an internal Illumina generated ID as the marker id. To match to `\gls{dbSNP}` for further analysis, the rsID of the marker is needed; this is supplied as a marker mapping file by the manufacturer. Failing to supply this file with data that contains Illumina generated internal marker ids will result in a low number of matched markers.

**(--comments)** Comments file: Path to a comments file. Some manifest files have an accompanying comments file which detail potential problems with the probes included on the chip such as a probe sequence that maps to multiple positions. Supplying this file to `\gls{remedy}` will cause the program to take this additional information into account when determining if a variant is valid or not. It is recommended to supply this file as default if it exists.

**(--comparison)** Comparison file: Path to a comparison manifest. Specifies another manifest to compare the target manifest

too. Discussed in more detail below. Useful when merging genotyping data from multiple sources after selecting a primary microarray to merge other datasets too.

**(--comparison-map)** Comparison map file: Path to rsID mapping file for comparison manifest.

The different processing stages of this sub-function are detailed in the following sections.

**3.1.7.1.1 Probe Screening** The manifest file is first screened for Single nucleotide variants (SNVs); variants with the *IlmnStrand* column in the manifest file marked as TOP or BOT are classed as single nucleotide variants whereas any other designations are marked as structural variation and removed from the final valid variant list.

The second stage of processing takes the comments file (if supplied) and uses it to filter potentially erroneous variants out. Common design problems with probes can include multiple position mapping and mapping to Pseudoautosomal regions (PARs). Probes are designed with a finite length which can result in a single probe matching to more than one position on the genome, which can cause incorrect base calls and produce noise in a GWAS. PAR-like regions are genomic regions in allosomes that are inherited in an autosomal manner. These areas of the genome can have regions that are homologs to autosomal regions and thus are another potential source of noise as the probe may map to either region.

Potential probe design problems that are identified on genotyping chips by the manufacturer are included in a comments file available as part of the chip documentation. Illumina comment files only contain negative comments; therefore, any probes which contain comment lines are logged and then filtered out by REMEDY.

**3.1.7.1.2 rsID Mapping** Once initial screening is complete the probes then need to be matched to dbSNP using REMEDY-DB for further processing. As discussed before, some manifest files contain internal Illumina ids so must be mapped to a rsID using a mapping file generated by the manufacturer.

If this map file is supplied as a command line argument, it will be loaded at this point and compared to the manifest file. Any variants which do not match to the rsID mapping file are logged as a high number of these may indicate that the wrong mapping file is being

used. In addition, some ids in the mapping file have no corresponding rsID as they are custom probes; these are also logged and filtered out from the final list of valid variants. At the end of this stage, all variants should have a valid rsID that can be used to search for a match in REMEDY-DB.

**3.1.7.1.3 Variant Matching** REMEDY uses the meta-data available from dbSNP to further assess variants for their potential quality in a GWAS. All valid variants from the previous steps are matched to dbSNP using REMEDY-DB. The outcomes of the variant matching stage are discussed below.

**3.1.7.1.3.1 Unmatched Variants** Microarray manifests are generated with the latest version of dbSNP at the time of design but then remain static; this results in situations where valid rsIDs do not match a valid variant in dbSNP. Assuming that REMEDY-DB is operating with a newer version of dbSNP, we can conclude that either the variant been merged with another rsID or that it has been removed from the database. In the case of removal, this variant is no longer considered valid and is removed from the analysis, in the case of merging, the merged rsID is located and substituted.

**3.1.7.1.3.2 Structural Variants** As previously discussed genotyped structural variants are unsuitable for GWAS and must be filtered out. During the initial probe screening, any structural variants marked in the manifest file are removed; this is repeated at this stage, but the structural variation designation is read from dbSNP instead. Some rsIDs which are described as SNVs in the manifest file are described as structural in dbSNP.

**3.1.7.1.3.3 Multiallelic Variants** As described in the introduction, multiallelic variants can be difficult to call in genotyping and therefore are a potential source of error. Despite this, they are regularly included on microarrays, perhaps because of coverage requirements in the region. The probe may also have been initially designed against a SNV, but in the latest version of dbSNP, this variant has now been reclassified as multiallelic. Whatever the reason for inclusion, any SNVs identified as being multiallelic in dbSNP will be filtered out by REMEDY.

It is important to note that this is different from having multiallelic SNVs in the local genotyping data. When processing individual datasets, genotypes are likely to be homogenous with respect to major and minor alleles; however, once the genotyping merging process is complete, there may still be some multiallelic SNVs in the merged dataset. These errors can be due to design differences in microarrays or experimental error; REMEDY processes

data on a per dataset basis and so is unable to catch downstream multiallelic errors such as these, a check must therefore be included in downstream pre-processing of data for GWAS.

**3.1.7.1.3.4 Strand Mismatches** A strand mismatch is defined as a situation where the minor and major alleles of a variant in dbSNP and probes target nucleotides match, but the strand is opposite. In order to calculate these errors, both the probe alleles and the dbSNP alleles are re-coded to FWD, and the alleles are then reverse matched to detect an inconsistency.

Strand mismatches can be the result of differences in the strand calling processes between Illumina and dbSNP. Both use the Blast-like alignment tool (BLAT) [215] but in differing pipelines; strand mismatches are the result of these slightly different approaches. The fact that there are very few of these supports this hypothesis. Strand mismatches are considered a potential source of noise and are filtered out from the final list of valid variants.

**3.1.7.1.3.5 Invalid Positions** The dbSNP database is not without its own errors; entries can be incomplete or contain erroneous data. One of these situations occurs when the genomic position in dbSNP is blank or invalid. These are relatively rare and are deemed as a source of risk even though the position from the manifest file could be used (this is supplied), they are therefore filtered out.

**3.1.7.1.4 Manifest Comparison** Manifest comparison is an optional function within REMEDY that allows users to compare two manifests together to identify design differences that may be potential sources of noise for a GWAS. When creating a merged genotyping dataset that contains a heterogeneous set of microarrays, it is advisable to select a primary microarray to which others will be compared. We recommend that this is the microarray used for the genotyping of internal project samples as this is usually the newest design given that the internal project samples will most likely have been genotyped for the project and any controls genotyped at an earlier date. In the case of multiple internal microarrays, it is recommended that the chip for the largest sample set is selected; we term this manifest the reference manifest.

When scanning a new manifest, if a reference manifest is supplied for comparison, three additional metrics will be calculated during the normal filtering process. The probe sequences for the probes of the two microarrays will be compared, and any differences will be logged; this helps to highlight if the probe design has changed between products. Differences in probe sequence for the same variant indicates that the probe sequence has been

optimised during the period between the generation of the two manifests; this may indicate that the genotyping calls for the older microarray may be less accurate. It is important to note that probe differences are merely logged and are not filtered out automatically.

If the probe sequences for the variant match, then the strand designations and SNP content will also be compared. Differences in strand or major/minor allele usually indicate that the manifest was generated with a different version of dbSNP. These again must be watched closely as potential noise sources but are not filtered out automatically.

**3.1.7.1.5 Logging Output Interpretation** We here describe the important aspects to note when analysing log file outputs for this sub-program (Figure 3.4, p.146). Once the manifest is first loaded, the logging will indicate how many loci were retrieved. This should match the expected number for the target microarray. The logging will next show the different strand types within the file and how many duplicate ids and structural loci there were overall. The number of loci remaining after these have been filtered out is shown highlighted in green.

The logging for the comments file (if provided) will show the counts and types of messages that are present. After filtering, the number of loci left is then shown, highlighted in green.

The logging for the mappings file (if provided) will show the total number of mappings loaded and how many manifest loci did not have a corresponding mapping; this number should be 0, a small number could mean that the manifest and mapping files are from different versions of the same microarray - a larger number could indicate that the wrong mapping file is being used. The "no. mapped rsID" field shows how many mappings had a "." rather than an rsID in the mapping file; this usually indicates a custom probe that doesn't have an entry in dbSNP. After filtering, the number of loci left is then shown, highlighted in green.

dbSNP matching is shown in the final portion of the log with detailed match statistics. The total number of loci (ignoring any previous filtering) with rsIDs in the manifest is shown first. The basic balance of matched and unmatched rsIDs is shown next followed by numbers for multiallelic, strand mismatches, invalid positions and then structural variant types. The final count of valid variants after all filtering steps is then shown highlighted in green; this represents the total number of variants which can be used to determine strand designation and that are considered valid for a GWAS.

```

[2019-04-04 09:19:31]-[SCAN_MANIFEST]-[HEADING]- Scanning manifest...
[2019-04-04 09:19:32]-[LOAD_MANIFEST]-[INFO]- Loading manifest: Multi-EthnicGlobal_D1_37.csv
[2019-04-04 09:19:34]-[TASK]-[INFO]- Loading manifest... - 10%
[2019-04-04 09:19:36]-[TASK]-[INFO]- Loading manifest... - 20%
[2019-04-04 09:19:37]-[TASK]-[INFO]- Loading manifest... - 30%
[2019-04-04 09:19:39]-[TASK]-[INFO]- Loading manifest... - 40%
[2019-04-04 09:19:41]-[TASK]-[INFO]- Loading manifest... - 50%
[2019-04-04 09:19:43]-[TASK]-[INFO]- Loading manifest... - 60%
[2019-04-04 09:19:45]-[TASK]-[INFO]- Loading manifest... - 70%
[2019-04-04 09:19:47]-[TASK]-[INFO]- Loading manifest... - 80%
[2019-04-04 09:19:49]-[TASK]-[INFO]- Loading manifest... - 90%
[2019-04-04 09:19:51]-[TASK]-[INFO]- Loading manifest... - 100%
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[IMPORTANT]- Loaded 1,748,250 loci from manifest
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[INFO]- Duplicate names: 0
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[INFO]- Strand type - BOT: 877,057
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[INFO]- Strand type - TOP: 846,738
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[INFO]- Strand type - MINUS: 11,493
[2019-04-04 09:19:51]-[LOAD_MANIFEST]-[INFO]- Strand type - PLUS: 12,962
[2019-04-04 09:19:52]-[SCAN_MANIFEST]-[INFO]- Structural loci: 24,455
[2019-04-04 09:19:52]-[SCAN_MANIFEST]-[IMPORTANT]- Valid loci remaining: 1,723,795
[2019-04-04 09:19:52]-[LOAD_COMMENTS]-[INFO]- Loaded 15,768 comments
[2019-04-04 09:19:52]-[LOAD_COMMENTS]-[INFO]- Comment type - Multiple mappings: 11,451
[2019-04-04 09:19:52]-[LOAD_COMMENTS]-[INFO]- Comment type - PAR-like region: 4,311
[2019-04-04 09:19:52]-[LOAD_COMMENTS]-[INFO]- Comment type - No probe mappings: 6
[2019-04-04 09:19:52]-[SCAN_MANIFEST]-[IMPORTANT]- Valid loci remaining: 1,708,643
[2019-04-04 09:19:54]-[LOAD_RSID]-[INFO]- Loaded 1,748,250 rsid mappings
[2019-04-04 09:19:54]-[SCAN_MANIFEST]-[INFO]- Mapping rsids to manifest...
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[INFO]- Total unfiltered manifest loci: 1,748,250
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[INFO]- Unmatched mappings: 0
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[INFO]- Matched mappings: 1,748,250
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[INFO]- No mapped rsid: 12,534
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[INFO]- Valid mappings: 1,735,716
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[IMPORTANT]- Valid loci remaining: 1,707,934
[2019-04-04 09:19:55]-[SCAN_MANIFEST]-[HEADING]- Matching identifiers to dbSNP...
[2019-04-04 09:21:22]-[TASK]-[INFO]- Variants matched to dbSNP: 164,744 of 166,084 - 10%
[2019-04-04 09:22:50]-[TASK]-[INFO]- Variants matched to dbSNP: 339,009 of 340,909 - 20%
[2019-04-04 09:24:11]-[TASK]-[INFO]- Variants matched to dbSNP: 512,785 of 515,734 - 30%
[2019-04-04 09:25:33]-[TASK]-[INFO]- Variants matched to dbSNP: 686,545 of 690,559 - 40%
[2019-04-04 09:26:54]-[TASK]-[INFO]- Variants matched to dbSNP: 860,268 of 865,384 - 50%
[2019-04-04 09:28:13]-[TASK]-[INFO]- Variants matched to dbSNP: 1,034,533 of 1,040,209 - 60%
[2019-04-04 09:29:28]-[TASK]-[INFO]- Variants matched to dbSNP: 1,209,236 of 1,215,034 - 70%
[2019-04-04 09:30:48]-[TASK]-[INFO]- Variants matched to dbSNP: 1,383,559 of 1,389,859 - 80%
[2019-04-04 09:32:04]-[TASK]-[INFO]- Variants matched to dbSNP: 1,558,166 of 1,564,684 - 90%
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Total loci with rsids: 1,735,716
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Unmatched : 7,280
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Matched : 1,728,436
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Multi-Allelic : 148,545
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Strand mismatches : 66,972
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Invalid Positions : 2
[2019-04-04 09:33:26]-[SCAN_MANIFEST]-[INFO]- Total valid variants from dbSNP (matched - invalid positions): 1,728,434
[2019-04-04 09:33:41]-[SCAN_MANIFEST]-[INFO]- SNV type - single: 1,708,072
[2019-04-04 09:33:41]-[SCAN_MANIFEST]-[INFO]- SNV type - deletion: 15,307
[2019-04-04 09:33:41]-[SCAN_MANIFEST]-[INFO]- SNV type - insertion: 5,023
[2019-04-04 09:33:41]-[SCAN_MANIFEST]-[INFO]- SNV type - in-del: 34
[2019-04-04 09:33:41]-[SCAN_MANIFEST]-[IMPORTANT]- Valid loci remaining: 1,565,402

```

Figure 3.4: Logging output for the manifest scan sub-program of REMEDY.

### 3.1.7.2 Genotype Data Scanning

This sub-function can be called using the keyword `scan-illumina-genotypes`. Its role is to scan a genotype data file in either Illumina matrix or final report format to test its match to the supplied manifest and to predict the strand encoding used when it was generated. The required inputs to the function are the genotyped data input file and the manifest file. The function does not output any summary files; it outputs all predictions and statistics to logging.

scan-illumina-genotypes option details and example syntax are shown below:

**(--input)** Input file/folder path - **required:** Specifies the file or folder path to the genotyping file(s) that require scanning. If a directory is supplied, the files inside will be enumerated and processed sequentially.

**(--manifest)** Manifest file path - **required:** Specifies the file path to the manifest file that provides data for the scanning process.

**(--delim)** File delimiter: Specifies the column delimiter for the input files. Defaults to tab.

**(--filter)** Filter: Path to the valid.txt output of the manifest scanning step. By supplying this optional file, only valid variants will be considered for encoding prediction. This can improve prediction accuracy and is recommended in a normal pipeline unless performing a custom quick scan of a genotyping file.

**(--ab)** AB Encoding: Specifies that the source data is Illumina AB encoded rather than GATC. The default outputs from Genome Studio are all AB encoded, a quick look at the top few lines of the input file will confirm to the user what the encoding is.

The different processing stages of this sub-function are detailed in the following sections.

**3.1.7.2.1 Genotype Scanning** Processing begins with a basic file scan; if the input file is in Illumina Final Report format, then this is first converted to matrix format before further processing. The matching engine then checks the variant is contained in the filter list (if supplied) and then checks that it also exists in the manifest. Large numbers of non-matching variants in the manifest are indicative that the genotyping dataset was not genotyped on the microarray for which the manifest was generated. After this, the final list of valid variants is subject to the encoding scheme detection stage.



**3.1.7.2.2 Strand Designation Detection** When merging disparate genotyping datasets, the information about how the dataset was formatted and processed is not always complete. If critical pieces of information are not known such as the source genotype strand designation, then this can cause significant errors later in the genomic analysis pipeline. REMEDY was developed to auto-detect the source strand encoding scheme; without summarising the information contained within the genotyping data file, it can be time-consuming to detect the source encoding manually.

To detect strand designation, REMEDY scans the manifest for each variant and calculates the TOP/BOT/FWD/REV encoding for the reference and alternate nucleotides; it then compares it to the genotyped alleles in the input data file. If the genotype matches one of the encoding schemes, a simple count tally is kept for that encoding. As the file is scanned, separate distributions over TOP/BOT and FWD/REV are calculated as percentages in real-time. The percentages can be interpreted to infer the source encoding of the genotyping file. For example, a file that was FWD encoded will have >95% of the genotyped alleles matching either the reference or alternate of a bi-allelic variant that has been FWD encoded. In the same way, genotypes that have a >95% match for TOP/BOT/REV will be encoded to those schemes respectively. If, however, there is a partial match across all encoding schemes then that can be interpreted as evidence that the source file was encoded in the DESIGN scheme; this is because DESIGN encoded data is encoding unspecific and is instead designated to the original design strand of the probe.

Source strand designation is outputted during the scanning stage of processing which can then be recorded by the user to set a source encoding for the file so that it can be appropriately re-encoded to the destination strand designation and encoding.

**3.1.7.2.3 Logging Output Interpretation** We here describe the important aspects to note when analysing log file outputs for this sub-program (Figure 3.5, p.149). Once the manifest is first loaded, the logging will indicate how many loci were retrieved. This should match the expected number for the target microarray. The logging will next show the different strand types within the file and how many duplicate ids and structural loci there were overall.

Genotyped input files will then be shown in sequence, separated by a blue header. The logging will show how many samples it detected for the new input file which should be sanity checked. The scan will output the proportions of different strand designation schemes during processing until the "Scan complete" message is displayed. The number of loci scanned will be shown before showing how many genotype file loci did not match the manifest; a large number here (over 10%) indicates that the wrong manifest file may

have been used. The loci used for strand detection will be a combination of the matching loci against the manifest and those that exist within the filtering file from the manifest scan (if supplied), this is shown in the logging accordingly.

The final important messages are shown in green that show the final proportion of strand designations and the subsequent predicted encoding. The predicted encoding should be used as the "source encoding" field when calling the re-coding sub-program.

```
[2019-04-04 10:07:14]-[SCAN_GENOTYPES]-[HEADING]- Scanning genotypes...
[2019-04-04 10:07:14]-[SCAN_GENOTYPES]-[INFO]- Loading filter...
[2019-04-04 10:07:15]-[SCAN_GENOTYPES]-[IMPORTANT]- Loaded 1,565,402 filter loci
[2019-04-04 10:07:15]-[LOAD_MANIFEST]-[INFO]- Loading manifest: Multi-EthnicGlobal_D1_37.csv
[2019-04-04 10:07:17]-[TASK]-[INFO]- Loading manifest... - 10%
[2019-04-04 10:07:19]-[TASK]-[INFO]- Loading manifest... - 20%
[2019-04-04 10:07:21]-[TASK]-[INFO]- Loading manifest... - 30%
[2019-04-04 10:07:23]-[TASK]-[INFO]- Loading manifest... - 40%
[2019-04-04 10:07:25]-[TASK]-[INFO]- Loading manifest... - 50%
[2019-04-04 10:07:27]-[TASK]-[INFO]- Loading manifest... - 60%
[2019-04-04 10:07:29]-[TASK]-[INFO]- Loading manifest... - 70%
[2019-04-04 10:07:31]-[TASK]-[INFO]- Loading manifest... - 80%
[2019-04-04 10:07:33]-[TASK]-[INFO]- Loading manifest... - 90%
[2019-04-04 10:07:35]-[TASK]-[INFO]- Loading manifest... - 100%
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[IMPORTANT]- Loaded 1,748,250 loci from manifest
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[INFO]- Duplicate names: 0
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[INFO]- Strand type - BOT: 877,057
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[INFO]- Strand type - TOP: 846,738
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[INFO]- Strand type - MINUS: 11,493
[2019-04-04 10:07:35]-[LOAD_MANIFEST]-[INFO]- Strand type - PLUS: 12,962
[2019-04-04 10:07:35]-[SCAN_GENOTYPES]-[HEADING]- Processing genotype_test_data.txt
[2019-04-04 10:07:35]-[SCAN_GENOTYPES]-[INFO]- Detected 7 samples
[2019-04-04 10:07:36]-[TASK]-[INFO]- TOP: 50.86% BOT: 49.14% FWD: 54.94% REV: 45.06% - 10%
[2019-04-04 10:07:37]-[TASK]-[INFO]- TOP: 51.38% BOT: 48.62% FWD: 53.9% REV: 46.1% - 20%
[2019-04-04 10:07:37]-[TASK]-[INFO]- TOP: 51.6% BOT: 48.4% FWD: 53.69% REV: 46.31% - 30%
[2019-04-04 10:07:38]-[TASK]-[INFO]- TOP: 51.75% BOT: 48.25% FWD: 54.49% REV: 45.51% - 40%
[2019-04-04 10:07:39]-[TASK]-[INFO]- TOP: 51.97% BOT: 48.03% FWD: 55.01% REV: 44.99% - 50%
[2019-04-04 10:07:40]-[TASK]-[INFO]- TOP: 52.1% BOT: 47.9% FWD: 54.94% REV: 45.06% - 60%
[2019-04-04 10:07:41]-[TASK]-[INFO]- TOP: 52.22% BOT: 47.78% FWD: 54.86% REV: 45.14% - 70%
[2019-04-04 10:07:42]-[TASK]-[INFO]- TOP: 52.27% BOT: 47.73% FWD: 54.81% REV: 45.19% - 80%
[2019-04-04 10:07:43]-[TASK]-[INFO]- TOP: 52.31% BOT: 47.69% FWD: 54.77% REV: 45.23% - 90%
[2019-04-04 10:07:44]-[TASK]-[INFO]- TOP: 52.29% BOT: 47.71% FWD: 54.72% REV: 45.28% - 100%
[2019-04-04 10:07:44]-[TASK]-[INFO]- TOP: 52.28% BOT: 47.72% FWD: 54.73% REV: 45.27% - 0%
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[INFO]- Scan complete
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[INFO]- Loci scanned: 1,779,819
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[INFO]- Loci not matching to manifest: 31,572
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[INFO]- Loci used for encoding prediction: 1,486,429
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[IMPORTANT]- TOP: 52.28% BOT: 47.72% FWD: 54.73% REV: 45.27%
[2019-04-04 10:07:44]-[SCAN_GENOTYPES]-[IMPORTANT]- Predicted encoding: Illumina Design Strand
```

**Figure 3.5:** Logging output for the genotype scan sub-program of REMEDY.

### 3.1.7.3 Data Re-coding and Conversion

This sub-function can be called using the keyword `recode-illumina-genotypes`. Its most fundamental role is to convert Illumina matrix files to the VCF standard genomic format. During this conversion process, the outputs from the manifest and genotyping scanning routines can be used to filter and re-code the data so that it can be successfully merged with other genotyping datasets for the purposes of GWAS.

recode-illumina-genotypes option details and example syntax are shown below:

**(--input)** Input file/folder path - **required**: Specifies the file or folder path to the genotyping file(s) that require scanning. If a directory is supplied, the files inside will be enumerated and processed sequentially.

**(--output)** Output path - **required**: Specifies the output folder to write data to.

**(--prefix)** Output file prefix: Prefixes filenames before writing to the output folder. Use if writing multiple report runs to the same folder so that the results are not overwritten.

**(--manifest)** Manifest file path: Specifies the file path to the manifest file. This is **required** if any re-coding has been selected in the other options.

**(--mapping)** rsID map file: Path to rsID map file. Some manifest files have an internal Illumina generated ID as the marker id. To match to \gls{dbsnp} for further analysis, the rsID of the marker is needed. This is supplied as a marker mapping file by the manufacturer.

**(--delim)** File delimiter: Specifies the column delimiter for the input files. Defaults to tab.

**(--filter)** Filter: Path to the valid.txt output of the manifest scanning step. By supplying this optional file, only valid variants will be considered for output to \gls{vcf} file.

**(--ab)** AB Encoding: Specifies that the source data is Illumina AB encoded rather than GATC. The default outputs from Genome Studio are all AB encoded, a quick look at the top few lines of the input file will confirm to the user what the encoding is.

**(--source-enc)** Source Encoding: The source encoding scheme of the data. This can be obtained by running the genotype scanning routine on the input data prior to running this function. Valid inputs are design, forward, reverse, top and bot.

**(--target-enc)** Destination Encoding: The target encoding scheme of the data. If this is different from the source encoding then `\gls{remedy}` will re-code the data to the target strand designation scheme. Valid inputs are design, forward, reverse, top and bot.

The different processing stages of this sub-function are detailed in the following sections.

**3.1.7.3.1 Manifest Scan Filtering** The first stage of processing is general conversion and filtering without re-coding. If the `—filter` flag is supplied, the input genotyping variants are checked against the list, and any non-matches are filtered out. If the rsID mapping option is supplied, the internal Illumina ids are swapped for rsIDs. If the `—ab` option is supplied, the genotypes are converted to GATC.

**3.1.7.3.2 Re-coding** REMEDY can re-code from and to either strand of the most popular encoding schemes. If the source and target encoding schemes are different when supplied to the routine, then this processing stage is triggered. The central premise of re-coding is that any encoding scheme can be reached from the probe DESIGN encoding. Any re-coding, therefore, will first be converted to design strand and then to the target scheme. If the source or target is DESIGN, then this simplifies the process as no extra work is required.

The key piece of information required for successful re-coding of a variant is shown in the manifest file in the *llmnID* column; this details in native DESIGN encoding, whether the variant is called as TOP or BOT and FWD or REV. The ID essentially specifies the rules for translation from that scheme to another. From DESIGN encoding, to convert to another scheme is trivial as the target scheme must simply be compared to the variants call for the scheme. For example, to convert to TOP when the variant is already TOP, nothing must be done, but if the strands are different, then the genotype letter must be reversed. To convert to DESIGN from another scheme is similarly trivial, if the strands are different between the

ID and the source scheme, the letter must be reversed. More information on switching between designation schemes can be found in the introduction (1.4.4.5, p.77).

**3.1.7.3.3 Genome Build Differences** Merging disparate genotyping data sets often results in the merging of datasets that were genotyped at different times. In each new genomic build, there is a possibility that a variant's position may change due to updates in that area of the genome. When combining temporally different genotyping output files, there is a possibility that the outputs were generated against different genome builds.

The genome build that the genotyping files were generated against will be available in the chip manifest file. Variant positions can be lifted to the new genome build by merely taking the position from the mapped dbSNP entry rather than the source file; this ensures that all variants that run through REMEDY have only one source of genomic position. This feature is automatically implemented when generating VCF files.

**3.1.7.3.4 VCF File Output** VCF is a text file format and is the global standard file for storing uncompressed, human-readable variant information data. Binary calling format (BCF) is the sister specification for storing this data in compressed binary format. The VCF file output was chosen as the final output for REMEDY as it allows users to easily link into the wealth of genomic command line tools in the public domain. The format was developed with large-scale genotyping and sequencing in mind and was developed as the primary file format for the 1000 genomes project. The format allows for extensive meta-data to be written alongside the primary variant and genotyping content while still showing the data in a compressed, human-readable format [221].

REMEDY creates a custom VCF header to show that it was processed by REMEDY along with a timestamp of when the pipeline was run and other runtime parameters. Output data is written in the simplest way allowed by the VCF specification.

**3.1.7.3.5 Logging Output Interpretation** We here describe the important aspects to note when analysing log file outputs for this sub-program (Figure 3.6, p.153). When the manifest is first loaded, the logging will indicate how many loci were retrieved. This should match the expected number for the target microarray. The logging will next show the different strand types within the file and how many duplicate IDs and structural loci there were overall.

Genotyped input files will then be shown in sequence, separated by a blue header. The logging will show how many samples it detected for the new input file which should be sanity

checked. The logging will show progress for the re-coding process before outputting how many loci did not match to the manifest; a high number indicates that the wrong manifest file is being used. Logging finally shows the total markers written to the VCF file.

```
[2019-04-04 10:50:22]-[RECODE_GENOTYPES]-[HEADING]- Recoding genotypes...
[2019-04-04 10:50:22]-[SCAN_GENOTYPES]-[INFO]- Loading filter...
[2019-04-04 10:50:24]-[SCAN_GENOTYPES]-[IMPORTANT]- Loaded 1,565,402 filter loci
[2019-04-04 10:50:25]-[LOAD_RSID]-[INFO]- Loaded 1,748,250 rsid mappings
[2019-04-04 10:50:25]-[LOAD_MANIFEST]-[INFO]- Loading manifest: Multi-EthnicGlobal_D1_37.csv
[2019-04-04 10:50:28]-[TASK]-[INFO]- Loading manifest... - 10%
[2019-04-04 10:50:30]-[TASK]-[INFO]- Loading manifest... - 20%
[2019-04-04 10:50:32]-[TASK]-[INFO]- Loading manifest... - 30%
[2019-04-04 10:50:35]-[TASK]-[INFO]- Loading manifest... - 40%
[2019-04-04 10:50:37]-[TASK]-[INFO]- Loading manifest... - 50%
[2019-04-04 10:50:39]-[TASK]-[INFO]- Loading manifest... - 60%
[2019-04-04 10:50:41]-[TASK]-[INFO]- Loading manifest... - 70%
[2019-04-04 10:50:44]-[TASK]-[INFO]- Loading manifest... - 80%
[2019-04-04 10:50:46]-[TASK]-[INFO]- Loading manifest... - 90%
[2019-04-04 10:50:48]-[TASK]-[INFO]- Loading manifest... - 100%
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[IMPORTANT]- Loaded 1,748,250 loci from manifest
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[INFO]- Duplicate names: 0
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[INFO]- Strand type - BOT: 877,057
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[INFO]- Strand type - TOP: 846,738
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[INFO]- Strand type - MINUS: 11,493
[2019-04-04 10:50:48]-[LOAD_MANIFEST]-[INFO]- Strand type - PLUS: 12,962
[2019-04-04 10:50:48]-[RECODE_GENOTYPES]-[HEADING]- Processing genotype_test_data.txt
[2019-04-04 10:50:48]-[RECODE_GENOTYPES]-[INFO]- Detected 7 samples
[2019-04-04 10:52:18]-[TASK]-[INFO]- Recoding... - 10%
[2019-04-04 10:53:39]-[TASK]-[INFO]- Recoding... - 20%
[2019-04-04 10:54:47]-[TASK]-[INFO]- Recoding... - 30%
[2019-04-04 10:56:04]-[TASK]-[INFO]- Recoding... - 40%
[2019-04-04 10:57:34]-[TASK]-[INFO]- Recoding... - 50%
[2019-04-04 10:59:02]-[TASK]-[INFO]- Recoding... - 60%
[2019-04-04 11:00:18]-[TASK]-[INFO]- Recoding... - 70%
[2019-04-04 11:01:58]-[TASK]-[INFO]- Recoding... - 80%
[2019-04-04 11:03:40]-[TASK]-[INFO]- Recoding... - 90%
[2019-04-04 11:04:58]-[TASK]-[INFO]- Recoding... - 100%
[2019-04-04 11:05:02]-[TASK]-[INFO]- Recoding... - 0%
[2019-04-04 11:05:02]-[RECODE_GENOTYPES]-[INFO]- Total manifest non-matches: 31,572
[2019-04-04 11:05:02]-[RECODE_GENOTYPES]-[IMPORTANT]- Total markers: 1,565,399
[2019-04-04 11:05:02]-[RECODE_GENOTYPES]-[INFO]- Complete
```

**Figure 3.6:** Logging output for the re-coding sub-program of REMEDY.

### 3.1.8 Data Merging

Each input file provided to REMEDY for re-coding and conversion results in a single VCF output file; in general, each input file will be the output from Genome Studio for a single well plate. In order to make data management, storage and further processing easier, it is recommended to merge the output VCF files into one VCF file. REMEDY does not provide functionality to do this as VCF Tools already provides this functionality.

VCF Tools is a software toolset design to work with VCF files created by the creators of the VCF specification [221]. There are three ways in which VCF files can be stored so that they can be interacted with by their software packages: native VCF format, BCF format and

GZIP with a tab index format. For native and GZIP with tab index interaction, VCF-Tools should be used while for BCF interaction, BCF Tools (produced by the same team) should be utilised.

GZIP is a standard compression format in Linux systems that is lossless. The tab index feature generates an additional index file which indexes genomic position within the compressed file to make navigation of large compressed VCF files faster; this is the primary storage format for most genomic data in VCF format and is the required input for many further analysis programs.

The BCF file format is a binary non-human readable, compressed VCF format. It was developed to increase the performance of some of the standard functions within VCF Tools. It allows a computer to perform operations at maximum performance without worrying about making the data human readable.

VCF Tools and BCF Tools provide functionality to merge VCF files to create one large output VCF file. This operation is strongly recommended to be included in any pipeline that uses REMEDY.

### **3.1.9 Installation and Setup**

To install REMEDY, download the latest version for the operating system of choice from the GitHub Wiki. Unzip the compressed folder to a directory and add the `bin` folder to the system path so that the toolset can be called from anywhere on the operating system. Completed releases can also be downloaded directly from Microsoft Azure or REMEDY can be updated automatically by triggering the updater included with the base software. The updater will check the file repository for a new version and download the relevant system update files. REMEDY builds are available for Windows 7, 10 and Linux; this ensures REMEDY can run optimally in a range of environments with or without an internet connection.

REMEDY requires REMEDY-DB to be downloaded and placed in a folder named `vardb` alongside the `bin` folder inside the main program folder. Multiple versions of REMEDY-DB can be downloaded from the GitHub Wiki for different versions of dbSNP. To update REMEDY-DB GitHub must be periodically checked and any new versions downloaded. Alternatively, we provide a sub-program within the REMEDY package to generate a new database from the UCSC base table file.

To verify installation, REMEDY can be called on the command line by typing `remedy`;

a welcome screen will be displayed with some version information and basic run options. It is recommended that a projects folder be created for all REMEDY data. For each new project create a new folder that contains all the data for REMEDY to complete processing for that particular project.

### **3.1.10 Software Performance**

We conducted performance testing on REMEDY using a high-end desktop computer running Windows and Linux in a dual boot configuration with a Samsung 950 Pro solid state drive, the full specification can be found in the appendices A.1, p.280. All three REMEDY sub-programs were testing using a subset of data from the SSNS GWAS project. The data-set consisted of seven samples which were genotyped on the Illumina Infinium Multi-Ethnic Global microarray detailed in the Steroid-sensitive nephrotic syndrome (SSNS) case study section below.

The manifest scanning took 14 minutes and 10 seconds to scan 1,748,250 variants, equating to 2056.7 variants per second for matching and processing. Genotype scanning on the seven sample test set took 30 seconds to complete a scan of 12,237,750 genotypes equating to 407,925 genotypes per second. Re-coding the data from DESIGN to FWD took 15 minutes and 20 seconds equating to 13,302 genotypes per second.

We found all testing to be more than acceptable in terms of quality and run-time to work inside of a GWAS pipeline, especially given that the REMEDY process would usually only be performed once on input data during a project and the result saved for future use. No significant differences in processing time or output were found between running REMEDY on Windows versus Linux operating systems. The processing time for genotype scanning and re-coding increased linearly with the number of samples included for analysis. Both the manifest scan and re-coding portions of REMEDY require variant matching to REMEDY-DB which occupies approximately 95% of the processing time during a run; therefore, increased sample sizes on the re-coding sub-program although increasing run-time linearly was negligible overall, equating to under 20 seconds even when scaled to a full well-plate of 96 samples. We also ran REMEDY on a similar test desktop, but with a standard magnetic hard drive, the full specification can be found in the appendices A.1, p.280. We found the processing times for manifest scanning and re-coding to be significantly longer; this was due to REMEDY-DB operating as an on-disk NoSQL database. NoSQL database performance is directly tied to storage drive performance, so this result was unsurprising. We strongly recommend REMEDY to be run on computers with solid-state drives for optimal performance.



We compared our re-coding output to Illumina Genome Studio by importing the same seven sample test file and exporting the data in TOP, BOT, FWD and REV strand designations. We then programmatically compared the four outputs to runs of REMEDY and found zero differences between the two programs. This confirmed that our re-coding algorithm produced the same results as Illumina software.

### **3.1.11 Steroid-Sensitive Nephrotic Syndrome Case Study**

We undertook an investigation into SSNS where we collaborated with several teams globally to collect a large cohort of over 1700 samples to perform a GWAS.

The large sample cohorts in our study, presented some additional technical challenges as we did not have the resources to generate proprietary control data; the microarray that was selected for the SSNS GWAS also made locating matching control data more difficult. We used the methodology shown in our methods section to build a data pipeline capable of handling the larger volume of genotyping data present, and that was designed to overcome the technical difficulties we faced. One of the primary challenges we encountered was the requirement to merge multiple large, disparate, genotyped datasets from different experiments in an accurate, repeatable way; we developed REMEDY to address these issues within our GWAS methodology.

Here we first present the pipeline design process for our SSNS GWAS, highlighting how REMEDY fits into the overall data flow. We then show a summary of the pipeline results to demonstrate the effectiveness of our tool. For more information on the SSNS study including detailed methods and results, we direct the reader to Dufek *et al.* [1].

#### **3.1.11.1 Introduction**

SNSS is the most common form of nephrotic syndrome in children with an incidence of approximately one to ten per 100,000; the majority of affected children experience a chronic relapsing course. The onset of disease manifestations is commonly associated with a preceding activation of the immune system, typically by an upper respiratory tract infection. As the name implies, SSNS is characterised by a therapeutic response to glucocorticoids, as well as to other immunosuppressants. The apparent triggering of the disease by infection and the therapeutic effect of immunosuppressive treatment have suggested that SSNS is an autoimmune disorder [1].

Investigating the genetic architecture of diseases through GWAS has proven success-

ful in many autoimmune diseases, the most common finding in GWAS is the identification of an association to the Human leukocyte antigen (HLA) region. Arguably, however, it is the regions outside HLA that are found to be associated that can provide the most meaningful insights into the disease mechanism. Our team previously showed that Membranous nephropathy (MN) had a significant association in the region of the gene *Anti-Phospholipase-A2-Receptor (PLA2R2)* outside of HLA which was separately shown to be crucial to the disease mechanism [177].

SSNS is a well-known disease that has been the subject of several studies, however, although a significant immune component for the disease had already been identified, further non-HLA causal regions were yet to be identified. In situations such as this, a GWAS of significant power can reveal new causal genomic regions leading to better understanding of the disease aetiology.

### **3.1.11.2 Project Planning**

The project premise for the SSNS project was to use a scientific collaboration to collect a large cohort of SSNS patient samples; this would potentially provide the necessary study power for a GWAS to reveal previously unidentified genomic regions of interest or show new significant variants in the current Region of interest (ROI). GWAS have historically been used to great effect in the study of rare disease to identify causal ROIs and variants, given the size of the cohort and the lack of large pedigree families available a linkage study would be infeasible in this situation; instead, we selected a GWAS as our method of positional cloning.

For our SSNS study, we collected 1698 Deoxyribonucleic acid (DNA) samples from patients with confirmed SSNS in a global collaboration with teams from Europe and Asia; the samples were collected as an input data source with no other associated clinical data. Our genomic analysis methodology was applied to create a data pipeline that would identify potential causal variants for SSNS when compared to a control cohort without SSNS. After causal variant identification, additional functional analysis would then be performed as required.

Samples genotyped on a high-density microarray was selected as the primary genetic data source for the project. The overall strategy for data collection was to collect samples in batches, label them according to our sample ID methodology and have them genotyped by an external lab. We would then collect the raw IDAT files as well as the Design output matrix files from Genome Studio as our data pipeline input data. We required a control cohort to compare against the SSNS samples which we did not have internally in our project; we,

therefore, obtained data from multiple public external sources to form a control set large enough to provide significant statistical power to the GWAS.

Given the multi-ethnic nature of the study cohort, the Illumina Infinium Multi-Ethnic Global BeadChip v.A1 [189] was selected as our microarray for the case samples. This relatively new chip contained over 1.7 million markers, which were spread over multiple ethnicities and was designed explicitly for multi-ethnic GWAS by a consortium. Given the excellent marker distribution and density and the multi-ethnicity of the variants, it was deemed that this chip would give the best probability of a successful GWAS given the study samples.

As discussed in the introduction, in a normal GWAS, we control for variance in the ethnicity of samples using Principal component analysis (PCA). Given the multi-ethnic characteristics of our patient cohort, we opted to perform a GWAS initially on our largest group of samples - the Europeans, thus retaining the remainder of the samples for meta-analysis or for creation of replicates; this maximises the power of the data while keeping study complexity to a minimum.

We opted to identify the European group of samples using PCA with standard deviation filtering of samples that were outliers when compared to the European cluster centre. Excluding samples based on ethnicity removes samples and thus lowers the power of a study, however, simply including all samples increases the genomic inflation factor and can introduce false positives in the data that are due to ethnic differences in samples rather than the target trait being measured.

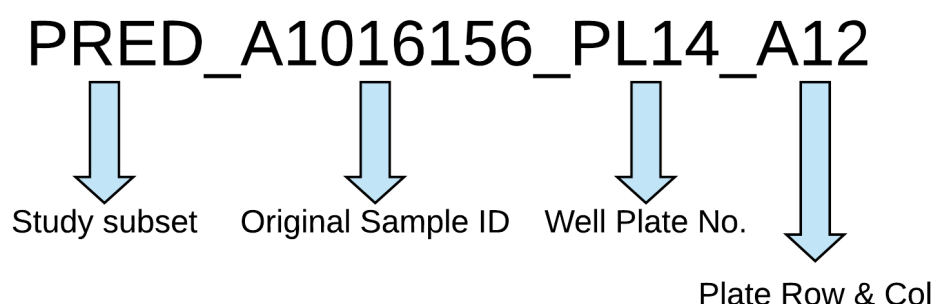
To control for ethnicity while retaining as many samples as possible in the study cohorts, we employed a two-pronged approach. First, we would perform PCA and standard deviation filtering with relatively relaxed filter parameters to remove samples which were from distant ethnic groups such as Chinese or Africans. We would then perform logistic regression association testing using the first 10 principal components from the PCA to control for any remaining deviations in ethnicity.

### **3.1.11.3 Sample Preparation and Genotyping**

Source data collection was problematic for both cases and controls, primarily due to the mosaic nature of both datasets; the samples for both cases and controls were collected from multiple sources with different processes, documentation and levels of pre-analysis.

**3.1.11.3.1 Case Samples** The patient samples were collected in biological form, our study was an international collaboration which resulted in samples being received in batches over several years with differing sample naming schemes. We developed a new internal sample naming system before they were sent for genotyping by creating a versioned mapping file between the two ID schemes (Figure 3.7, p.159). The DNA samples were sent for genotyping at the Institute for Child Health after being prepared on 96 well plates with the required excel documentation and their original source identifiers. As the samples were returned, all data was matched to the internal sample IDs, catalogued and saved to drives that were part of our data redundancy infrastructure. Any misformatting of data or missing samples were dealt with at the point of receipt of data — the final genotyped input dataset for SSNS comprised 1698 samples from 20 separate sample sets.

We manually filtered these samples to a set of 712 samples of reported European ancestry based on the location of the global centre from which they came; this comprised the input data for cases to our data pipeline.



**Figure 3.7:** Internal Sample ID scheme for our SSNS GWAS.

**3.1.11.3.2 Control Samples** We identified potential sources of control data by searching online repositories such as dbGaP and EGA as well as large-scale projects such as the 1000 genomes project [225] and also control samples provided by Illumina based on samples from the Hap Map project [186]. Our final list of control data comprised of a study named the 'Oxford' dataset, Illumina controls based on the HapMap project, and a set of controls from the Wellcome trust case control consortium (WTCCC):

- The Oxford dataset is available through the European Genome Archive (EGAD00010000144 and EGAD00010000520), it provided data from 432 Individuals [226] [227].

- CEU Illumina ethnicity controls were obtained from Illumina and provided data from 90 individuals [228].
- WTCCC controls. This is a combined dataset comprising the 1958 UK birth cohort controls and the UK blood service control group controls (Consortium 251), a total of 5,604 individuals. Data is available through the WTCCC website [187].

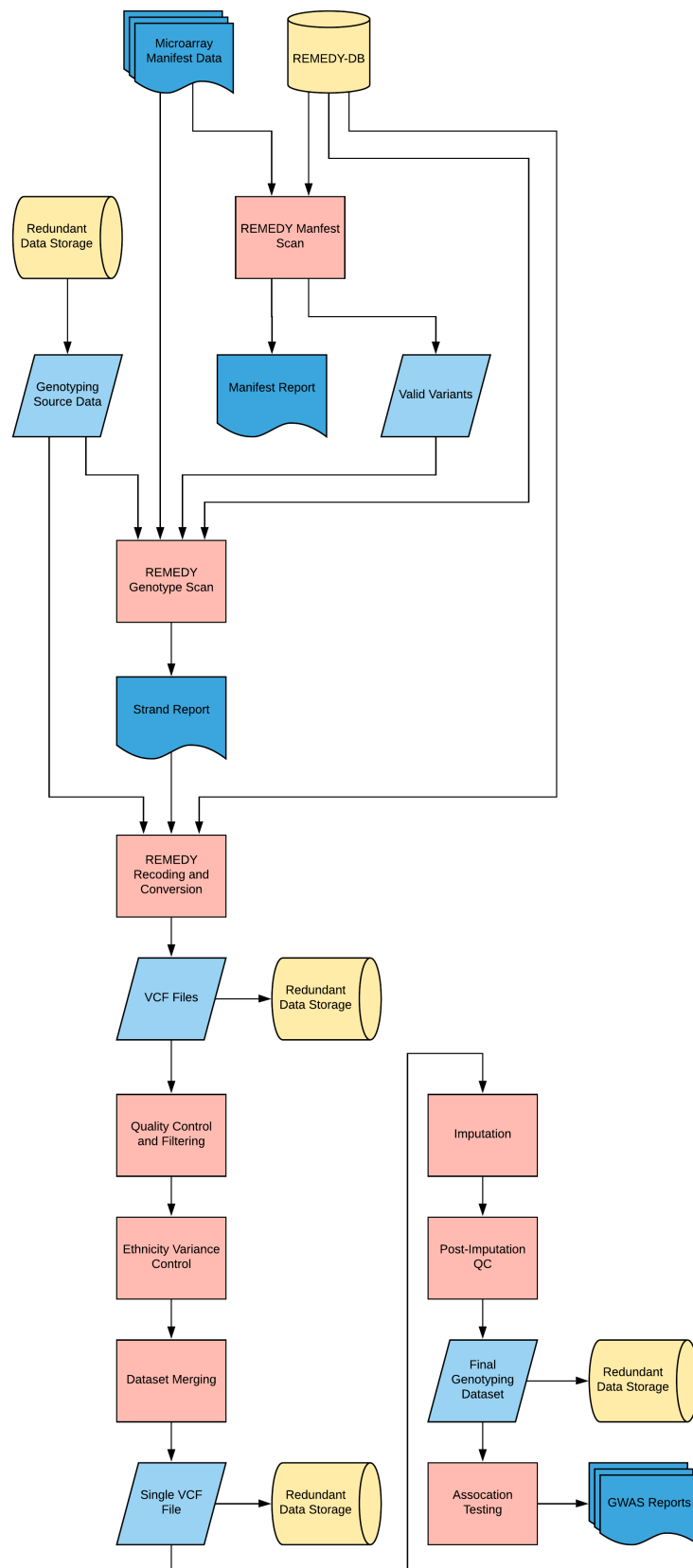
Despite the control data already being genotyped, pre-processing presented a unique set of challenges. Identifying what level of pre-processing had already been applied to the data combined with the challenges inherent with merging the case and control datasets to each other, presented a unique set of challenges. We discuss how we overcame these difficulties in detail in our methods section (2.3.1, p.107). Our control sets were genotyped on a range of microarrays, strand designation schemes and input file formats (Table 3.1, p.160).

Dataset	Microarray	Sample No.	SNV Count	Encoding	File Format
<b>Illumina Controls</b>	HumanOmniExpress-12v1-C	270	711,320	FWD	Final Report GATC
<b>Oxford Controls</b>	HumanOmniExpress-12v1-J	432	712,893	FWD	Matrix GATC
<b>WTCCC-1958</b>	Human1-2M-DuoCustom-v1-A	2867	1,106,184	DESIGN	Final Report AB
<b>WTCCC-NBS</b>	Human1-2M-DuoCustom-v1-A	2737	1,106,188	DESIGN	Final Report AB

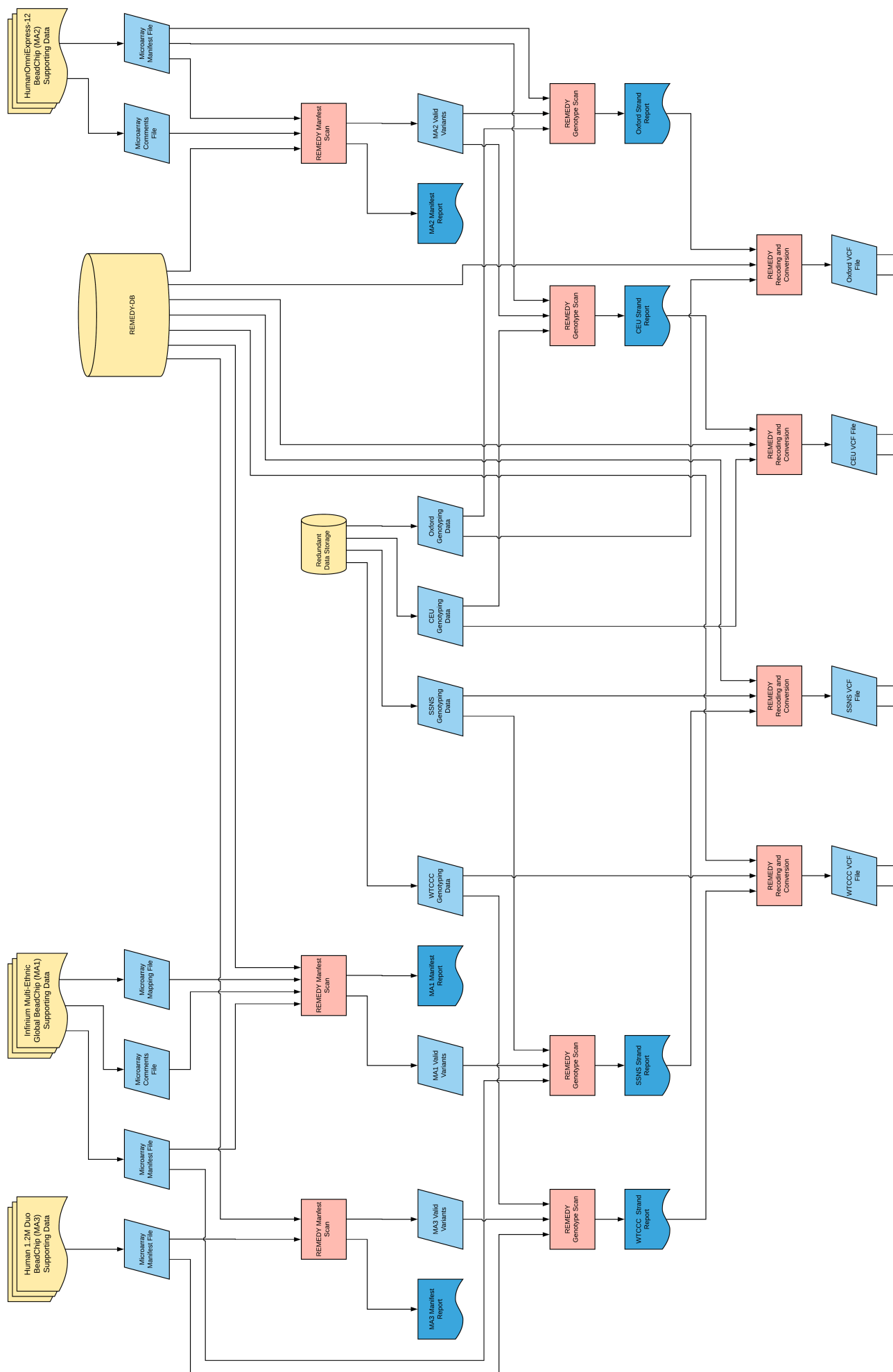
**Table 3.1:** Control data subsets for the SSNS GWAS. The table shows the range of formats and encoding schemes encountered.

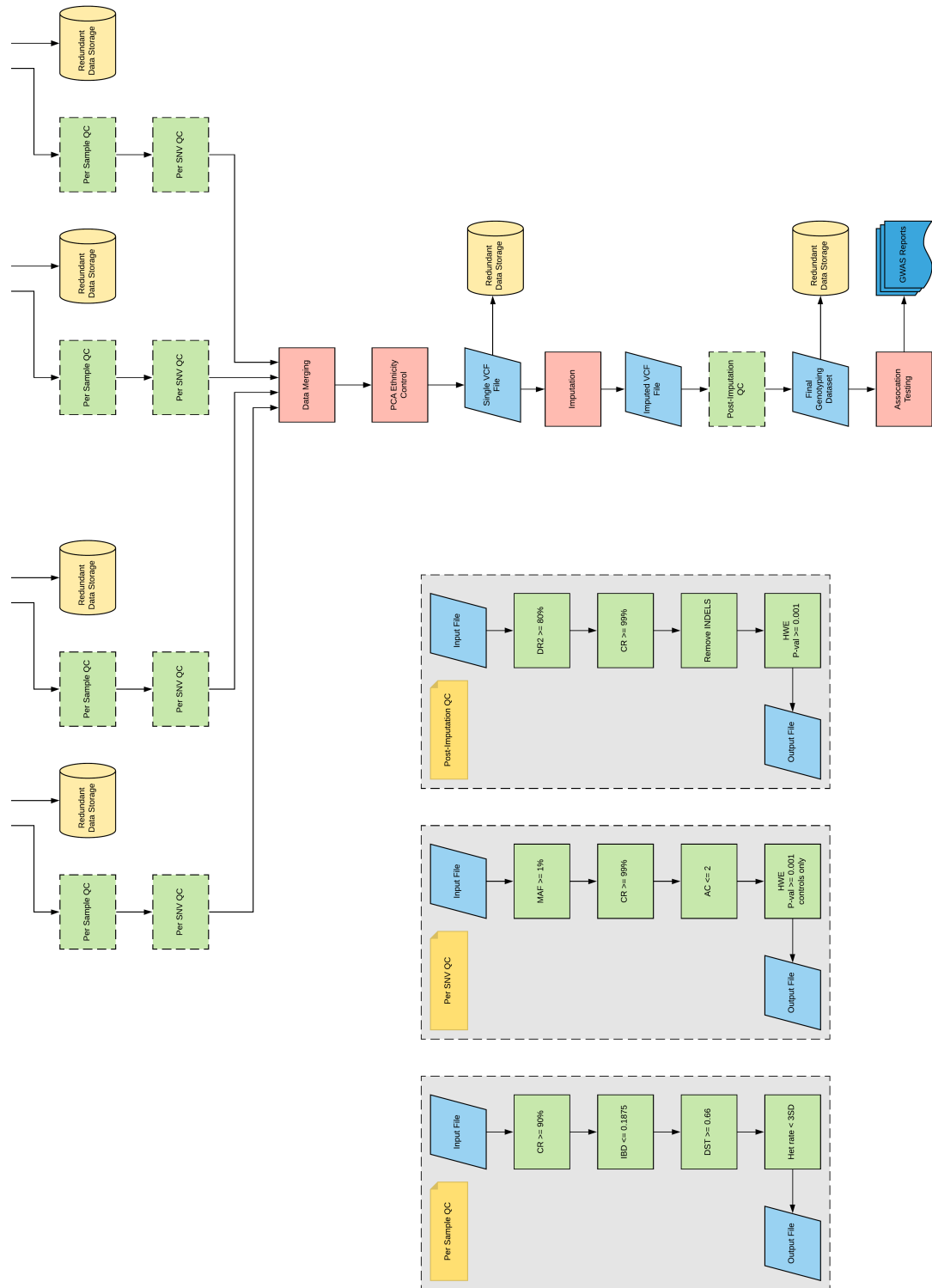
#### 3.1.11.4 Pipeline Design

Here we first present our generalised pipeline design for a GWAS based on genotyping data (Figure 3.8, p.161); we show the flow of data from first inputs through to the primary result required (the association testing results). We then expand on this template and apply the full pipeline design principles shown in the methodology to develop a processing strategy for our SSNS GWAS (Figure 3.9, p.163). We discuss the full pipeline in detail below, highlighting each node and edge as per our methodology.



**Figure 3.8:** A generalised pipeline design for a GWAS using REMEDY with genotyping input data.





**Figure 3.9:** Full SSNS GWAS pipeline data graph.



**3.1.11.4.1 Source Data** The input data for our SSNS pipeline comprises the SSNS case dataset in the form of one file per well-plate in Genome Studio matrix format, which was requested to be designated to the Design strand, and the three control datasets from our selected online resources. The control datasets were also contained within a series of files in the case of the WTCCC and Oxford controls or a single file as was the case with the CEU Illumina data.

It is important to note that the SSNS case dataset is not merged before entry into the pipeline; although the case data was processed by the same lab and the same machine, it was generated by different technicians at different times; therefore, the data must be subject to extensive genotyping data consistency checks before merging. REMEDY was designed with this in mind; therefore, each SSNS data node is, in fact, one file per batch. We extended REMEDY to handle folder processing as well as single files; therefore, the processing steps do not change; the data merging step simply has more files to merge.

**3.1.11.4.2 Microarray Supporting Data** We accessed the support file repository on the Illumina website and downloaded all available manifest, comment and rsID mapping files for human genome version GRCh37 and dbSNP version 150 for each of the three microarrays used to genotype our source data. While the Multi-Ethnic Global chip had a comments file and required a mapping file, other arrays did not need a mapping file (rsIDs were used in the manifest instead of Illumina internal identifiers); comments files were available for two out of three of the microarrays. Given that the Human 1.2M Duo BeadChip is a relatively old array, it is possible that comments files did not exist at the time of manufacture.

**3.1.11.4.3 REMEDY-DB** As discussed in our methods section, REMEDY-DB is a custom file-based representation of dbSNP that has fast tree-based indexing for searching by rsID. Variant data is used the manifest scanning and re-coding portions of REMEDY; we show it as a separate item in our pipeline as it operates as a stand-alone database that has its own space and performance requirements.

**3.1.11.4.4 Redundant Data Storage** We describe in our methodology that project data (especially source data) should be stored in a redundant storage configuration. We show in our data pipeline points at which data is retrieved or transferred from long-term storage; this aids in storage planning and also when making edge calculations. We show our actual storage configuration in the pipeline implementation section below.

**3.1.11.4.5 Manifest Scan** We perform a manifest scan for each of the three target microarrays in the pipeline. We define a manifest report node to represent the various reporting outputs for this stage (including logging) that should be manually checked by the user to ensure the correct files are being used and output is as expected. The primary output for the data pipeline for the manifest scan is the list of valid variants; this is used by the other portions of REMEDY to filter for variants that are valid for GWAS only.

**3.1.11.4.6 Genotype Scan** REMEDY scans the genotyping data, the primary output is a logged strand designation prediction which is needed to select the correct command line inputs for the re-coding and conversion section. We show this in our pipeline as a strand report node that feeds into subsequent nodes.

**3.1.11.4.7 Re-coding and Conversion** The re-coding and conversion system is the last stage of the REMEDY processing block. It takes inputs in various input file formats and strand designation schemes and re-encodes them to a common scheme in VCF file format; only variants technically valid for GWAS are included in the final data files. The VCF files are saved to redundant storage before further processing.

**3.1.11.4.8 Per Sample QC** Per-sample quality control is designed to identify low-quality samples, duplicates and related individuals. We break out the process into a separate box for space considerations but also to show that it is a modular process. The full methods for quality control can be found in Dufek *et al.* [1].

In summary, quality control per sample included removal of individuals with a call rate of < 90%, removal of duplicates or related individuals *via* Identity by descent (IBD), removal of samples with average identity by state distance (DST) of < 0.66 and heterozygosity rates of more than three standard deviations below or above the mean.

**3.1.11.4.9 Per SNV QC** Per-SNV quality control is designed to identify low-quality variants and variants which are unsuitable for GWAS. We break out the process into a separate box for space considerations but also to show that it is a modular process. The full methods for quality control can be found in Dufek *et al.* [1].

In summary, quality control per variant included the removal of markers with a call rate of < 99%, markers with more than one alternate allele and a minor allele frequency of < 1%. Markers in control datasets with a significant deviation (p-value < 0.001) from Hardy-Weinberg equilibrium (HWE) were all removed.

**3.1.11.4.10 Data Merging** The data merging stage is responsible for combining all of the single genotyping files into one dataset. At this point, it is essential to include extra information into the dataset to indicate which dataset the sample came from and whether it is a case or control. We used SNP & Variation Suite (SVS) to perform the merging and extra annotation.

**3.1.11.4.11 PCA Ethnicity Control** It is essential to control for population stratification prior to performing an association test. We pre-selected input samples for European ancestry based on collection location; we add in process nodes for more precise PCA-based stratification control here.

**3.1.11.4.12 Imputation** Imputation is a standard method in GWAS for increasing the marker density and thus the resolution of association testing; we add process nodes for Imputation at this point. Post-imputation quality control is designed to apply similar principles to that of per-SNV Quality control (QC) on our imputed dataset, which now contains many SNVs which did not pass through our initial QC and filtering checks. We break out the process into a separate box for space considerations but also to show that it is a modular process. The full methods for post-imputation quality control can be found in Dufek *et al.* [1].

In summary, we remove those markers with Dosage R-Squared (DR2) of  $< 80\%$ , minor allele frequencies of  $< 1\%$ , call rates of  $< 99\%$  and Hardy-Weinberg deviation for p-values of  $< 0.001$ . We also remove all structural variants, in-line with our GWAS methodology.

**3.1.11.4.13 Association Testing** Association testing is the primary result generator for our data pipeline. It is responsible for producing a list of variants included in the analysis with their associated p-values that represent how different the cases are from controls at that loci. We add a report node for all reports that can be generated from the base result set including Manhattan plots and other related reporting data.

### **3.1.11.5 Node Resolution**

Using the completed pipeline and following our design methodology, we next resolved our nodes in terms of selecting software and toolsets to complete the process tasks. The initial stages of the pipeline up to the quality control section are handled by REMEDY. Several options exist for quality control; SVS for all our GWAS analysis. SVS is a GUI based analysis

suite which has options for importing VCF files into an internal storage scheme from which operations can be performed including QC, merging and association testing. We used SVS as our base for data processing during these stages; for any external operations, we exported to file, performed the operation and then re-imported back into SVS. Per-sample, per-SNV and data merging were performed on the imported VCF files from within SVS except for the IBD and DST calculations for which we selected the PRIMUS software suite [229]. Ethnicity control was conducted from within SVS before we exported our dataset for imputation by Beagle 5.0 [230]. During initial testing, SVS was unable to handle the number of SNPs generated by imputation; we, therefore for post-imputation QC, used bcftools for DR2, Minor allele frequency (MAF) and CR filtering [221] and PLINK v1.90 beta [231] for HWE thresholding to reduce the total number of SNVs prior to re-importing back into SVS for association testing.

### **3.1.11.6 Edge Analysis**

Edge resolution further resolves the data graph to predict data storage requirements during processing and identify potential performance bottlenecks. Following our pipeline design methodology, we used the known data volumes at the input nodes combined with knowledge of the selected node programs to create a prediction of storage at key edges on the graph (Table 3.2, p.168).

While it is impossible to know the exact volume of data until pipeline run-time, we predicted values based on approximate known ratios between different file formats and on test data. We estimated we needed at least 7.6 TB of working storage to store all data in a pipeline run, but only 1.1 TB of redundant storage for storing the final results. Based on the edge information we created a new 40 TB storage server specifically to deal with the large volume of imputed data which was predicted by our pipeline design. In terms of performance considerations, we identified that imputation of 8000 samples could be a significant bottleneck given the large volumes of data predicted by the edge analysis.

Dataset	Final Report	Matrix Conversion	Re-coding	Imputed	Compressed	Total
Illumina	14 GB	~0.5 GB	~0.7 GB	~90 GB	~5 GB	
Oxford	36 GB	~1.2 GB	~1.4 GB	~180 GB	~10 GB	
WTCCC-1958	311 GB	~10 GB	~12 GB	~1600 GB	~90 GB	
WTCCC-NBS	298 GB	~10 GB	~12 GB	~1600 GB	~90 GB	
SSNS	N/A	18 GB	~20 GB	~3100 GB	~180 GB	
<b>Total Working</b>	659 GB	~40 GB	~46 GB	~6500 GB	~400 GB	~7645 GB
<b>Total Stored</b>	659 GB	~40 GB	~46 GB	N/A	~400 GB	~1145 GB

**Table 3.2:** Storage predictions for key parts of the SSNS pipeline. these values were calculated as part of the edge analysis stage of our data pipeline methodology. The predictions were based on approximate known ratios between different file formats and on test data. We estimated we needed at least 7.6 TB of working storage to store all data in a pipeline run, but only 1.1 TB of redundant storage for storing the final results.

### 3.1.11.7 Pipeline Implementation

We first identified the required operating systems based on the node resolution steps performed in previous sections. REMEDY is capable of running on either Windows or Linux based systems while all other tools with the exception of SVS run on Linux only. SVS is a windows based software suite; therefore, we identified the need to use at least two machines running Windows and Linux separately. Given the performance requirements of the Linux part of the computation from our edge analysis, we opted to run all of our Linux tools on an in-house compute server. REMEDY and SVS processing nodes were run on a high-performance windows desktop custom created for the project.

We created a 40 TB RAID 10 redundant storage array in-house using a custom built server, we then additionally stored all raw data onto portable hard disks and stored them off-site to give multiple-location redundancy. We installed a  $1 \text{ Gbit s}^{-1}$  network between the storage server, Windows desktop and Linux compute servers to enable high-speed data transfer between nodes in the pipeline. Data transfer could have been automated between the servers to better adhere to our design practices; however, in this instance, we opted to manually storage and transfer data between servers.

### 3.1.11.8 REMEDY Processing

We present here the results for all REMEDY based processing in the initial stages of the pipeline.

**3.1.11.8.1 Manifest Scanning** We scanned each of the three manifests pertaining to the set of microarrays utilised in our cases and controls; additional manifests from three other popular human genotyping arrays manufactured by Illumina were also assessed by REMEDY to provide more test samples for comparison at the manifest scanning stage. Details of the microarrays assessed and their column alias for the results summary table are shown in Table 3.3, p.169, the REMEDY processing results are summarised in Table 3.4, p.172.

Microarray	Summary Table Alias
HumanOmniExpress-12 v1.0	Omni Ex
Human1-2M-DuoCustom v1.0	Duo Custom
Multi-Ethnic Global Array	MEGA
InfiniumOmni2.58 v1.3	Duo Custom
HumanCore-24 v1.0	Core
Infinium Global Screening v3.0	Screening

**Table 3.3:** List of Illumina microarrays assessed using REMEDY.

The initial loaded SNV count for each microarray falls broadly in line with the age of the array, with older products such as the *Duo custom* and the *Omni Ex* having lower numbers of SNVs than current products such as the *MEGA* and *Omni 2.5* (Illuminas flagship microarray with 2.5m SNVs). The exceptions are the *Infinium core*, which is a more economical but newer array and the *Global screening*, which is targeted more for pharmacogenomics studies, and precision medicine research.

None of the arrays contained duplicate Ids, which was expected as the manifests were downloaded directly from the manufacturer's websites. This feature was added to REMEDY when the origin of the manifest could not be confirmed and is a sanity check step only.

The strand analysis stage showed that the majority of SNVs on all arrays were designed to TOP or BOT. MINUS and PLUS designations denote probes that are targeted to structural variation on the 3' and 5' biological strands. The Omni range of arrays were designed with very little structural variants included as opposed to the *MEGA*, *Infinium core* and *Global screening* arrays, which have a number of structural variants included that increases approximately in proportion to their size. The outlier is the *Duo custom*, which has a disproportional amount of structural variants configured for the 5' strand. This array is the oldest of the group therefore, this could be due to older design considerations; however, the true reason cannot be deduced from the source data and therefore, is not discussed further here.

Most of the arrays were supplied with comment files except for the *Duo custom*; this is most likely due to the age of the array. The number of comments in each category fell broadly in line with the size of the array except for the *MEGA*, where there was a significantly larger number of SNVs included on the array that mapped to multiple or PAR-like regions. This, again, is most likely due to the intended function of the array that is specifically designed to target ethnicity-specific variants. The design decision to include a variant onto the textitMEGA array relied upon further considerations in addition to genomic coverage and therefore, technically less desirable SNVs were included; specifically, additional variants were included by an external consortium.

The rsID mapping stage attempts to find a valid rsID with which to map the variant to dbSNP using a mapping file supplied by the manufacturer. The *Omni Ex* and *Duo Custom* arrays had the manifest row ids already encoded as non-Illumina internal ids and thus required no mapping file. For the other arrays, there were a number of different counts of invalid mappings included in the mapping file. REMEDY attempts to find a valid rsID for every variant; therefore, at this stage, if a custom variant is included on an array that does not have an id that begins with 'rs', it is listed as invalid at this stage. If no mapping file exists, then this check falls through to the dbSNP matching stage; no direct comparison can be drawn between the arrays without mapping files and the arrays with at this stage of processing. For the arrays that do include mapping files, there is no discernible pattern between the numbers as this is simply due to design considerations as to how many custom variants were included on the chip.

The most inter-array comparison can be performed at the dbSNP matching and meta-data retrieval stages. The *Duo custom* has a large number of SNVs (>100k) that do not match to dbSNP; this is unsurprising as it is the oldest array and, as REMEDY uses the latest version of dbSNP for the most up-to-date meta-data, many more SNVs have been removed or merged since the design of the array. On the other arrays, there is a small number of unmatched variants in proportion to the size of the array versus the age and amount of structural variants included in the design.

The number of multi-allelic SNVs removed by REMEDY is directly proportional to the size of the array and does not have any correlation with age. This at first seems counter-intuitive until the release schedule of dbSNP is taken into consideration. Recent releases of dbSNP have significantly increased the number of submissions used to create the rsIDs due to the explosion of genetic information being generated by the scientific community. This has resulted in a situation whereby many variants have gone multi-allelic within the space of a small number of recent dbSNP releases; consequently, the number of multi-allelic SNVs on an array becomes a function of size.

The number of strand mismatches is a more complex situation. The counts between arrays are broadly a function of both size and age; the *Duo custom* for example, has the largest count of mismatches due to the age of design versus the latest version of dbSNP while the Core, being one of the smallest yet newest arrays, has the lowest amount. The outlier is the Global screening array, this is due to the number of structural variants such as CNVs that are included on the array. This is related to the intended function such as screening for pharmacogenomics sites; these variants are usually less-well described in dbSNP and are therefore subject to strand changes as more submissions are collected that affect the rsID calculations.

In conclusion, across all metrics that are not a result of design considerations, the number of 'lost' or filtered variants by REMEDY is a function of size and the age of the array. Old arrays fall out of touch with dbSNP and therefore find many variants have been removed, merged with other Ids or have been designated multi-allelic. Larger arrays have statistically more chance of having problematic variants; the counts are then skewed by array age. Arrays which have more specific functions other than genome-wide population genetics may or may not fall into this model dependant on how many structural variants or areas of the genome that are difficult to genotype are included in the array design.

In terms of the first three arrays included in the analysis that were used by sample sets in our study dataset, the results show that many thousands of variants on each microarray are unsuitable for GWAS; these SNVs all have the potential to cause noise in subsequent result sets, but are detected and removed by REMEDY.



Phase	Metric	Omni Ex	Duo Custom	MEGA	Omni 2.5	Core	Screening
<b>Manifest Load</b>	Loaded SNVs	719,665	1,238,733	1,748,250	2,372,784	654,027	306,670
	Duplicates	0	0	0	0	0	0
	<b>Strict Loci Rem.</b>	<b>719,665</b>	<b>1,238,733</b>	<b>1,748,250</b>	<b>2,372,784</b>	<b>654,027</b>	<b>306,670</b>
<b>Strand Analysis</b>	No. BOT	349,809	572,048	877,057	1,183,883	322,354	143,436
	No. TOP	369,855	591,763	846,738	1,188,860	321,555	150,571
	No. MINUS	0	237	11,493	11	4,868	5,743
	No. PLUS	1	74,685	12,962	30	5,250	6,920
	<b>Strict Loci Rem.</b>	<b>719,664</b>	<b>1,163,811</b>	<b>1,723,795</b>	<b>2,372,743</b>	<b>643,909</b>	<b>294,007</b>
<b>Comments</b>	Comments Loaded	1,363	-	15,768	4,675	1,388	801
	Multiple Mappings	546	-	11,451	1,860	507	683
	PAR-like Region	722	-	4,311	2,558	879	106
	No Probe Mapping	77	-	6	257	2	12
	<b>Strict Loci Rem.</b>	<b>718,302</b>	<b>1,163,811</b>	<b>1,708,643</b>	<b>2,368,070</b>	<b>642,531</b>	<b>293,587</b>
<b>rsID Mapping</b>	Unmatched to Manifest	-	-	0	0	0	0
	Matched to Manifest	-	-	1,748,250	2,372,784	654,027	306,670
	Invalid Mappings	-	-	12,534	2,491	5,700	23,859
	Valid Mappings	-	-	1,735,716	2,370,293	648,327	282,811
	<b>Strict Loci Rem.</b>	<b>718,302</b>	<b>1,163,811</b>	<b>1,707,934</b>	<b>2,367,697</b>	<b>638,225</b>	<b>282,284</b>
<b>rsID Matching</b>	Loci with rsIDs	719,665	1,238,733	1,735,716	2,370,293	648,327	282,284
	Unmatched to dbSNP	1,198	100,757	7,280	5,831	6,859	535
	Matched to dbSNP	718,467	1,137,573	1,728,436	2,364,462	641,468	282,276
	Multi-allelic	45,018	72,016	148,545	160,296	45,332	17,800
	Strand Mismatches	115,616	184,351	66,972	112,589	39,476	43,116
	Invalid Positions	5	50	2	0	1	0
	Deletions	0	189	15,307	1	4,191	249
	Insertions	0	206	5,023	12	1,379	97
	Indels	0	8	34	0	67	0
	<b>Strict Loci Rem.</b>	<b>672,361</b>	<b>1,065,696</b>	<b>1,565,402</b>	<b>2,201,986</b>	<b>591,669</b>	<b>264,182</b>
	Total Removed	47,304	173,037	182,848	170,798	62,358	42,488
	<b>Total Valid SNVs</b>	<b>672,361</b>	<b>1,065,696</b>	<b>1,565,402</b>	<b>2,201,986</b>	<b>591,669</b>	<b>264,182</b>

**Table 3.4:** Summary of the core output from the manifest scan stage of the REMEDY toolset when run inside the SSNS GWAS pipeline. The *Strict Loci Rem.* rows show how many valid variants are left in the dataset after each filter pass.

**3.1.11.8.2 Genotype Scanning** Each dataset was made up of multiple individual files that we scanned as a batch. We show the results for the Illumina controls in Table 3.5, p.173, for the Oxford controls in Table 3.6, p.173, for the WTCCC birth cohort in Table 3.7, p.174, for the WTCCC blood cohort in Table 3.8, p.174 and the SSNS cases in Table 3.9, p.175.

Filename	Strand Prediction	TOP (%)	BOT (%)	FWD (%)	REV (%)
47CHB_11162012	FWD	49.92	50.08	99.75	0.25
47JPT_11162012	FWD	49.94	50.06	99.79	0.21
92YRI_11162012	FWD	49.97	50.03	99.84	0.16
95CEU_11162012	FWD	49.91	50.09	99.73	0.27

**Table 3.5:** Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the Illumina controls dataset.

Filename	Strand Prediction	TOP (%)	BOT (%)	FWD (%)	REV (%)
Oxf1	DESIGN	51.29	48.71	49.85	50.15
Oxf2	DESIGN	51.3	48.7	49.86	50.14

**Table 3.6:** Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the Oxford controls dataset.

Filename	Strand Prediction	TOP (%)	BOT (%)	FWD (%)	REV (%)
20100416_10	TOP	96.81	3.19	48.71	51.29
20100416_11	TOP	96.85	3.15	48.73	51.27
20100416_12	TOP	96.57	6.43	47.09	52.91
20100416_13	TOP	96.62	3.38	48.62	51.38
20100416_1	TOP	96.57	3.43	48.59	51.41
20100416_2	TOP	96.61	3.39	48.61	51.39
20100416_3	TOP	96.64	3.36	48.63	51.37
20100416_4	TOP	96.77	3.23	48.69	51.31
20100416_5	TOP	97.06	2.94	48.84	51.16
20100416_6	TOP	96.24	7.76	46.41	53.59
20100416_7	TOP	96.88	3.12	48.75	51.25
20100416_8	TOP	95.92	4.08	48.27	51.73
20100416_9	TOP	96.43	3.57	48.52	51.48

**Table 3.7:** Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the WTCCC birth controls dataset.

Filename	Strand Prediction	TOP (%)	BOT (%)	FWD (%)	REV (%)
FCR_NBS_1_1	TOP	95.49	4.51	48.16	51.74
FCR_NBS_1_2	TOP	95.68	4.32	48.26	51.74
FCR_NBS_2_1	TOP	95.46	4.54	48.15	51.85
FCR_NBS_2_2	TOP	95.6	4.4	48.22	51.78
FCR_NBS_3_1	TOP	94.97	5.03	47.89	52.11
FCR_NBS_3_2	TOP	95.44	4.56	48.14	51.86
FCR_NBS_4_1	TOP	95.75	4.25	48.29	51.71
FCR_NBS_4_2	TOP	95.86	4.14	48.35	51.65
FCR_NBS_5_1	TOP	95.59	4.41	48.21	51.79
FCR_NBS_5_2	TOP	95.49	4.51	48.16	51.84
FCR_NBS_6_1	TOP	95.13	4.87	47.98	52.02
FCR_NBS_8_1	TOP	96.03	3.97	48.44	51.56
FCR_NBS_8_2	TOP	95.23	4.77	48.03	51.97

**Table 3.8:** Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the WTCCC blood controls dataset.

Filename	Strand Prediction	TOP (%)	BOT (%)	FWD (%)	REV (%)
Kleta 10 Full Data Table	DESIGN	52.35	47.65	54.67	45.33
Kleta 11 Full Data Table	DESIGN	52.31	47.69	54.66	45.34
Kleta 12 Full Data Table	DESIGN	52.31	47.69	54.66	45.34
Kleta 13 Full Data Table	DESIGN	52.35	47.65	54.67	45.33
Kleta 14 Full Data Table	DESIGN	52.33	47.67	54.67	45.33
Kleta 15 Full Data Table	DESIGN	52.41	47.59	54.68	45.32
Kleta 16 Full Data Table	DESIGN	52.31	47.69	54.66	45.34
Kleta 17 Full Data Table	DESIGN	52.39	47.61	54.67	45.33
Kleta 18 Full Data Table	DESIGN	52.29	47.71	54.66	45.34
Kleta 19 Full Data Table	DESIGN	52.34	47.66	54.67	45.33
Kleta 20 Full Data Table	DESIGN	52.33	47.67	54.67	45.33
Kleta 21 Full Data Table	DESIGN	52.3	47.7	54.66	45.34
Kleta 22 Full Data Table	DESIGN	52.29	47.71	54.66	45.34
Kleta 23 Full Data Table	DESIGN	52.29	47.71	54.66	45.34
Kleta 24 Full Data Table	DESIGN	52.31	47.69	54.67	45.33
Kleta 25 Full Data Table	DESIGN	52.31	47.69	54.67	45.33
Kleta 26 Full Data Table	DESIGN	52.28	47.72	54.66	45.34
Kleta 27 Full Data Table	DESIGN	52.29	47.71	54.66	45.34
Kleta 28 Full Data Table	DESIGN	52.28	47.72	54.67	45.33
Kleta 4 Full Data Table	DESIGN	53.02	46.98	54.69	45.31
Kleta 5 Full Data Table	DESIGN	52.55	47.45	54.69	45.31
Kleta 6 Full Data Table	DESIGN	52.35	47.65	54.66	45.34
Kleta 7 Full Data Table	DESIGN	52.32	47.68	54.67	45.33
Kleta 8 Full Data Table	DESIGN	52.28	47.72	54.66	45.34
Kleta 9 Full Data Table	DESIGN	52.31	47.69	54.67	45.33
Kleta Plate 3 Full Data Table	DESIGN	52.45	47.55	54.67	45.33
Kleta Plate 2 Full Data Table	DESIGN	52.36	47.64	54.67	45.33
Kleta1 Full Data Table	DESIGN	52.32	47.68	54.67	45.33
MRC MN 1 Full Data Table	DESIGN	52.29	47.71	54.67	45.33
MRC MN 2 Full Data Table	DESIGN	52.36	47.64	54.68	45.32
MRC MN 3 Full Data Table	DESIGN	52.33	47.67	54.68	45.32
Nijmegen 1 Full Data Table	DESIGN	52.36	47.64	54.66	45.34
Nijmegen 2 Full Data Table	DESIGN	52.43	47.57	54.66	45.34
Plate 29 Full Data Table	DESIGN	52.31	47.69	54.67	45.33
UKM8121 Full Data Table	DESIGN	52.33	47.67	54.67	45.33
UKM8122 Full Data Table	DESIGN	52.3	47.7	54.67	45.33

**Table 3.9:** Processing results for the REMEDY genotype scan run inside the SSNS GWAS pipeline for the SSNS cases dataset.

**3.1.11.8.3 Re-coding and Conversion** Each file shown in the genotype scanning stage was subject to re-coding by REMEDY. For each dataset, the predicted strand designation from the genotype scan stage was supplied to the re-coding REMEDY sub-program as the source encoding; the destination encoding for every file was set to FWD. A list of valid SNVs from the manifest scan stage for each dataset was also supplied so that only valid GWAS SNVs were included in the final VCF file.

### **3.1.11.9 Quality Control, Filtering and merging**

Per-sample and per-SNV QC were performed in SVS per case or control dataset. After re-coding and conversion, we merged all batch file sets into a single dataset for each overarching dataset classification of SSNS cases, Oxford, Illumina CEU and WTCCC. We formed QC on these four datasets individually as per the pipeline design. Per-sample QC results were as follows:

- For the SSNS cases (n=712), samples were removed because of call rate (CR) <90% (n=3), IBD >0.1875 (n=58), average DST <0.66 (n=0) and heterogeneity >3SD (n=26). Remaining were 625 individuals.
- For the Oxford controls (n=432), samples were removed because of CR <90% (n=0), IBD >0.1875 (n=6), average DST <0.66 (n=0) and heterogeneity >3SD (n=6). Remaining were 420 individuals
- For the Illumina CEU controls (n=90), samples were removed because of CR <90% (n=0), IBD >0.1875 (n=30), average DST <0.66 (n=0) and heterogeneity >3SD (n=1). Remaining were 59 individuals.
- For the WTCCC controls (n=5604), samples were removed because of CR <90% (n=55), IBD >0.1875 (n=67), average DST <0.66 (n=0) and heterogeneity >3SD (n=82). Remaining were 5400 individuals.

After QC of the samples, the total number of cases was 625, and the total number of combined controls remaining was 5879. Per-SNV QC results were as follows:

- For the cases (autosomal markers n=1,512,983), markers were removed because of CR <99% (n=222,889), MAF <1% (n=700,397) and multiallelic (n=0). Remaining were 669,943 markers.
- For the Oxford controls (autosomal markers n=653,959), markers were removed because of CR <99% (n=13,310), MAF <1% (n=67,724), multiallelic (n=0), HWE  $p < 0.001$  (n=2,889). Remaining were 571,616 markers.

- For the Illumina CEU controls (autosomal markers  $n=653,959$ ), markers were removed because of CR  $<99\%$  ( $n=79,777$ ), MAF  $<1\%$  ( $n=68,185$ ), multiallelic ( $n=0$ ), HWE  $p<0.001$  ( $n=1,795$ ). Remaining were 516,372 markers.
- For the WTCCC controls (autosomal markers  $n=1,025,437$ ), markers were removed because of CR  $<99\%$  ( $n=133,778$ ), MAF  $<1\%$  ( $n=135,992$ ), multiallelic ( $n=0$ ), HWE  $p<0.001$  ( $n=16,941$ ). Remaining were 788,849 markers.

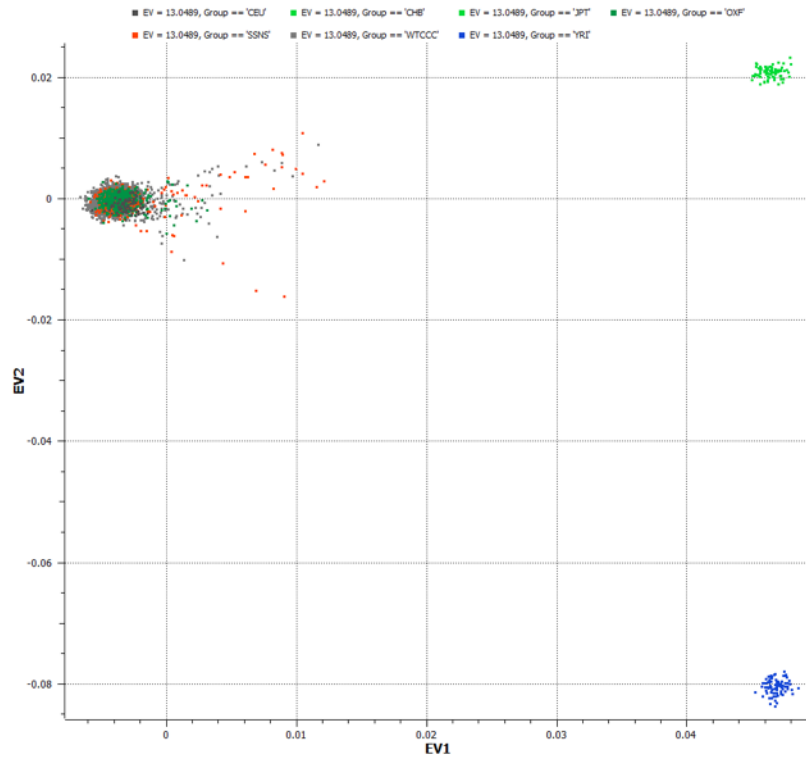
We combined the control sets at this point: the number of overlapping markers in the combined control dataset was 372,137. After combining with the case dataset, the overlapping marker count was 158,314. Subsequent QC was performed on the combined case-control dataset with HWE testing on controls only. The final number of markers in the combined dataset was 158,217.

#### **3.1.11.10 Principal Component Analysis**

We performed PCA on our combined cases and controls dataset using the first ten eigenvectors as our measure of variance. We subsequently used the eigenvectors to remove outliers from our dataset of greater than three standard deviations from the mean. We specify that outliers in terms of a PCA represent samples that are significantly different genetically than the average sample and are therefore of a different ethnicity (non-European); as discussed in our introduction and methods, large variances in ethnicity are a source of noise for GWAS and must be controlled. We visualised the two most significant eigenvectors as a 2-dimensional graph for reporting purposes (Figure 3.10, p.178); after filtering the dataset comprised 422 cases and 5642 controls with 158,217 markers.

#### **3.1.11.11 Imputation**

Imputation requires a source reference panel to make inferences about missing data; we selected the current industry standard, the 1000 Genomes Phase 3 data (version 5a) [232]. We exported our dataset from SVS before running imputation with the recommended settings for Beagle, we then performed the post-imputation QC described in our data pipeline immediately after. Our final dataset contained 5,216,266 markers, we then imported the output file back into SVS for further analysis.



**Figure 3.10:** PCA of our SSNS combined cases and controls after filtering outliers of more than three standard deviations. We plot the first and second principle components against each other. We include the full Illumina ethnicity controls in the input data for the PCA calculation so the ethnicity clustering is more clearly shown. The Asian cluster in the top right in light green and the African cluster in the bottom right have no additional sample dots. The European cluster in black is hidden by the sample dots from our case/control set. The plot shows our case/control samples are clustered in the European sector indicating that they are all European in genetic origin.

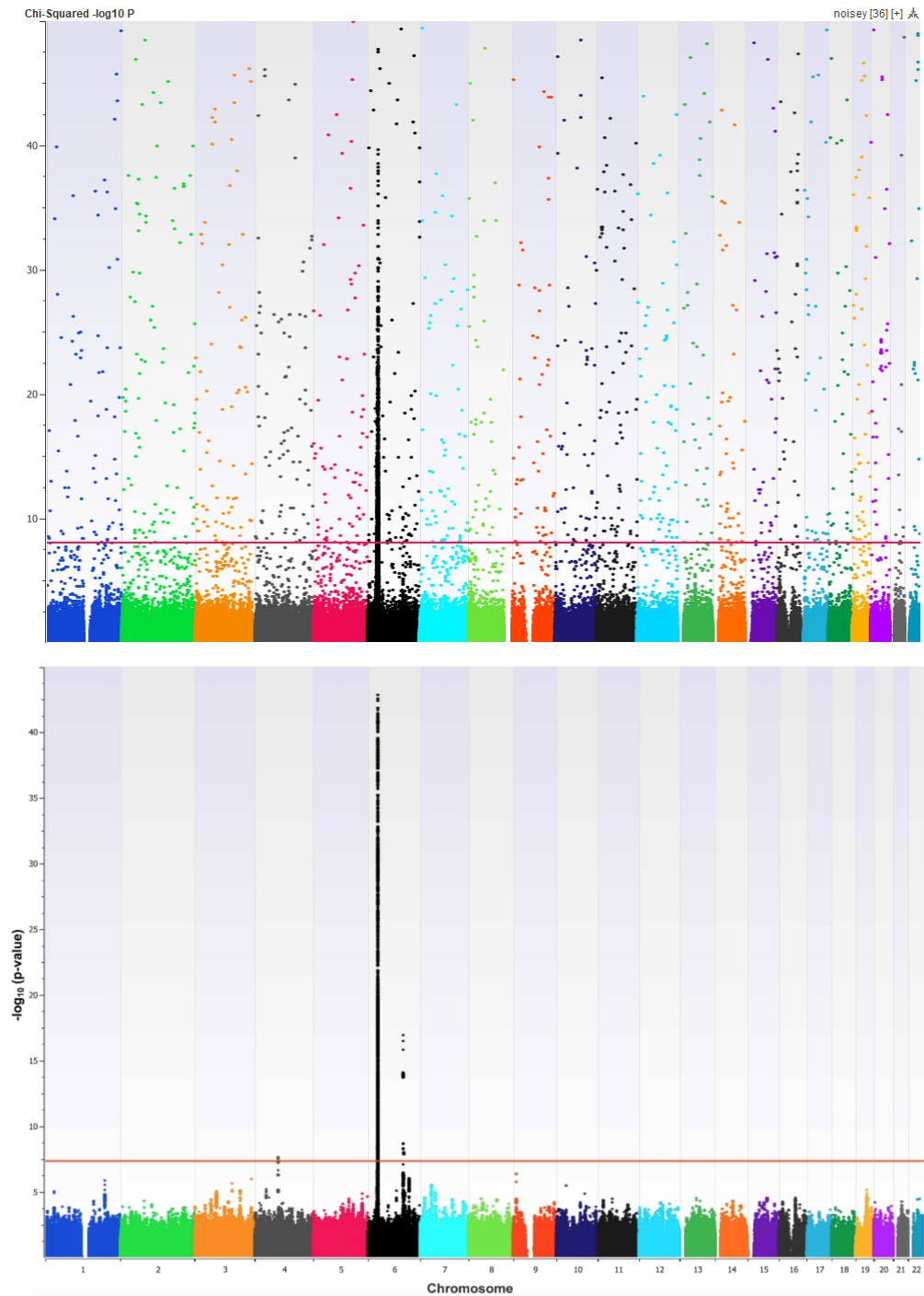
### 3.1.11.12 Association Testing

We performed association testing under a logistic regression model using the first ten principal components of a PCA performed without filtering out any samples on our final dataset. By using a regression model rather than basic allele testing only, we can minimise the effect of population stratification on our final results. The widely accepted p-value threshold of  $P < 5 \times 10^{-8}$  was used to declare genome-wide significance [153]; we generated a Manhattan plot to visualise our results for reporting purposes using the in-built functionality of SVS (Figure ??, p.??). The genomic inflation factor was calculated to quantify potential inflation of type I error due to stratification or technical artefacts. No substantial inflation was noted ( $\lambda = 1.027$ ). We also show the original Manhattan plot generated before REMEDY was developed and integrated into the SSNS analysis pipeline (Figure ??, p.??). We also show overlaid Q-Q plots, empirically showing the reduction in noise as a result of REMEDY processing (Figure 3.12, p.181).

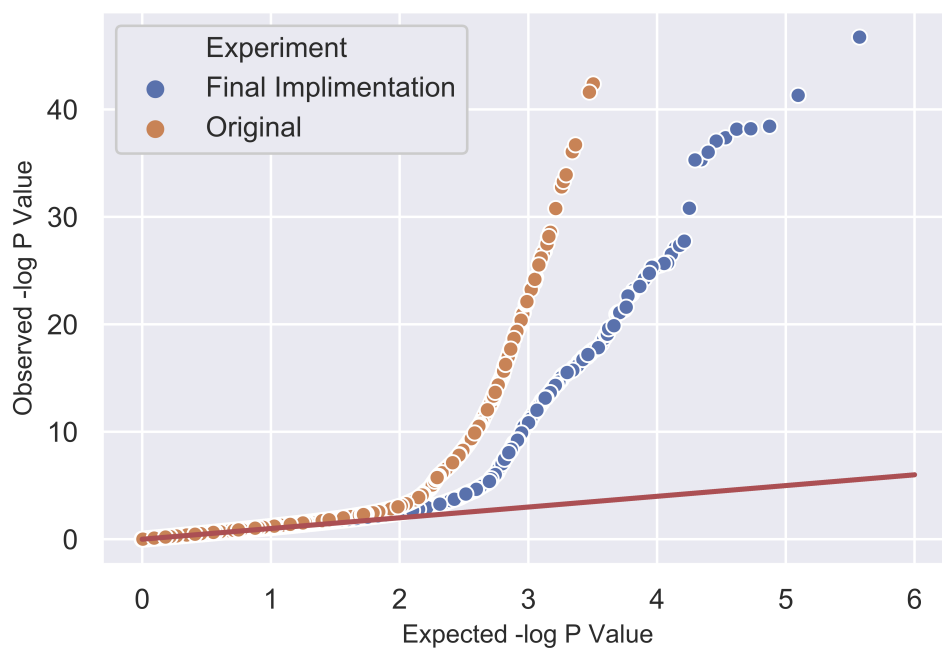
Association testing revealed three genetic loci achieving genome-wide significance: a

broad peak in the classical HLA region at band 6p21.32, some significant markers around a gene known as *Calcium Homeostasis Modulator Family Member 6* (*CALHM6*) and an intronic peak inside the *Prostate Androgen-Regulated Mucin-Like Protein 1* (*PARM1*) gene. The discovered associations stood up to rigorous statistical testing and were present with a low signal to noise ratio; this demonstrates the effectiveness of our GWAS and pipeline design methodology and our REMEDY software suite. For more detailed results and discussion of our findings, see Dufek *et al.* [1].





**Figure 3.11: Top:** Manhattan plot showing the original noisy plot with many artefact's. REMEDY was primarily responsible for cleaning and merging the data so that a clear plot could be generated (shown in the bottom diagram). **Bottom:** Manhattan plot of a basic allele test (BAT) performed on our case/control SSNS dataset. Each plot point shows the BAT calculated for one variant. The X axis shows the genomic position of the SNP and the Y axis shows the p-value of the BAT. The smaller p-values show higher on the graph. The plot shows several large peaks in the HLA region on chromosome 6 and some smaller significant peaks in other regions. The graph is clean with a high signal to noise ratio indicating that the cleaning and QC was done correctly.



**Figure 3.12:** Q-Q plot showing expected negative log p-values versus observed negative log p-values from a basic allele test performed on the SSNS GWAS. The plot shows an improved Q-Q plot that only deviates away from the expected at lower p-values. There is still a significant deviation away from the expected at high p-values as the signal on chromosome 6 is so strong.

## 3.2 LOOPER

Our proprietary software, LOOPER, was developed in two parts; the loop prediction algorithms were developed during the initial Hyperinsulinism with polycystic kidney disease (HIPKD) project while the loop visualisation portion was developed as a result of maturing chromatin confirmation experiments triggering a revisit of the project data.

We first present our algorithms for both loop prediction and loop visualisation before introducing the HIPKD project as a case study. We then apply both modules of LOOPER to the HIPKD region of interest and study the results to show what inferences can be made about chromatin looping using the toolset. Finally, we show how LOOPER can be integrated with other similar tools to increase further the value of insights gained from the input data.

We do not present any design specifications for LOOPER as the program is more a collection of scripts which grew organically with the HIPKD project. Given their experimental nature, they are designed to be used from code rather than be wrapped up into a program such as was the case with REMEDY.

### 3.2.1 Software Overview

LOOPER is a collection of small programs and scripts which aid in the prediction and visualisation of localised chromatin loop architecture. LOOPER integrates data from several different experimental sources that target analysis of chromatin interactions such as Chromatin immunoprecipitation with next generation sequencing (ChIP-Seq) [200] [201] [202] [197], Hi-C [70] and Chromatin interaction analysis by paired-end tag sequencing (ChIA-PET) [233].

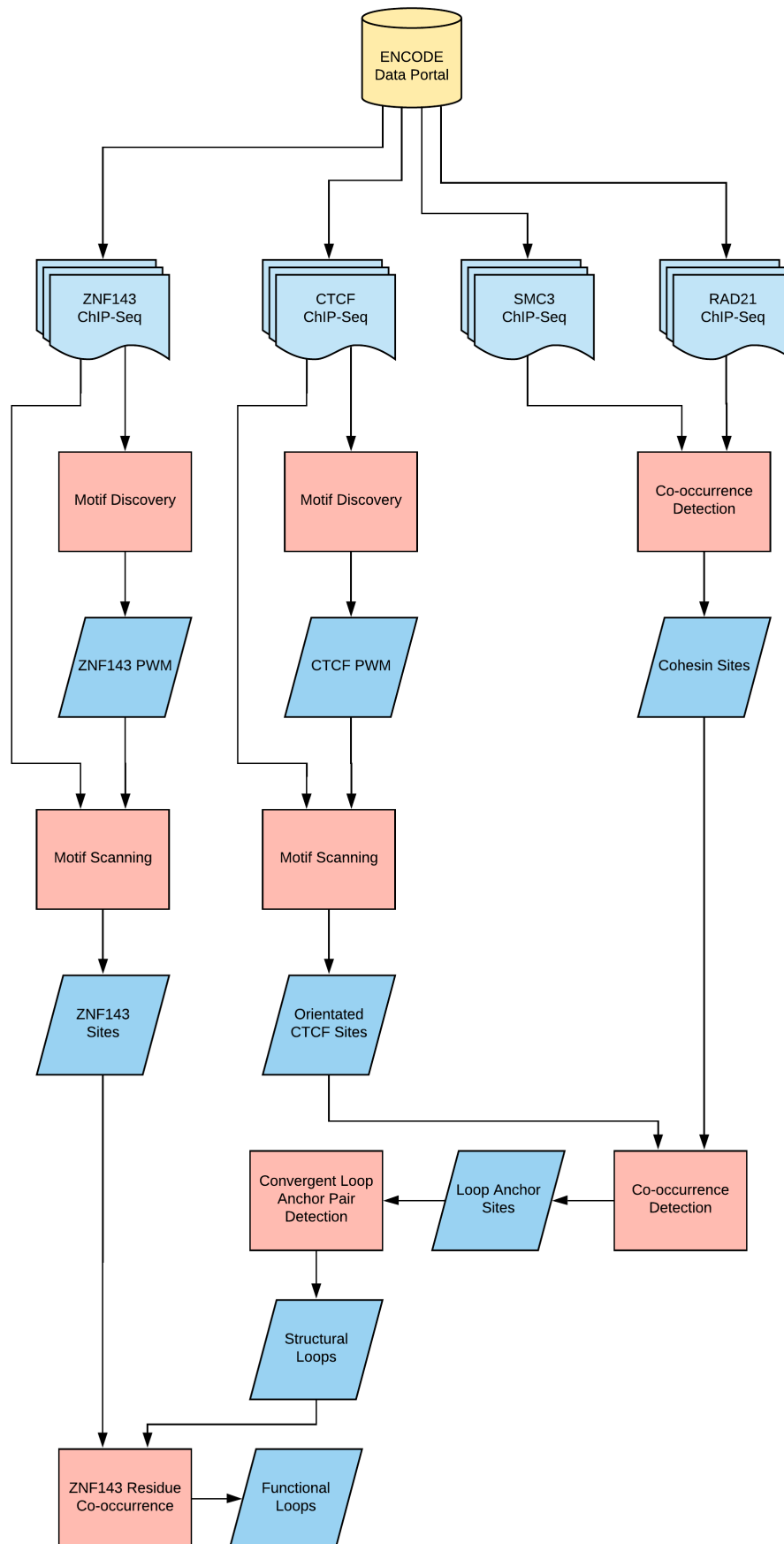
The first portion of the software suite aims to predict structural CCCTC-binding factor (CTCF)/Cohesin chromatin loops and functional Zinc-finger protein 143 (ZNF143) loops within a region of interest and display the results linearly in the UCSC browser as a custom track; this is achieved through integration of ChIP-Seq data from the ENCODE project [234] [235] combined with transcription factor binding motif discovery which is then used to predict chromatin loops in a region of interest. Analysing potential chromatin loops in a region of interest can aid in understanding the regulatory architecture present in a region and thus how non-coding mutations may affect gene transcription.

The second portion of the suite enables visualisation of chromatin loops as a force dir-

ected graph. The program output can be directly loaded by Cytoscape [236], a user-friendly force directed graph visualisation program. Loops are visualised as connections between transcription factor binding sites which are shown as nodes with connecting edges showing the relationships between different regulatory elements. The combination of LOOPER and Cytoscape enables end users to understand better the potential loop configurations that exist in their region of interest.

### **3.2.2 Loop Prediction Algorithm**

We developed a CTCF/Cohesin loop prediction software algorithm that outputs predicted loops to custom tracks which can be visualised in the UCSC genome browser. As described in the introduction (1.2.11.1, p.55), chromatin loops are anchored via DNA binding Transcription factors (TFs). The first step in loop prediction, therefore, is identifying high probability binding sites for TFs involved in 3D chromatin architecture. To achieve this, we take existing ChIP-Seq data from the ENCODE project (2.4.7.1, p.123) for a range of cell lines and use the experimental binding sequence data to generate a Position weight matrices (PWM) (2.4.8.1, p.124) for the TF, which can then be used to scan the region of interest for high probability binding sites [206]. We do not use the experimental binding data directly as this is only available for a small subset of cell lines; instead, we use the data to create a statistical prediction model for binding which can then be applied to any cell line. Once the TF sites are predicted, further analysis regarding site orientation and co-occurrence of signals is performed before final loop prediction (Figure 3.13, p.184).



**Figure 3.13:** LOOPER loop prediction program workflow.

### 3.2.2.1 ChIP-Seq Data

Given the abundance of publicly available ChIP-Seq data in the ENCODE project, we opted not to perform *de novo* experiments. Instead, we obtained ChIP-Seq summary data from the UCSC browser ENCODE Transcription factor binding site (TFBS) track which contains ChIP-Seq data for a range of TFs across different cell lines (2.4.7.1, p.123).

To obtain ChIP-Seq data for input to LOOPER, we filtered the flat-file TFBS table for our target TF to give a list of chromosome number, start position, end position and strength of the ChIP-Seq block signature. The blocks represent a genomic region that the TF was bound to with a confidence value proportional to a strength value of 0-1000; the regions are typically 200-400bp.

### 3.2.2.2 Motif Discovery

To build a statistical model that we can use to predict TFBS for a given sequence, we obtained ChIP-Seq data for each target TF in the loop prediction algorithm. We then used the hg19 reference genome to pull all of the sequences predicted to be bound by a target TF. We achieved this by constructing a small script that loaded each chromosome DNA sequence into memory before sequentially traversing a list of ChIP-Seq sites for that chromosome ordered by start position. The program then extracts the correct DNA sequence for each site and stores it in FASTA format. Once the set of sequences were obtained we passed them through the MEME-Chip [211] motif discovery pipeline to obtain a PWM for the TF. If more than one motif was shown in results, we selected the highest confidence (lowest p-value) logo (2.4.8.1, p.124).

We here acknowledge that there is in fact now a well-known tool for obtaining pre-calculated motifs for specific cell lines obtained using ENCODE ChIP-Seq data call Factorbook [212].

### 3.2.2.3 Motif Scanning and Site Orientation

Once a PWM for the target TF has been obtained we then use another MEME suite tool known as Motif alignment search tool (MAST) to scan the sequence from our region of interest for probable TFBS [213]. MAST returns a list of sites with a p-value and importantly, site binding orientation; a TFBS can be on either strand and thus effects the direction with which a protein binds to DNA. Orientation information is essential for predicting CTCF

structural loops as they rely on convergent CTCF binding sites to form. After this stage of processing we have a list of TFBS in our region of interest for our target TFs with associated p-values and site orientation.

#### **3.2.2.4 Structural Loop Prediction**

CTCF/Cohesin loop formation is hypothesised to follow the loop extrusion theory (1.2.11.1, p.55); this involves Cohesin forming a handcuff configuration around DNA in two adjacent sections of DNA before the DNA extrudes through the Cohesin until it meets a correctly orientated CTCF motif with a bound protein. We define the CTCF site pairs as loop anchors and a single CTCF site as 'half-anchor'.

The first stage of structural loop prediction is to identify the half-anchor sites in the region of interest. CTCF is implicated in a wide range of regulatory activities as well as chromatin architecture; therefore, is it not sufficient to use all CTCF binding sites within the ROI as half-anchors. In the loop-extrusion model, Cohesin does not bind directly to DNA but binds indirectly through CTCF. ChIP-Seq is configured to identify DNA binding proteins; however, when the target TF is pulled down using immunoprecipitation, it does not distinguish if the protein is bound directly or indirectly to DNA, meaning that Cohesin can still be targeted and studied via ChIP-Seq.

Cohesin is a large protein that consists of several sub-units, instead of targeting cohesin as a whole with antibodies there are two antibodies which target different sub-units: SMC3 and RAD21. Searching for co-occurrence (overlapping ChIP-Seq signals) of both SMC3 and RAD21 provides strong evidence for cohesin indirect DNA binding. Given that a loop anchor must contain both CTCF and cohesin, we can filter for potential CTCF half-anchors by searching for SMC3 and RAD21 signals which overlap with CTCF signals and a predicted PWM-based CTCF binding site (the PWM is used for further CTCF site confirmation and for site orientation).

Once loop half-anchors have been identified, we then use this data to identify potential loop anchor pairs in the ROI. The loop extrusion theory requires that CTCF sites be convergent in their orientation and loops are predicted to be less than 500Kb in size; we search an ROI using these parameters to identify predicted loop anchor pairs. Given that loops can exist within other loops and multiple loops can anchor to the same CTCF sites, the total space of predicted loops in the region is the exhaustive combination of all convergent half-anchors within 500Kb of each other.

ChIP-Seq experiments are performed not only on specific TFs but also on a selected

cell line; hence, we sought to identify cell lines within ENCODE for ChIP-Seq that fitted with the observed symptoms for HIPKD, notably in the kidney, pancreas and liver. Unfortunately, no kidney or pancreatic cell lines were included in the March 2016 ENCODE release for CTCF or Cohesin ChIP-Seq data; furthermore, HEPG2, an immortalised human liver cancer cell (the closest potential candidate cell model for HIPKD) although available, only had CTCF data available and thus was unsuitable to be used for structural loop prediction.

Given the lack of cell-specific data, we opted to generalise our loop prediction model for structural loops by selecting ubiquitous CTCF and Cohesin ChIP-Seq sites based on all available cell lines in ENCODE.

### **3.2.2.5 Functional Loop Prediction**

Functional chromatin loops are more difficult to predict as they are less well described in the literature. The current best hypothesis for functional looping involving ZNF143 (our primary functional looping TF) is that it binds to a promoter on one side and then to a mediator complex formed on an enhancer at loop anchors to facilitate gene transcription (1.2.11.1, p.55). Given this hypothesis and using the TF residue principle of indirect DNA protein binding in ChIP-Seq experiments we would expect there to be cohesin residue at ZNF143 sites in promoters and ZNF143 residue at loop anchors for ZNF143 proteins actively involved in functional loop recruitment.

To predict functional loops, we first search gene promoters defined by the genome segmentation track in UCSC browser in our ROI (2.4.7.2, p.124); we extracted predicted promoters by pulling all genome segments with the promoter tag listed in the region. We then use MAST to identify high probability ZNF143 sites within promoters; we also searched for SMC3 and RAD21 signals to indicate that the ZNF143 site was in contact with a loop anchor at the time of the experiment. Next, the predicted loop anchors in the structural loop prediction stage were searched for ZNF143 ChIP-Seq signals to identify loop anchors that the ZNF143 promotor site may have been in contact with. Using this information we then generated an exhaustive model of possible functional loop pair combinations between promoters and loop anchors.

### **3.2.2.6 Predicted Loop Visualisation**

The loop predictions outputted by LOOPER were defined by chromosome number, a start position and an end position that represented the effective loop domain. We designed LOOPER to output the predicted loops in Browser extensible data (BED) format and coded



so the loops would appear as black blocks on the screen to show the loop domain in UCSC browser [215]. BED format files can be uploaded to UCSC browser as custom tracks and shown on-screen alongside other tracks to aid further analysis of chromatin looping in an ROI.

### **3.2.3 Loop Visualisation Algorithm**

The primary function of regulatory chromatin looping in the genome is to bring distal genomic elements such as enhancers and promoters into proximity so that they can act on each other to regulate gene transcription. If the genomic elements are represented as nodes on a graph, we can say that two nodes are connected if they have some relationship in terms of physical proximity with each other, whether this is positional proximity on the genome or proximity as a result of chromatin interactions.

Force directed graphs are a simple, intuitive way to visualise connected data; we were unable to find any programs which could generate force directed graphs from chromatin confirmation experiments therefore, we added additional scripts to LOOPER that generate data for Cytoscape [236] to render force directed graphs.

#### **3.2.3.1 Source Data**

ChIA-PET is ideal for studying looping as different loop types can be targeted by selecting the TFs involved in that particular structure. The ENCODE project has a repository of ChIA-PET data which was generated from one protocol, making individual experiments comparable with each other. We selected ChIA-PET as our data source for visualising interactions in a force directed graph with LOOPER (2.4.9.1, p.129).

#### **3.2.3.2 Pre-processing**

We downloaded FASTA format files from the ENCODE data portal for each target TF experiment, we then fed the files into the Mango processing suite to obtain a list of chromatin interactions (2.4.9.2, p.130). Pre-processing must be performed for each cell-line/TF combination used for analysis. The results from Mango are then collected into one folder and fed into the graph generation algorithm. LOOPER effectively generates a graph for each ChIA-PET experiment but then merges the graphs to form a coherent picture of interaction in a region of interest.

### 3.2.3.3 Graph Generation Algorithm

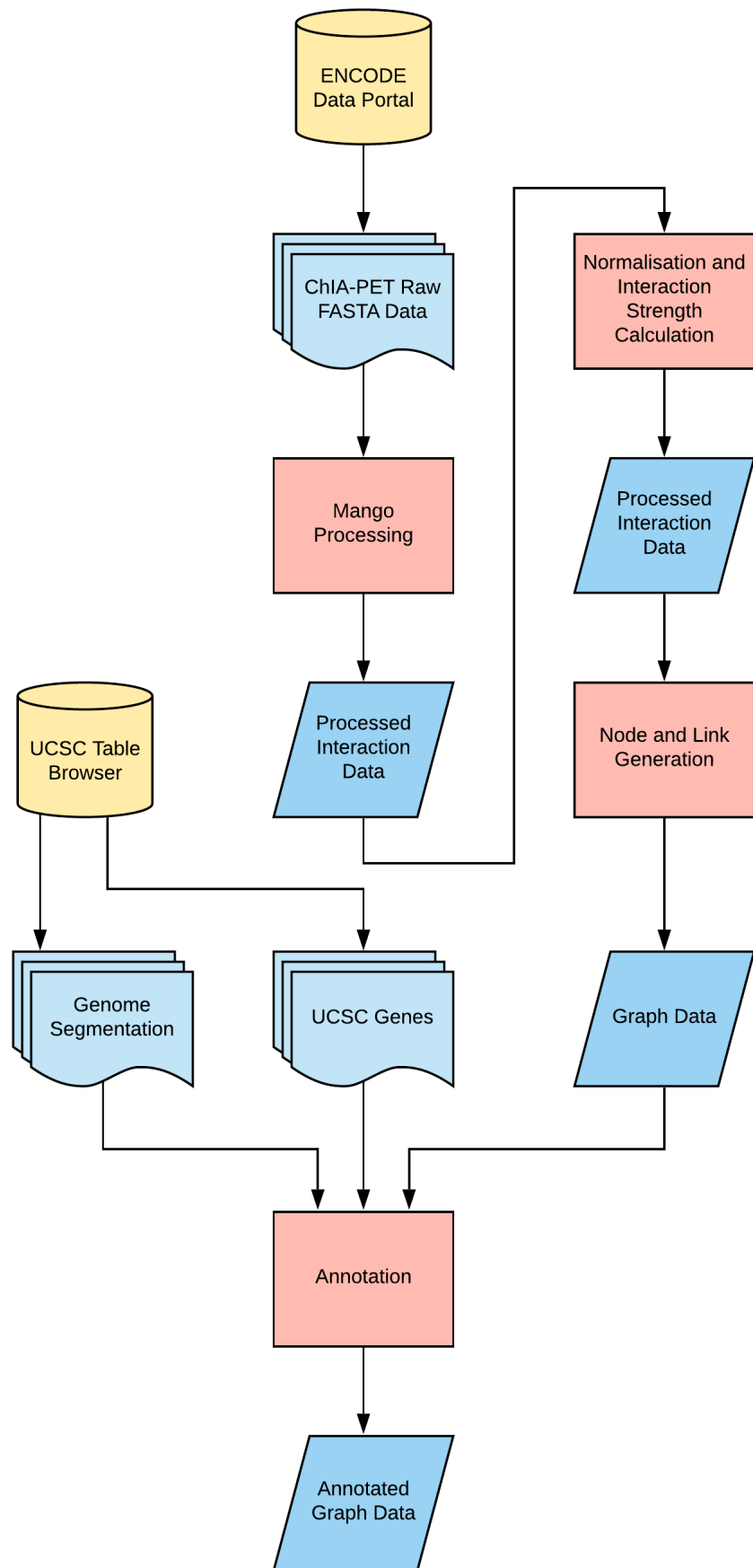
Our graph generation algorithm was written as a C# script and currently must be run from within Visual Studio or another such development environment capable of compiling and running C# code; file paths and other parameters must be changed in code and recompiled before running the code. LOOPER is targeted at existing developers as experimental software only; given the rate at which the chromatin loop research field is currently moving, we concluded that packaging LOOPER up in the same way as REMEDY would be inappropriate at the time of writing.

We show the full LOOPER visualisation pipeline in Figure 3.14, p.190. The first stage of processing is the loading of the interaction data described in the previous section into C# data structures. The target TF and cell-line must be supplied when loading the files so that multiple layers of interaction data can be loaded from different experiments; for example for structural loop analysis, we would ideally load ChIA-PET data for CTCF, RAD21 and SMC3. LOOPER provides functionality to layer all of this data onto one graph.

Cytoscape represents interaction strength in a graph by the thickness of the vertex joining two nodes. After interaction loading, the interaction strength from the Mango output is normalised to the inter-quintile range, as the strength of the interaction p-value in the Mango output falls in a large range that can contain outliers. We first normalised all the data between 0 and 1; however, we found the outliers skewed the visualisation of the strength. Instead, we normalise using the bottom 20% and 80% range of the strength data as the minimum and maximum; this removes outliers from the normalisation calculation while still ensuring that the majority of the variance contained within the data is still captured.

After normalisation, we proceed to add artificial vertices that represent physical proximity. During early testing, we found that although viewing interactions was interesting, without the context of which elements were close to each other on the genome, the graph was difficult to interpret. To solve this LOOPER provides quick functions to add in additional vertices based on a threshold distance. Multiple thresholds can be supplied to show multiple levels of proximity that is then mapped on top of the interaction data as additional vertices; this ensures that when the force directed graph is calculated, that physically proximal elements are positioned close to each other on the graph rather than being on opposite sides.

During development, we found the graph for a whole chromosome to be too complex, and so included functionality to filter for objects within or connected to an ROI. We started by filtering for nodes inside a region of interest but retaining any distal long-range contacts outside of the ROI but found the resultant graph was still too complicated, especially in



**Figure 3.14:** LOOPER loop visualisation program workflow.

highly-connected regions; we, therefore, included additional filters to show only the long-range contacts that fell within a Topologically-associated domain (TAD). We reasoned that most regulatory interaction occurred intra-TAD and therefore, any inter-TAD communication would either be spurious or subject to broader chromosome-wide regulatory processes that are still not well understood [237]. Also, we placed some additional filtering constraints on the graph, enabling users to filter based on strength and distance of interaction should the graph complexity still be too large on generation.

The final stage of processing before the generation of the node and link files required for Cytoscape is graph annotation. During testing, we found that although the graph was initially useful, there were many visual aspects of our generated graphs that were left unused. For example, nodes were all the same colour, size and shape; we expanded LOOPER by integrating gene and genome segmentation data from UCSC browser in order to place the graph nodes in a better functional physical context (2.4.7.2, p.124). We first downloaded a list of genes from the UCSC gene track and used this as a static reference to annotate the labels of all overlapping nodes. We found that this extra data contextualised the graph to a user by showing locations of known elements such as genes and *cis*-regulatory elements.

Twenty-five states are used to segment the genome in the UCSC genome segmentation track (2.4.7.2, p.124), these states were then grouped and coloured to highlight predicted functional elements including promoters and enhancers; we selected a subset of these states that were relevant for chromatin looping. The genome segments can be narrow when compared to ChIP-Seq sites and are not 100% accurate; therefore we decided to expand the range of effect for any sites of interest such as promoters and enhancers via a user-specified parameter. Any expanded segments overlapping with a node are annotated to that element. Joining functional annotation to nodes on the data graph adds additional context and is especially useful when viewing the graph in terms of regulatory interactions.

After final processing, all of the data for the graph must be generated so that it is readable by Cytoscape. Nodes and links must be placed into separate files with their respective annotations so that they can be loaded correctly; nodes are written to file with Id, Position Id, Segment Type and Label. The position ID is specified so that it can be easily copied into the location finder box in UCSC browser; this allows for easy context switching between LOOPER and other genomic analysis programs or references. Segment type and label represent the annotation performed in the previous step which can be used to colour and shape the generated graph in Cytoscape. The segment annotation corresponds to a priority labelling algorithm which, in the case of multiple overlapping segments will annotate the 'most interesting' label as per the order listed in Table 3.10, p.192.

Interactions are written to file with Id-A, Id-B, Interaction Type, Distance and Strength.

Priority Level	Returned Label	Label Description	Comment
1	P	Promoter	Promoter only
2	PF	Promoter flanking	Promoter flanking region only
3	E	Enhancer	Enhancer only
4	WE	Weak Enhancer	Weak enhancer only
5	I	Insulator	Insulator only
6	T	Transcribed region	Transcribed region only
7	E	Enhancer	Enhancer and weak enhancer
8	P	Promoter	Promoter and promoter flanking
9	PE	Promoter/Enhancer	Promoter elements and enhancer elements

**Table 3.10:** List of annotation priorities when assessing genome segments which overlap a node.

The two ID fields represent the IDs of the nodes to which the link is connecting; the interaction type corresponds to the TF which the ChIA-PET experiment targeted which allows for layering multiple experiments in a single graph. The distance column corresponds to the genomic distance between the two nodes and can be used as a parameter when generating a force directed graph. The strength column represents the calculated interaction strength based on several output parameters from the Mango pipeline. Force-directed graphs as a default aim to keep all edge lengths the same; however, by setting a strength for the edge, the graph can be distorted to better represent the interactions in the underlying data. LOOPER provides two methods for achieving this: genomic distance can be used to distort the graph to better represent the physical genomic position of the data, or the interaction strength parameter can be used to distort the graph to represent the statistical strengths of the contacts. We developed an in-house interaction strength formula which is discussed in more detail below.

**3.2.3.3.1 ChIA-Pet Interaction Strength Calculation** There are two primary variables which govern the strength of a ChIA-PET interaction as outputted from Mango: the read depth of the interaction at the two anchors and the number of PETs that support the interaction (2.4.9.2, p.130). The read depth governs the confidence in the quality of the read sequence at either end of an interaction; a low read depth may indicate potential inaccuracies in the mapping of a tag and thus the genomic positioning of an interaction anchor point. We define that the total quality of an interactions read-depth can be measured as a function of both the total read depth for both paired-ends and the ratio of one read-depth to the other; we can define a high-quality interaction in terms of read-depth as a high and balanced versus low and unbalanced. For example, three interactions have total read depths

of 60, 60 and 20; the first interaction has read depths of 30/30 while the second has 5/55 and the third 10/10. Initially we may have designated the interaction with a read depth of 20 as the lowest quality; however, we can see that one of the anchors in the second interaction has a read-depth of just 5 and thus could be considered to be lower-quality. The number of supporting PETs is also of critical importance when assessing the strength of interaction; it essentially defines the level of supporting experimental evidence for a true interaction between the two anchor points. By combining the total read-depth, anchor read-depth ratio and the supporting PETs into a metric for interaction strength, we can use this to distort our force directed graph into a representation of the most important regulatory interactions in a region of interest.

We first define the weights  $w_d$ ,  $w_t$  and  $w_s$  for depth ratio, total depth and PET support respectively. The weights are set to the open unit interval between 0 and 1 (Equation 3.1, p.194) and can be configured in LOOPER to give an importance weighting to the concepts discussed in the previous paragraph. We default the weightings to  $w_d = 0.5$ ,  $w_t = 0.1$  and  $w_s = 1$ .

For a set of  $n$  interactions we denote the  $i$ -th interaction strength as  $x_i$ , the read-depth at anchor A as  $a_i$ , the read-depth at anchor B as  $b_i$  and the supporting PETs as  $s_i$ . We then define our formula for interaction strength as the product of the weighted formulas for depth ratio, total depth and supporting PETs with intermediate values for each criteria defined as  $v_d$ ,  $v_t$  and  $v_s$  respectively (Equation 3.2, p.194). Each input value is normalised before being fed into the weighting calculation; the type of normalisation performed is dependant on the characteristics of the dataset. For example,  $v_{si}$  and  $v_{ti}$  both had a possibility of zero values, requiring corrections to stop divide by zero errors, while  $v_{di}$  also required log scaling to reduce the influence of outliers.

$$\begin{aligned}
w_d &= \{x \in \mathbb{R} \mid 0 \leq x \leq 1\} \\
w_t &= \{x \in \mathbb{R} \mid 0 \leq x \leq 1\} \\
w_s &= \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}
\end{aligned} \tag{3.1}$$

$$\begin{aligned}
v_{di} &= 1 - \frac{\min\{a_i, b_i\}}{\max\{a_i, b_i\}} \\
v_{ti} &= 1 - \frac{1}{1 + \log_{10}(a_i + b_i)} \\
v_{si} &= 1 - \frac{1}{1 + s_i} \\
x_i &= (1 - w_d v_d)(1 - w_t v_t)(1 - w_s v_s)
\end{aligned} \tag{3.2}$$

### 3.2.4 Case Study Introduction

We developed LOOPER to explore chromatin structure as a result of a discovered point mutation in a newly characterised disease, HIPKD [2]. LOOPER was developed and extended in tandem with the HIPKD project and applied to the single ROI obtained from positional cloning and sequencing.

In this section we present our analysis of the HIPKD ROI using LOOPER. We first discuss the initial research which led to the discovery of the putative causal point mutation, we then show the further analysis which led to the implication of chromatin architecture playing a potential role in the pathogenesis of HIPKD. Finally, we apply both the loop prediction and force directed graph loop analysis modules of LOOPER to explore the regulatory landscape of the local genomic region.

The introduction and initial results presented below were generated by our team and was not the work of the writer. The results presented in the bioinformatic analysis section were parts of the project on which the writer directly worked.

### 3.2.5 Initial Research

#### 3.2.5.1 Clinical Presentation

Autosomal recessive polycystic kidney disease (ARPKD) is a rare disorder with an estimated incidence of 1:20,000, it is characterised by a combination of polycystic kidneys and hepatic fibrosis. Causative mutations in *PKHD1* are found in approximately 85% of cases; other mutations in ciliary genes are also known to phenocopy the disease. Our team investigated patients with ARPKD-like clinical presentation but had no previously described mutations for ARPKD; they additionally presented with a concurrent diagnosis of hyperinsulinemic hypoglycemia (HI). HI in itself is also a rare disorder, with an estimated incidence of 1:50,000 and most commonly associated with mutations in the *ABCC8* and *KCNJ11*, genes involved in the regulation of insulin release from pancreatic  $\beta$  cells; however, no previously described mutations were found in patients within these genes [2].

The co-occurrence of these two rare disorders in multiple patients and without identified mutations in known disease genes strongly suggested the existence of a yet undescribed Mendelian disorder with recessive inheritance. Our team noted the co-occurrence of the clinical diagnoses of HI and polycystic kidney disease (HIPKD) in 17 patients from 11 families of European descent, including a consanguineous family with four affected individuals. Multiple siblings were affected in three further pedigrees, consistent with autosomal recessive inheritance [2].

Eight patients from seven families had been referred for genetic testing for HI through an international study of 1250 families. All patients had been tested for *ABCC8* and *KCNJ11* mutations, which are the most common cause of HI, yet none were found. Mutations in *PKHD1*, the causative gene for ARPKD, had been excluded by linkage and/or sequencing analysis in 14 patients [2].

#### 3.2.5.2 Genetic Analysis

Autozygosity analysis of genotyping data in a consanguineous family showed a homozygous 2.5 Mb ROI on chromosome 16p.13.2. Whole genome multi-point parametric linkage analysis in the consanguineous and four other informative families confirmed and refined the ROI to a single significant locus of 2.3 Mb on chromosome 16 in region 16p.13.2 with a combined LOD score of 6.5 [2].

The refined 2.5Mb single genomic ROI identified via linkage analysis while a significant



LOD score was a comparatively large region to search for causal variants at 2.5Mb with 14 genes lying within this region and significant numbers of known variants. Next-generation sequencing on patient samples identified a non-coding variant in the promoter region for *Phosphomannomutase 2 (PMM2)*, c.-167G>T; this was a previously undocumented variant in common databases. Subsequent Sanger sequencing confirmed the mutation in all patients; it was present in a homozygous state in four patients in the consanguineous family, and in a compound heterozygous state with previously described mutations in *PMM2* in all others [2].

### 3.2.5.3 Functional Analysis

The identified Guanine to Thymine mutation lies in a predicted promoter region of *PMM2* [238]. The effect of the promoter variant on the transcription of *PMM2* was assessed *in vitro*, using human kidney and pancreatic  $\beta$  cells. In both models, a significant reduction in gene transcription was observed in mutated cells versus wild-type. Kidney cells from a patient that was compound heterozygous for the promoter mutation and a missense mutation in *PMM2* were also assessed for transcription levels; digital quantitative PCR showed reduced expression for the allele containing the promoter variant when compared with the other allele.

After linking the promoter mutation to *PMM2* activity, our team then went on to analyse the mechanism of action by which the promoter mutation controls expression. Given the target was a promoter, the most likely path of action was through transcription factor binding; we found our mutation to lie within a recognised binding site for the transcription factor ZNF143. Our team assessed binding of ZNF143 to the promoter region experimentally using an Electrophoretic mobility shift assay (EMSA), which revealed significantly reduced binding in the mutation versus the wild-type. The consequences of decreased *PMM2* activity on insulin production were also investigated, and a direct link was established between glycosylation and insulin secretion.

In summary, functional analysis revealed a chain of causality beginning with a promoter mutation in *PMM2*, leading to reduced binding of ZNF143 to the promoter region which in turn leads to reduced transcription of the *PMM2* gene; subsequent proteins produced from *PMM2* transcripts were linked to glycosylation of proteins involved with insulin secretion (one of the key parts of the HIPKD phenotype). The mechanism by which ZNF143 affects *PMM2* transcription remained hidden, until a paper showing ZNF143 loop formation presented a possible hypothesis whereby reduced ZNF143 binding may affect local chromatin-loop structure [239]; we present our bioinformatic analysis that resulted from this new hypothesis in the next section.

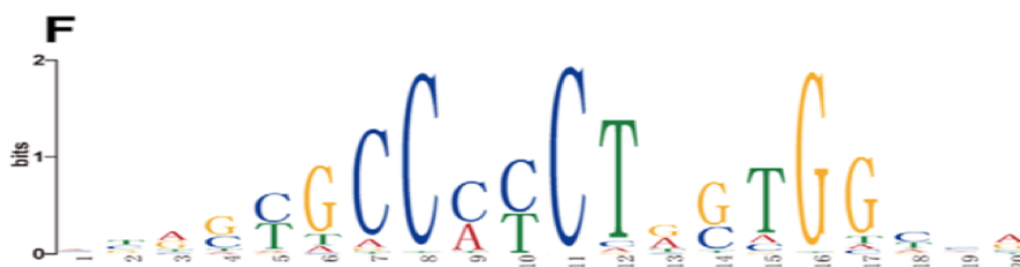
### 3.2.6 Bioinformatic Analysis

We present here the bioinformatic analysis performed in the genomic region around *PMM2* including application of the LOOPER software suite to characterise the chromatin looping architecture.

#### 3.2.6.1 Binding Motif Discovery

The first stage of our loop prediction algorithm requires the generation of PWMs for the target TFs for structural and functional loop prediction: CTCF, RAD21, SMC3 and ZNF143. We used ChIP-Seq data as described in the sections above to provide the binding sequences required for motif discovery.

**3.2.6.1.1 CTCF** CTCF was used as a positive control for our motif discovery pipeline as the PWM was already well described in the literature [240] [241]. We obtained the same base ChIP-Seq data and ran this through our custom pipeline, the resulting motif is equivalent to the published 2011 motif [240] and was also in consensus with other literature, indicating the pipeline was working correctly (Figure 3.15, p.197).



**Figure 3.15:** Consensus binding motif for CTCF from [240]. The sequence is 20 base pairs long and contains a GCCCCCT rich central motif combined with a secondary GTGT motif with a one base pair gap. The overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position.

The UCSC genome browser at the time of analysis had a newer 2012 ENCODE data release with more ChIP-Seq data from additional cell lines for CTCF. We ran the new data through the same pipeline to see if an update to the CTCF motif was required. The resulting motifs were filtered for motif sub-sets, and then the highest p-value motif was selected as the comparison (Figure 3.16, p.198). Our results showed no difference between the 2011 and 2012 motifs; this shows that CTCF is highly conserved in the human genome across cell types (Figure 3.17, p.198). The reverse complement of the primary motif was also

calculated so that CTCF site orientation could be deduced during the motif scanning phase (Figure 3.18, p.199).

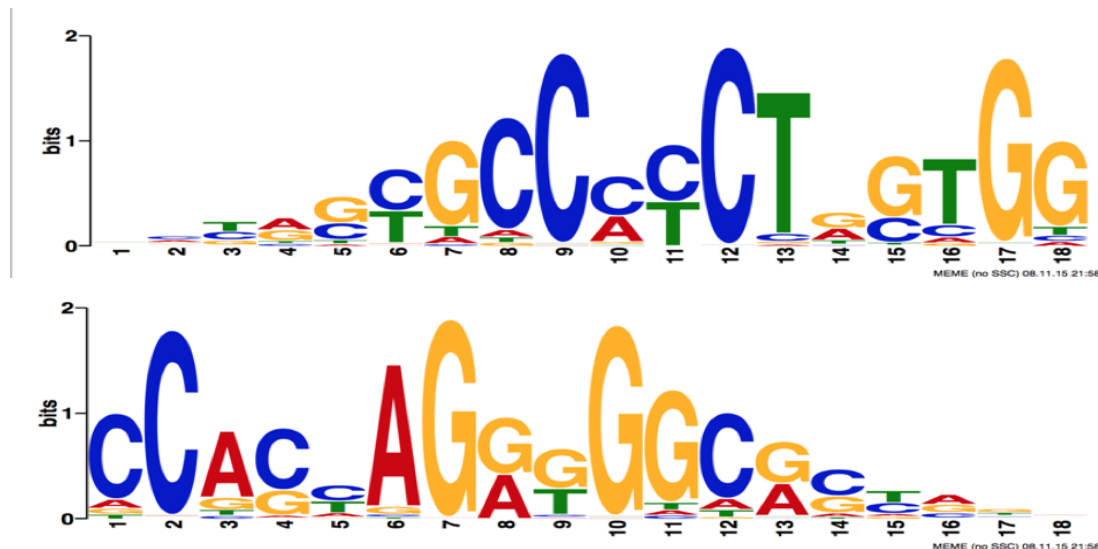


**Figure 3.16:** MEME-ChIP web result output for CTCF run on 2012 ENCODE data release from the UCSC genome browser. The output shows three motifs with their calculated web-logos and p-values (third column). The bottom motif is a subset of the larger motifs and so was discounted. We selected the central motif as our final result as it had the highest p-value. We compared the 2012 motif with the previously calculated consensus motif from 2011 and found there to be no difference.



**Figure 3.17:** CTCF binding motif calculated from 2011 ENCODE release data used in [240]. ChIP-Seq data was run through a custom pipeline that selected and transformed data for input into MEME-ChIP. The sequence above was the consensus motif from the MEME-ChIP output. Overall letter stack height indicates the sequence conservation at that position, while the height of the symbol within the stack shows the relative frequency of each base at that position. The motif contains the same two-part binding motif as the consensus motif and has no other extra motifs; therefore, they can be considered to be equivalent. When using the consensus motif as a positive control, this indicates that the custom pipeline was functioning correctly.

To rule out any cell-specific changes to the CTCF motif, we filtered our dataset for cell line which is present in organs affected by HIPKD: HEPG2 (a liver cell-line) and PANC1 (a pancreatic cell-line); We found the binding motif to be identical to the ubiquitous motif in both cases.



**Figure 3.18: Top:** Calculated CTCF binding motif based on 2012 ENCODE data selected for use in our loop prediction pipeline. The sequence is 18 base pairs long and contains a two-part binding motif. **Bottom:** Reverse complement of the original sequence which was used to detect binding orientation for CTCF.

Overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position.

**3.2.6.1.2 Cohesin** Cohesin can be pulled down in antibody-based experiments such as ChIP-Seq using antibodies that target two sub-units of the protein complex: SMC3 and RAD21. We filtered our UCSC ENCODE TFBS summary dataset for SMC3 and RAD21 and ran the subsequent datasets through our motif discovery pipeline. Interestingly the results obtained matched the CTCF motif exactly, suggesting that all three transcription factors have the same binding motif. The loop extrusion theory used in LOOPER states that cohesion factors bind indirectly to chromatin through CTCF forming a complex; our results are in line with this theory.

As SMC3 and RAD21 do not bind directly to DNA in this context, we could not calculate a binding motif for either meaning that we only had a single motif for identifying all three binding site types. We mitigated this by altering the loop anchor identification strategy slightly; instead of looking for the co-occurrence of the motif and ChIP-Seq region separately, then finding overlapping sites of all three proteins, we instead searched for CTCF binding sites with both motif and ChIP-Seq and then combined this with co-occurrence of both SMC3 and RAD21 ChIP-Seq residue.

**3.2.6.1.3 ZNF143** ZNF143 has well-defined antibodies with experimental data available from the ENCODE portal as described in our methodology. We filtered our UCSC EN-

CODE TFBS summary dataset for ZNF143 and ran the subsequent data through our motif discovery pipeline (Figure 3.19, p.200).



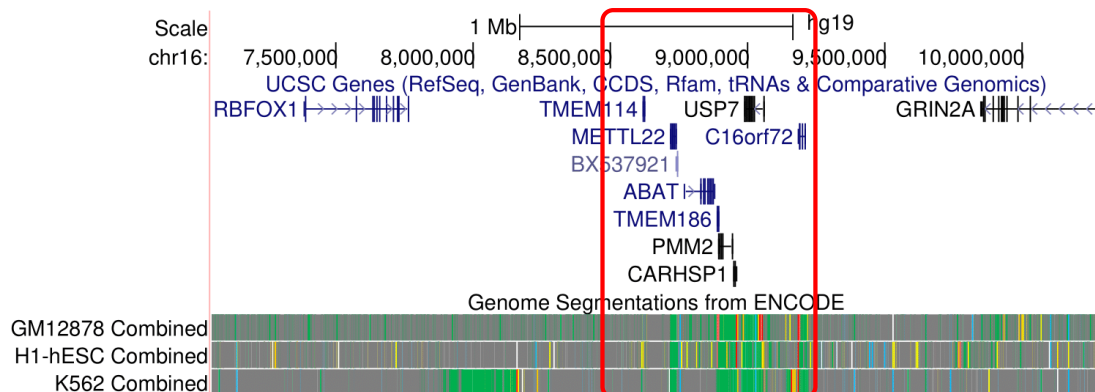
**Figure 3.19:** ZNF143 binding motif calculated from 2012 ENCODE release data. Overall letter stack height indicates the sequence conservation at that position, while the height of symbol within the stack shows the relative frequency of each base at that position. The motif is 22 base pairs long and contains several islands.

### 3.2.6.2 Region of Interest Initial Analysis

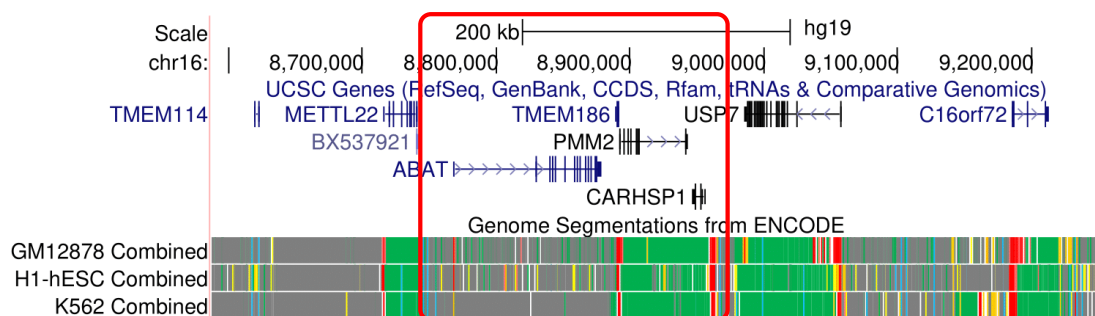
After generation of PWMs for our looping TFs was complete, we first performed an initial annotation of our region of interest around the *PMM2* gene. Using UCSC browser, we added the UCSC genes track that shows the predicted positions, orientations and structures of genes on the genome, and the Genome Segmentation track (2.4.7.2, p.124).

Zooming into the genomic region around *PMM2*, the gene is clustered inside a group of genes that form a ‘gene island’ (Figure 3.20, p.201). The island is surrounded by gene depleted areas of mostly closed chromatin which would suggest that these areas are transcriptionally inactive. Conversely, the gene island is rich in open chromatin as well as promoter regions and enhancers. Zooming into the local genomic area further, we see a four-gene cluster approximately 200Kb in length (Figure 3.21, p.201 and Figure 3.22, p.201). This area contains *PMM2* and *Transmembrane Protein 186 (TMEM186)* in the centre both facing away from a non-coding region where the mutation is located. Flanking this area are two other genes: *4-Aminobutyrate Aminotransferase (ABAT)* and *Calcium Regulated Heat Stable Protein 1 (CARHSP1)*. Importantly, this region is flanked by strong blue bands suggesting that the area is transcriptionally insulated and therefore could be forming a TAD with loops inside. The region where the mutation was positioned was annotated as a promoter region, given that the promotor region stretched unbounded between the two adjacent genes and that their transcription direction faced away from the promoter, we classified this as a bidirectional promoter.

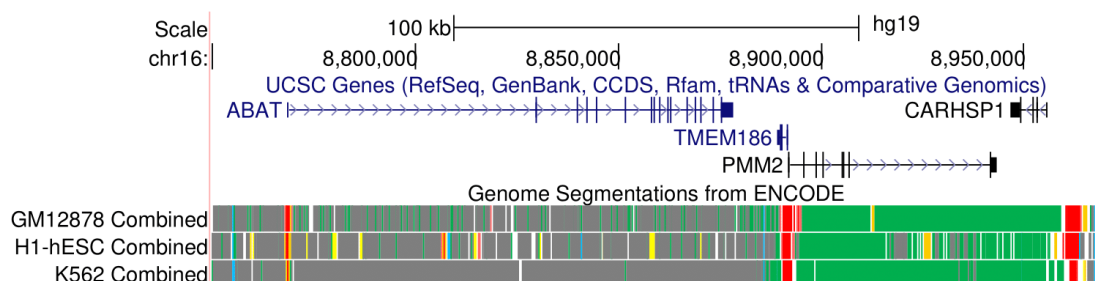
We present a summary of each gene in the cluster below, we obtained our information from three established sources of gene and protein information: RefSeq [242], GTEx [243] and UniProt [244].



**Figure 3.20:** A 3.2Mb region on chromosome 16 displayed in the UCSC genome browser, with the targeted region of interest centred inside the red box. The genomic position is shown at the top, genes are displayed in the centre. The three tracks shown at the bottom are visual representations of the genome segmentation ChromHMM project data; shown are predicted areas of euchromatin (green), heterochromatin (grey), promoter regions (red), insulator regions (blue) and enhancer/regulatory regions (yellow). The three lines represent data from three different cell lines for which this track has been calculated for (GM12878, H1-hESC and K562) [205]. The gene rich area in the centre is clearly flanked by regions of gene depleted, closed chromatin. The red box represents the zoomed in area shown in (Figure 3.21, p.201)



**Figure 3.21:** A 600Kb region centred on the HIPKD ROI. *PMM2* is shown in the centre with *TMEM186* (the other half of the bidirectional promoter for *PMM2*) shown to the left. There are finer grained areas of open and closed chromatin located within some of the genes. The red box represents the zoomed in area shown in (Figure 3.22, p.201).



**Figure 3.22:** A 200Kb genomic region showing a four-gene local cluster with *PMM2* and *TMEM186* flanking a bidirectional promoter whilst themselves being flanked by *ABAT* and *CARHSP1*. Promoter regions (red bands) are clearly shown for all genes with some insulator regions (blue) flanking the whole region. Some enhancer activity (yellow) can also be seen towards the periphery of the region

**3.2.6.2.1 PMM2** *PMM2* codes for an enzyme called phosphomannomutase 2. This enzyme is involved in glycosylation, which attaches groups of sugar molecules (oligosaccharides) to proteins. The *PMM2* enzyme is involved in one of the early steps of the glycosylation process.

**3.2.6.2.2 TMEM186** *TMEM186* is a potential transmembrane protein expressed across the body, but most significantly in the brain, liver and pancreas. Little else is known about this protein.

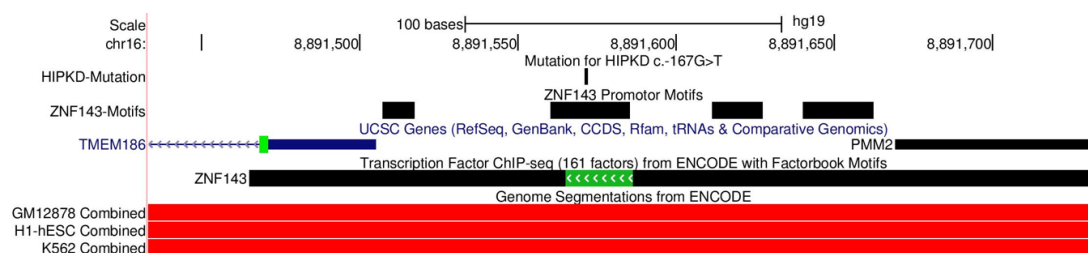
**3.2.6.2.3 CARHSP1** *CARHSP1* is a calcium-regulated, heat stable protein that binds mRNA and regulates its stability in vitro. It is expressed primarily in the brain, testis and some muscular tissue. The promotor for *CARHSP1* is located close to regional CTCF sites and is near to several sites enriched in enhancers.

**3.2.6.2.4 ABAT** *ABAT* (4-aminobutyrate aminotransferase) is a nuclear gene encoding a mitochondrial protein. It is responsible for the catabolism of gamma-aminobutyric acid (GABA), an important inhibitory neurotransmitter in the central nervous system. It is expressed in the liver, pancreas, brain, heart and placenta. Mutations in this gene are linked to neurological defects such as psychomotor retardation, hypotonia and hyperreflexia.

### **3.2.6.3 ZNF143 Binding Analysis**

During the functional analysis of our promoter mutation we found that it lied within a region significantly enriched for ZNF143 in ChIP-Seq data. ChIP-Seq binding regions are approximately 400bp whereas a typical protein binding site is less than 20bp.

To narrow down the actual predicted binding site(s) for ZNF143 and following our LOOPER methodology, we obtained the sequence from UCSC browser that overlapped the ZNF143 ChIP-Seq signal and passed the sequence and our PWM for ZNF143 into MAST from the MEME-suite to scan the binding region for statistically probable binding sites. Several sites were predicted within the ChIP-Seq region, indicating that multiple ZNF143 binding sites exist in the promoter (Figure 3.23, p.203). The mutation was located inside the centre of the lowest p-value predicted ZNF143 binding site, suggesting that a mutation at this location would affect binding.



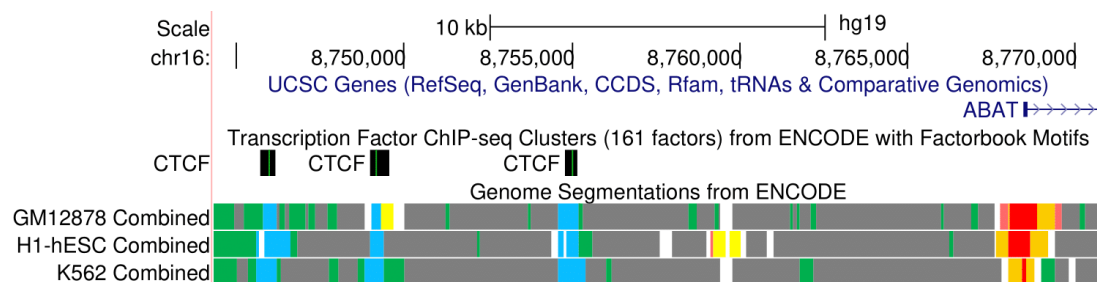
**Figure 3.23:** UCSC genome browser view of the bidirectional promoter containing the HIPKD mutation. The top row shows the location of the one base mutation while the second row shows the predicted ZNF143 sites from MAST. The third row shows the transcription start regions of both *PMM2* (left) and *TMEM186* (right). The fourth row shows binding site prediction for ZNF143; the black region representing strong ZNF143 ChIP-Seq binding while the green area shows the predicted strongest binding site.

### 3.2.6.4 Binding Site Prediction and Analysis

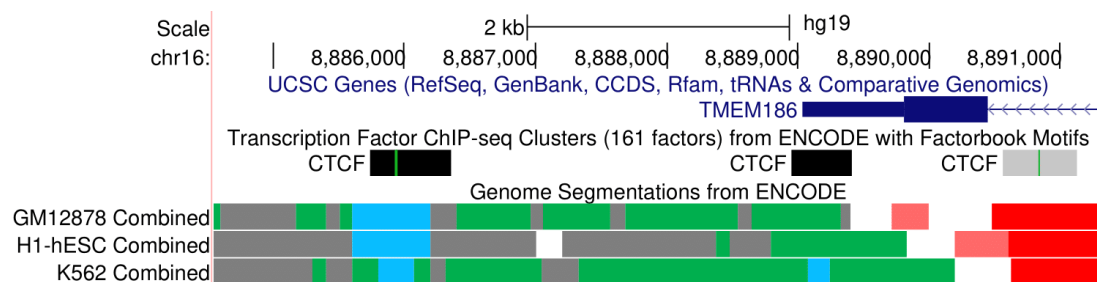
After initial analysis, we proceeded to use our generated PWMs to scan for CTCF and ZNF143 motifs.

**3.2.6.4.1 CTCF** we scanned for well conserved, ubiquitous CTCF sites. Outside the *ABAT* promoter, we identified a cluster of three strong CTCFs with predicted motifs and with chromatin states predicted as insulators by ChromHMM (Figure 3.24, p.204). We found the left-most site to be orientated to the 3' strand, the centre to the 5' strand and the right-most to the 3' strand. We identified one CTCF site with a 5' orientation to the left of *TMEM186* (Figure 3.25, p.204). To the right of our region of interest we identified a cluster of four sites to the right of the *CARHSP1* promoter (Figure 3.26, p.204). The left most site was of medium strength, had a 3' orientated predicted binding site and was classified as a predicted enhancer by ChromHMM. The second site was also 3' orientated and was in a region classified as promoter/enhancer/insulator. We discounted the third site as it did not have a predicted CTCF binding site; the fourth site was 5' orientated and predicted to be an insulator by ChromHMM.

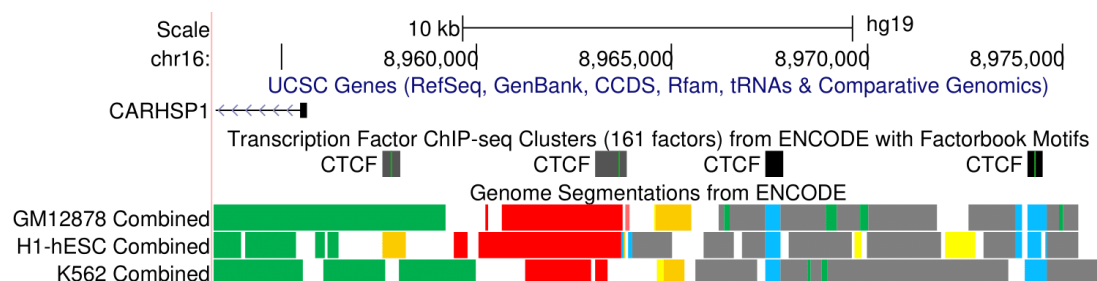




**Figure 3.24:** UCSC genome browser view of three CTCF sites located in an inter-geneic region to the left of the *ABAT* promoter. All three sites have a strong ChIP-Seq signature (black boxes) with predicted binding sites that lie within the ChIP-Seq block. The ChromHMM track shows the TFBS as predicted insulators. The sites are orientated from left to right as 3', 5' and 3' respectively.



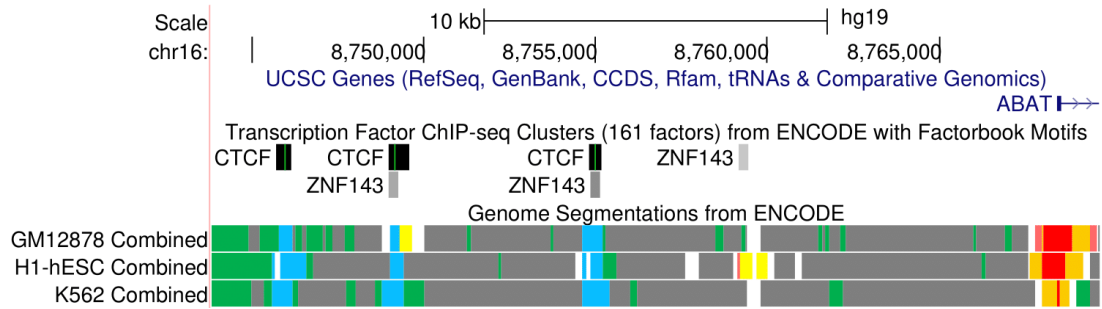
**Figure 3.25:** UCSC genome browser view of three CTCF sites located near the *TMEM186* gene. The left most CTCF site was selected as a target loop anchor while the other two were discarded. The centre motif was only partly well-conserved and had no predicted motif while the right most was cell group specific and relatively weak.



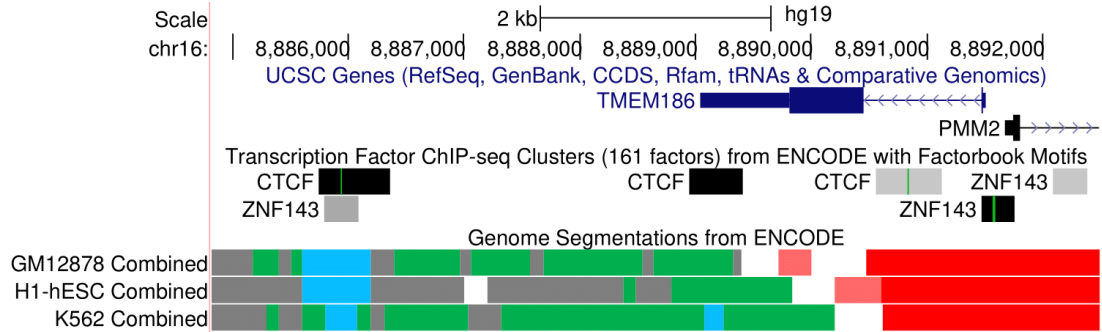
**Figure 3.26:** UCSC genome browser view of four CTCF sites located to the right of the *CARHSP1* promoter region. The left two sites both orientated to the 3' strand, the third site has no predicted binding motif and the right most site is 5' orientated.

**3.2.6.4.2 ZNF143** we scanned for ZNF143 sites within the HIPKD ROI. Outside of the *ABAT* promoter region, we identified several ZNF143 'residue' marks comprising of a weak ZNF143 signal with no binding motif (Figure 3.27, p.205). Two of these sites overlapped with previously identified CTCF sites and one was located in a region predicted to be a cell specific enhancer; these sites are characteristic of potential indirect binding of ZNF143. We also identified one weak ZNF143 site overlapping a CTCF insulator to the right of the *CARHSP1* promoter region; this site had an off-centre predicted binding motif (Figure 3.28, p.205).

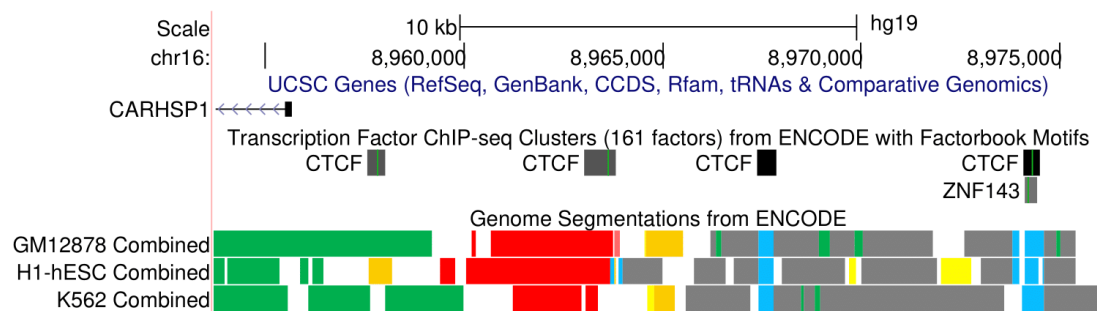
We identified several ZNF143 sites within the region around *PMM2* (Figure 3.29, p.206); two sites were characteristic of weak ZNF143 residue binding, one overlapped with our previously identified CTCF site from this region and one was positioned within the bidirectional promoter for *PMM2* and *TMEM186*. The third ZNF143 site was confirmed as the putative mutation ZNF143 binding site; we noted it was different from the other sites in our region of interest as it had a strong ChIP-Seq signature with a low p-value that is characteristic of direct DNA binding.



**Figure 3.27:** UCSC genome browser view of ZNF143 sites to the left of the *ABAT* promoter region. We additionally show the CTCF sites identified in the previous section. We show several weak ZNF143 sites with no binding motif, two of which overlap with previously identified CTCF sites.

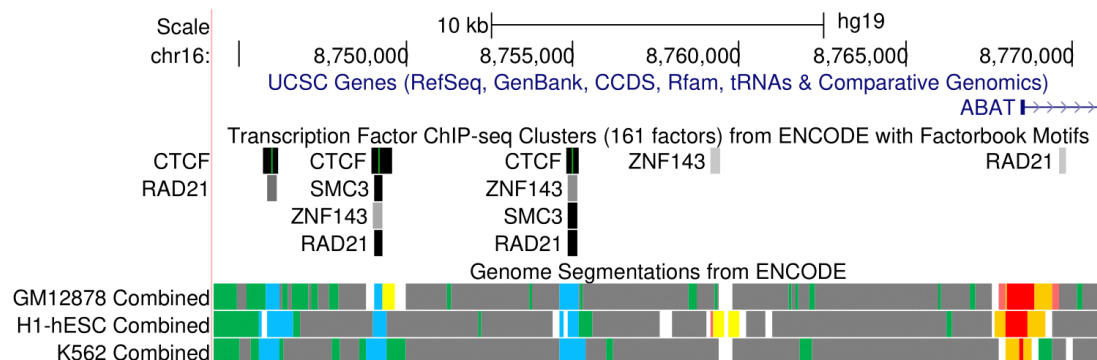


**Figure 3.28:** UCSC genome browser view of ZNF143 sites to the right of *CARHSP1*. We show the CTCF sites identified in the previous section and a single weak ZNF143 site overlapping a CTCF insulator with an off-centre predicted binding motif.

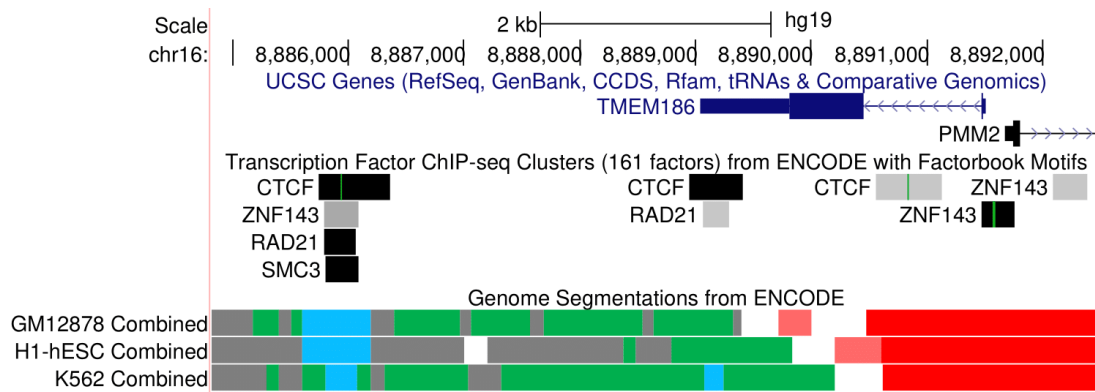


**Figure 3.29:** UCSC genome browser view of ZNF143 sites located near the *TMEM186* gene. We show the CTCF sites identified in the previous section and two 'residue' style sites, one overlapping a CTCF insulator site and one with in the *PMM2* promoter region. The third site shows a strong ChIP-Seq signature with a binding motif; this was confirmed to be the TFBS identified in our functional analysis overlapping with the putative mutation.

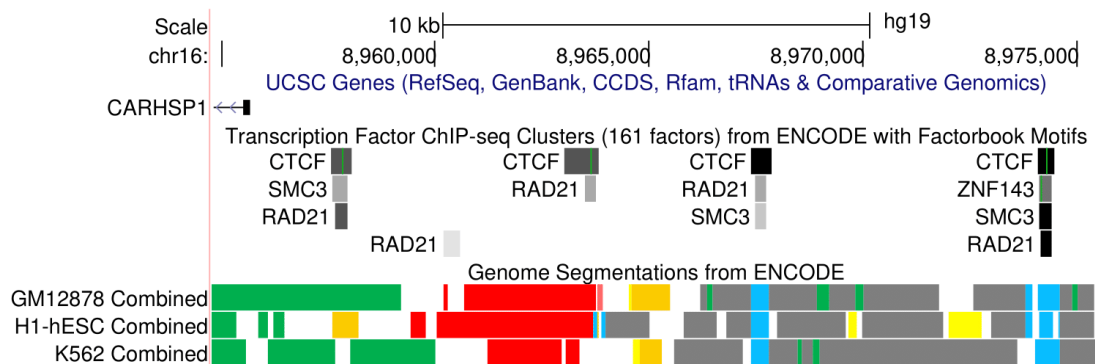
**3.2.6.4.3 Cohesin** For cohesin analysis we searched for ChIP-Seq 'residue' marks only, indicating indirect binding to CTCF at loop anchor sites. Outside of the *ABAT* promoter region, we identified strong ChIP-Seq marks for SMC3 and RAD21 indicating the presence of cohesin on the right most two CTCF sites that we identified earlier (Figure 3.30, p.206). In our centre region to the left of *TMEM186*, we also confirmed a strong cohesin signature over our target CTCF site (Figure 3.31, p.207). To the right of *CARHSP1*, we show that the left most CTCF site identified earlier had a medium cohesin signature while the centre two CTCF sites have a weak cohesin marks. The right-most site which was orientated to the 5' strand, has a strong cohesin signature (Figure 3.32, p.207).



**Figure 3.30:** UCSC genome browser view of cohesin sites to the left of the *ABAT* promoter region. We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that the left most CTCF had only weak RAD21 binding when compared to the other two sites which have strong SMC3 and RAD21 cohesin ChIP-Seq marks.



**Figure 3.31:** UCSC genome browser view of cohesin sites located near the *TMEM186* gene. We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that our target CTCF site in the region has strong SMC3 and RAD21 ChIP-Seq marks indicating the presence of cohesin.

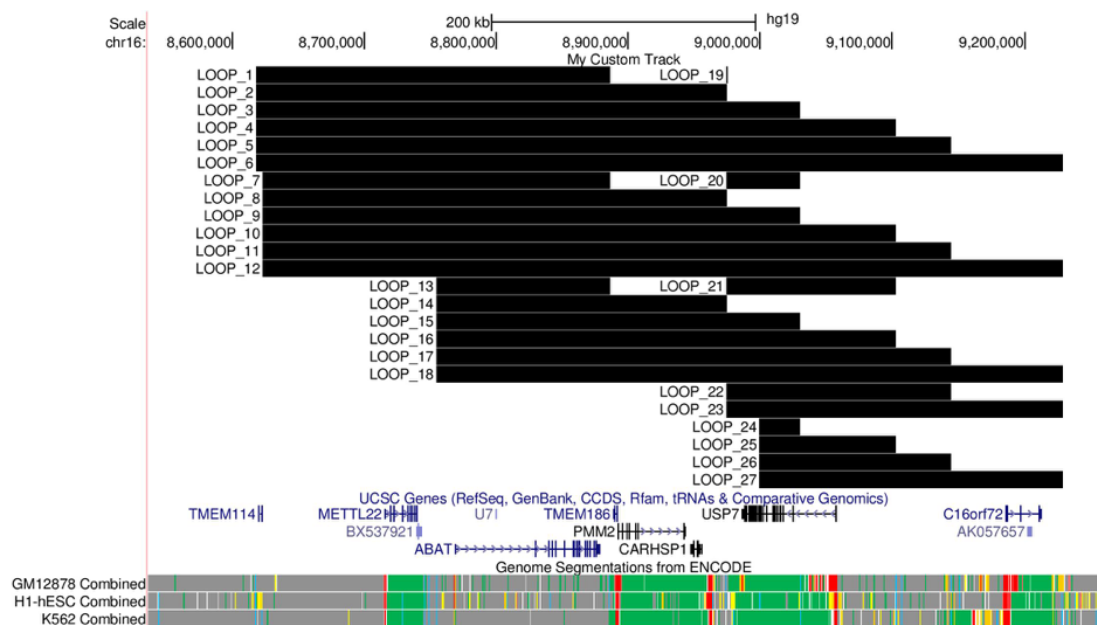


**Figure 3.32:** UCSC genome browser view of cohesin sites sites to the right of *CARHSP1*. We additionally show the CTCF and ZNF143 sites identified in the previous sections. We show that the left most site has a medium CTCF and cohesin signature while the centre two CTCF sites have a weak cohesin mark. The right most site has strong SMC3 and RAD21 ChIP-Seq marks indicating the presence of cohesin.

### 3.2.6.5 Structural Loop Prediction

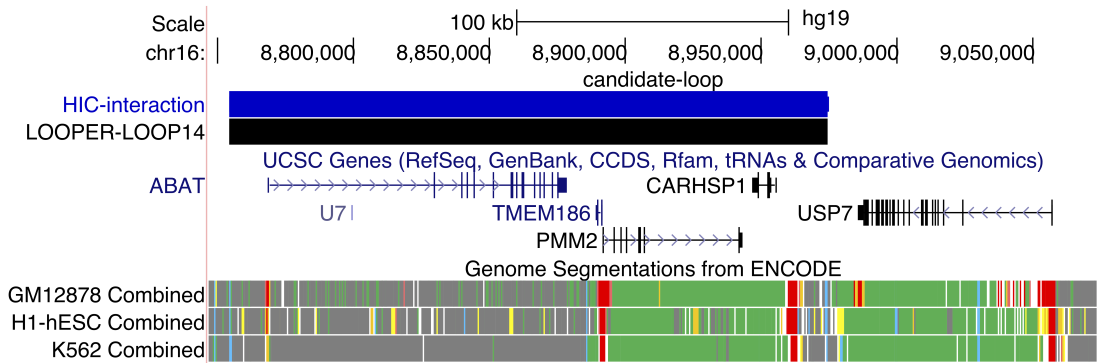
After the initial analysis of our region of interest was complete, we then fed the TFBS ChIP-Seq dataset and our MAST motif scanning results (including site orientation) into our LOOPER chromatin loop prediction algorithm. LOOPER first identifies potential loop half-anchors by searching for highly-conserved CTCF sites with binding motifs that overlap with RAD21 and SMC3 ChIP-Seq signatures. An exhaustive set of potential loops is subsequently generated from all the combinations of convergent half-anchors in the region of interest. LOOPER automatically converts the final loop set into a BED track for display in UCSC browser (Figure 3.33, p.208).

LOOPER predicted 18 possible CTCF structural loops in our wider region of interest. Given the number of potential loops, identifying the 'true' loop(s) was challenging; we sought to obtain further experimental data to confirm one loop over another. After the



**Figure 3.33:** LOOPER chromatin loop prediction results visualised as a bed track in UCSC browser. The potential loops are shown with IDs at the top as black blocks showing the loop domain; the loop anchors for each loop are located at the ends of each block. Below we show the gene and genome segmentation tracks for localisation.

construction of the LOOPER pipeline, the chromatin confirmation experiment Hi-C was developed to provide the first genome-wide experimental confirmation of chromatin loops. We obtained Hi-C data from an early paper [245] and visualised the predicted loops in UCSC browser alongside our loop prediction results. The Hi-C data matched loop ID 14 accurately, confirming the existence of a potential structural loop that spanned from the *ABAT* and the *CARHSP* anchor regions described earlier in our results (Figure 3.34, p.208).



**Figure 3.34:** UCSC browser diagram showing the LOOPER predicted loop number 14 (out best loop candidate). We overlay the Hi-C interaction experimentally detected from Belton *et al*, lending confirmation that LOOPER does indeed predict valid chromatin loops.

### 3.2.6.6 Functional Loop Prediction

Our hypothetical functional looping model states that ZNF143 binds to promoter regions before interacting with other regulatory elements such as mediator complexes bound to enhancers. We show in our motif analysis that there is a single strong ZNF143 site with a binding motif in the bidirectional promoter of *PMM2* and *TMEM186*. Weak ZNF143 ChIP-Seq signals are also detected at both loop half-anchors of the putative structural loop in the ROI. Our results suggest that there is a functional loop that is pulling the *PMM2* promoter down towards the identified primary loop anchor.

### 3.2.7 LOOPER Graph Analysis

We present here our results from the analysis of the *PMM2* ROI using our LOOPER graph visualisation tool. We chose to layer both CTCF and POLR2A (a sub-unit marker for DNA Polymerase 2) ChIA-PET experiments into the same graph to capture both structural and functional looping interactions. Given that we could not obtain ZNF143 ChIA-PET data, we hypothesised that an indirect POLR2A signature would be present from any functional looping in our ROI. We present our pre-processing results before showing the resultant graph structure; we then show additional results from other similar tools which add context to our graph.

#### 3.2.7.1 Source Data and Pre-Processing

We obtained ChIA-PET data from ENCODE for the K562 (human immortalised myelogenous leukaemia) cell line; this was the only dataset that had both CTCF and POLR2A data to compare. Both experiments were completed under the ENCODE ChIA-PET protocol, the raw FASTQ data for each experiment was downloaded from the ENCODE data repository from experiment ENCSR000BZY [246] and ENCSR000CAC [247] for POLR2A and CTCF respectively. We obtained TFBS, genome segmentation and TF motif position data from the UCSC browser database for graph annotation.

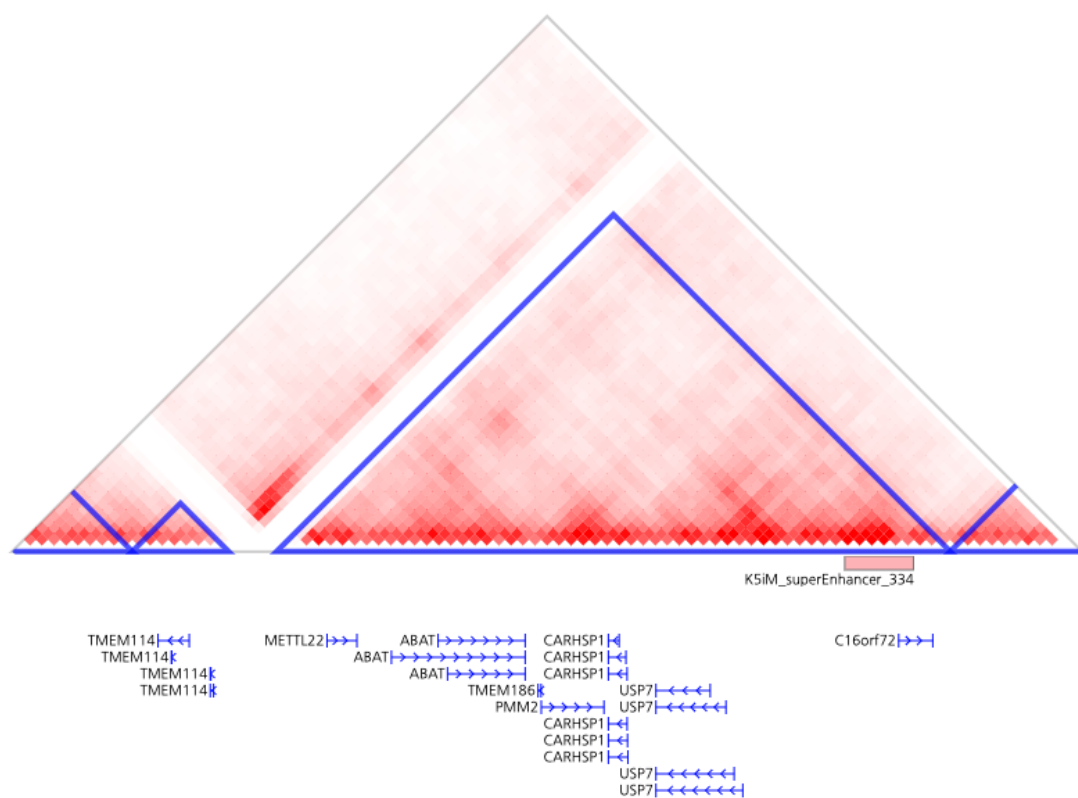
We subsequently ran the two paired-end FASTQ files for each experiment through the Mango pipeline (2.4.9.2, p.130). After processing, we obtained a list of 37,522 CTCF and 21,400 POLR2A interactions for graph generation.

### 3.2.7.2 Graph Analysis

We used the pre-processed files from Mango as input to LOOPER, filtering for the ROI. To analyse functional and structural looping, we wished to visualise the graph for all intra-TAD interactions for CTCF and POLR2A inside the local TAD for *PMM2*. To define our TAD, we used the genome interaction tool 3DIV [248] where we generated a TAD diagram using K562 and *PMM2* as the target (Figure 3.35, p.211). The resultant graph shows a double TAD in the ROI comprising one large and one small TAD. Given the methodology in which 3DIV generates TADs we argue that the area of missing data shown in white causes a double TAD to be erroneously shown; we also argue that given the additional signal above the whited out area within the primary TAD, the TAD should actually encompass both TADs. We therefore define our region of interest TAD as the area spanning from the left of *Transmembrane Protein 114 (TMEM114)* to the right of *Chromosome 16 Open Reading Frame 72 (C16orf72)* - chr16:8,568,000-9,265,000; we set LOOPER to filter for contacts within this area. The resulting graph shows a network of CTCF and POLR2A interactions spanning the whole TAD (Figure 3.36, p.212). We also show the region as viewed in the UCSC browser to show the linear distribution of genes in the TAD (Figure 3.37, p.212). The full run parameters and output for the graph generation are listed in Appendix C, p.283.

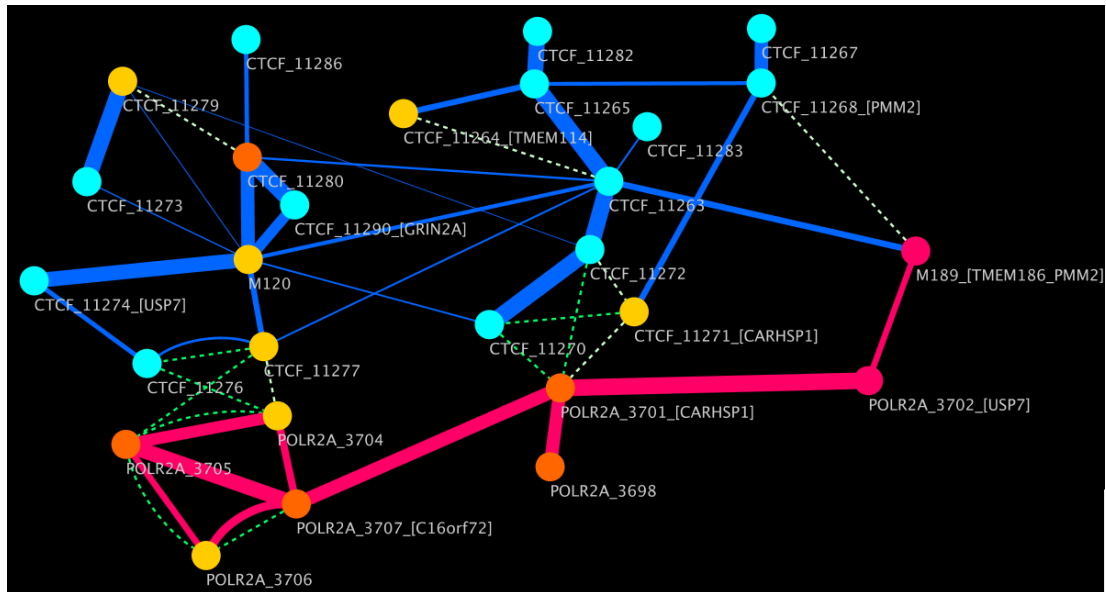
The LOOPER graph has two distinct regions on the left and right with distinctly fewer and weaker contacts linking the two together. The genes on the left side of the graph such as *Ubiquitin Specific Peptidase 7 (USP7)* and *C16orf72* are all located towards the right side of the TAD and have little connectivity with the target *PMM2* promoter. To reduce complexity, we only discuss the right hand side of the graph (left side of the TAD) in detail in this section, mentioning where needed other elements for completeness.

We first centre our analysis on the bidirectional promoter for *PMM2* and *TMEM186*, this is defined by the node labelled M189\_[*TMEM186\_PMM2*] (Figure 3.38, p.213). The node has three connections: one proximal connection to another contact site (green dotted line), one CTCF ChIA-PET contact (blue line) and one POLR2A (red line) ChIA-PET contact. We also show the analysis of the ChIA-PET region covered by the contact in the UCSC browser for context.

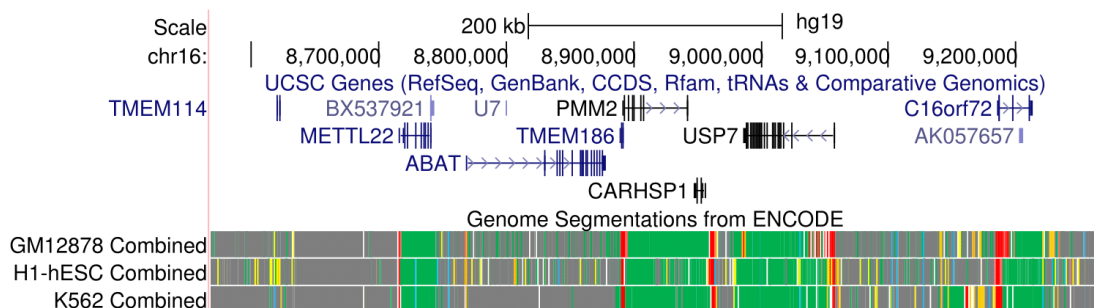


**Figure 3.35:** Diagram of the predicted TAD around *PMM2* in the K562 cell line. The blue lines demark the TAD boundaries. Each red square has a red scale in proportion to the number of Hi-C contacts present at the intersection between the two sites on the x-axis. We propose that the actual TAD surrounding *PMM2* comprises both the primary TAD and the small TAD adjacent to the left.

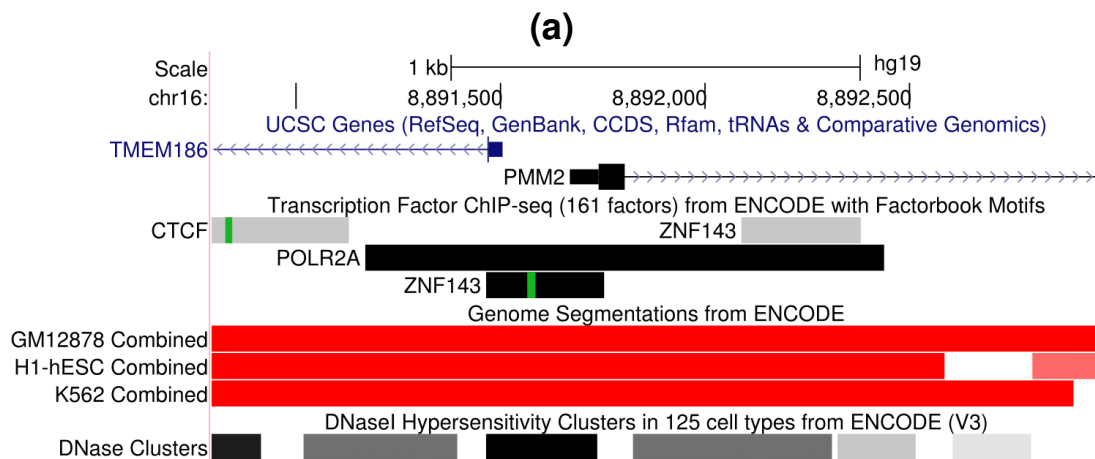
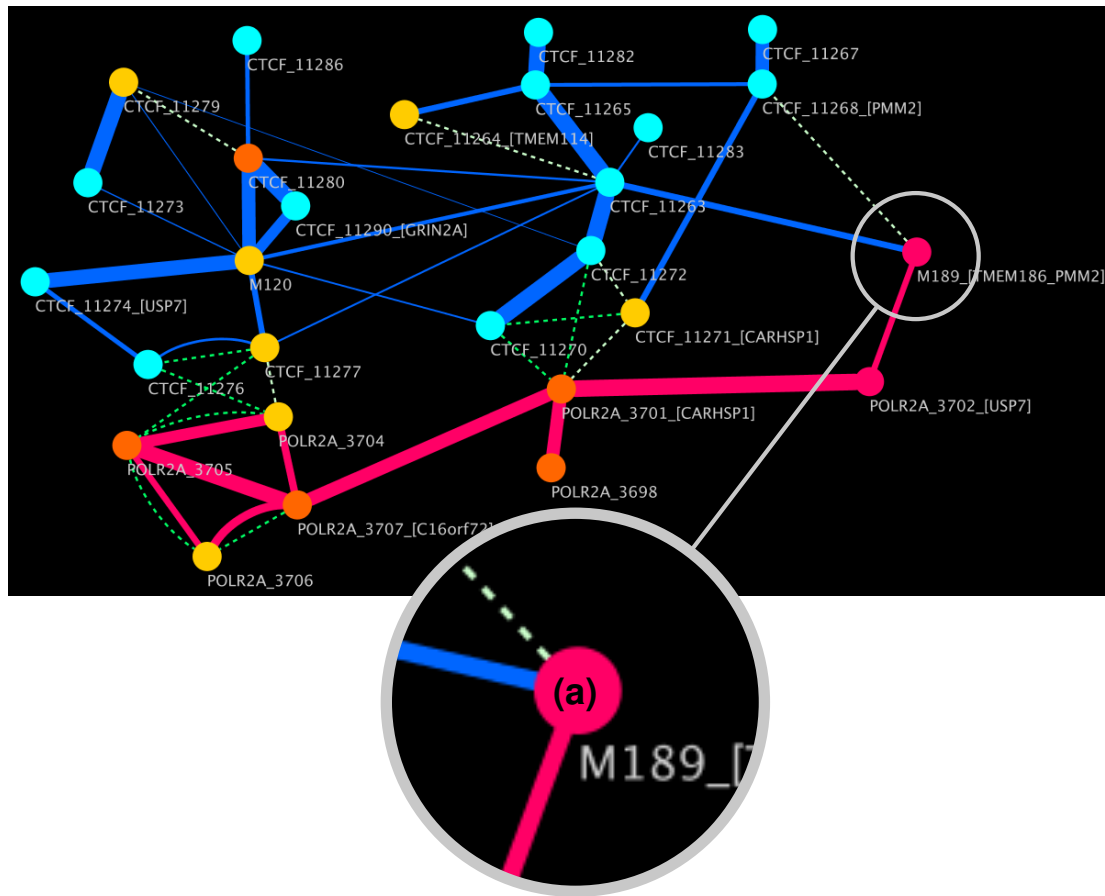




**Figure 3.36:** LOOPER Graph output for the *PMM2* TAD region. The nodes represent points of contact from ChIA-PET data, the edges show which contact points were in contact with each other. The edge colour shows blue for CTCF ChIA-PET and red for POLR2A ChIA-PET. The edge thickness shows the relative interaction strength of the contact. The node colour shows blue for insulators, red for promoters, yellow for enhancers and orange for regions which contain both promoters and enhancers. The green dotted arcs represent nodes which are physically close to each other on the genome. The node labels show the contact site type, the ID and the gene promoters which are in proximity to the contact point.



**Figure 3.37:** Target region for LOOPER graph analysis shown in the UCSC browser.

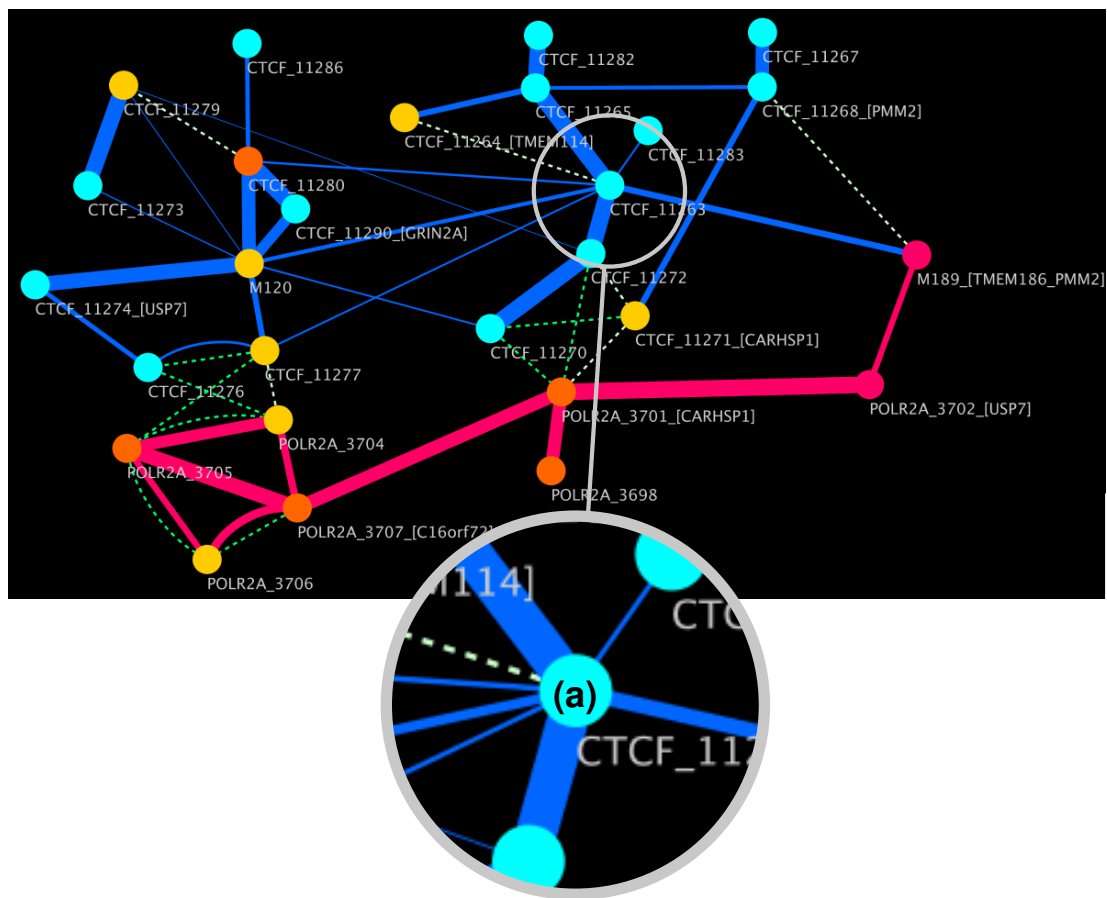


**Figure 3.38:** A Highlighted section of the LOOPER analysis graph. **(a)**, *PMM2* promoter node - has proximal connection to another contact site (green dotted line), one CTCF ChIA-PET contact (blue line) and one POLR2A ChIA-PET contact (red line). UCSC browser of **(a)** shows the *PMM2/TMEM186* bidirectional promoter that overlaps with the ZNF143 ChIP-Seq block which contains the HIPKD mutation. The genome segmentation track shows red for promoter. There is a strong, well conserved POLR2A signature, indicating that the region is transcriptionally active. Refer to Figure 3.37, p.212 for positional context of **(a)** within the TAD.

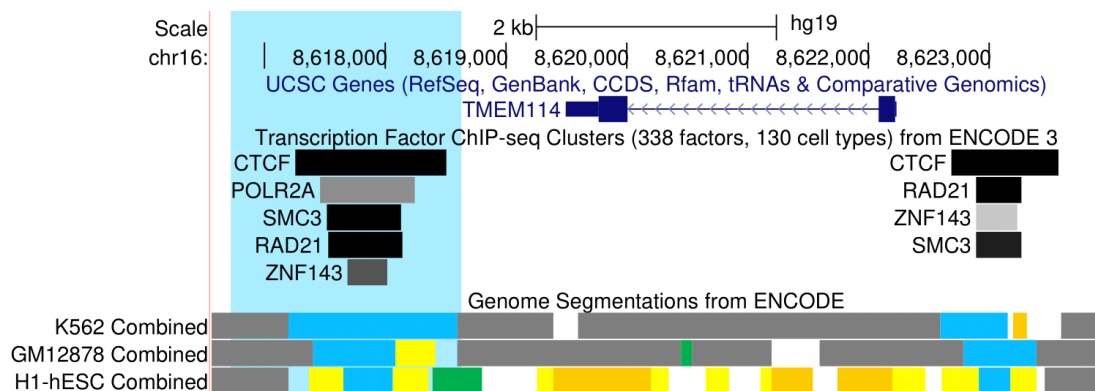
M189\_[TMEM186\_PMM2] is connected *via* a weak CTCF connection to the CTCF\_11263 node. CTCF\_11265 is the most connected node in the TAD in terms of CTCF contacts, indicating that it is a major loop anchor hub (Figure 3.39, p.215). Analysis of the node in UCSC browser shows evidence for a loop anchor with strong CTCF and cohesin marks. The genome segmentation track shows several enhancers nearby in the intronic regions of *TMEM114*; the enhancers only show on one cell-line suggesting that they are not ubiquitous and may exhibit tissue specificity. The anchor also shows evidence for indirect binding of ZNF143 and POLR2A. The POLR2A signature indicates that one or more promoters are in contact with the anchor. The ZNF143 indirect binding signature strongly suggests that the *PMM2* promoter is in contact with the loop anchor, as it is the only ZNF143 TFBS in the region. We believe this may be a significant regulatory region for this TAD.

CTCF\_11263 has a strong CTCF connection to CTCF\_11265 and CTCF\_11282 (Figure 3.40, p.216). CTCF\_11263 is a loop anchor that is intronic to *Methyltransferase Like 22* (*METTL22*), it has a strong connection to the left TAD boundary enhancer cluster and has strong evidence for indirect binding of POLR2A and ZNF143, suggesting contact with the *PMM2* promoter, most likely through contact at the TAD boundary through CTCF\_11263. CTCF\_11282 is a loop anchor at the extreme right of the TAD that has a strong connection to CTCF\_11265. The node also has weak ZNF143 and POLR2A signatures, giving tantalising evidence that the two TAD ends are in contact with each other.

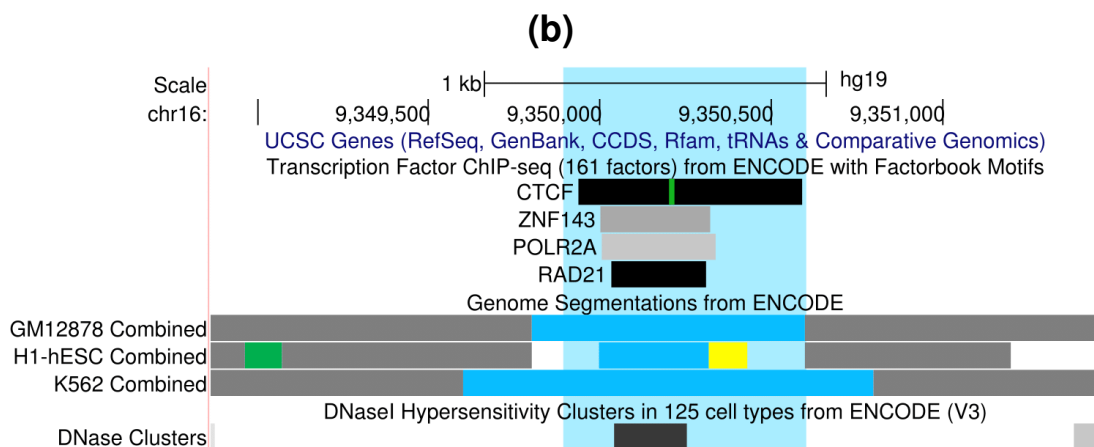
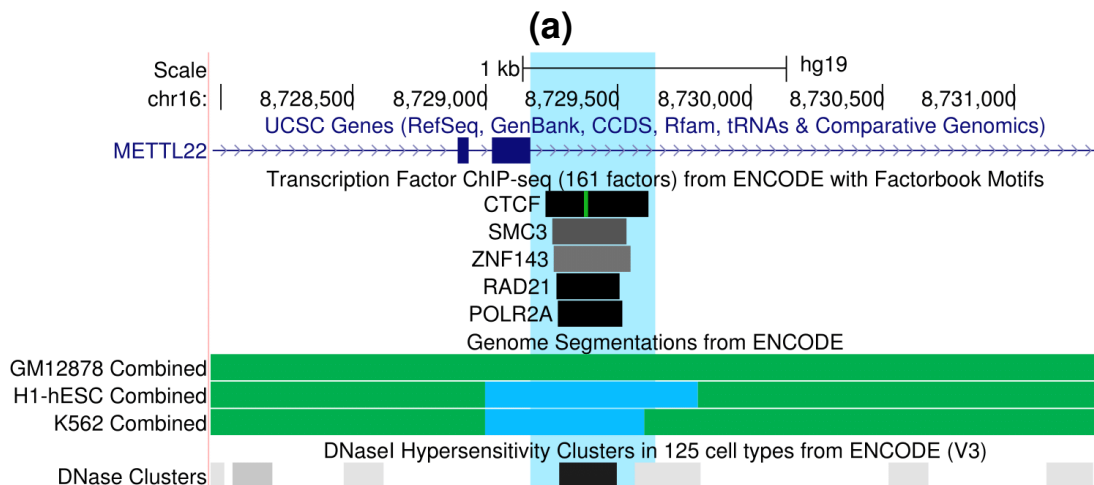
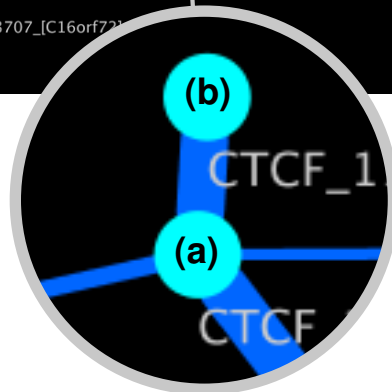
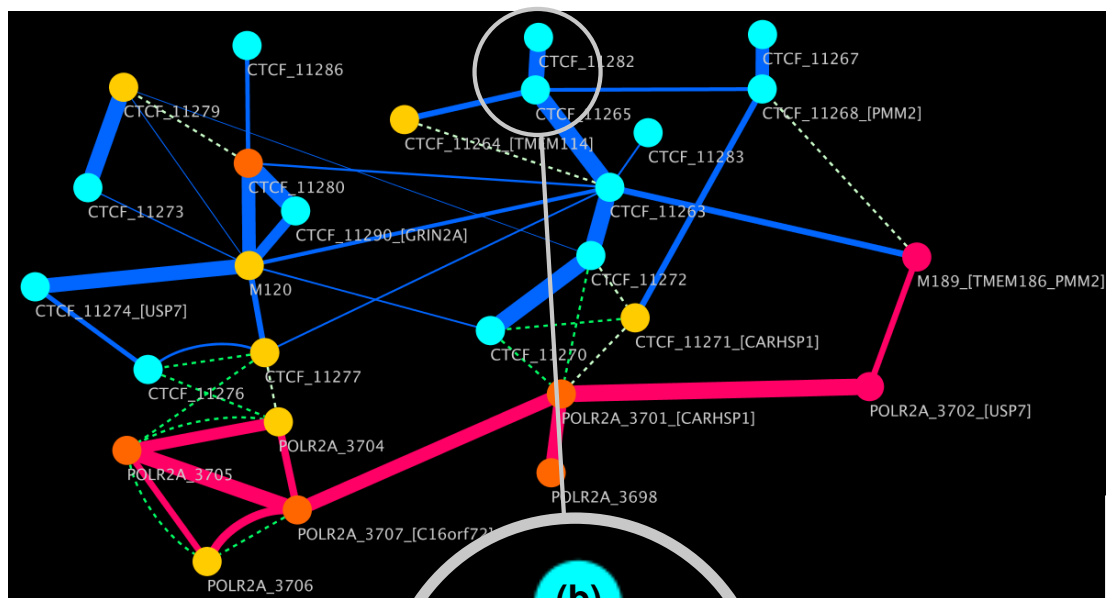
Interestingly, the three nodes discussed above span the whole TAD (one on the left TAD boundary, one in the centre left and one on the right TAD boundary), suggesting both the ends of the TAD and points along the region all occupy the same physical space. This is inline with the CTCF/Cohesin loop theory that TADs are major structural loops that anchor at the boundaries, thus bringing the two sides of a TAD into proximity (Figure 1.9, p.58). CTCF\_11265 acts in the middle of the nodes suggesting a more complex looping pattern than a single chromatin loop.



(a)

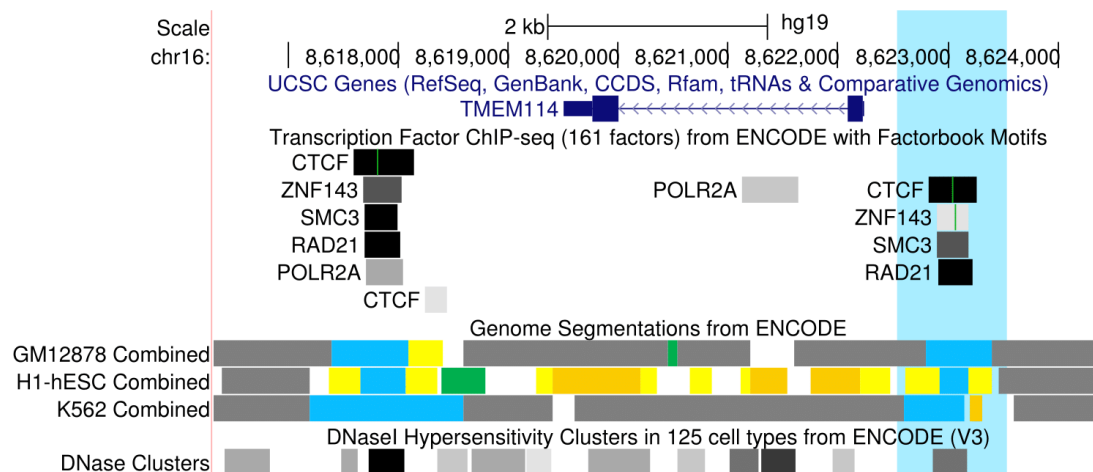


**Figure 3.39:** A Highlighted section of the LOOPER analysis graph. **(a)**, highly connected loop anchor to the left of *TMEM114* (CTCTF\_11263 highlighted in blue). Strong evidence for contact with the *PMM2* promoter through the graph connection and indirect binding of ZNF143 and POLR2A. The site has several enhancers downstream in the intronic regions of *TMEM114*. Refer to Figure 3.37, p.212 for positional context of **(a)** within the TAD.



**Figure 3.40:** A Highlighted section of the LOOPER analysis graph. **(a)**, a loop anchor situated in an intron of *METTL22* with a strong connection to the enhancer cluster at the left TAD boundary (highlighted in blue). Evidence for indirect ZNF143 and POLR2A binding is present. **(b)**, a loop anchor situated on the right TAD boundary (highlighted in blue). Evidence for indirect ZNF143 and POLR2A is present. Refer to Figure 3.37, p.212 for positional context of **(a)** and **(b)** within the TAD.

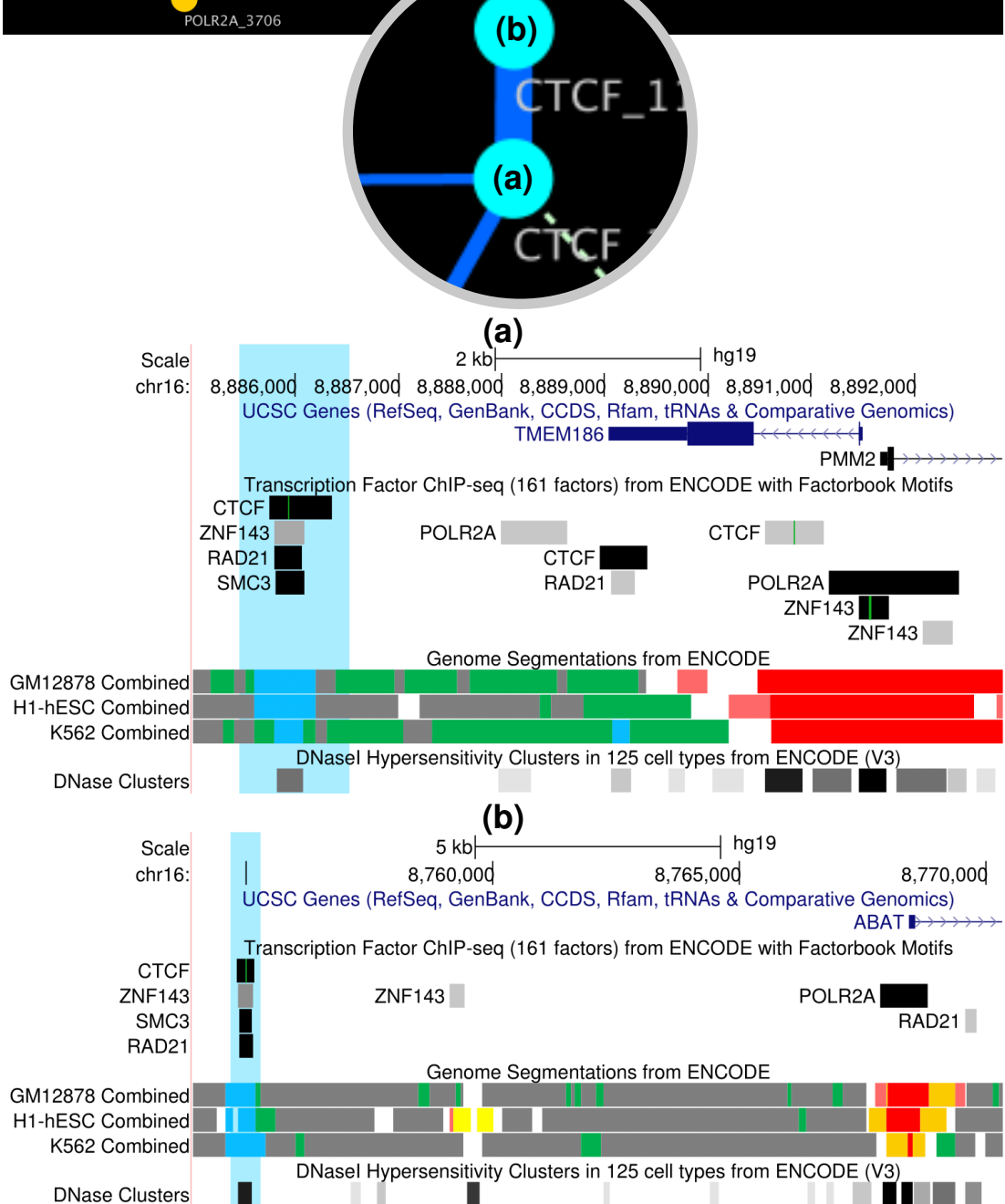
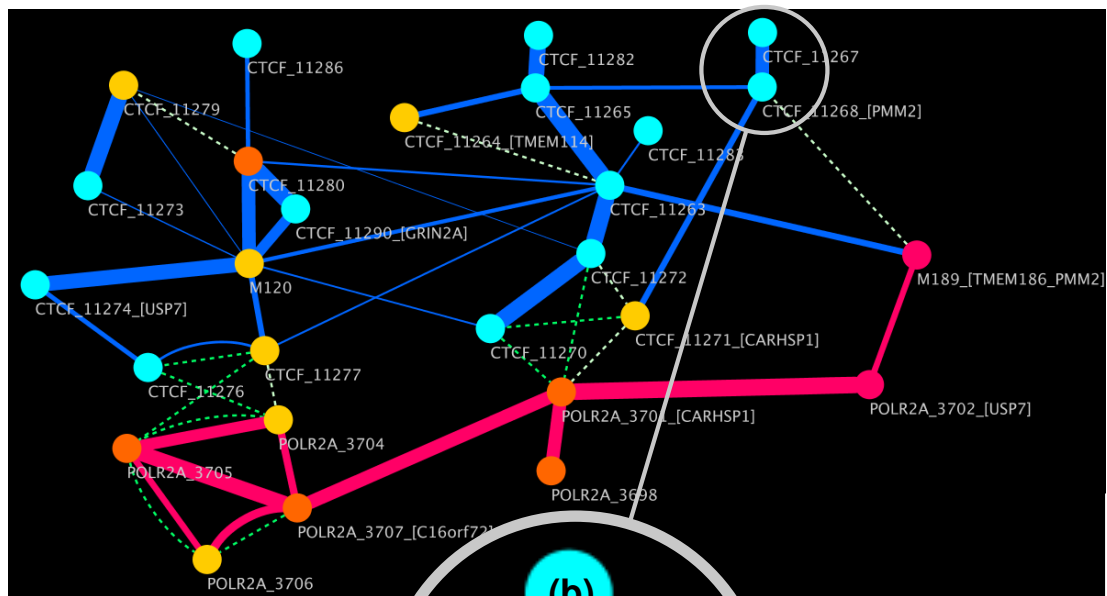
CTCF\_11264\_[TMEM114], another loop anchor on the left TAD boundary located on the opposite side of *TMEM114* (Figure 3.41, p.217), has a weak CTCF connection to CTCF\_11265 which would bring this site into contact with the TAD anchor regulatory region in the same way as the connections shown in Figure 3.40, p.216.



**Figure 3.41:** UCSC browser capture of the CTCF\_11264\_[TMEM114]. The node region is highlighted in blue. We show that the site is a loop anchor with several predicted enhancers clustered around the *TMEM114* region. The site has evidence for indirect binding of ZNF143 and is located near the promoter for *TMEM114*.

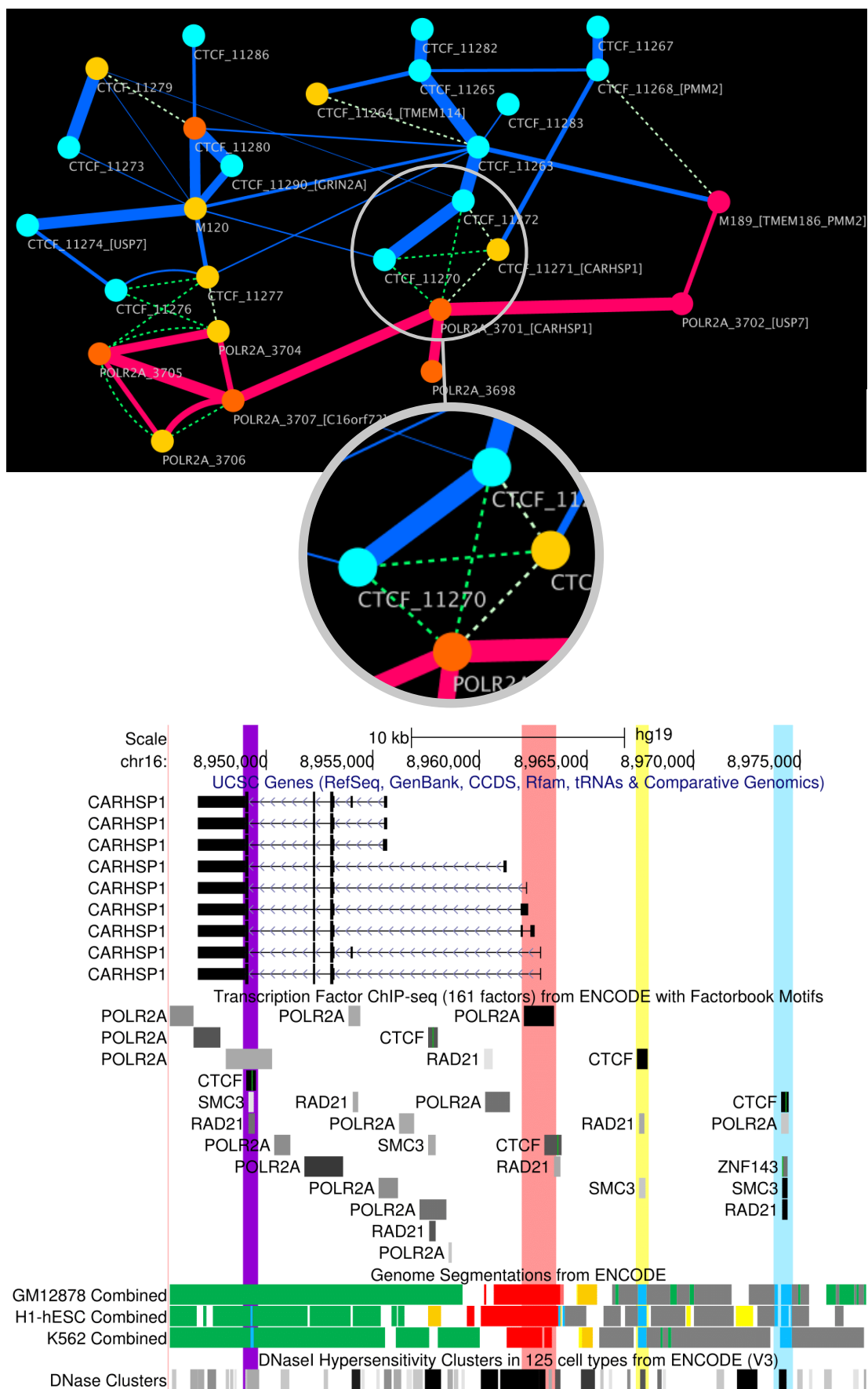
CTCF\_11268\_[PMM2] is a physically proximal node (not linked by ChIA-PET data but by proximity on the genome) to M189\_[TMEM186\_PMM2] and has a strong CTCF connection to CTCF\_11267 (Figure 3.42, p.218). Both sites are loop anchors with ZNF143 residue. Interestingly, CTCF\_11267 is the left loop half-anchor predicted to the best candidate loop for HIPKD in our loop prediction algorithm; this is evidence that this loop exists but also suggests the loop is part of the wider looping structure for the TAD. CTCF\_11268\_[PMM2] has two CTCF contacts with other nodes; CTCF\_11265 is a putative loop anchor in *METTL22* and CTCF\_11271\_[CARHSP1] is discussed below.

The next graph section we focus on is the cluster of contact nodes around *CARHSP1* (Figure 3.43, p.219). CTCF\_11272 and CTCF\_11270 are weak loop anchors located in the *CARHSP1* region; the four nodes in proximity of this region while having no other detected CTCF contacts between each other are located in a region with several enhancers. we suggest these nodes form an interaction 'backbone' with CTCF\_11263, CTCF\_11265 and CTCF\_11282; these sites span the TAD from left to right, contacting at key promoters and enhancer clusters.



**Figure 3.42:** A Highlighted section of the LOOPER analysis graph. **(a)**, a loop anchor situated to the left of *TMEM186* (highlighted in blue), it is the closest anchor to the *PMM2* promoter. Evidence for indirect ZNF143 binding is present. **(b)**, a loop anchor situated to the left of *ABAT* (highlighted in blue). Evidence for indirect ZNF143 binding is present. Refer to Figure 3.37, p.212 for positional context of **(a)** and **(b)** within the TAD.

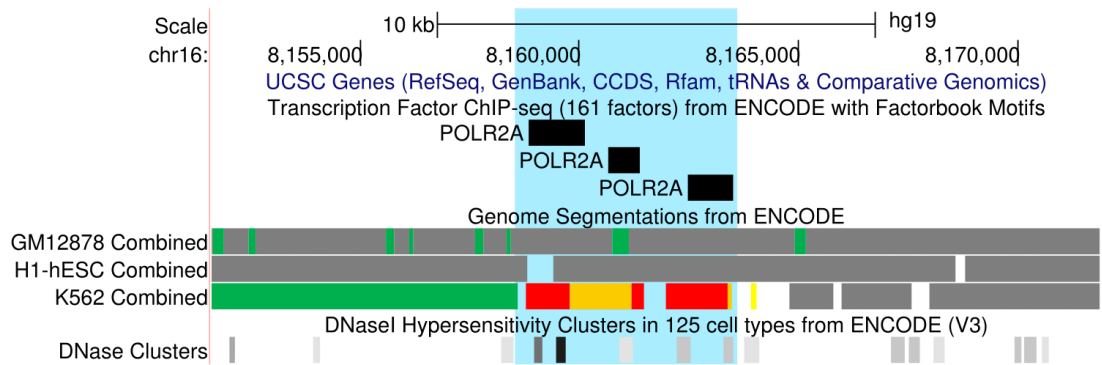




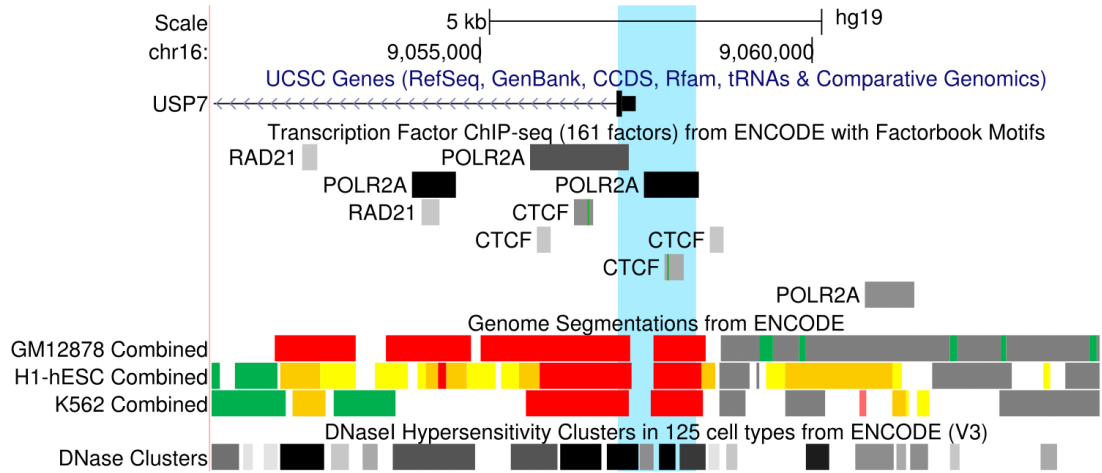
**Figure 3.43:** A Highlighted section of the LOOPER analysis graph. Four nodes are physically located in the same region while CTCF\_11272 and CTCF\_11270 have connectivity to the regulatory hub at CTCF\_11263. *CARHSP1* is shown with all splice variants in the UCSC browser. Two connected loop anchors: CTCF\_11272 (blue highlight) and CTCF\_11270 (purple highlight) are located at either end of the diagram. POLR2A\_3701\_[*CARHSP1*] is located in the promoter region for *CARHSP1* (red highlight) with the enhancer CTCF\_11271[*CARHSP1*] located to the right (yellow highlight). Refer to Figure 3.37, p.212 for positional context of *CARHSP1* within the TAD.



We now focus on the strong POLR2A connections seen on the right hand side of the graph that link the promoters of several genes in the TAD. POLR2A\_3698 is defined as an enhancer/promoter despite being located at the extreme left of the TAD in a gene depleted region (Figure 3.44, p.220). The UCSC browser analysis shows a triplet of POLR2A indirect binding signatures, suggesting a triple enhancer site. More detailed analysis of TFs binding to this site show three definitive enhancers with over 100 different transcription factors binding to three distinct enhancers. Interestingly, the ChIP-Seq data suggests this site is cell specific to the K562 cell line. POLR2A\_3702\_[USP7] is shown to be the promoter for *USP7* in an area highly populated with regulatory elements; it has also has a functional connection to the *PMM2* promoter region (Figure 3.45, p.220).



**Figure 3.44:** UCSC browser capture of the POLR2A\_3698 node. The site is located at the extreme end of the LHS TAD boundary and shows a predicted enhancer element with several strong POLR2A signatures suggesting promoter contact.



**Figure 3.45:** UCSC browser capture of the POLR2A\_3702\_[USP7]. The site is located at the *USP7* promoter which shows a cluster of regulatory elements including CTCF signatures and several enhancers.

### 3.2.7.3 Additional Tool Assessment

The NIH 4D nucleosome project [249] has a repository of assessed software tools that can be used to process data from interaction experiments such as Hi-C and ChIA-PET. LOOPER was designed to sit within this existing ecosystem of databases, pipelines and chromatin architecture visualisation tools. We performed an assessment of the existing software ecosystem to demonstrate that LOOPER is unique in this space and to find other visualisation tools with synergies to LOOPER that could be used to perform additional analysis on the HIPKD region of interest.

The 4DN software list was used as a guide to assess the tool ecosystem as this list is up-to-date and maintained by a consortium of research groups. Tools were only selected for analysis if they produced visualisations (not just processing pipelines) and if they processed Hi-C or ChIA-PET data (3C and 4C were not counted). Our results are summarised in Table 3.11, p.222.

UCSC browser is used extensively by the scientific community and can integrate large amounts of experimental data displayed in a linear manner. We do not discuss UCSC browser further as we already show the integration of LOOPER results with UCSC browser information earlier in this section.

Many of the other tools in the ecosystem concentrate on Hi-C data visualisation either using heatmaps, linear genome annotation or a combination of both. Only GIVE and 3DIV visualised interaction data in a way similar to LOOPER, albeit on a linear genome. Both of these tools highlight a key difference between LOOPER and other visualisation software assessed: all interactions are anchored to a linear genome so that it is comfortable for the user to visualise the genomic location and so that other annotations such as ChIP-Seq binding data can be easily shown in parallel. In LOOPER, the linear genome is removed and the interactions are displayed in a force-directed graph; thus, the true pattern of interactions is revealed in an intuitive way. This view is extremely useful when exploring chromatin looping in a region of interest; however, LOOPER is perhaps best used in conjunction with a tool that has a connection to the linear genome so that both views can be considered simultaneously. GIVE was also the only tool other than LOOPER that we found was able to display ChIA-PET data in a reasonable way by comparing two linear genomes on top of each other and using banding to link areas of interaction between the two.

In terms of data availability, many of the tools do not have any built-in datasets, including LOOPER. 3DIV, GIVE and Juicebox all have integrated datasets; 3DIV has the most comprehensive set of Hi-C with many different cells and tissue types available through the browser.

We selected 3DIV to perform additional analysis on the HIPKD region of interest as it displayed Hi-C data, had a large selection of integrated datasets and produced filterable interactions on a linear genome as well as a 2D heatmap. Integrating Hi-C analysis alongside ChIA-PET visualisation from LOOPER allows for the coarser-grained analysis of the overall interaction landscape for all proteins rather than the single targets of ChIA-PET; subsequently, this allows for major structures such as TAD boundaries to be identified and for sanity checking of interactions identified in ChIA-PET.

Tool Name	Data Type	Visualisation Type	Data Availability	Comments
UCSC browser	All	Linear visualisation	All	Classic go-to genome visualiser
3DIV	Hi-C	2D Heat map over linear gene annotation	315 experiments from 80 human samples	Easy to use with integrated data and intuitive interaction mapping
HiCtool	Hi-C	2D Heat map	None	Full pipeline with contact map visualisation options using python
GIVE	Hi-C, ChIA-PET, ChIP-Seq	Linear gene annotation	ENCODE and up-loadable	Excellent linear visualisation tool but limited built-in data
Hi-C data browser	Hi-C	2D Heat map	None	Older browser, superseded by Hi-glass and Juicebox
Hi-C Plotter	Hi-C	2D Heat map over linear gene annotation	None	Python based visualisation generating data similar to 3DIV
HiGlass	Hi-C	2D Heat map over linear gene annotation	None	Web-based heatmap and gene annotation tool
Juicebox	Hi-C	2D Heat map	Much of the classical Hi-C data	Web-based heatmap generation and comparison

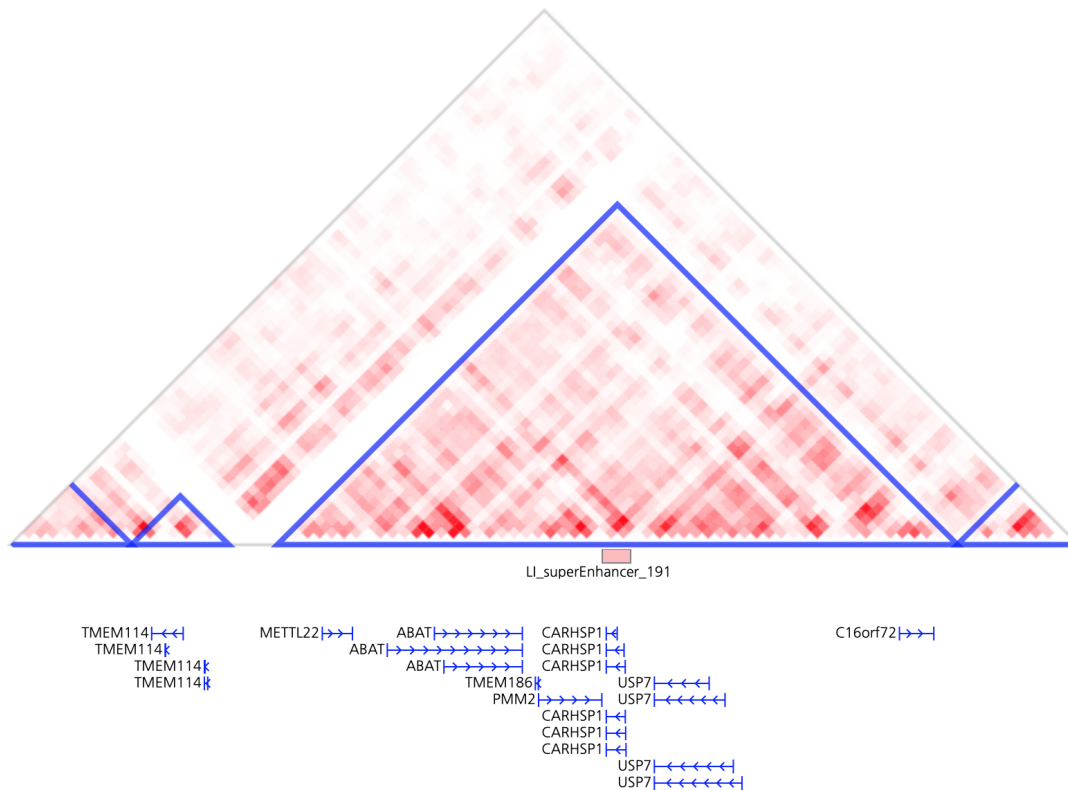
**Table 3.11:** List of Hi-C and ChIA-PET visualisation tools with comparison comments from the 4DN software portal.

### 3.2.7.4 Further Analysis

In this section, we now conduct additional analysis of our ROI using a one of these tools to further our understanding of the regulatory network identified by LOOPER.

We used the 3DIV tool [248] to generate a TAD analysis plot for a liver sample to better match the phenotype of HIPKD. We show that the predicted TADs are the same for liver tissue samples are the same. 3DIV uses H3K27ac histone modification data to identify en-

hancers and super-enhancers; we show that 3DIV identifies a liver-specific super-enhancer within our *PMM2* TAD (Figure 3.46, p.223).

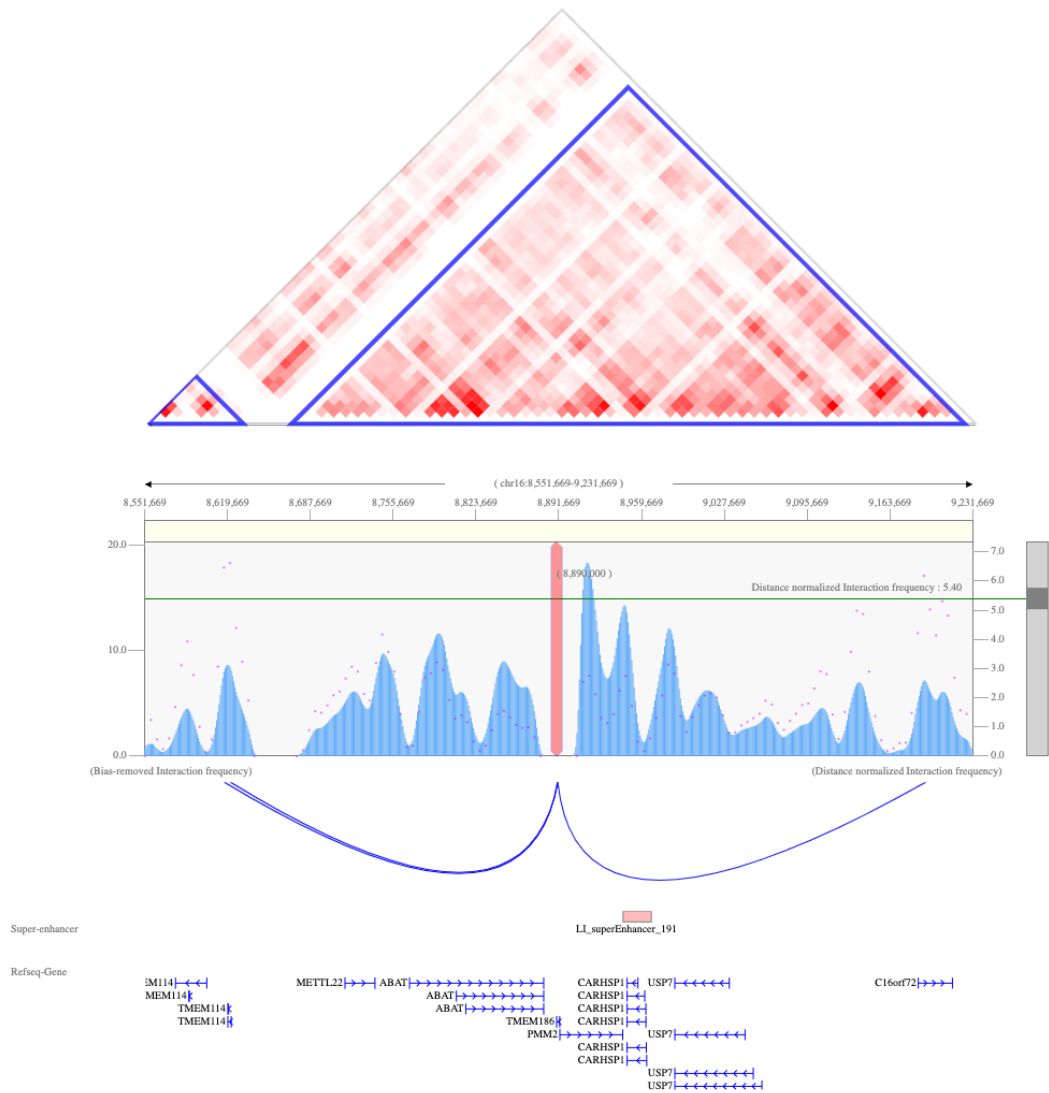


**Figure 3.46:** 3DIV capture of the *PMM2* TAD in a liver tissue sample. We show that the TAD structure is identical to one generate for K562 presented earlier (Figure 3.35, p.211). The TAD shows one liver specific super-enhancer: 'LI\_superEnhancer\_191'.

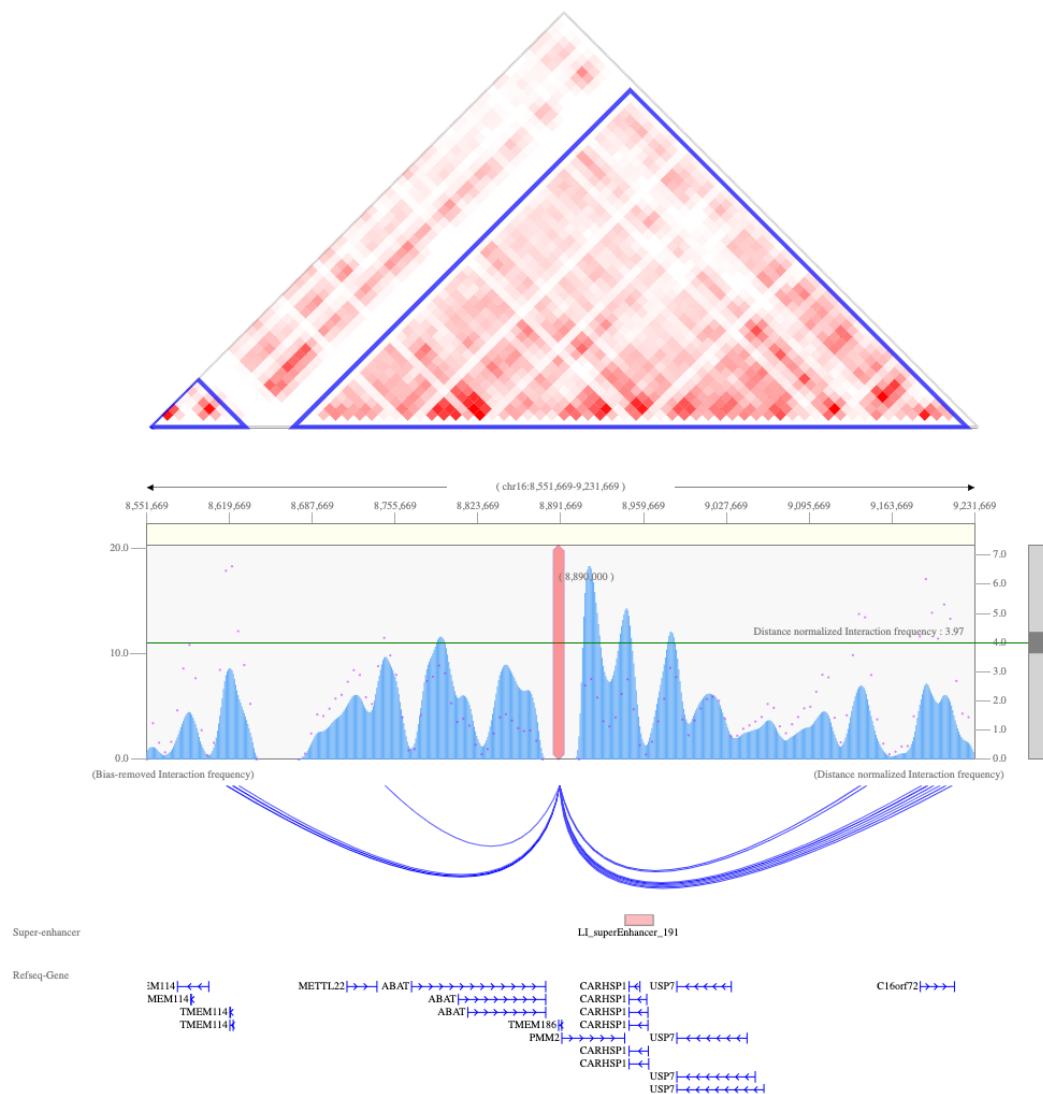
The liver-specific super enhancer is located in the region spanning the *CARHSP1* gene (chr16:8941669-8971669) (Figure 3.43, p.219). As described earlier, this region connects to the CTCF 'backbone' to which the *PMM2* promoter also appears to interact with. Several ZNF143 residue marks in the enhancer region also suggest contact with *PMM2* and the TAD boundaries.

We next used the Hi-C contact analysis feature of 3DIV to show which areas of the genome were in contact with the *PMM2* promoter in liver tissue; we used the HIPKD mutation as bait to give a one to many contact view. In our initial analysis, we filtered for the strongest contacts only - a fold change of above 5.40 the normal (Figure 3.47, p.224); we show interactions with both ends of the TAD to two regulatory hotspots confirmed in LOOPER with ChIA-PET data at the *TMEM114* and *C16orf72* genes. Lowering the fold change threshold to 3.97, we see increased levels of contacts to the same areas and to a loop anchor located between *METTL22* and *ABAT* (Figure 3.48, p.225). Reducing our threshold down to the minimum limit of 2, we see additional interactions to many of the enhancer areas identified by LOOPER including the liver super enhancer and the intergenic

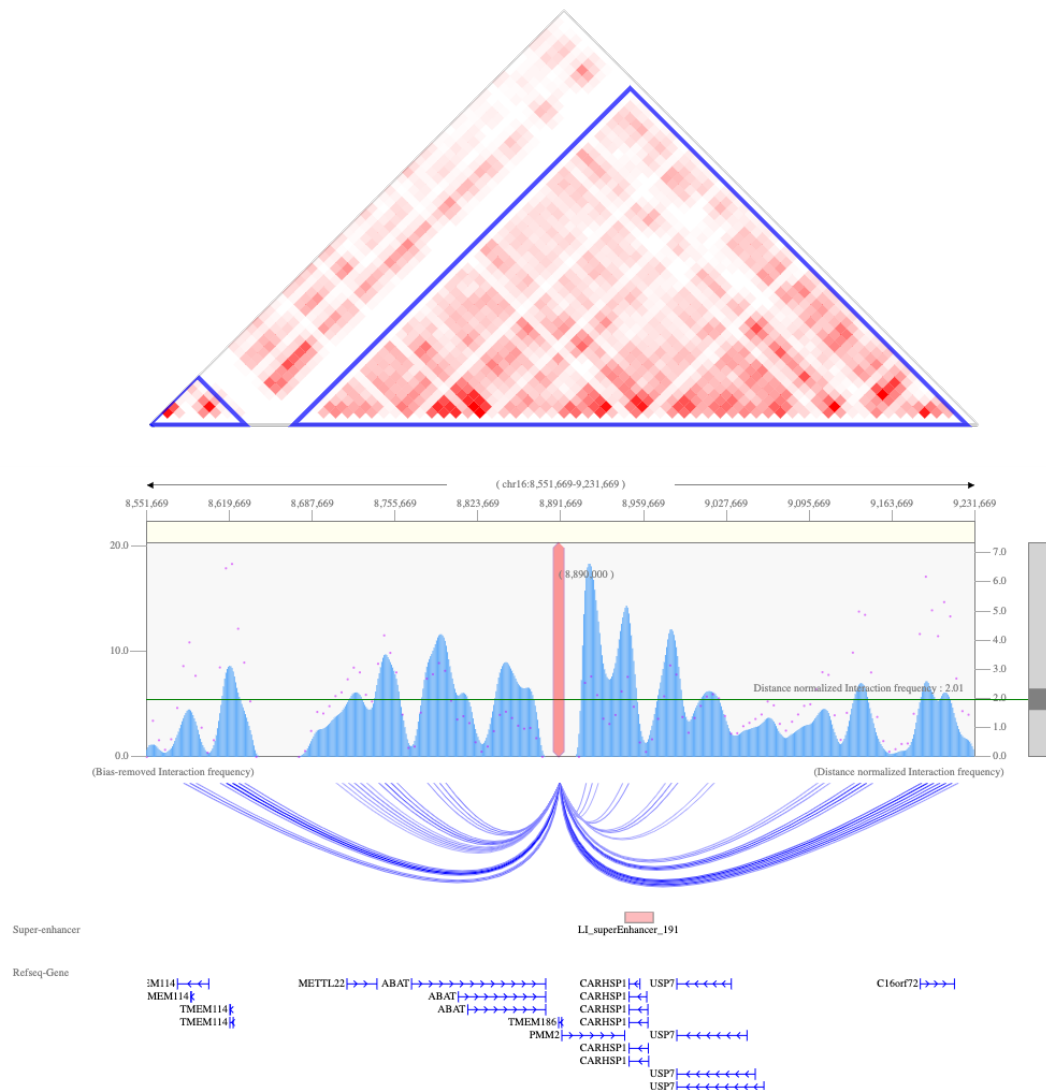
regulatory region between *USP7* and *C16orf72* which contacts the hub node 'M120' in our graph (Figure 3.49, p.226).



**Figure 3.47:** 3DIV capture of the *PMM2* TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. We show strong contacts with the TAD boundaries with previously identified regulatory regions.



**Figure 3.48:** 3DIV capture of the *PMM2* TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. We show strong contacts with the TAD boundaries with previously identified regulatory regions and with loop anchors in intergenic regions.



**Figure 3.49:** 3DIV capture of the *PMM2* TAD in a liver tissue sample showing Hi-C contacts with our HIPKD mutation. Show contacts with many of the regulatory elements show using ChIA-PET data in LOOPER including a liver specific super enhancer.

### 3.2.7.5 Summary

In summary, we have shown that experimental evidence from ChIP-Seq, ChIA-PET and Hi-C all suggest that the *PMM2* promoter is connected to a complex TAD-wide regulatory network. In the context of the HIPKD mutation we believe that the *PMM2* promoter ultimately links to a liver specific super-enhancer through structural and functional interactions at the TAD boundaries. We showed that there are ZNF143 ChIP-Seq signatures at key sites throughout the TAD despite there being only one ZNF143 TFBS at the HIPKD mutation; this suggests that the *PMM2* promoter is in physical proximity to many sites simultaneously. Our results show that there is TAD-wide, strongly connected chain of CTCF/cohesin loop anchors which we term the 'backbone' that link together two enhancer clusters and other structural sites. One enhancer cluster is at the left TAD boundary around *TMEM114* and one is downstream of the *CARHSP1* promoter, where the putative liver specific super-enhancer is located.

The Hi-C data from liver tissue samples visualised in 3DIV confirmed many of the regulatory interactions shown with the CTCF and POLR2A ChIA-PET data visualised using LOOPER. We additionally identified a liver-specific super-enhancer that we showed to interact with our HIPKD mutation.

We propose that the TAD is composed of a primary structural loop which spans the whole region. There are then several sets of nested CTCF/cohesin loops which cluster their anchors around the primary loop anchor at the TAD boundaries. The looping brings the two enhancer clusters we have defined into contact with the primary loop anchor. The *PMM2* promoter is brought into contact with the primary loop anchor *via* ZNF143 functional looping which brings it into contact with various *cis*-regulatory elements including the liver specific super-enhancer. This mechanism can be visualised in the same way as a single loop (Figure 1.11, p.59), but extrapolated to many loops where the loop anchors bunch together at the same physical base [72].



## Chapter 4

# Discussion

In this thesis, we have presented two different software tools which were developed to solve problems which we encountered during bioinformatic investigations undertaken by our research team.

We first presented our methodology for Genome-wide association study (GWAS) where we applied computer science practices such as version control, naming conventions and data pipeline analysis to standard GWAS methodology to build robust, repeatable and transferable genomic analysis pipelines. We defined the route from source data to results as a data pipeline that can be represented as a graph, where the edges are routes that data flows along and the nodes are transformations performed on data. A complete pipeline consists of a set of transformation nodes and connecting edges that converts input data into the final set of reports and results that satisfy the project goals. We subsequently demonstrated the effectiveness of this framework by creating a data pipeline for a GWAS based on genotyping data to investigate the genetic architecture of Steroid-sensitive nephrotic syndrome (SSNS).

Our initial GWAS results for the SSNS investigation contained significant noise in the form of false-positive associations. Subsequent detailed analyses showed that the input datasets were genotyped on different microarrays and subject to different analyses leading to a mosaic case-control cohort that had inherent errors, primarily due to strand mismatching. We developed Re-coding For Merging of Genotyping Data (REMEDY) to counter these issues when literature research revealed no viable alternatives; we then modified our data pipeline to include the REMEDY toolset without needing to change the entire methodology. We then re-ran our pipeline to produce the low signal to noise ratio Manhattan plots shown in the results section, demonstrating that REMEDY functioned correctly but also that our data graph methodology was able to adapt to problems we encountered.

Our second tool, LOOPER, was developed to solve a different type of problem than the data cleaning and quality control issues which REMEDY was designed to correct. Analysis of a newly characterised rare renal disorder Hyperinsulinism with polycystic kidney disease (HIPKD), led to the hypothesis that a change in chromatin looping at a specific genomic locus may be involved in the aetiology of the disease. A single point mutation in the promoter of *Phosphomannomutase 2 (PMM2)*, located in a binding site of the transcription factor Zinc-finger protein 143 (ZNF143) was implicated in the dysregulation of *PMM2* through the alteration of the local chromatin architecture; specifically, a functional chromatin loop made between the *PMM2* promoter and a CTCF/Cohesin loop anchor mediated by bound ZNF143 protein.

We developed LOOPER, a software suite capable of predicting chromatin loops from Chromatin immunoprecipitation with next generation sequencing (ChIP-Seq) data to explore the possible confirmations of chromatin architecture in the HIPKD Region of interest (ROI). LOOPER predicted several interesting functional and structural loops that supported our hypothesis. After data for new experimental methods that targeted chromatin confirmation were released, we extended LOOPER to visualise Chromatin interaction analysis by paired-end tag sequencing (ChIA-PET) and ChIP-Seq data as a force-directed graph to show experimentally confirmed structural and functional chromatin interactions. Next, we re-analysed the HIPKD ROI with the extended LOOPER software to show experimentally validated chromatin interactions. We first confirmed our original predicted loops and subsequently discovered that the local genomic region has many more chromatin features than first thought. The force directed graph output was layered on top of other existing tools to enhance our understanding of chromatin architecture in the HIPKD region of interest and successfully improve our hypothesis regarding the aetiology of HIPKD.

## 4.1 REMEDY

REMEDY was developed in response to the false-positive associations that we came across during our initial analysis on our SSNS dataset. Given the large number of case samples that we had, it was deemed infeasible to identify, collect and process control samples in-house as well as our cases. Instead, we identified a mosaic of genotyped datasets from online repositories such as European genome archive (EGA) that we merged to produce a single control set for GWAS. The datasets collected were sourced from a range of different studies and were genotyped on different microarrays and had been subject to different pre and post-processing methodologies. After investigation of the merged case-/control dataset, the source of the GWAS noise was deemed to be due to four primary areas: differences in the formatting and strand designation of the genotyping datasets, dif-

ferences in the variant databases used and quality of variant used for analysis, differences in genotyping chip designs and genome version changes.

We first manually attempted to fix these problems before realising that an automated script would be a better solution to the problem. After several script versions, further requests for our team to conduct other GWAS and literature surveys showing very little in the way of tooling to aid in these types of merging problems, we decided to turn our scripts into a fully featured tool, REMEDY. We adapted our GWAS methodology data graph to accommodate the three primary REMEDY sub-programs that assess, filter and perform quality control on input genotyping data into a mergable standard format ready for GWAS. The toolset fits into the data cleaning and Quality control (QC) portion of our pipeline; however, it does not replace more traditional QC such as call rate filtering. Instead, it adds additional stringent filtering specifically designed to remove false-positives from genotyping data used for association studies.

We packaged REMEDY into a deployable software tool so that it could be used by clinicians with little computing experience. REMEDY was successfully applied to several GWAS, of which one is presented in the results section as our case study.

#### **4.1.1 Manifest Scanning Evaluation**

During our investigation of the false-positives in our SSNS GWAS results, we encountered several problems that we traced back to the quality of the variants (from a GWAS perspective) that were included on the microarray. We located the primary reference file called the manifest that contained all probe design and strand designation details. We initially implemented an algorithm that utilised all of the available information in the manifest file to screen for potential Single nucleotide variant (SNV)s that would cause false-positives before expanding the filtering potential by integrating The single nucleotide polymorphism database (dbSNP) data to provide more fine-grained variant analysis. We discuss our evaluation of the different parts of the manifest scanning sub-program in the sections below.

##### **4.1.1.1 Structural Variation**

The first major source of noise identified was the utilisation of genotyped structural variants in a GWAS. A basic allele test is based on the chi-squared test - a simple comparison of categorical data, it is therefore not technically incorrect to perform the test on the binary outcome of whether a genomic position contains one structural variant over another. We argue that the problem lies in the genotyping process itself. The Illumina BeadChip process

uses fluorescent tagging to identify single nucleotide complement extensions to the probe sequence. Probes are designed so that the target variant overhangs the end of the probe sequence by one. The probe sequence design must be a balance between uniqueness to the target sequence where longer is more desirable, and hybridisation performance, where longer probes can be problematic.

When considering structural variants, the probe design must also somehow use single nucleotide variation to indicate which multi-nucleotide variant is present in the sample genome. Targeting areas of the structural variant that are different between the two target sequences is the obvious starting point [142]; by using single or multiple probes to genotype the differences one can distinguish between the two. However, this is probabilistic as by their multi-nucleotide nature, structural variants have many different possible combinations that cannot be detected unless a probe is specifically designed for it.

During our analysis, we encountered several probes targeting structural variation that were causing false-positive associations. It was difficult to ascertain whether the structural variants were being miscalled, or whether the differences in genotyping processing between our different datasets was the root cause; we, therefore, decided to eliminate all structural variation from our output. This decision was further compounded by the relatively low number of structural variants included on our target microarrays ( 1-2%); while more markers are important for association (structural or not), genotyped markers are not causally linked to the phenotype and thus, reducing the marker density by a small amount is unlikely to impact the final results.

As detection of structural variation becomes more accurate, it is being implicated in many disease mechanisms, especially so with Copy number variation (CNV)s and cancer [250] [251]. Despite the errors we encountered, popular genotyping manufacturers such as Illumina and Affymetrix as well as literature [142] all claim that some types of structural variation can be accurately detected on microarrays. With further research, is it conceivable that developing algorithms for merging structural variation from multiple microarray designs is possible; however, it would certainly not be a trivial problem. Given the rise of Whole genome sequencing (WGS) that is able to detect structural variation much more accurately [252], it is becoming more unlikely that microarrays will be used for detection of structural variation of the quality required for GWAS in the future; REMEDY is designed for use with genotyping data therefore, we believe our decision to remove structural variants was the best solution.

#### **4.1.1.2 Comments File**

During our assessment of the available data alongside the manifest file for each microarray, we noticed that there was a comments file included with some of the newer designs. Subsequent analysis revealed that the file contained a record of comments made against probes on the microarray. The majority of the comments related to probes that matched to Pseudoautosomal regions (PARs) as well as their intended target site, or probes which matched to multiple regions. All of the probe comments were negative in nature which lead us to the belief that the comments file was a record of design errors for the probes. Probe designs that match to multiple genomic regions are an apparent source of noise in a GWAS and given that the number was small ( 1%) we again decided as with structural variation, to blanket remove any variant which had an entry in the comments file. Of course, this leads to the potential error that as yet unseen commented variants that do not require filtering are removed, thus causing the potential for false-negatives. At the time of design, this was deemed an acceptable risk; however, we accept that this is a limitation of REMEDY in its current version and is scheduled to be fixed in a forthcoming release. Our proposed solution uses a configuration file that recognises each comment and has a keep/remove binary setting that the program uses to assess each probe on a case by case basis.

#### **4.1.1.3 dbSNP Variant Assessment**

During our investigation, we found several problems that could not be solved using the manifest file alone; foremost of these was multi-allelic variation. Given that Illumina genotyping microarrays can only distinguish between two variations because of the red/green labelling system (this is reflected in the A/B encoding of Illumina data), any variant which contains more than two options has the potential to be misread by the genotyping system and is therefore a possible source of noise.

Some multi-allelic variation can be detected by scanning within the dataset itself, by searching for variants which have more than two different nucleotides present. This type of local multi-allelic analysis is already performed as standard in GWAS methodology; however, it is not the whole story. A variant may be multi-allelic yet, in the data we only see two of the genotypes, thus it would pass a standard allele count test; yet, if one of those genotypes was, in fact, a miscall, this could still produce a false association where there was none. Merging data from multiple microarrays also presents the problem of different manifests using different alternate alleles of a multi-allelic variant, thus only causing noise if these sets are merged. In addition, variants can begin life bi-allelic and are subsequently upgraded to multi-allelic at a later time without this change being updated on the microar-

ray. The broader problem of multi-allelic variants in GWAS cannot, therefore, be solved with the manifest file alone.

We developed REMEDY-DB, a custom, file-based database as a high-speed, local representation of dbSNP initially, to provide the additional data we required to filter out multi-allelic variants. We used the latest record of multi-allelic variation that dbSNP provides combined with the standard allele count filter in our GWAS methodology to ensure that only strictly bi-allelic variants were included in our analysis. We found filtering for these removed a significant amount of noise in our GWAS, while the integration of dbSNP data opened several new doors for quality control given the amount of additional meta-data that we now had at our disposal.

In addition to multi-allelic variant filtering, we also developed several other methods for removing noise based on dbSNP data. The first was the removal of variants that did not match in dbSNP; while this was necessary for the correct operation of REMEDY, removing non-matching variants had several benefits. First, it filtered out custom probes that are included on a microarray; these probes are included in the microarray design not because of their suitability to being a genotype-able variant, but rather because of project-based driving factors, thus they have the potential to be sources of noise. Given these factors, we removed them as a precautionary measure despite not detecting any direct sources of error.

The other problem that variant matching solves is one of removed or merged rsIDs. During the evolution of dbSNP, variants are merged or removed much more frequently than microarray versions; this leads to a kind of variation drift where the genotyping data is no longer 'in-date' on the microarray. We solved this problem by removing any non-matching variants from our output data, thus removing the possibility that a variant that was no longer valid was included in a GWAS. However, merged dbSNP variants are still valid, they have been merged for admin purposes with another variant; we treat merged and deleted dbSNP ID's equally and delete both. While not a source of noise, this is a limitation of REMEDY as the correct operation would be to remove deleted variants but in the case of merged, to find the new variant ID and use this instead. This problem could be corrected with a software update; however, this data is not readily available in the reduced version of dbSNP which we obtained from the UCSC browser. We would instead, need to build a new REMEDY-DB from the full version of the dbSNP database that contains the history of all the merges and deletions. Given that the number of unmatched variants was small in our testing (<1%), this was deemed to be a feature for a future version of REMEDY.

The inclusion of dbSNP meta-data also allowed us to scan for variants that appeared to have corrupt information in dbSNP. Genotyping probes are designed from sources of

information such as dbSNP; therefore, the quality of the probe is directly related to the quality of information available for a variant. All variants are not created equal in dbSNP: some are based on high numbers of submission counts, are well established and have correct meta-data while others are the opposite. During our analysis, we encountered several variants where their genomic position was blank or had impossible positions; we deemed these to be an unacceptable risk to include them in our analysis and were removed.

The final and most complex dbSNP meta-data based filtering we performed was the strand mismatch. To our knowledge, this was a novel discovery and has not been mentioned in previous literature. Towards the end of our investigation, there were some variants that we knew were producing errors, yet we could not find the reason, i.e. they appeared to be normal. We eventually found that these variants had matching major and minor alleles between the manifest and dbSNP, but that their strand designations were different. For example, a variant has A/G in both manifest and dbSNP, but the manifest has it called TOP and dbSNP, BOT. This problem was difficult to detect as manifest files are DESIGN encoded; thus, these differences appeared normal unless the manifest file and dbSNP variant were re-coded to same strand designation. Once detected, we also found the reason behind the error complex to ascertain. Our most favoured hypothesis is that there are fundamental differences in the strand calling algorithms between Illumina and dbSNP; in particular, dbSNP uses several additional in-house tools to designate strand than Illumina. Without knowing what the exact algorithms are, we cannot correct these errors, and so we filter them out. Unfortunately, this represents a much large number of variants in our analysis ( 3-4%) and so is a limitation of REMEDY; however, it was still deemed acceptable given the time constraints of the project.

#### **4.1.2 Strand Designation Evaluation**

Our most significant breakthrough regarding our investigation of our GWAS errors was regarding strand designation. We noticed early on that there appeared to be several different strand designation schemes mentioned in our data. Illumina termed their strands TOP/BOT while dbSNP referenced FWD/REV; in literature, we also came across POS/NEG. We initially concluded that these were different terms for the 5' and 3' strands of Deoxyribonucleic acid (DNA). We found there to be a lack of verifiable information on the subject of strand designation in research, we eventually resorted to an obscure Illumina data sheet that detailed their TOP/BOT strand designation scheme.

From this document, we were able to ascertain that indeed, despite the strand scheme being denoted TOP and BOT, these keywords had no relation to the biological DNA strand. This, in turn, meant that the schemes were not all the same and TOP/BOT was different

from POS/NEG (which we had already found was equivalent to 5' and 3') and FWD/REV. After a detailed investigation into the origins of strand designation we began to find more papers on the subject through better domain knowledge search terms. The subject had been acknowledged as necessary terminology to reference the correct strand when discussing a variant; however, few had encountered problems when merging data from different encoding schemes. We concluded that the problem of merging large mosaic genotyping datasets was relatively new, given that the size of datasets and genotyping microarrays has been steadily rising. We identified this as a problem which required a novel solution; after a literature search for tools to help us we came across the software tool we refer to as the 'Oxford pipeline' [191], but after assessment, we deemed it insufficient for our needs.

After defining the different strand designation schemes and what they represented, we realised that merging two different genotype datasets that were on opposite strands would cause catastrophic noise in any subsequent association testing. Interestingly, although we were encountering significant noise levels, when we tested our data, we found that not all of the genotypes were on opposite strands, the level fluctuated at around 50%. We ultimately determined that our data, although sometimes on the opposite strand, was usually designated in a different scheme altogether. Two datasets that are encoded to TOP and FWD will have, by chance an overlap of variants that match and do not cause errors thus obscuring the real problem. We found this to be the most difficult challenge that we overcame during our development of REMEDY.

We next needed a method to transcode between the different schemes so that our data could be converted into the same designation scheme and strand for merging. Given that there was very little in the way of information on strand conversion, this was more challenging than perhaps it should have been. This was made more frustrating by the discovery that the Illumina tool, Genome Studio was able to convert between all schemes, yet it was closed source, so the solution was obscured. After several early software versions, we found a key clue in the Illumina manifest file - an undocumented legend hidden within the ID of the variant. This small-scale Rosetta stone enabled us to convert to and from the Illumina DESIGN scheme to TOP/BOT and FWD/REV and thus to and from TOP/BOT and FWD/REV by using DESIGN as a middle-man.

After finding a solution to the strand designation problem, we then encountered more frustration as our data still had large-scale noise problems. We subsequently identified that the cause of this was that some source datasets had their strand misreported in the documentation. We expanded REMEDY to counter this by including a major sub-program that's function was to scan the manifest and genotyping data to auto-detect the source strand designation scheme.



To detect strand designation, REMEDY scans the manifest for each variant and calculates the TOP/BOT/FWD/REV encoding for the reference and alternate nucleotides; it then compares it to the genotyped alleles in the input data file. If the genotype matches one of the encoding schemes, a simple count tally is kept for that encoding. As the file is scanned, separate distributions over TOP/BOT and FWD/REV are calculated as percentages in real-time. The percentages can be interpreted to infer the source encoding of the genotyping file. For example, a file that was FWD encoded will have greater than 95% of the genotyped alleles matching either the reference or alternate of a bi-allelic variant that has been FWD encoded. In the same way, genotypes that have a >95% match for TOP/BOT/REV will be encoded to those schemes respectively. If, however, there is a partial match across all encoding schemes then that can be interpreted as evidence that the source file was encoded in the DESIGN scheme; this is because DESIGN encoded data is encoding unspecific and is instead designated to the original design strand of the probe.

The strand auto-detection algorithm is novel to our knowledge and represents a significant improvement in the field towards properly managing the merging of disparate genotyping datasets. REMEDY does not currently convert between the POS/NEG encoding scheme as this scheme is more difficult to define than the others. POS/NEG may refer to the biological 5' and 3' strands, but it may also refer to the HapMap project [186] initial attempts to produce a stable strand designation for a genome build. We found there was no real way to distinguish between the two and that the scheme was uncommon; therefore, we omitted this from the final functionality of the program.

### **4.1.3 Re-coding Evaluation**

The final sub-program we developed for REMEDY used the results from the manifest scan and the genotype strand designation scan to perform the actual re-coding and filtering steps on input data before outputting to a common file format, Variant calling format (VCF). We separated this stage of REMEDY from the others so that the initial two scanning stages could be run without converting any data for diagnostic purposes. We also favoured modularity in our software design so that different portions of the program could be built into pipelines if needed. The sub-program was broken into three parts; we first employed our proprietary algorithm for converting between strand encoding schemes to re-encode from a source to a destination scheme (supplied by the user). Variants are then filtered based on the manifest scan so that only those deemed valid for GWAS are included, we then finally convert all data to the VCF file format.

The decision to convert files to VCF was primarily based on the fact that both SNP & Variation Suite (SVS) [188] and PLINK [253] (the main GWAS software packages) both

import and process data in the VCF file format. Rather than re-inventing the wheel, we also wanted to integrate with the existing software ecosystem by merging data after processing by REMEDY using other tools. VCF-tools contains several functions to merge VCF files reliably and is considered an industry standard; therefore, we selected VCF as our output format. In addition, we also wished to distance ourselves from the Illumina proprietary file formats, preferring open source alternatives. It is a limitation of REMEDY that VCF is the only output file format, in future versions, we would wish to include more outputs such as Illumina Matrix and simple tabular formats.

The VCF file format has two compressed sub-formats which allow for faster processing. The tabindex format compresses a VCF file to a tarball [254] file before indexing the contents in a separate tabix file. The Binary calling format (BCF) format is a binary representation of the VCF file design for fast processing by glsbcf-Tools [255]. We found that after testing with PLINK and SVS, that although they took VCF as an input format, they required the VCF files to be either in glsbcf or tabix formats. REMEDY provides no functionality to output to these formats; however, as part of a pipeline, one would be able to add in functions to convert the raw VCF files into a compressed format for further processing. In the interest of not re-inventing the wheel, we felt that it was more correct to integrate with other tooling rather than provide functions for converting to compressed formats. We recognise, however, that if REMEDY was used to process very large files, the ability to compress VCF files to GZIP format would be useful to save on storage space during pipeline processing.

#### **4.1.4 REMEDY-DB Evaluation**

REMEDY-DB was developed in response to the requirement to integrate dbSNP data into the REMEDY workflow so that more stringent filtering strategies could be employed using the additional variant meta-data. As mentioned in our results, we evaluated several strategies for obtaining and processing dbSNP data before developing REMEDY-DB. Given the number of variants present on modern microarrays, requesting data for each variant from National centre for biotechnology information (NCBI) via an internet connection would have been prohibitive. We were aware of the NCBI dbSNP batch query service [256], however, at the time of development the service was scheduled to be replaced and its successor, an Application programming interface (API)-based service was still in the implementation phase where it was unclear what the performance characteristics of the new service would be.

After ruling out online queries, we were left with several options for interfacing with a downloaded representation of dbSNP. The primary candidate was a direct download of the dbSNP MySQL database that contained all dbSNP information, we also located a reduced

version of the database, pared down to critical information only in the UCSC browser. We first assessed interfacing with the MySQL database but found querying to be complicated and slow, we, therefore, favoured the UCSC single flat-file as our input data source as it contained all the data required to perform the extended filtering REMEDY required in a single table.

We also found a paper detailing a web-based package that provided an interface to the dbSNP SQL database [257]. Unfortunately, the detailed links were non-functional (the tool was no longer maintained) and we did not assess this further given that we could not download the software.

The primary function of the dbSNP interface was search and retrieve. The design specification required submission of an RSID and retrieval all data regarding a variant in the fastest possible time. The simplest form of search is the linear search where items are sequentially traversed in a list until the correct item is found; this at worst takes  $\mathcal{O}(n)$  comparisons where  $n$  is the total number of items in the list. Given that the dbSNP database currently contains over 650 million records [258] and the largest microarray used for the SSNS project has over 1.7 million variants on it, a linear search for a single microarray would require at worst  $1.105 \times 10^{-15}$  searches. At the time of writing, the fastest solid-state drives deliver 500,000 random read operations per second [259]. In the best case scenario, where performing one search comparison requires one read (a simplification), this would take 613,888 hours or 1,681 years to complete. It is possible to load the table into memory for faster searching in RAM; however, the memory required for this would also be infeasible to implement.

NoSQL databases have been shown to be better than relational databases in storage and retrieval of genomic information [260]. To solve the search problem, we designed a custom on-disk, NoSQL database that employed an efficient binary search strategy. Binary search is a search algorithm that finds the position of a target value within a sorted array via a divide and conquer strategy [261] [262]; it has a computational complexity of  $\mathcal{O}(\log_2 n)$  where  $n$  is the total number of items in the list. In terms of the timings shown above, the worst case number of operations would be  $1.498 \times 10^{-7}$ , 13 orders of magnitude better than a linear search in this situation. We implemented a binary search algorithm using a Binary Tree, which is an efficient option to represent a binary search in programming as the connections between branches on the tree are created using pointers; each bifurcation on the tree represents a stage of binary search. The leaf nodes on the tree were linked to file positions where the data for the variant resided. We then constructed a simple payload for storing variant data that allowed variable length variants to be sequentially written by including the length of the next payload in a fixed length header. The data was encoded in JavaScript object notation (JSON), a standard programmatic data format.

We believe the file-based binary tree representation of dbSNP to be novel in the field and provides a high-speed search database for variant data which is unparalleled. NCBI has recently released a new batch API service based on up to date API technology; however, while convenient, it will always be slower than direct access to file data. REMEDY-DB follows the web paradigm concept of local caching. There are several web-caching databases such as CouchDB [263], where their role is to provide local fast access key-value store of information that sits in front of a more complex database. We view REMEDY-DB as a similar database, providing a fast access point to dbSNP without the performance considerations of a web connection or a relational database.

The primary limitation of REMEDY-DB is that it is dependant on new versions of both dbSNP and the UCSC flat-file table representation, thus requiring a level of maintenance. The database could be improved by the removal of the dependency on UCSC, by building it directly from the dbSNP MySQL database. In addition, this would enable access to more detailed meta-data regarding a variant, thus enabling the potential for more features in REMEDY.

In terms of the database storage footprint, improvements could be made to the storage of the variant payload. JSON is structured and therefore has a level of redundancy in the format of the text; if the variant payload were designed as a sequential list of data for each field of information it would be smaller as the formatting constraints of JSON would not be required. Further compaction would require compression of the text data which would severely impact read performance.

Another potential area of improvement for REMEDY-DB lies with the search algorithm and indexing of data. Binary trees, while efficient to represent in computing terms, are an imperfect binary search as the search tree has the potential to become unbalanced; this leads to slightly worse performance than the theoretical maximum. Hashing is a well-used search and indexing method that is proven to be better than binary search in problems where best-guess matches are not allowed [264] [265]. In dbSNP, a search for an exact match to rsID is required, therefore, using hash tables instead of Binary Tree's while more complicated to implement, may offer better performance. While hashing's worst-case complexity is  $\mathcal{O}(n)$ , the average is  $\mathcal{O}(1)$ , making it on average faster than binary search [266].

A more popular method for computing based indexing and search is the B-Tree [267] [257]. The B-Tree is a generalisation of a binary tree where nodes can have more than two children, and the tree is self-balancing. B-Tree's take advantage of the way in which storage disks operate to improve search times; algorithms use sparse indexes across disk blocks to reduce the amount of disk that needs to be searched. B-Trees improve on the

search time from  $\mathcal{O}(\log_2 n)$  to  $\mathcal{O}(\log_b n)$  where  $b$  is known as the blocking factor. Swapping the REMEDY-DB search algorithm to a B-Tree based approach would most likely have a considerable positive performance impact on search times for variants.

#### 4.1.5 Software Performance

We conducted performance testing on REMEDY using a high-end desktop computer running Windows and Linux in a dual boot configuration with a Samsung 950 Pro solid state drive, the full specification can be found in the appendices A.1, p.280. All three REMEDY sub-programs were testing using a subset of data from the SSNS GWAS project. The data-set consisted of seven samples which were genotyped on the Illumina Infinium Multi-Ethnic Global microarray detailed in the SSNS case study section.

We found the performance of REMEDY to be most affected by choice of storage drive; this is unsurprising considering the most processing is done by REMEDY-DB when matching variants stored on-disk. The difference in run-times for operations between a solid state drive versus a standard magnetic drive was several order of magnitude faster. We would therefore strongly recommend using a solid state drive for any REMEDY processing.

#### 4.1.6 Oxford Pipeline Comparison

The Oxford pipeline was an unpublished software tool for re-coding genotyping data developed by the Wrayner group at Oxford University [191]. The tool is split into two parts; the first generates a conversion file set using the input manifest file of the target microarray and the second uses the conversion file to convert a genotyping file in Illumina TOP format to a user-specified genome build in FWD designation. To build a conversion file, each probe sequence from the selected manifest is matched to a target reference genome using the Blast-like alignment tool (BLAT) to obtain the genomic position and biological strand. The program assumes the input data is in Illumina TOP format and then applies a standard conversion from TOP to FWD using the strand designation from BLAT and the TOP alleles of the manifest file. Conversion files for most Illumina and Affymetrix microarrays and genome builds 35, 36 and 37 are available. The website also states that more can be generated on request. There are some significant differences between REMEDY and the Oxford pipeline, which we will refer to as the OP from now on. We discuss these in detail below.

To transcode between strand schemes, an algorithm must know the start strand, the destination strand and a method for re-coding between the two. The OP achieves this

through the requirement that all input data be encoded to Illumina TOP while the output is fixed to FWD, thus eliminating the need to transcode between multiple schemes. Strand re-coding is achieved using BLAT to map the probe sequence to a strand on a target genome build which allows re-mapping of the TOP encoded alleles to FWD. The use of BLAT allows the correct genomic position to be outputted for any genome build, essentially performing both a lift over and re-coding at the same time. REMEDY uses the strand information embedded within the manifest file and a target version of dbSNP to transcode between different designation schemes.

Illumina provides strand designation for the genome build for which the microarray was designed for, hence merging data from multiple microarrays on different genome builds becomes impossible. The OP solves this by calculating the strand for a target genome build using a custom BLAT pipeline that is internal to the program. REMEDY obtains genome build specific data and strand information from dbSNP through REMEDY-DB.

While both methods of transcoding between strand designation schemes are valid, the OP requires that all input data be in a single scheme which is left to the user to calculate. This requirement presents potential problems as many datasets are not in Illumina TOP format before re-coding, yet it provides no solution other than a custom conversion file being generated by the OP team on request. REMEDY was developed to solve such situations; this reveals a fundamental difference between the two tool sets. The OP was developed with correcting lift over to different genome builds as the primary goal with strand re-coding as an additional feature. REMEDY was designed with strand transcoding and quality filtering as the primary goals; this has led to different design priorities during the development of the tools.

The OPs reliance on the BLAT pipeline forces a new conversion file to be generated for every microarray/genome build required, hence generating a significant maintenance cost associated with managing the tool. Furthermore, if the OP supported transcoding between multiple schemes, a new file would need to be generated for each microarray/genome build/source strand/destination strand combination. REMEDY uses the reference file directly rather than generating an intermediate conversion file and can transcode between TOP, BOT, FWD and REV; however, it is not entirely maintenance free. REMEDY-DB must be generated against a target dbSNP build; thus, for each target dbSNP build, a new database must be generated. While this allows the latest version of dbSNP to be used (the OP uses the manifest file dbSNP version), it does require extra maintenance. We have mitigated this by providing tooling to create new databases from the UCSC flat file packaged with REMEDY as a utility program; this allows users to create a custom REMEDY-DB against any target dbSNP version without contact with the REMEDY developers.

The reliance on custom generated conversion files, and the fixed TOP->FWD transcoding was one of two primary drivers behind the choice to develop new software rather than using the OP for our in-house GWAS. The second was that the OP only provides lift over and re-coding. Although strand designation mismatching was our most significant source of noise that we encountered, other causes such as structural variants, multi-allelic variants and poorly designed probes were also issues. REMEDY seeks not only to correct strand designation but also to perform quality control using the manifest file and dbSNP. A feature comparison table between the OP and REMEDY is shown in Table 4.1, p.242.

Feature	Oxford Pipeline	REMEDY
Core strand designation reference	BLAT	dbSNP
Method of strand conversion	Manifest + BLAT	Manifest + dbSNP
Supported transcoding pairs	TOP->BOT	TOP/BOT/FWD/REV
Required conversion file for re-coding	Custom file	Manifest file
Output format	Custom	VCF
Auto-strand scheme detection	✗	✓
Supports lift over	✓	✓
Support for Illumina microarrays	✓	✓
Support for Affymetrix microarrays	✓	✗
Supports multiple dbSNP versions	✗	✓
Windows Support	✗	✓
Linux Support	✓	✓
Structural variant filtering	✗	✓
Multi-Allele filtering	✗	✓
Strand mismatch filtering	✗	✓
Invalid position filtering	✗	✓
Deleted variant filtering	✗	✓
Multi-position alignment filtering	✓	✓

**Table 4.1:** REMEDY and Oxford pipeline feature comparison table.

We performed a direct comparison between the OP and REMEDY using the same test data set detailed in the performance testing section above. REMEDY was not provided with a list of variants to filter consequently, effectively switching off the variant cleaning parts of the software tool leaving just the re-coding. The seven sample test file was inputted into both pipelines for conversion from TOP to FWD. The output files had a match percentage of 99.75%. We then assessed the 4227 non-matching variants to identify the cause of

the mismatch. 50% of the differences were attributed to differences in the strand calling between the manifest and the OP BLAT pipeline and 50% were due to inconsistencies between the OP BLAT pipeline and dbSNP. During the development of REMEDY, we also encountered strand mismatches between the Illumina alignment process and dbSNP. The testing setup above adds in the OP alignment process as a, which is the reason that two different classes of error are seen in the results. We conclude that these variations are due to differences in the alignment pipelines between Illumina, the OP and dbSNP; therefore, we can treat the outputs of the OP and REMEDY as equivalent.

In summary, the OP pipeline and REMEDY both perform strand re-coding to equivalent outputs but with different internal pipelines. The OP's reliance on BLAT for alignment overall leads to a large maintenance cost and limited transcoding options. REMEDY's reliance on dbSNP produces a lower maintenance cost overall, which we have significantly mitigated by publishing the database generation algorithms; this also allows for transcoding between all strand designation schemes. REMEDY also provides a much richer set of features for filtering variants for GWAS based on meta-data from both the manifest file and dbSNP. Perhaps REMEDY's most substantial advantage is its ability to auto-detect the source strand designation and encoding scheme of the input data. When combined with the capability to transcode between all major encoding schemes it essentially negates any potential source of error from strand encoding when merging two disparate genotyping datasets.

#### **4.1.7 Future Work**

REMEDY is feature complete and tested to a high standard. It has already been used in two different GWAS, one of which has been published [1] and one is in pre-publishing; both are published or targeted to their respective top of field journals. However, the various minor improvements suggested in the evaluation section, such as options for auto compressing to VCF tabix format could be packaged up into a new 'quality of life' release to improve usability. One of the significant limitations of REMEDY currently, is that it can only process Illumina microarrays; the inclusion of algorithms to process Affymetrix microarrays would be a notable improvement to REMEDY.

REMEDY at its core relies on the flat-file generated by UCSC browser that targets a specific version of dbSNP. As discussed in the evaluation section, removing the dependency on this file by building the REMEDY-DB directly from the dbSNP MySQL database would remove a significant point of failure from the REMEDY maintenance plan. Also, this would enable additional features based on the richer source of variant data such as the ability to distinguish between deleted and merged variants. While a major programming project, we conclude that this would be a significant boon for the REMEDY program and



would constitute our next major release.

Utilising the full power of dbSNP would also enable more complex quality control filtering to be developed and tested. We, for example, propose a variant quality score for suitability to a GWAS based on the number and quality of submissions for the target variant. Researchers may wish to distinguish between a variant which is based on 100 submissions from major projects versus a variant which has one submission from an independent group. Other data, such as ethnicity frequency could also be used to filter variants for an ethnic-focused study. dbSNP also provides links to other NCBI databases such as ClinVar to provide additional filtering possibilities based on disease status, for example.

REMEDY-DB is a powerful tool in its own right. Despite being developed to sit alongside REMEDY, there are currently no high-speed, document-based versions of dbSNP available. For users developing programs that require fast, programmatic access to variant data, REMEDY-DB would be a compelling alternative to working with the raw MySQL database. We propose to develop REMEDY-DB into an independent software package, that self-updates from dbSNP via either the downloaded database or through the web API to provide programmers with a robust library for accessing variant data in C#.

GWAS are a type of positional cloning that also includes linkage studies. Linkage analysis performs positional cloning via the study of families; however, it is likely that variants which are unsuitable for GWAS are also unsuitable for linkage. It is possible that REMEDY could be tested on linkage studies to assess if any improvements to the output can be made by filtering out unwanted variants.

Finally, while genotyping is currently the most ubiquitous method in the field, WGS is the new gold standard for sequencing. REMEDY in its current form is incompatible with WGS given that many of the filtering and strand re-coding techniques are designed specifically for handling genotyping data. WGS, while having its own set of problems does not suffer from many of the hurdles corrected by REMEDY. Genotyping can still be used to potent effect in rare, Mendelian disease genomic analysis, for a much lower cost than WGS; therefore, we expect the requirement for REMEDY to exist for the foreseeable future. Furthermore, by separating REMEDY-DB into an independent toolset, we believe that this will be useful for many more types of genomic analysis beyond positional cloning.

## 4.2 LOOPER

### 4.2.1 Loop Prediction Evaluation

The loop prediction portion of LOOPER was the first set of scripts to be developed. At the time of development, chromatin conformation experiments were relatively new, with sparse amounts of low-resolution Hi-C data available for analysis [237]. While useful in determining genome-wide Topologically-associated domain (TAD)s and the basis for CCCTC-binding factor (CTCF)/Cohesin structural loops, this data was unsuitable for exploring mega-base sized regions of interest; we, therefore, sought to develop a prediction algorithm for different types of chromatin looping in the HIPKD region of interest. We opted to focus on two types of loops: the structural CTCF/Cohesin loop because it is the most well-characterised chromatin loop and the functional ZNF143 loop, given that our mutation resided in a binding site for ZNF143.

#### 4.2.1.1 CTCF/Cohesin Loop Prediction

The current best hypothesis regarding CTCF/Cohesin is the loop extrusion model which proposes that DNA is extruded through pairs of convergent CTCF Transcription factor binding site (TFBS) bound by cohesin until the CTCF sites meet and form a loop anchor complex [75]. The resultant DNA trapped in the 'pinch' is known as the loop. We do not discuss the merits of this model of loop formation in this thesis; we take it to be canon.

We hypothesised that a reasonable model for predicting potential loops could be constructed by identifying convergent CTCF sites which had some evidence that cohesin had also been in contact. We selected ChIP-Seq data as our primary source of TFBS data given the large amount of data available for the experiment through the ENCODE project [234], where the experimental methodology for ChIP-Seq and the strict adherence to data cleaning and management was considered to be best in the field.

To predict structural loops, we required both CTCF and Cohesin ChIP-Seq data. Ideally, performing ChIP-Seq analysis for CTCF and the Cohesin sub-units SMC3 and RAD21 on cells in different tissues obtained directly from patients with HIPKD would have been optimal; however, this was infeasible for our project. Hence, we sought to identify cell lines within ENCODE for ChIP-Seq that fitted with the observed symptoms for HIPKD; notably in the kidney, pancreas and liver. Unfortunately, no kidney or pancreatic cell lines were included in the March 2016 ENCODE release for CTCF or Cohesin ChIP-Seq data;

furthermore, HEPG2, an immortalised human liver cancer cell (the closest potential candidate cell model for HIPKD) only had CTCF data available and thus was unsuitable to be used for structural loop prediction.

Given the lack of cell-specific data, we opted to generalise our loop prediction model for structural loops by selecting ubiquitous CTCF and Cohesin ChIP-Seq sites based on all available cell lines in ENCODE. We recognise this as a limitation of our methodology as this model only includes highly conserved CTCF and Cohesin sites when the HIPKD disease model is tissue-specific. This effect was somewhat mitigated in the HIPKD ROI as the size of the region was relatively small, and there were no tissue-specific CTCF loop anchor sites anywhere; we, therefore, do not believe that our results were expressly affected in this project by the exclusion of cell-specific TFBS.

To increase the accuracy of the ChIP-Seq binding data and reduce false-positives, we opted to scan the TFBS sites for binding motifs so that we only included CTCF sites which had both a statistical and experimental backing [268] [269]. While ChIP-Seq can determine the presence of a transcription factor, it does not provide evidence of actual binding to DNA. As we have seen in our analysis, cohesin likely indirectly binds through CTCF to DNA, but this is indistinguishable in the ChIP-Seq data. Consequently, we calculated Position weight matrices (PWMs) for our target transcription factors and used them to detect any probable binding sites within the ChIP-Seq predicted binding site in order to separate actual DNA binding from residual binding through other proteins.

We recognise a limitation in this approach as the data used to generate the PWMs was the same source of data that the experimental binding sites were calculated from, hence introducing potential bias. Despite this drawback, we believe this approach still helps to remove false-positive binding sites as sites where indirect binding or where residual protein was present in the formaldehyde fixing complex are removed due to there being no statistical predicted binding site present in these sites.

PWMs also suffer from some principle drawbacks [270]. They are at their core, a simple statistical model of the frequency of base pairs at a position in a motif; consequently, they oversimplify actual protein-DNA binding processes. The molecular dynamics of protein interactions are not considered as each nucleotide is considered as independent from its neighbours. Furthermore, PWMs do not consider that Transcription factors (TFs) may bind to multiple motifs and to motifs of varying lengths. They are, however, a useful tool for separating parts of DNA that have a higher likelihood for binding than their surrounding areas and the PWMs themselves are calculated using experimental results from assays such as ChIP-Seq i.e. they are not statistical models based purely on theory. Given their simplicity, they are a useful tool for *de novo* motif discovery if one is aware of their specific

limitations.

Our loop prediction algorithm was based on the loop extrusion model where we identified pairs of convergent CTCF sites with SMC3 and RAD21 residue (indirect binding identified by a ChIP-Seq signature with no binding motif) that represented probable loop anchor sites. As we used no experimental evidence to confirm the loops, we enumerated all possible loops from all valid loop half-anchors; while not incorrect, we accept that not all predicted loops will exist. To increase the loop prediction specificity, we could indeed include some measure of likelihood; we propose an improvement to our loop prediction algorithm where the strength of the ChIP-Seq binding sites and the p-value of the PWM binding motif are included in a measure of loop strength.

Our results showed several sets of overlapping structural loops present in the HIPKD ROI. One of the loops was later confirmed experimentally using ENCODE data; this was our most favoured loop based on our hypothesis and provided evidence to support the accuracy of our loop prediction model.

#### **4.2.1.2 ZNF143 Loop Prediction**

The exact mechanism of ZNF143 mediated chromatin interactions is still poorly understood [76]. Rather than developing a specific algorithm for detection, we opted for a simpler approach by distinguishing between direct and indirect binding. As described in the last section, indirect binding is characterised by ChIP-Seq data with a binding motif where indirect binding can be defined by weak ChIP-Seq binding with no binding motif. In the immediate HIPKD ROI, only one ZNF143 site was detected, with multiple binding motifs within it. We then detected indirect binding in several key predicted loop anchors, thus supporting evidence in the literature to suggest that ZNF143 mediates contacts between promoters and loop anchors [76].

The evidence that we detected for ZNF143 mediated interactions is circumstantial; much more experimental evidence would be required to define further how ZNF143 mediated chromatin contacts form and what kinds of algorithms could be developed for prediction. We do believe, however, that the presence of indirect ZNF143 at so many key loop anchors while observing a singular strong ZNF143 site in a promoter is compelling in convincing us that the transcription factor has a role in 3D chromatin architecture.

## 4.2.2 Loop Visualisation Evaluation

After the initial LOOPER analysis for HIPKD was completed, the volume of available chromatin confirmation data increased dramatically. New Hi-C experimental data and analysis tools combined with new experiments such as ChIA-PET made the prospect of using actual data rather than predictive models to analyse an ROI a reality. This is not to say that the predictive part of LOOPER is obsolete, there are still many cell lines and loop types for which experimental loop data still does not exist, leaving a space for loop prediction algorithms to be still useful; however, a window for visualisation tools based on experimental chromatin interactions was open. Many new visualisation tools were developed by other groups, but from previous research, our team favoured using force-directed graphs to show connected data, but no such tools existed; consequently, we sought to develop in-house software to visualise chromatin interactions as a force-directed graph.

### 4.2.2.1 Chromatin Interaction Experiment Selection

We selected ChIA-PET as our data source for chromatin interactions due to the finer-grained interaction analysis afforded by targeting specific transcription factors; this is achieved by using chromatin immunoprecipitation, cross-linking and next-gen sequencing [195]. In addition to the higher resolution that ChIA-PET provides over Hi-C, the ability to target specific transcription factors enables selection of the type of interactions that are to be analysed. For example, to target CTCF/Cohesin loops, one may target CTCF and RAD21 in two separate experiments and then combine the data while to target a Yin-yang 1 (YY1) mediated loop, antibodies for YY1 could be used to pull down only those interactions specific to those loops. Selective targeting of different loop types allowed for greater control over bioinformatic analysis, which we utilised in building layers of interactions into our output graph.

The selection of ChIA-PET also allowed for close integration into existing ChIP-Seq data as the experimental protocols have some fundamental similarities in their use of chromatin immunoprecipitation; hence binding sites identified with ChIP-Seq can be closely correlated with the interaction sites of ChIA-PET data. The straightforward merging of these two sets of experimental data allowed us to accurately annotate the ChIA-PET interaction sites represented as nodes in our graph; we believe this was key to making the force directed graph understandable.

#### 4.2.2.2 Source Data

At the time of analysis, there was a limited set of ChIA-PET data available in the ENCODE project. The only viable cell line with enough usable data was K562 (human immortalised myelogenous leukaemia cells); we accept this as a severe limitation of our HIPKD analysis given that cancer cells have been shown to have markedly different chromatin architectures with respect to healthy cells [271].

The K562 ChIA-PET data repository contained several useful experiments for studying both structural and functional loops. Initially, CTCF and RAD21 data for K562 were selected for analysing structural loops; however, the RAD21 experiment had very few actual interactions after processing *via* Mango [219]. We believe this was due to poor experimental quality and resulted in a key transcription factor for structural loop identification being unavailable for processing by our pipeline. We consequently had to rely on CTCF data only; CTCF has a number of functions in the genome but has to date only been shown to interact with cohesin at loop anchors [237]. Removing cohesin detection from the interaction model disabled a key filter in excluding non-looping CTCF interactions. We accept this as a limitation, but we deemed this as acceptable given that CTCF to date has only been shown to participate in chromatin interactions with regards to forming chromatin loops.

Locating source data for studying ZNF143 functional interactions proved to be challenging as no ChIA-PET experiments existed for ZNF143 at the time of analysis. Instead, we hypothesised that given the literature supported a model of ZNF143 mediated interaction between promoters and *cis*-regulatory elements, that identifying contact sites between promoters and regulatory elements from ChIA-PET data and then filtering for ZNF143 specific loops using ChIP-Seq data, would give a reasonable estimate of the true chromatin architecture. To identify promoter interactions, we used DNA Polymerase II (POL2) data to show points of contact between actively transcribing genes and *cis*-regulatory elements. Of course, identifying actual ZNF143 ChIA-PET would be preferable, and we recognise this a limitation of our HIPKD ROI analysis.

#### 4.2.2.3 Graph Generation Algorithm Evaluation

Our graph generation algorithm integrated several sources of data to produce output files for nodes and links in a graph that was in the correct format to be loaded by Cytoscape - our selected visualisation program. In the first iteration of the LOOPER graph algorithm, we sought to display ChIA-PET data only with the contacts anchors as nodes and the edges as the contacts derived from the output of the Mango pipeline [219]. The primary

challenge we faced was how to aggregate the contacts together to produce a graph that was understandable to the end user. The Mango output format produces a contact for each line but does not perform any aggregation of contacts; therefore, we first grouped the contacts by location using the Mango internal ID system to merge the data points.

The resultant graph contained a large number of contacts that were unreadable due to the amount of information displayed. Reducing the field of view was not possible as we required a view of the whole HIPKD ROI; therefore we developed implemented a hierarchical clustering algorithm in order to group the contacts under a threshold distance together as one node. The simple algorithm recursively merged the closest nodes together, until no two nodes were under the threshold distance for merging. Having a merged set of data both in terms of nodes and edges presented two challenges: how to properly annotate merged nodes and how to show the relative strength of merged contacts on an edge.

For calculating the strength of a set of merged contacts, we identified three key pieces of information: sequencing depth ratio (between the paired-end contacts), total sequencing depth and PET support. We borrowed from data science practices by converting all values into a normalised value between 0 and 1 before combining them as a weighted sum after cleaning. Developing the formula for interaction strength was one of the most challenging areas of developing the graph generation algorithm.

The methods of normalisation for each value was selected based on the statistical characteristics of each value. The depth ratio as a bounded mix of two read depths required only simple min-max normalisation [272]; however, when calculating both the total read depth and PET support, we encountered several zero values which, if min-max normalised, would have caused infinite numbers. Instead, we employed a standard mathematical technique to avoid dividing by zero by placing all values in the pattern  $\frac{1}{1+x}$  where  $x$  is the value to be normalised (Equation 3.2, p.194). For the total read depth, we were faced with significant outliers in our dataset, we, therefore, sought to 'squash' these down before they were used in normalisation by taking the log of the raw value. While no novel mathematical techniques were used here, we believe our strength formula as a whole to be novel because we incorporate the depth ratio into a pair of contacts; thus, we only give high strength to contacts which have a read depth on both sides. This is not represented in any other work to our knowledge. We represented the calculated strength as the thickness of the edge as this is ubiquitously recognised in graph representations as a visual measure of strength.

We solved the challenge of annotating merged nodes by using meta-data downloaded from UCSC browser from the Chrom-HMM project [205]. The data allowed annotation of genomic segments as several categories, including enhancers and promoters. Where

different genomic segment types had been merged, we developed a colouring system that distinguished these from regions that were all enhancer or all promoter. We believe that the Chrom-HMM annotation gave critical visual feedback to the user regarding the map of enhancers and promoter interactions.

#### 4.2.2.4 Case Study Graph Evaluation

Graph analysis of the HIPKD ROI revealed that the broader TAD within which the *PMM2* bi-directional promoter is located had a complex regulatory network made from distinct areas of interaction between CTCF and POLR2A contacts (Figure 3.36, p.212). The regulatory network had two distinct sections with only weak CTCF interactions between the two; these regions were also separated physically in the TAD, with the right-most genes occupying the left of the graph and the centre and left areas of the TAD (containing *PMM2*) occupying the right section of the graph. We shall only discuss the right side of the graph directly related to *PMM2* in the following paragraphs.

The CTCF network on the right of the graph (in blue), shows a highly interconnected set of loop anchors, with no CTCF contact sites showing without a cohesin signature, confirming that in this genomic region, CTCF mediated contacts are most likely all from CTCF/Cohesin structural looping. Analysis of the CTCF sites in our prediction model showed many possible loops, and it is clear from the graph analysis that several loops interconnect and share common loop anchors within the region. Several recent papers proposed a nested loop configuration of chromatin when multiple CTCF loops appear to be present [72] [73]; we propose that a set of nested loops is present in this TAD region, resulting in many loop anchors bunching up into one large loop super-anchor.

Analysis of the *PMM2* promoter shows a strong interaction with the super-anchor; this is further confirmed by weak ZNF143 residue being present on several key loop anchor points. We argue that the *PMM2* promoter is being pulled down to the super-anchor to interact with *cis*-regulatory elements clustered in this region. Evidence for this is supported by the 3DIV program [248] showing a liver-specific super-enhancer that has strong links to the super-anchor, suggesting that the loop configuration places this element at the base of a loop, in proximity to the *PMM2* promoter (Figure 3.46, p.223).

Without representing the ChIA-PET data as a force-directed graph, we argue that the observations made above would have been difficult to ascertain from other software tools given the highly connected nature of the genomic elements in the HIPKD ROI. Our assessment of the LOOPER visualisation package is that it provides a novel and critical view of regulatory action, which was essential in exploring the potential regulatory pathways of



*PMM2* gene expression.

#### 4.2.2.5 Comparison with software ecosystem

The HIPKD project identified a point mutation in a putative ZNF143 TFBS that was hypothesised to interact with other regulatory elements in the local TAD to regulate the *PMM2* gene. During the initial analysis, it became clear that the location on the linear genome was largely inconsequential to the analysis of the looping structure of the chromatin interactions. We identified that the primary data points for looping analysis were properly annotated loci (nodes) and their interactions with each other (edges).

LOOPER was developed out of a requirement for a visualisation which concentrates on the pattern of interactions, rather than the mapping of loci to other loci on a linear genome. We applied force-directed graphs to map the annotated nodes and interactions in an easy to visualise way using colours to show annotations. Applying these graphs to the HIPKD region of interest, we were able to see the patterns of interaction between the ZNF143 TFBS and the rest of the TADs interaction structure.

When compared with other tools assessed in the results section (3.2.7.3, p.221), there are several key differences that set LOOPER apart, but that also show potential synergies with other existing tools for a more complete analysis. The tools mentioned in the following paragraphs are already summarised in Table 3.11, p.222.

One of the key enablers that allows for fine-grained interaction analysis is the ChIA-PET experiment. The majority of tools for interaction visualisation focus on Hi-C, which shows coarse-grained interactions with a resolution of approximately 5Kb for all proteins. This kind of high-level analysis is excellent for defining genome-wide structures such as TADs. We found the identification of TADs to be essential in bounding our analysis to a specific region; experimental data is inherently noisy, therefore, restricting the analysis to a specific region can reduce the number of interactions for analysis. Hi-C is unable to detect finer grain interaction sites in the same resolution as other non-interaction based experiments such as ChIP-Seq which can detect sites in the region of 400bp.

ChIA-PET uses chromatin immunoprecipitation to pull down proteins of interaction and is, therefore, has a much higher resolution that makes it ideal for visualising interactions in a TAD. Perhaps because ChIA-PET requires selection of a specific protein and because of the relative lack of experimental data in the public domain, there are many fewer tools able to visualise ChIA-PET when compared to Hi-C. As a result of the HIPKD study, we wished to concentrate on the CTCF loop and the functional ZNF143 loop; therefore, there was

a clearly defined set of proteins which we required analysis for; this made ChIA-PET the clear choice for interaction analysis. In summary, LOOPER already sets itself aside from many other interaction tools as a virtue of the input data type that it processes, allowing for fine-grained visualisation of intra-TAD chromatin interactions.

When comparing the type of interactive visualisation which is generated, we believe LOOPER is unique in this space. All other tools that we assessed attempt to map interactions to a linear genome so that annotations can be easily added and so users are grounded in the genome position of the interaction loci. We argue that by tying the visualisation to a 1D line, much of the nuance of the interaction pattern is lost. Force-directed graphs are generated procedurally such that the graph attempts to keep strongly interacting, highly connected items close together. When applied to ChIA-PET data, this results in strongly interacting genomic elements occupying the same section of the graph regardless of their actual genomic position. LOOPER also provides options to generate a graph that is halfway between abstract interactions and absolute genomic position by including 'dummy' interactions between elements that are physically close to each other. In this mode of operation, LOOPER produces a hybrid graph where the position of each node is a weighting of the strength of interaction and physical position; this is also unique in the field.

Displaying interactions as a force-directed graph allows for the strength of the interaction between two loci to be intuitively displayed; this is something that is much more difficult when using a linear genome for reference. 3DIV provides features to filter out lower strength interactions and tools such as GIVE use strength of colour and opacity to give the same effect; however, we believe the thickness of the interaction line as well as baking the interaction strength into the graph generation algorithm itself produces much more informative visualisations. In summary, LOOPER is unique in the field as it allows for loci-free interaction patterns to be visualised.

In terms of computing knowledge required and ease of use, LOOPER has an immediate disadvantage, as it is not polished into a browser and is in reality still a collection of scripts. The tools we assessed had varying levels of computing skill required ranging from a full python pipeline that is required to be run from the command-line to a fully hosted browser-based tool that simply required knowledge of the experiment performed to interpret the results. We would say LOOPER sits at the bottom of the list because some of the file paths are hard-coded into the script thus requiring some alteration of code to be able to generate the required results. This is a key disadvantage of LOOPER; to improve this, we would seek to first parameterise the scripts so that they could be run from the command line and eventually integrate the graph generation into a browser tool using existing force-directed graph javascript libraries.

One of the key factors that we found affected usability, was the availability of data. The majority of tools we assessed, including LOOPER required input data to be able to function. This adds significant amounts of effort on the user side before interpretable plots can be generated. Some tools we assessed such as 3DIV, GIVE and Juicebox had integrated experimental data that could be selected without needing to upload any data. We found this to be hugely beneficial and was a primary reason for selecting 3DIV as our further analysis tool in the HIPKD analysis. Again, LOOPER is at a disadvantage as it has no integrated data; however, we argue that although very useful, maintaining a database of existing data in such a fast-moving field adds a significant amount of upkeep time to the maintenance of the tool.

Genomic analysis relies heavily on integrating multiple sources of information, especially when creating human-readable visualisations. One of the primary reasons that so many of the tools we assessed are tied to the linear genome, is so that other sources of annotation can be easily mapped on top of the interactive visualisation to provide additional context. LOOPER has a disadvantage as all annotation has to be converted into the graph 'space'. For example, we used node colouring to show putative enhancers, promoters and insulators and edge-colouring to show which type of experimental data was shown. Even though LOOPER displays the most important annotations, there is limited scope to include further annotations on a force-directed graph without overcrowding the visualisation. For this reason, we believe LOOPER is best used in conjunction with other linear visualisation tools such as 3DIV. In a future browser-based version of LOOPER, we would seek to show a linear genome alongside the force-directed graph with the interaction between the two diagrams for maximum analysis power.

#### **4.2.2.6 Future Work**

Chromatin confirmation is currently a research focus, including structures such as the chromatin loop, due to the discovery that 3D chromatin architecture heavily influences both normal and abnormal gene regulation [273]. In normal gene regulation, regulatory processes in cell differentiation, response to extracellular stimuli and apoptosis have all been linked to chromatin architecture [274]. In abnormal situations, chromatin loop malformation and regulatory network disruption has been linked to cancer [271] as well as many other disorders such as the discovered disorder, HIPKD [2].

The rush of new experiments to study chromatin confirmation underpins the intense research focus on this subject. Despite this, there is still some way to go before the theory, experimental protocols and volume of experimental data are available to model the 3D chromatin landscape completely. While we believe our chromatin visualisation tool

LOOPER was essential in developing our hypothesis for the regulation of *PMM2*, there is much additional work that could be done with the availability of additional datasets.

While inferring functional loop interactions from POLR2A was useful, and we believe that we would now include this data as standard in any LOOPER analysis, to study ZNF143 interactions directly, a ZNF143 specific ChIA-PET experiment would have been crucial to understanding the role this transcription factor has to play in the local TAD regulatory network. In addition, layering data from RAD21 or SMC3 ChIA-PET for more accurate CTCT/Cohesin loop analysis would have been extremely beneficial. We would seek in the future to re-analyse this ROI if this data were to be published.

Another limiting factor that would also trigger a re-analysis of the HIPKD ROI would be if more cell lines such as HEPG2 for ChIA-PET were made available through the ENCODE portal. As we mentioned in the previous section, the reliance on a cancer cell line as our disease model is sub-optimal at best, we would seek to move to a different cell-line as a matter of priority in our future work.

In addition to improving on the existing loop analysis quality in LOOPER, we would also see to layer additional loop type data as it became available. YY1 has recently been shown to be a potential loop mediator in a similar way to ZNF143 [275]; we propose that there are many more loop types to be discovered. As other teams publish their ChIA-PET or other chromatin interaction data, that we could layer this information onto our LOOPER data graph to further unravel the regulatory network within the HIPKD TAD.

We also propose expanding LOOPER to analyse other genomic areas. Graphs of different genomic regions could be compared and, and perhaps a taxonomy of force-directed graph regulatory networks could be built. We would employ machine learning techniques for graph pattern recognition to group the graph topologies. We believe this vein of research to be a potentially important one and would require no extra data; therefore, we would pursue this as a matter of priority in future work.

Finally, in terms of the LOOPER software suite, the first major improvement we would pursue would be to remove the dependency on Cytoscape for the graph generation. We propose to migrate the LOOPER algorithm into a web-based tool that uses the D3 JavaScript library [276] for creating annotated force-directed graphs. This change would enable, richer more featured graphs and would mean little or no configuration on the user side to use the tool. This approach is also in line with other software tools in the field.

#### 4.2.2.7 Combined REMEDY and LOOPER Opportunities

REMEDY and LOOPER were presented in the same thesis as we believe that they occupy different portions of the same bioinformatic workflow when studying the aetiology of disease. REMEDY occupies the initial portion of the workflow where raw experimental data leads to the identification of genetic loci associated with the target disease. LOOPER broadly occupies the later stages of analysis where the loci or regions of interest are linked to the disease aetiology. Given this knowledge, we here present a candidate study where both REMEDY and LOOPER could have been used together to improve the workflow and/or results.

Work on inflammatory bowel disease (IBD) published by Jostins *et al* 2012 [277] and Meddens *et al* 2016 [278] was found to be a candidate for re-analysis with REMEDY and LOOPER. In the first paper, a GWAS meta-analysis combined with new genotyped data is presented that identified 163 SNVs associated with IBD. The second paper identified 92 of the 163 loci as being in non-coding regions of the genome that contained regulatory elements. They then performed 4C chromatin interaction analysis to identify candidate genes that were regulated by the 92 regulatory elements; thus, implicating those genes as candidates affecting the aetiology of IBD.

The first paper combines meta-data from different studies we believe primarily because the source datasets were genotyped across a multitude of different arrays making the merging of the source data problematic. This resulted in a situation where only summary statistics were used to produce a meta-analysis which may have lost some of the nuances in the data that would have been retained if the data had been merged into one large GWAS. We argue that by employing REMEDY on the source datasets, the multi-array source data could have been successfully merged and a high power GWAS could have been performed on the super-set of data; consequently, more genome-wide significant SNVs could have been identified.

In the second paper, 4C is performed on the non-coding loci identified to build an interactive map of the data. In the results, a force-directed graph of 4C interactions is shown alongside a linear genome for a specific candidate gene. We feel that the force-directed graph they show is not informative as it does not take into account any structural looping. LOOPER could have been employed alongside this analysis to show the broader looping structure at this locus and thus, how the 4C interactions fit into a broader regulatory structure in the region.

#### 4.2.2.8 Research Impact Summary

Many of the challenges presented by the vast amount of experimental data involved in modern genomic analysis are overcome with software tools. By developing two different pieces of software that specifically address two real problems which were encountered by our research group, our work has contributed to at least three papers at different stages of publication. Furthermore, our software has the potential to impact the broader research community and has contributed to a better understanding of rare renal genetic diseases.

Our first tool REMEDY sought to aid in the pre-processing stages of preparing genotyping datasets for genome-wide association studies (GWAS). We used the software to successfully merge data in two different GWAS projects, which would have been impossible without the technical research that is presented in this thesis. The first project, a study on steroid-sensitive nephrotic syndrome (SSNS) [1], has contributed to our understanding of the aetiology of this rare autoimmune kidney disease in children and may lead to better diagnosis and treatment in the future. The second GWAS project that REMEDY has contributed to is a rare blood disorder; our results again point to genomic regions which we are currently investigating to better understand the disease mechanism.

REMEDY was developed to be a deployable package, usable by non-technical clinicians to aid the wider research community in processing large genotyping based datasets. We believe our software can be useful to any genotyping data-based GWAS, especially where large control datasets are required; hence, REMEDY has the potential to have a significant impact in the study of the genetic architecture of many different diseases. Standardisation of study protocol is an important aspect of study design; we think REMEDY contributes to the standardisation of GWAS practice and therefore, the reliability and repeatability of this class of experiments.

Our second tool, LOOPER, was designed to aid in the analysis and visualisation of chromatin interactions in the genome. We successfully used the tool to explore a specific genomic region where we previously found a mutation which we showed to be important in a newly characterised disease, Hyperinsulinism with polycystic kidney disease (HIPKD). The loop predictions and later visualisation helped our team to build a hypothesis around the mechanism of change in the chromatin architecture which was contributing to the genetic component of the disease, and was later published in a high-impact journal [2].

The 3D regulatory architecture of the genome affects crucial cellular processes and is implicated in a wide range of diseases. 3D regulatory networks have been shown to be extremely complex, even in a small area; consequently, tools to process and visualise the raw data that allows researchers to explore this information in a concise and understand-

able manner are a critical component to understanding how the regulatory architecture of the genome functions. LOOPER provides both a prediction framework for the analysis of un-observed chromatin loops and provides a novel way of viewing existing chromatin confirmation data as a force-directed graph, which we believe is novel in the field.

# References

1. Dufek, S. *et al.* Genetic Identification of Two Novel Loci Associated with Steroid-Sensitive Nephrotic Syndrome. en. *Journal of the American Society of Nephrology*, ASN.2018101054. ISSN: 1046-6673, 1533-3450 (July 2019).
2. Cabezas, O. R. *et al.* Polycystic Kidney Disease with Hyperinsulinemic Hypoglycemia Caused by a Promoter Mutation in Phosphomannomutase 2. en. *Journal of the American Society of Nephrology*, ASN.2016121312. ISSN: 1046-6673, 1533-3450 (Apr. 2017).
3. Paterson, A. H. & Li, Z.-K. Paleo-Green Revolution for rice. en. *Proceedings of the National Academy of Sciences* **108**, 10931–10932. ISSN: 0027-8424, 1091-6490 (July 2011).
4. Wood, R. & Orel, V. *Genetic Prehistory in Selective Breeding: A Prelude to Mendel* ISBN: 978-0-19-850584-6 (Oxford University Press, Oxford, New York, July 2001).
5. Wilczynski, J. Z. On the Presumed Darwinism of Alberuni Eight Hundred Years before Darwin. *Isis* **50**, 459–466. ISSN: 0021-1753 (Dec. 1959).
6. Mendel, G. Experiments in Plant Hybridization. *Verhandlungen des naturforschenden Vereines in Brünn, Bd. IV für das Jahr 1865, Abhandlungen*, 3–47. (1866).
7. Morgan, T. H. What are “Factors” in Mendelian Explanations? en. *Journal of Heredity* **os-5**, 365–367. ISSN: 0022-1503 (Jan. 1909).
8. Benson, K. R. T. H. Morgan’s resistance to the chromosome theory. en. *Nature Reviews Genetics* **2**, 469–474. ISSN: 1471-0056, 1471-0064 (June 2001).
9. Silberschatz, A., Korth, H. F. & Sudarshan, S. *Database Systems Concepts* 3rd (ed Tucker, A. B.) ISBN: 978-0-07-044756-1 (McGraw-Hill, Inc., New York, NY, USA, 1997).
10. International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. en. *Nature* **409**, 860–921. ISSN: 1476-4687 (Feb. 2001).



11. Chargaff, E., Vischer, E., Doniger, R., Green, C. & Misani, F. THE COMPOSITION OF THE DESOXYPENTOSE NUCLEIC ACIDS OF THYMUS AND SPLEEN. en. *Journal of Biological Chemistry* **177**, 405–416. ISSN: 0021-9258, 1083-351X (Jan. 1949).
12. Elson, D. & Chargaff, E. On the desoxyribonucleic acid content of sea urchin gametes. eng. *Experientia* **8**, 143–145. ISSN: 0014-4754 (Apr. 1952).
13. Watson, J. D. & Crick, F. H. C. Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid. en. *Nature* **171**, 737–738. ISSN: 0028-0836 (Apr. 1953).
14. Osborne, T. B. *The vegetable proteins* eng. <http://archive.org/details/vegetableprotein00osbouoft> (2019) (London, Longmans, 1909).
15. Hartley, H. Origin of the Word ‘Protein’. En. *Nature* **168**, 244. ISSN: 1476-4687 (Aug. 1951).
16. Perrett, D. From ‘protein’ to the beginnings of clinical proteomics. en. *PROTEOMICS – Clinical Applications* **1**, 720–738. ISSN: 1862-8354 (2007).
17. Fruton, J. S. Contrasts in Scientific Style. Emil Fischer and Franz Hofmeister: Their Research Groups and Their Theory of Protein Structure. *Proceedings of the American Philosophical Society* **129**, 313–370. ISSN: 0003-049X (1985).
18. Sanger, F. & Tuppy, H. The amino-acid sequence in the phenylalanyl chain of insulin. 2. The investigation of peptides from enzymic hydrolysates. *Biochemical Journal* **49**, 481–490. ISSN: 0264-6021 (Sept. 1951).
19. Kendrew, J. C. *et al.* A Three-Dimensional Model of the Myoglobin Molecule Obtained by X-Ray Analysis. En. *Nature* **181**, 662. ISSN: 1476-4687 (Mar. 1958).
20. Allen, F. W. The Biochemistry of the Nucleic Acids, Purines, and Pyrimidines. *Annual Review of Biochemistry* **10**, 221–244. ISSN: 0066-4154 (June 1941).
21. Brenner, S., Jacob, F. & Meselson, M. An Unstable Intermediate Carrying Information from Genes to Ribosomes for Protein Synthesis. En. *Nature* **190**, 576. ISSN: 1476-4687 (May 1961).
22. Gros, F. *et al.* Unstable Ribonucleic Acid Revealed by Pulse Labelling of Escherichia Coli. En. *Nature* **190**, 581. ISSN: 1476-4687 (May 1961).
23. Crick, F. H. On protein synthesis. eng. *Symposia of the Society for Experimental Biology* **12**, 138–163. ISSN: 0081-1386 (1958).
24. Simoni, R. D., Hill, R. L. & Vaughan, M. The Discovery of the Amino Acid Threonine: the Work of William C. Rose. en. *Journal of Biological Chemistry* **277**, e25–e25. ISSN: 0021-9258, 1083-351X (Sept. 2002).
25. Gamow, G. Possible Relation between Deoxyribonucleic Acid and Protein Structures. En. *Nature* **173**, 318. ISSN: 1476-4687 (Feb. 1954).

26. Brenner, S. ON THE IMPOSSIBILITY OF ALL OVERLAPPING TRIPLET CODES IN INFORMATION TRANSFER FROM NUCLEIC ACID TO PROTEINS. *Proceedings of the National Academy of Sciences of the United States of America* **43**, 687–694. ISSN: 0027-8424 (Aug. 1957).
27. Nirenberg, M. W. & Matthaei, J. H. The dependence of cell-free protein synthesis in *E. coli* upon naturally occurring or synthetic polyribonucleotides. en. *Proceedings of the National Academy of Sciences* **47**, 1588–1602. ISSN: 0027-8424, 1091-6490 (Oct. 1961).
28. Kim, P. S. & Baldwin, R. L. Specific Intermediates in the Folding Reactions of Small Proteins and the Mechanism of Protein Folding. *Annual Review of Biochemistry* **51**, 459–489. ISSN: 0066-4154 (June 1982).
29. Johannsen, W. *Arvelighedslærens Elementer* Danish. OCLC: 61069746 (Gyldendal, Kbh., 1905).
30. Benzer, S. FINE STRUCTURE OF A GENETIC REGION IN BACTERIOPHAGE. *Proceedings of the National Academy of Sciences of the United States of America* **41**, 344–354. ISSN: 0027-8424 (June 1955).
31. Benzer, S. ON THE TOPOLOGY OF THE GENETIC FINE STRUCTURE. eng. *Proceedings of the National Academy of Sciences of the United States of America* **45**, 1607–1620. ISSN: 0027-8424 (Nov. 1959).
32. Gilbert, W. Why genes in pieces? eng. *Nature* **271**, 501. ISSN: 0028-0836 (Feb. 1978).
33. Amunts, A. *et al.* Structure of the yeast mitochondrial large ribosomal subunit. eng. *Science (New York, N.Y.)* **343**, 1485–1489. ISSN: 1095-9203 (Mar. 2014).
34. Henry, N. L., Sayre, M. H. & Kornberg, R. D. Purification and characterization of yeast RNA polymerase II general initiation factor g. eng. *The Journal of Biological Chemistry* **267**, 23388–23392. ISSN: 0021-9258 (Nov. 1992).
35. Kornberg, R. D. The molecular basis of eukaryotic transcription. en. *Proceedings of the National Academy of Sciences* **104**, 12955–12961. ISSN: 0027-8424, 1091-6490 (Aug. 2007).
36. Nguyen, M. & Haenni, A.-L. Expression strategies of ambisense viruses. eng. *Virus Research* **93**, 141–150. ISSN: 0168-1702 (June 2003).
37. Winkler, H. *Verbreitung und Ursache der Parthenogenesis im Pflanzen- und Tierreiche* / <https://www.biodiversitylibrary.org/item/16372> (G. Fischer, Jena :).
38. Alberts, B. *Molecular Biology of the Cell: Reference edition* en. Google-Books-ID: iepqmRfP3ZoC. ISBN: 978-0-8153-4111-6 (Garland Science, 2008).

39. Sadakierska-Chudy, A. & Filip, M. A comprehensive view of the epigenetic landscape. Part II: Histone post-translational modification, nucleosome level, and chromatin regulation by ncRNAs. eng. *Neurotoxicity Research* **27**, 172–197. ISSN: 1476-3524 (Feb. 2015).
40. Wang, G. G., Allis, C. D. & Chi, P. Chromatin remodeling and cancer, part II: ATP-dependent chromatin remodeling. *Trends in molecular medicine* **13**, 373–380. ISSN: 1471-4914 (Sept. 2007).
41. Mueller-Planitz, F., Klinker, H. & Becker, P. B. Nucleosome sliding mechanisms: new twists in a looped history. en. *Nature Structural & Molecular Biology* **20**, 1026–1032. ISSN: 1545-9993 (Sept. 2013).
42. The Biology of Chromatin Remodeling Complexes. *Annual Review of Biochemistry* **78**, 273–304 (2009).
43. Cutter, A. & Hayes, J. J. A Brief Review of Nucleosome Structure. *FEBS letters* **589**, 2914–2922. ISSN: 0014-5793 (Oct. 2015).
44. *Packaging of DNA: Nucleosomes and Chromatin* <http://www.nature.com/scitable/topicpage/dna-packaging-nucleosomes-and-chromatin-310> (2017).
45. Brown, K. E. *et al.* Association of transcriptionally silent genes with Ikaros complexes at centromeric heterochromatin. eng. *Cell* **91**, 845–854. ISSN: 0092-8674 (Dec. 1997).
46. Lee, W., Haslinger, A., Karin, M. & Tjian, R. Activation of transcription by two factors that bind promoter and enhancer sequences of the human metallothionein gene and SV40. eng. *Nature* **325**, 368–372. ISSN: 0028-0836 (Jan. 1987).
47. Lambert, S. A. *et al.* The Human Transcription Factors. English. *Cell* **172**, 650–665. ISSN: 0092-8674, 1097-4172 (Feb. 2018).
48. Calo, E. & Wysocka, J. Modification of Enhancer Chromatin: What, How, and Why? *Molecular Cell* **49**, 825–837. ISSN: 1097-2765 (Mar. 2013).
49. Wittkopp, P. J. & Kalay, G. Cis-regulatory elements: molecular mechanisms and evolutionary processes underlying divergence. eng. *Nature Reviews. Genetics* **13**, 59–69. ISSN: 1471-0064 (Dec. 2011).
50. Gerstein, M. B. *et al.* Architecture of the human regulatory network derived from ENCODE data. en. *Nature* **489**, 91–100. ISSN: 0028-0836, 1476-4687 (Sept. 2012).
51. Vernimmen, D. & Bickmore, W. A. The Hierarchy of Transcriptional Activation: From Enhancer to Promoter. *Trends in Genetics* **31**, 696–708. ISSN: 0168-9525 (Dec. 2015).

52. Long, H. K., Prescott, S. L. & Wysocka, J. Ever-Changing Landscapes: Transcriptional Enhancers in Development and Evolution. English. *Cell* **167**, 1170–1187. ISSN: 0092-8674, 1097-4172 (Nov. 2016).
53. Spitz, F. & Furlong, E. E. M. Transcription factors: from enhancer binding to developmental control. en. *Nature Reviews Genetics* **13**, 613–626. ISSN: 1471-0056 (Sept. 2012).
54. Schoenfelder, S. & Fraser, P. Long-range enhancer–promoter contacts in gene expression control. En. *Nature Reviews Genetics*, 1. ISSN: 1471-0064 (May 2019).
55. Elgin, S. C. R. & Reuter, G. Position-effect variegation, heterochromatin formation, and gene silencing in *Drosophila*. eng. *Cold Spring Harbor Perspectives in Biology* **5**, a017780. ISSN: 1943-0264 (Aug. 2013).
56. Larson, J. L. & Yuan, G.-C. Chromatin States Accurately Classify Cell Differentiation Stages. *PLoS ONE* **7**. ISSN: 1932-6203. doi:10.1371/journal.pone.0031414. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3282719/> (2017) (Feb. 2012).
57. Van Steensel, B. Chromatin: constructing the big picture. *The EMBO Journal* **30**, 1885–1895. ISSN: 0261-4189 (May 2011).
58. Lobanenko, V. V. *et al.* A novel sequence-specific DNA binding protein which interacts with three regularly spaced direct repeats of the CCCTC-motif in the 5'-flanking sequence of the chicken c-myc gene. eng. *Oncogene* **5**, 1743–1753. ISSN: 0950-9232 (Dec. 1990).
59. Kim, S., Yu, N.-K. & Kaang, B.-K. CTCF as a multifunctional protein in genome regulation and gene expression. en. *Experimental & Molecular Medicine* **47**, e166. ISSN: 2092-6413 (June 2015).
60. Ohlsson, R., Renkawitz, R. & Lobanenko, V. CTCF is a uniquely versatile transcription regulator linked to epigenetics and disease. English. *Trends in Genetics* **17**, 520–527. ISSN: 0168-9525 (Sept. 2001).
61. Harper, P. S. E.A. Carlson (ed) Mendel's Legacy: the origin of classical genetics (2004). en. *Human Genetics* **115**, 346–346. ISSN: 1432-1203 (Sept. 2004).
62. Dekker, J. & Misteli, T. Long-Range Chromatin Interactions. *Cold Spring Harbor Perspectives in Biology* **7**. ISSN: 1943-0264. doi:10.1101/cshperspect.a019356. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4588061/> (2019) (Oct. 2015).
63. Pennisi, E. ENCODE Project Writes Eulogy for Junk DNA. en. *Science* **337**, 1159–1161. ISSN: 0036-8075, 1095-9203 (Sept. 2012).

64. Lanctôt, C., Cheutin, T., Cremer, M., Cavalli, G. & Cremer, T. Dynamic genome architecture in the nuclear space: regulation of gene expression in three dimensions. eng. *Nature Reviews. Genetics* **8**, 104–115. ISSN: 1471-0056 (Feb. 2007).
65. Croft, J. A. *et al.* Differences in the localization and morphology of chromosomes in the human nucleus. eng. *The Journal of Cell Biology* **145**, 1119–1131. ISSN: 0021-9525 (June 1999).
66. Gonzalez-Sandoval, A. & Gasser, S. M. On TADs and LADs: Spatial Control Over Gene Expression. English. *Trends in Genetics* **32**, 485–495. ISSN: 0168-9525 (Aug. 2016).
67. Lupiáñez, D. G. *et al.* Disruptions of Topological Chromatin Domains Cause Pathogenic Rewiring of Gene-Enhancer Interactions. *Cell* **161**, 1012–1025. ISSN: 0092-8674 (May 2015).
68. Dixon, J. R. *et al.* Topological Domains in Mammalian Genomes Identified by Analysis of Chromatin Interactions. *Nature* **485**, 376–380. ISSN: 0028-0836 (Apr. 2012).
69. Dixon, J. R. *et al.* Chromatin Architecture Reorganization during Stem Cell Differentiation. *Nature* **518**, 331–336. ISSN: 0028-0836 (Feb. 2015).
70. Rao, S. S. P. *et al.* A 3D Map of the Human Genome at Kilobase Resolution Reveals Principles of Chromatin Looping. *Cell* **159**, 1665–1680. ISSN: 0092-8674 (Dec. 2014).
71. Fraser, J. *et al.* Hierarchical folding and reorganization of chromosomes are linked to transcriptional changes in cellular differentiation. eng. *Molecular Systems Biology* **11**, 852. ISSN: 1744-4292 (Dec. 2015).
72. Tang, Z. *et al.* CTCF-Mediated Human 3D Genome Architecture Reveals Chromatin Topology for Transcription. *Cell* **163**, 1611–1627. ISSN: 0092-8674 (Dec. 2015).
73. Chetverina, D. *et al.* Boundaries of loop domains (insulators): Determinants of chromosome form and function in multicellular eukaryotes. eng. *BioEssays: News and Reviews in Molecular, Cellular and Developmental Biology* **39**. ISSN: 1521-1878. doi:10.1002/bies.201600233 (2017).
74. Vietri Rudan, M. *et al.* Comparative Hi-C Reveals that CTCF Underlies Evolution of Chromosomal Domain Architecture. *Cell Reports* **10**, 1297–1309. ISSN: 2211-1247 (Feb. 2015).
75. Fudenberg, G. *et al.* Formation of Chromosomal Domains by Loop Extrusion. *Cell Reports* **15**, 2038–2049. ISSN: 2211-1247 (May 2016).
76. Bailey, S. D. *et al.* ZNF143 provides sequence specificity to secure chromatin interactions at gene promoters. *Nature communications* **2**, 6186. ISSN: 2041-1723 (Feb. 2015).

77. Wang, J. *et al.* YY1 Positively Regulates Transcription by Targeting Promoters and Super-Enhancers through the BAF Complex in Embryonic Stem Cells. *Stem Cell Reports* **10**, 1324–1339. ISSN: 2213-6711 (Apr. 2018).
78. Hall, B. K. & Hallgrímsson, B. *Strickberger's Evolution* en. Google-Books-ID: CrJsNQ5wX8oC. ISBN: 978-1-4496-6390-2 (Jones & Bartlett Publishers, June 2011).
79. Marieb, E. N. *Essentials of human anatomy and physiology* English. OCLC: 41266267. ISBN: 978-0-8053-4940-5 978-0-8053-0885-3 978-0-8053-4938-2 (Benjamin Cummings, San Francisco, 2000).
80. Darwin, C. *The Origin of Species: By Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life* 6th ed. doi:10.1017/CB09780511694295 (Cambridge University Press, 2009).
81. Noordwijk, A. J. V. Maynard Smith, J. 1989. Evolutionary genetics. Oxford University Press, Oxford, xii + 325 pp. pbk. £16.95. en. *Journal of Evolutionary Biology* **3**, 313–315. ISSN: 1420-9101 (1990).
82. Lichten, M. & Goldman, A. S. H. Meiotic recombination hotspots. *Annual Review of Genetics* **29**, 423–444. ISSN: 0066-4197 (Dec. 1995).
83. Bustin, S. Molecular Biology of the Cell, Sixth Edition; ISBN: 9780815344643; and Molecular Biology of the Cell, Sixth Edition, The Problems Book; ISBN 9780815344537. *International Journal of Molecular Sciences* **16**, 28123–28125. ISSN: 1422-0067 (Nov. 2015).
84. Griffiths, A. J., Miller, J. H., Suzuki, D. T., Lewontin, R. C. & Gelbart, W. M. Penetrance and expressivity. en. *An Introduction to Genetic Analysis. 7th edition*. <https://www.ncbi.nlm.nih.gov/books/NBK22090/> (2019) (2000).
85. Neel, J. V. The Inheritance of Sickle Cell Anemia. en. *Science* **110**, 64–66. ISSN: 0036-8075, 1095-9203 (July 1949).
86. Cutting, G. R. Cystic fibrosis genetics: from molecular understanding to clinical application. *Nature reviews. Genetics* **16**, 45–56. ISSN: 1471-0056 (Jan. 2015).
87. Boyd, K. P., Korf, B. R. & Theos, A. Neurofibromatosis type 1. *Journal of the American Academy of Dermatology* **61**, 1–16. ISSN: 0190-9622 (July 2009).
88. Cantor, R. M. in *Emery and Rimoin's Principles and Practice of Medical Genetics* (eds Rimoin, D., Pyeritz, R. & Korf, B.) 1–9 (Academic Press, Oxford, Jan. 2013). ISBN: 978-0-12-383834-6. doi:10.1016/B978-0-12-383834-6.00010-0. <http://www.sciencedirect.com/science/article/pii/B9780123838346000100> (2019).
89. Kanitz, R., Guillot, E. G., Antoniazza, S., Neuenschwander, S. & Goudet, J. Complex genetic patterns in human arise from a simple range-expansion model over continental landmasses. en. *PLOS ONE* **13**, e0192460. ISSN: 1932-6203 (Feb. 2018).

90. Saint Pierre, A. & Génin, E. How important are rare variants in common disease? en. *Briefings in Functional Genomics* **13**, 353–361. ISSN: 2041-2649 (Sept. 2014).
91. Coyne, J. A., Barton, N. H. & Turelli, M. PERSPECTIVE: A CRITIQUE OF SEWALL WRIGHT'S SHIFTING BALANCE THEORY OF EVOLUTION. eng. *Evolution; International Journal of Organic Evolution* **51**, 643–671. ISSN: 1558-5646 (June 1997).
92. Collins, D. W. & Jukes, T. H. Rates of transition and transversion in coding sequences since the human-rodent divergence. eng. *Genomics* **20**, 386–396. ISSN: 0888-7543 (Apr. 1994).
93. Deaton, A. M. & Bird, A. CpG islands and the regulation of transcription. *Genes & Development* **25**, 1010–1022. ISSN: 0890-9369 (May 2011).
94. Feuk, L., Carson, A. R. & Scherer, S. W. Structural variation in the human genome. eng. *Nature Reviews. Genetics* **7**, 85–97. ISSN: 1471-0056 (Feb. 2006).
95. Magadum, S., Banerjee, U., Murugan, P., Gangapur, D. & Ravikesavan, R. Gene duplication as a major force in evolution. eng. *Journal of Genetics* **92**, 155–161. ISSN: 0973-7731 (Apr. 2013).
96. Zhang, F., Gu, W., Hurles, M. E. & Lupski, J. R. Copy number variation in human health, disease, and evolution. eng. *Annual Review of Genomics and Human Genetics* **10**, 451–481. ISSN: 1545-293X (2009).
97. Sharp, A. J. *et al.* Segmental Duplications and Copy-Number Variation in the Human Genome. *American Journal of Human Genetics* **77**, 78–88. ISSN: 0002-9297 (July 2005).
98. Rabkin, C. S. & Janz, S. Mechanisms and consequences of chromosomal translocation. eng. *Cancer Epidemiology, Biomarkers & Prevention: A Publication of the American Association for Cancer Research, Cosponsored by the American Society of Preventive Oncology* **17**, 1849–1851. ISSN: 1055-9965 (Aug. 2008).
99. Taki, T. & Taniwaki, M. Chromosomal translocations in cancer and their relevance for therapy. eng. *Current Opinion in Oncology* **18**, 62–68. ISSN: 1040-8746 (Jan. 2006).
100. Kazemi, M., Salehi, M. & Kheirollahi, M. Down Syndrome: Current Status, Challenges and Future Perspectives. *International Journal of Molecular and Cellular Medicine* **5**, 125–133. ISSN: 2251-9637 (2016).
101. Kirkpatrick, M. How and Why Chromosome Inversions Evolve. *PLoS Biology* **8**. ISSN: 1544-9173. doi:10.1371/journal.pbio.1000501. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2946949/> (2019) (Sept. 2010).
102. Heather, J. M. & Chain, B. The sequence of sequencers: The history of sequencing DNA. *Genomics* **107**, 1–8. ISSN: 0888-7543 (Jan. 2016).

103. Holley, R. W. *et al.* STRUCTURE OF A RIBONUCLEIC ACID. eng. *Science (New York, N.Y.)* **147**, 1462–1465. ISSN: 0036-8075 (Mar. 1965).
104. Wu, R. & Kaiser, A. D. Structure and base sequence in the cohesive ends of bacteriophage lambda DNA. *Journal of Molecular Biology* **35**, 523–537. ISSN: 0022-2836 (Jan. 1968).
105. Wu, R. Nucleotide sequence analysis of DNA: I. Partial sequence of the cohesive ends of bacteriophage  $\lambda$  and 186 DNA. *Journal of Molecular Biology* **51**, 501–521. ISSN: 0022-2836 (Aug. 1970).
106. Padmanabhan, R. & Wu, R. Nucleotide sequence analysis of DNA. IX. Use of oligonucleotides of defined sequence as primers in DNA sequence analysis. eng. *Biochemical and Biophysical Research Communications* **48**, 1295–1302. ISSN: 0006-291X (Sept. 1972).
107. Sanger, F., Donelson, J. E., Coulson, A. R., Kössel, H. & Fischer, D. Use of DNA polymerase I primed by a synthetic oligonucleotide to determine a nucleotide sequence in phage  $\phi$ 1 DNA. eng. *Proceedings of the National Academy of Sciences of the United States of America* **70**, 1209–1213. ISSN: 0027-8424 (Apr. 1973).
108. Padmanabhan, R., Jay, E. & Wu, R. Chemical synthesis of a primer and its use in the sequence analysis of the lysozyme gene of bacteriophage T4. eng. *Proceedings of the National Academy of Sciences of the United States of America* **71**, 2510–2514. ISSN: 0027-8424 (June 1974).
109. Sanger, F. & Coulson, A. R. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *Journal of Molecular Biology* **94**, 441–448. ISSN: 0022-2836 (May 1975).
110. Maxam, A. M. & Gilbert, W. A new method for sequencing DNA. eng. *Proceedings of the National Academy of Sciences of the United States of America* **74**, 560–564. ISSN: 0027-8424 (Feb. 1977).
111. Sanger, F. *et al.* Nucleotide sequence of bacteriophage  $\phi$ X174 DNA. eng. *Nature* **265**, 687–695. ISSN: 0028-0836 (Feb. 1977).
112. Sanger, F., Nicklen, S. & Coulson, A. R. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America* **74**, 5463–5467. ISSN: 0027-8424 (Dec. 1977).
113. Hunkapiller, T., Kaiser, R. J., Koop, B. F. & Hood, L. Large-scale and automated DNA sequence determination. eng. *Science (New York, N.Y.)* **254**, 59–67. ISSN: 0036-8075 (Oct. 1991).
114. Staden, R. A strategy of DNA sequencing employing computer programs. eng. *Nucleic Acids Research* **6**, 2601–2610. ISSN: 0305-1048 (June 1979).



115. Anderson, S. Shotgun DNA sequencing using cloned DNase I-generated fragments. eng. *Nucleic Acids Research* **9**, 3015–3027. ISSN: 0305-1048 (July 1981).
116. *Human Genome Overview - Genome Reference Consortium* <https://www.ncbi.nlm.nih.gov/grc/human> (2019).
117. Benson, D. A. *et al.* GenBank. eng. *Nucleic Acids Research* **41**, D36–42. ISSN: 1362-4962 (Jan. 2013).
118. Cartwright, R. A. & Graur, D. The multiple personalities of Watson and Crick strands. *Biology Direct* **6**, 7. ISSN: 1745-6150 (Feb. 2011).
119. Hradecna, Z. & Szybalski, W. Fractionation of the complementary strands of coliphage lambda DNA based on the asymmetric distribution of the poly I,G-binding sites. eng. *Virology* **32**, 633–643. ISSN: 0042-6822 (Aug. 1967).
120. Taylor, K., Hradecna, Z. & Szybalski, W. Asymmetric distribution of the transcribing regions on the complementary strands of coliphage lambda DNA. *Proceedings of the National Academy of Sciences of the United States of America* **57**, 1618–1625. ISSN: 0027-8424 (June 1967).
121. Rhee, S., Han, Z. j., Liu, K., Miles, H. T. & Davies, D. R. Structure of a triple helical DNA with a triplex-duplex junction. eng. *Biochemistry* **38**, 16810–16815. ISSN: 0006-2960 (Dec. 1999).
122. Bailly, C., Møllegaard, N. E., Nielsen, P. E. & Waring, M. J. The influence of the 2-amino group of guanine on DNA conformation. Uranyl and DNase I probing of inosine/diaminopurine substituted DNA. eng. *The EMBO journal* **14**, 2121–2131. ISSN: 0261-4189 (May 1995).
123. Cherry, J. M. *et al.* SGD: Saccharomyces Genome Database. eng. *Nucleic Acids Research* **26**, 73–79. ISSN: 0305-1048 (Jan. 1998).
124. Sherry, S. T. *et al.* dbSNP: the NCBI database of genetic variation. eng. *Nucleic Acids Research* **29**, 308–311. ISSN: 1362-4962 (Jan. 2001).
125. Kitts, A. & Sherry, S. *The Single Nucleotide Polymorphism Database (dbSNP) of Nucleotide Sequence Variation* en. <https://www.ncbi.nlm.nih.gov/books/NBK21088/> (2019) (National Center for Biotechnology Information (US), Feb. 2011).
126. Information, N. C. f. B., Pike, U. S. N. L. o. M. 8. R., MD, B. & Usa, 2. *Sequence Formatting in dbSNP Reports* en. <https://www.ncbi.nlm.nih.gov/books/NBK44414/> (2019) (National Center for Biotechnology Information (US), 2005).
127. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. eng. *Journal of Molecular Biology* **215**, 403–410. ISSN: 0022-2836 (Oct. 1990).
128. De la Cruz, O. & Raska, P. Population structure at different minor allele frequency levels. *BMC Proceedings* **8**, S55. ISSN: 1753-6561 (June 2014).

129. The International HapMap Consortium. A second generation human haplotype map of over 3.1 million SNPs. en. *Nature* **449**, 851–861. ISSN: 1476-4687 (Oct. 2007).
130. Kent, W. J. BLAT—the BLAST-like alignment tool. eng. *Genome Research* **12**, 656–664. ISSN: 1088-9051 (Apr. 2002).
131. *How to interpret DNA strand and allele information for Infinium genotyping array data* <http://emea.support.illumina.com/bulletins/2017/06/how-to-interpret-dna-strand-and-allele-information-for-infinium-.html> (2019).
132. *DNA Strand Designations* <http://emea.support.illumina.com/bulletins/2016/06/dna-strand-designations-.html> (2019).
133. Wwcore (webteam). *Zebrafish Genome Project* en-GB. <https://www.sanger.ac.uk/science/data/zebrafish-genome-project> (2019).
134. Mukhopadhyay, R. DNA sequencers: the next generation. *Analytical Chemistry* **81**, 1736–1740. ISSN: 0003-2700 (Mar. 2009).
135. Ragoussis, J. Genotyping technologies for genetic research. eng. *Annual Review of Genomics and Human Genetics* **10**, 117–133. ISSN: 1545-293X (2009).
136. Li, G. A new model calling procedure for Illumina BeadArray data. *BMC Genetics* **17**. ISSN: 1471-2156. doi:10.1186/s12863-016-0398-x. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4921002/> (2019) (June 2016).
137. Li, H. Toward better understanding of artifacts in variant calling from high-coverage samples. en. *Bioinformatics* **30**, 2843–2851. ISSN: 1367-4803 (Oct. 2014).
138. *BeadArray Microarray Technology* <https://emea.illumina.com/science/technology/beadarray-technology.html> (2019).
139. *GenomeStudio Software* <https://emea.illumina.com/techniques/microarrays/array-data-analysis-experimental-design/genomestudio.html> (2019).
140. Morita, A. et al. Genotyping of triallelic SNPs using TaqMan PCR. eng. *Molecular and Cellular Probes* **21**, 171–176. ISSN: 0890-8508 (June 2007).
141. Hickey, G. et al. Genotyping structural variants in pangenome graphs using the vg toolkit. en. *bioRxiv*, 654566 (June 2019).
142. Eichler, E. E. Genome structural variation discovery and genotyping—sequencing versus arrays. *Pathology. Australasian Division of the International Academy of Pathology Abstracts 36th Annual Scientific Meeting 2011* **44**, S29. ISSN: 0031-3025 (Jan. 2012).

143. Lau, Q. C., Chow, V. T. & Poh, C. L. Polymerase chain reaction and direct sequencing of *Neisseria gonorrhoeae* protein IB gene: partial nucleotide and amino acid sequence analysis of strains S4, S11, S48 (serovar IB4) and S34 (serovar IB5). eng. *Medical Microbiology and Immunology* **182**, 137–145. ISSN: 0300-8584 (July 1993).
144. Ott, J., Wang, J. & Leal, S. M. Genetic linkage analysis in the age of whole-genome sequencing. en. *Nature Reviews Genetics* **16**, 275–284. ISSN: 1471-0064 (May 2015).
145. Visscher, P. M. *et al.* 10 Years of GWAS Discovery: Biology, Function, and Translation. English. *The American Journal of Human Genetics* **101**, 5–22. ISSN: 0002-9297, 1537-6605 (July 2017).
146. Stringer, S., Wray, N. R., Kahn, R. S. & Derks, E. M. Underestimated Effect Sizes in GWAS: Fundamental Limitations of Single SNP Analysis for Dichotomous Phenotypes. en. *PLOS ONE* **6**, e27964. ISSN: 1932-6203 (Nov. 2011).
147. Clarke, G. M. *et al.* Basic statistical analysis in genetic case-control studies. en. *Nature Protocols* **6**, 121–133. ISSN: 1750-2799 (Feb. 2011).
148. Zondervan, K. T. & Cardon, L. R. Designing candidate gene and genome-wide case-control association studies. eng. *Nature Protocols* **2**, 2492–2501. ISSN: 1750-2799 (2007).
149. Pettersson, F. H. *et al.* Marker selection for genetic case-control association studies. *Nature protocols* **4**, 743–752. ISSN: 1754-2189 (2009).
150. Anderson, C. A. *et al.* Data quality control in genetic case-control association studies. eng. *Nature Protocols* **5**, 1564–1573. ISSN: 1750-2799 (Sept. 2010).
151. F.R.S, K. P. X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **50**, 157–175. ISSN: 1941-5982 (July 1900).
152. Bonferroni, C. E. *Teoria statistica delle classi e calcolo delle probabilità* it (Libreria internazionale Seeber, 1936).
153. McCarthy, M. I. *et al.* Genome-wide association studies for complex traits: consensus, uncertainty and challenges. en. *Nature Reviews Genetics* **9**, 356–369. ISSN: 1471-0064 (May 2008).
154. Barsh, G. S., Copenhaver, G. P., Gibson, G. & Williams, S. M. Guidelines for Genome-Wide Association Studies. en. *PLOS Genetics* **8**, e1002812. ISSN: 1553-7404 (July 2012).

155. Panagiotou, O. A., Ioannidis, J. P. A. & Genome-Wide Significance Project. What should the genome-wide significance threshold be? Empirical replication of borderline genetic associations. eng. *International Journal of Epidemiology* **41**, 273–286. ISSN: 1464-3685 (Feb. 2012).
156. Fadista, J., Manning, A. K., Florez, J. C. & Groop, L. The (in)famous GWAS *P*-value threshold revisited and updated for low-frequency variants. en. *European Journal of Human Genetics* **24**, 1202–1205. ISSN: 1476-5438 (Aug. 2016).
157. Wigginton, J. E., Cutler, D. J. & Abecasis, G. R. A Note on Exact Tests of Hardy-Weinberg Equilibrium. *American Journal of Human Genetics* **76**, 887–893. ISSN: 0002-9297 (May 2005).
158. *The Hardy-Weinberg Principle | Learn Science at Scitable* <https://www.nature.com/scitable/knowledge/library/the-hardy-weinberg-principle-13235724> (2019).
159. F.R.S, K. P. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2**, 559–572. ISSN: 1941-5982 (Nov. 1901).
160. Price, A. L. *et al.* Principal components analysis corrects for stratification in genome-wide association studies. eng. *Nature Genetics* **38**, 904–909. ISSN: 1061-4036 (Aug. 2006).
161. De la Chapelle, A. Disease gene mapping in isolated human populations: the example of Finland. eng. *Journal of Medical Genetics* **30**, 857–865. ISSN: 0022-2593 (Oct. 1993).
162. Li, Y., Willer, C., Sanna, S. & Abecasis, G. Genotype Imputation. *Annual review of genomics and human genetics* **10**, 387–406. ISSN: 1527-8204 (2009).
163. *MEGA Genotyping Array Consortium* <https://emea.illumina.com/science/consortia/human-consortia/multi-ethnic-genotyping-consortium.html> (2019).
164. Wilk, M. B. & Gnanadesikan, R. Probability plotting methods for the analysis of data. eng. *Biometrika* **55**, 1–17. ISSN: 0006-3444 (Mar. 1968).
165. *Git* <https://git-scm.com/> (2019).
166. *GitKraken* en. <https://www.gitkraken.com/> (2019).
167. Butler, S., Wermelinger, M., Yu, Y. & Sharp, H. *Exploring the Influence of Identifier Names on Code Quality: An Empirical Study in 2010* 14th European Conference on Software Maintenance and Reengineering (Mar. 2010), 156–165. doi:10.1109/CSMR.2010.27.

168. Hove, S. E. & Anda, B. *Experiences from conducting semi-structured interviews in empirical software engineering research in 11th IEEE International Software Metrics Symposium (METRICS'05)* (Sept. 2005), 10 pp.–23. doi:10.1109/METRICS.2005.24.
169. Arisholm, E., Briand, L. C., Hove, S. E. & Labiche, Y. The impact of UML documentation on software maintenance: an experimental evaluation. *IEEE Transactions on Software Engineering* **32**, 365–381. ISSN: 0098-5589 (June 2006).
170. Patterson, D. A., Gibson, G. & Katz, R. H. *A Case for Redundant Arrays of Inexpensive Disks (RAID)* in *Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data* event-place: Chicago, Illinois, USA (ACM, New York, NY, USA, 1988), 109–116. ISBN: 978-0-89791-268-6. doi:10.1145/50202.50214. <http://doi.acm.org/10.1145/50202.50214> (2019).
171. *EUGDPR – Information Portal* en-GB. <https://eugdpr.org/> (2019).
172. Chow, T. & Cao, D.-B. A survey study of critical success factors in agile software projects. *Journal of Systems and Software. Agile Product Line Engineering* **81**, 961–971. ISSN: 0164-1212 (June 2008).
173. *Microsoft Visio | Flowchart Maker & Diagram Software* en-gb. <https://products.office.com/en-gb/visio/flowchart-software> (2019).
174. *Enterprise Application Container Platform* en. <https://www.docker.com/> (2019).
175. Turnbull, J. *The Docker Book: Containerization Is the New Virtualization* en. Google-Books-ID: 4xQKBAAQBAJ. ISBN: 978-0-9888202-0-3 (James Turnbull, July 2014).
176. Leipzig, J. A review of bioinformatic pipeline frameworks. en. *Briefings in Bioinformatics* **18**, 530–536. ISSN: 1467-5463 (May 2017).
177. Stanescu, H. C. *et al.* Risk HLA-DQA1 and PLA2R1 Alleles in Idiopathic Membranous Nephropathy. *New England Journal of Medicine* **364**, 616–626. ISSN: 0028-4793 (Feb. 2011).
178. Zuvich, R. L. *et al.* Pitfalls of Merging GWAS Data: Lessons Learned in the eMERGE Network and Quality Control Procedures to Maintain High Data Quality. *Genetic epidemiology* **35**, 887–898. ISSN: 0741-0395 (Dec. 2011).
179. Van Iperen, E. P. A., Hovingh, G. K., Asselbergs, F. W. & Zwinderman, A. H. Extending the use of GWAS data by combining data from different genetic platforms. *PLoS ONE* **12**. ISSN: 1932-6203. doi:10.1371/journal.pone.0172082. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5330464/> (2019) (Feb. 2017).
180. Turner, S. *et al.* Quality control procedures for genome-wide association studies. eng. *Current Protocols in Human Genetics* **Chapter 1**, Unit1.19. ISSN: 1934-8258 (Jan. 2011).

181. Evangelou, E. & Ioannidis, J. P. A. Meta-analysis methods for genome-wide association studies and beyond. en. *Nature Reviews Genetics* **14**, 379–389. ISSN: 1471-0064 (June 2013).
182. Lappalainen, I. *et al.* The European Genome-phenome Archive of human data consented for biomedical research. en. *Nature Genetics* **47**, 692–695. ISSN: 1546-1718 (June 2015).
183. Mailman, M. D. *et al.* The NCBI dbGaP database of genotypes and phenotypes. eng. *Nature Genetics* **39**, 1181–1186. ISSN: 1546-1718 (Oct. 2007).
184. *Wellcome Trust Case Control Consortium - WTCCC* <https://www.wtccc.org.uk/> (2019).
185. *HumanOmniExpress-12 BeadChip Product Files* [http://emea.support.illumina.com/array/array\\_kits/humanomniexpress-12-beadchip-kit/downloads.html](http://emea.support.illumina.com/array/array_kits/humanomniexpress-12-beadchip-kit/downloads.html) (2019).
186. The International HapMap Project. En. *Nature* **426**, 789. ISSN: 1476-4687 (Dec. 2003).
187. Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. eng. *Nature* **447**, 661–678. ISSN: 1476-4687 (June 2007).
188. *SNP & Variation Suite (SVS) - Golden Helix* [http://goldenhelix.com/products/SNP\\_Variation/index.html](http://goldenhelix.com/products/SNP_Variation/index.html) (2019).
189. *Multi-Ethnic Global-8 Kit Product Files* [https://support.illumina.com/array/array\\_kits/infinium-multi-ethnic-global-8-kit/downloads.html](https://support.illumina.com/array/array_kits/infinium-multi-ethnic-global-8-kit/downloads.html) (2019).
190. *Infinium Genotyping Manifest Column Headings* <http://emea.support.illumina.com/bulletins/2016/05/infinium-genotyping-manifest-column-headings.html> (2019).
191. *Wrayner group Oxford Strand Software* <https://www.well.ox.ac.uk/~wrayner/strand/index.html> (2019).
192. Elliott, E. *The Forgotten History of OOP* Nov. 2018. <https://medium.com/javascript-scene/the-forgotten-history-of-oop-88d71b9b2d9f> (2019).
193. *xUnit.net is a free, open source, community-focused unit testing tool for the .NET Framework.: xunit/xunit* original-date: 2013-05-06T16:37:13Z. Apr. 2019. <https://github.com/xunit/xunit> (2019).
194. *Azure Cloud Computing Platform & Services* en. <https://azure.microsoft.com/en-gb/> (2019).

195. Mishra, A. & Hawkins, R. D. Three-dimensional genome architecture and emerging technologies: looping in disease. *Genome Medicine* **9**. ISSN: 1756-994X. doi:10.1186/s13073-017-0477-2. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5623062/> (2019) (Sept. 2017).
196. Solomon, M. J., Larsen, P. L. & Varshavsky, A. Mapping protein-DNA interactions in vivo with formaldehyde: evidence that histone H4 is retained on a highly transcribed gene. *eng. Cell* **53**, 937–947. ISSN: 0092-8674 (June 1988).
197. Kidder, B. L., Hu, G. & Zhao, K. ChIP-Seq: Technical Considerations for Obtaining High Quality Data. *Nature immunology* **12**, 918–922. ISSN: 1529-2908 (Sept. 2011).
198. Blat, Y. & Kleckner, N. Cohesins bind to preferential sites along yeast chromosome III, with differential regulation along arms versus the centric region. *eng. Cell* **98**, 249–259. ISSN: 0092-8674 (July 1999).
199. Ren, B. *et al.* Genome-wide location and function of DNA binding proteins. *eng. Science (New York, N.Y.)* **290**, 2306–2309. ISSN: 0036-8075 (Dec. 2000).
200. Barski, A. *et al.* High-resolution profiling of histone methylations in the human genome. *eng. Cell* **129**, 823–837. ISSN: 0092-8674 (May 2007).
201. Mikkelsen, T. S. *et al.* Genome-wide maps of chromatin state in pluripotent and lineage-committed cells. *eng. Nature* **448**, 553–560. ISSN: 1476-4687 (Aug. 2007).
202. Johnson, D. S., Mortazavi, A., Myers, R. M. & Wold, B. Genome-wide mapping of in vivo protein-DNA interactions. *eng. Science (New York, N.Y.)* **316**, 1497–1502. ISSN: 1095-9203 (June 2007).
203. Park, P. J. ChIP-Seq: advantages and challenges of a maturing technology. *Nature reviews. Genetics* **10**, 669–680. ISSN: 1471-0056 (Oct. 2009).
204. *Txn Fac ChIP V2 Track Settings* [http://genome-euro.ucsc.edu/cgi-bin/hgTrackUi?hgsid=231718976\\_uuK7jL4qziUkuzL5RJ5L7MrTSGoq&c=chr21&g=wgEncodeRegTfbsClusteredV2](http://genome-euro.ucsc.edu/cgi-bin/hgTrackUi?hgsid=231718976_uuK7jL4qziUkuzL5RJ5L7MrTSGoq&c=chr21&g=wgEncodeRegTfbsClusteredV2) (2019).
205. Ernst, J. & Kellis, M. ChromHMM: automating chromatin-state discovery and characterization. *eng. Nature Methods* **9**, 215–216. ISSN: 1548-7105 (Feb. 2012).
206. Stormo, G. D., Schneider, T. D., Gold, L. & Ehrenfeucht, A. Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Research* **10**, 2997–3011. ISSN: 0305-1048 (May 1982).
207. Crooks, G. E., Hon, G., Chandonia, J.-M. & Brenner, S. E. WebLogo: a sequence logo generator. *eng. Genome Research* **14**, 1188–1190. ISSN: 1088-9051 (June 2004).
208. Matys, V. *et al.* TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. *eng. Nucleic Acids Research* **34**, D108–110. ISSN: 1362-4962 (Jan. 2006).

209. Sandelin, A., Alkema, W., Engström, P., Wasserman, W. W. & Lenhard, B. JASPAR: an open-access database for eukaryotic transcription factor binding profiles. eng. *Nucleic Acids Research* **32**, D91–94. ISSN: 1362-4962 (Jan. 2004).
210. Bailey, T. L. *et al.* MEME SUITE: tools for motif discovery and searching. eng. *Nucleic Acids Research* **37**, W202–208. ISSN: 1362-4962 (July 2009).
211. Machanick, P. & Bailey, T. L. MEME-ChIP: motif analysis of large DNA datasets. en. *Bioinformatics* **27**, 1696–1697. ISSN: 1367-4803, 1460-2059 (June 2011).
212. Wang, J. *et al.* Sequence features and chromatin structure around the genomic regions bound by 119 human transcription factors. en. *Genome Research* **22**, 1798–1812. ISSN: 1088-9051, 1549-5469 (Sept. 2012).
213. Bailey, T. L. & Gribskov, M. Combining evidence using p-values: application to sequence homology searches. eng. *Bioinformatics (Oxford, England)* **14**, 48–54. ISSN: 1367-4803 (1998).
214. *Genome Browser FAQ* <https://genome.ucsc.edu/FAQ/FAQformat.html> (2019).
215. Kent, W. J. *et al.* The Human Genome Browser at UCSC. *Genome Research* **12**, 996–1006. ISSN: 1088-9051 (June 2002).
216. De Wit, E. & de Laat, W. A decade of 3C technologies: insights into nuclear organization. *Genes & Development* **26**, 11–24. ISSN: 0890-9369 (Jan. 2012).
217. *BLAST TOPICS* [https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE\\_TYPE=BlastDocs&DOC\\_TYPE=BlastHelp](https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=BlastHelp) (2019).
218. Phanstiel, D. H., Boyle, A. P., Heidari, N. & Snyder, M. P. Mango: a bias-correcting ChIA-PET analysis pipeline. *Bioinformatics* **31**, 3092–3098. ISSN: 1367-4803 (Oct. 2015).
219. Phanstiel, D. *chia pet analysis software. Contribute to dphansti/mango development by creating an account on GitHub* original-date: 2014-02-27T00:31:59Z. Jan. 2019. <https://github.com/dphansti/mango> (2019).
220. Zhang, Y. *et al.* Model-based analysis of ChIP-Seq (MACS). eng. *Genome Biology* **9**, R137. ISSN: 1474-760X (2008).
221. Danecek, P. *et al.* The variant call format and VCFtools. *Bioinformatics* **27**, 2156–2158. ISSN: 1367-4803 (Aug. 2011).
222. Elmasri, R. & Navathe, S. *Fundamentals of Database Systems* 6th. ISBN: 978-0-13-608620-8 (Addison-Wesley Publishing Company, USA, 2010).
223. Venkatraman, S., Fahd, K., Kaspi, S. & Venkatraman, R. International Journal of Information Technology and Computer Science(IJITCS). en. *International Journal of Information Technology and Computer Science(IJITCS)* **8**, 59.
224. *The most popular database for modern apps* en. <https://www.mongodb.com/index> (2019).



225. The 1000 Genomes Project Consortium. A global reference for human genetic variation. en. *Nature* **526**, 68–74. ISSN: 0028-0836 (Oct. 2015).
226. Fairfax, B. P. *et al.* Genetics of gene expression in primary immune cells identifies cell type-specific master regulators and roles of HLA alleles. en. *Nature Genetics* **44**, 502–510. ISSN: 1546-1718 (May 2012).
227. Fairfax, B. P. *et al.* Innate immune activity conditions the effect of regulatory variants upon monocyte gene expression. eng. *Science (New York, N.Y.)* **343**, 1246949. ISSN: 1095-9203 (Mar. 2014).
228. *Global Screening Array-24 Kit | Population-scale genetics* <https://emea.illumina.com/products/by-type/microarray-kits/infinium-global-screening.html> (2019).
229. Staples, J. *et al.* PRIMUS: rapid reconstruction of pedigrees from genome-wide estimates of identity by descent. eng. *American Journal of Human Genetics* **95**, 553–564. ISSN: 1537-6605 (Nov. 2014).
230. Browning, B. L., Zhou, Y. & Browning, S. R. A One-Penny Imputed Genome from Next-Generation Reference Panels. eng. *American Journal of Human Genetics* **103**, 338–348. ISSN: 1537-6605 (Sept. 2018).
231. Purcell, S. *et al.* PLINK: a tool set for whole-genome association and population-based linkage analyses. eng. *American Journal of Human Genetics* **81**, 559–575. ISSN: 0002-9297 (Sept. 2007).
232. *1000 Genomes Phase 3 Reference Panel (version 5a)* <ftp://ftp.1000genomes.ebi.ac.uk/vol11/ftp/release/20130502/> (2019).
233. Fullwood, M. J. & Ruan, Y. ChIP-based methods for the identification of long-range chromatin interactions. *Journal of cellular biochemistry* **107**, 30–39. ISSN: 0730-2312 (May 2009).
234. ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. eng. *Nature* **489**, 57–74. ISSN: 1476-4687 (Sept. 2012).
235. Davis, C. A. *et al.* The Encyclopedia of DNA elements (ENCODE): data portal update. eng. *Nucleic Acids Research* **46**, D794–D801. ISSN: 1362-4962 (Jan. 2018).
236. *Cytoscape: An Open Source Platform for Complex Network Analysis and Visualization* <https://cytoscape.org/> (2019).
237. Rao, S. S. P. *et al.* Cohesin Loss Eliminates All Loop Domains. eng. *Cell* **171**, 305–320.e24. ISSN: 1097-4172 (Oct. 2017).
238. Anno, Y.-N. *et al.* Genome-wide evidence for an essential role of the human Staf/ZNF143 transcription factor in bidirectional transcription. *Nucleic Acids Research* **39**, 3116–3127. ISSN: 0305-1048 (Apr. 2011).

239. Bailey, T. L., Johnson, J., Grant, C. E. & Noble, W. S. The MEME Suite. *Nucleic Acids Research* **43**, W39–W49. ISSN: 0305-1048 (July 2015).
240. Chen, H., Tian, Y., Shu, W., Bo, X. & Wang, S. Comprehensive Identification and Annotation of Cell Type-Specific and Ubiquitous CTCF-Binding Sites in the Human Genome. *PLoS ONE* **7**. ISSN: 1932-6203. doi:10.1371/journal.pone.0041374. <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3400636/> (2017) (July 2012).
241. Kim, T. H. *et al.* Analysis of the Vertebrate Insulator Protein CTCF-Binding Sites in the Human Genome. *Cell* **128**, 1231–1245. ISSN: 0092-8674 (Mar. 2007).
242. O'Leary, N. A. *et al.* Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *eng. Nucleic Acids Research* **44**, D733–745. ISSN: 1362-4962 (Jan. 2016).
243. GTEx Consortium. Genetic effects on gene expression across human tissues. *en. Nature* **550**, 204–213. ISSN: 1476-4687 (Oct. 2017).
244. Apweiler, R. *et al.* UniProt: the Universal Protein knowledgebase. *eng. Nucleic Acids Research* **32**, D115–119. ISSN: 1362-4962 (Jan. 2004).
245. Belton, J.-M. *et al.* Hi-C: a comprehensive technique to capture the conformation of genomes. *eng. Methods (San Diego, Calif.)* **58**, 268–276. ISSN: 1095-9130 (Nov. 2012).
246. ENCSR000BZY – ENCODE <https://www.encodeproject.org/experiments/ENCSR000BZY/> (2019).
247. ENCSR000CAC – ENCODE <https://www.encodeproject.org/experiments/ENCSR000CAC/> (2019).
248. Yang, D. *et al.* 3DIV: A 3D-genome Interaction Viewer and database. *en. Nucleic Acids Research* **46**, D52–D57. ISSN: 0305-1048 (Jan. 2018).
249. 4DN Software <https://www.4dnucleome.org/software.html> (2019).
250. Shlien, A. & Malkin, D. Copy number variations and cancer. *Genome Medicine* **1**, 62. ISSN: 1756-994X (June 2009).
251. Standfuß, C., Parczyk, J., Ruhnau, J. & Klein, A. Genome reorganization in different cancer types: detection of cancer specific breakpoint regions. *Molecular Cytogenetics* **12**. ISSN: 1755-8166. doi:10.1186/s13039-019-0435-3. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6585066/> (2019) (June 2019).
252. Kosugi, S. *et al.* Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing. *Genome Biology* **20**, 117. ISSN: 1474-760X (June 2019).
253. Purcell, S. *PLINK: Whole genome data analysis toolset* <http://zzz.bwh.harvard.edu/plink/contact.shtml> (2019).

254. Deutsch, P. GZIP file format specification version 4.3. <http://dl.acm.org/citation.cfm?id=RFC1952> (2019) (May 1996).
255. *Bcftools by samtools* <https://samtools.github.io/bcftools/> (2019).
256. *dbSNP Batch Query* <https://www.ncbi.nlm.nih.gov/projects/SNP/batchquery.html> (2019).
257. Saccone, S. F. *et al.* New tools and methods for direct programmatic access to the dbSNP relational database. en. *Nucleic Acids Research* **39**, D901–D907. ISSN: 0305-1048 (Jan. 2011).
258. Staff, N. *dbSNP database doubles in size twice in 13 months* en. July 2018. <https://ncbiinsights.ncbi.nlm.nih.gov/2018/07/02/dbsnp-database-doubles-size-twice-13-months/> (2019).
259. *Samsung SSD 970 PRO | Samsung V-NAND Consumer SSD* en. <http://www.samsung.com/semiconductor/minisite/ssd/product/consumer/970pro/> (2019).
260. Schulz, W. L., Nelson, B. G., Felker, D. K., Durant, T. J. S. & Torres, R. Evaluation of relational and NoSQL database architectures to manage genomic annotations. *Journal of Biomedical Informatics* **64**, 288–295. ISSN: 1532-0464 (Dec. 2016).
261. Williams Jr., L. F. *A Modification to the Half-interval Search (Binary Search) Method* in *Proceedings of the 14th Annual Southeast Regional Conference* event-place: Birmingham, Alabama (ACM, New York, NY, USA, 1976), 95–101. doi:10.1145/503561.503582. <http://doi.acm.org/10.1145/503561.503582> (2019).
262. Knuth, D. E. *The Art of Computer Programming, Volume 3: (2Nd Ed.) Sorting and Searching* ISBN: 978-0-201-89685-5 (Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1998).
263. *Apache CouchDB* <http://couchdb.apache.org/> (2019).
264. Dietzfelbinger, M. *et al.* *Dynamic perfect hashing: upper and lower bounds* in *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science* (IEEE, White Plains, NY, USA, 1988), 524–531. ISBN: 978-0-8186-0877-3. doi:10.1109/SFCS.1988.21968. <http://ieeexplore.ieee.org/document/21968/> (2019).
265. Sedgewick, R. & Wayne, K. *Algorithms* 4th. ISBN: 978-0-321-57351-3 (Addison-Wesley Professional, 2011).
266. Beame, P. & Fich, F. E. Optimal Bounds for the Predecessor Problem and Related Problems. *Journal of Computer and System Sciences* **65**, 38–72. ISSN: 0022-0000 (Aug. 2002).

267. Bayer, R. *Binary B-trees for Virtual Memory in Proceedings of the 1971 ACM SIG-FIDET (Now SIGMOD) Workshop on Data Description, Access and Control* event-place: San Diego, California (ACM, New York, NY, USA, 1971), 219–235. doi:10.1145/1734714.1734731. <http://doi.acm.org/10.1145/1734714.1734731> (2019).
268. Pickrell, J. K., Gaffney, D. J., Gilad, Y. & Pritchard, J. K. False positive peaks in ChIP-seq and other sequencing-based functional assays caused by unannotated high copy number regions. *Bioinformatics* **27**, 2144–2146. ISSN: 1367-4803 (Aug. 2011).
269. Mundade, R., Ozer, H. G., Wei, H., Prabhu, L. & Lu, T. Role of ChIP-seq in the discovery of transcription factor binding sites, differential gene regulation mechanism, epigenetic marks and beyond. eng. *Cell Cycle (Georgetown, Tex.)* **13**, 2847–2852. ISSN: 1551-4005 (2014).
270. Simcha, D., Price, N. D. & Geman, D. The Limits of De Novo DNA Motif Discovery. *PLOS ONE* **7**, e47836. ISSN: 1932-6203 (Nov. 2012).
271. See, Y. X., Wang, B. Z. & Fullwood, M. J. Chromatin Interactions and Regulatory Elements in Cancer: From Bench to Bedside. English. *Trends in Genetics* **35**, 145–158. ISSN: 0168-9525 (Feb. 2019).
272. Patro, S. G. K. & Sahu, K. K. Normalization: A Preprocessing Stage. *arXiv:1503.06462 [cs]*. arXiv: 1503.06462. <http://arxiv.org/abs/1503.06462> (2019) (Mar. 2015).
273. Robson, M. I., Ringel, A. R. & Mundlos, S. Regulatory Landscaping: How Enhancer-Promoter Communication Is Sculpted in 3D. English. *Molecular Cell* **74**, 1110–1122. ISSN: 1097-2765 (June 2019).
274. Stadhouders, R., Filion, G. J. & Graf, T. Transcription factors and 3D genome conformation in cell-fate decisions. En. *Nature* **569**, 345. ISSN: 1476-4687 (May 2019).
275. YY1 Facilitates Enhancer-Promoter Contacts to Promote Gene Expression. eng. *Cancer Discovery* **8**, 135. ISSN: 2159-8290 (2018).
276. Bostock, M. *D3.js - Data-Driven Documents* <https://d3js.org/> (2019).
277. Jostins, L. *et al.* Host-microbe interactions have shaped the genetic architecture of inflammatory bowel disease. eng. *Nature* **491**, 119–124. ISSN: 1476-4687 (Nov. 2012).
278. Meddens, C. A. *et al.* Systematic analysis of chromatin interactions at disease associated loci links novel candidate genes to inflammatory bowel disease. *Genome Biology* **17**, 247. ISSN: 1474-760X (Nov. 2016).

## Appendix A

# Hardware Specification

REMEDY and LOOPER were developed on the hardware specification below. All performance tests used the same hardware except where explicitly defined.

Component	Make/Model
<b>Motherboard</b>	Asus Intel Z170-WS Skylake ATX Workstation
<b>Processor</b>	Intel Core i7-6700K 4.0GHz (Skylake) Socket LGA1151 Processor
<b>RAM</b>	Corsair Dominator Platinum Series 64GB DDR4 DRAM 3333MHz
<b>Storage</b>	Samsung 950 Pro 512GB M.2 PCI-e 3.0 x 4 NVMe SSD
<b>Graphics Card</b>	2 x Asus Poseidon-GTX980TI

**Table A.1:** Test and development hardware specification.

## Appendix B

# Software and Source Code

All source code and downloads for REMEDY and LOOPER are available from Github.

REMEDY source code is available from <https://github.com/chris-cheshire/project-priestley>.

LOOPER source code is available from <https://github.com/chris-cheshire/project-banner>.

The source code for this thesis including digital copies of all images shown is available from <https://github.com/chris-cheshire/phd-thesis>.

The images and diagrams shown in this thesis were generated using the following programs:

- LucidChart - <https://www.lucidchart.com>
- BioRender - <https://biorender.com/>

## Appendix C

# Graph Visualisation

The following parameters in the LOOPER graph generation program were used in the production of all graphs shown in the results. Items in *italic* are variable names.

- *targetChr*=chr16
- *targetStart*=8551669
- *targetEnd*=9231669
- *targetCell*=K562
- *segmentExpansion*=2000
- *geneExpansion*=10000
- *proximalThreshold1*=2000
- *proximalThreshold2*=10000
- *proximalThreshold3*=20000
- *distanceThreshold*=1500
- *strengthThreshold*=0.00

ID	PosID	Seg-Type	Label
<b>CTCF_11263</b>	chr16:8617164-8618549	I	CTCF_11263
<b>CTCF_11265</b>	chr16:8728778-8730031	I	CTCF_11265
<b>M189</b>	chr16:8890795-8892964	P	M189_[TMEM186_PMM2]
<b>CTCF_11272</b>	chr16:8973570-8974916	I	CTCF_11272
<b>M120</b>	chr16:9102190-9103495	E	M120
<b>CTCF_11277</b>	chr16:9143850-9145097	E	CTCF_11277
<b>CTCF_11280</b>	chr16:9232490-9233658	PE	CTCF_11280
<b>CTCF_11283</b>	chr16:9351911-9353102	I	CTCF_11283
<b>CTCF_11264</b>	chr16:8622468-8623631	E	CTCF_11264_[TMEM114]
<b>CTCF_11268</b>	chr16:8885350-8886518	I	CTCF_11268_[PMM2]
<b>CTCF_11282</b>	chr16:9349574-9350741	I	CTCF_11282
<b>CTCF_11267</b>	chr16:8754373-8755583	I	CTCF_11267
<b>CTCF_11271</b>	chr16:8967049-8968171	E	CTCF_11271_[CARHSP1]
<b>CTCF_11270</b>	chr16:8948771-8949998	I	CTCF_11270
<b>CTCF_11279</b>	chr16:9228251-9229448	E	CTCF_11279
<b>CTCF_11273</b>	chr16:8998635-8999810	I	CTCF_11273
<b>CTCF_11274</b>	chr16:9029348-9030559	I	CTCF_11274_[USP7]
<b>CTCF_11276</b>	chr16:9126900-9128085	I	CTCF_11276
<b>CTCF_11290</b>	chr16:10274138-10276178	I	CTCF_11290_[GRIN2A]
<b>POLR2A_3698</b>	chr16:8159239-8160968	PE	POLR2A_3698
<b>POLR2A_3701</b>	chr16:8961792-8963678	PE	POLR2A_3701_[CARHSP1]
<b>POLR2A_3702</b>	chr16:9056978-9058311	P	POLR2A_3702_[USP7]
<b>POLR2A_3707</b>	chr16:9184379-9187388	PE	POLR2A_3707_[C16orf72]
<b>POLR2A_3704</b>	chr16:9146490-9147898	E	POLR2A_3704
<b>POLR2A_3705</b>	chr16:9161426-9163040	PE	POLR2A_3705
<b>POLR2A_3706</b>	chr16:9172349-9173820	E	POLR2A_3706

**Table C.1:** Full details of the graph nodes generated from the LOOPER visualisation algorithm.



Source	Target	Interaction	Distance	Strength
CTCF_11263	CTCF_11265	CTCF	111.55	1
CTCF_11263	M189	CTCF	273.75	0.35
CTCF_11263	CTCF_11272	CTCF	356.39	1
CTCF_11263	M120	CTCF	484.96	0.21
CTCF_11263	CTCF_11277	CTCF	526.62	0
CTCF_11263	CTCF_11280	CTCF	615.22	0.1
CTCF_11263	CTCF_11283	CTCF	734.65	0
CTCF_11264	CTCF_11265	CTCF	106.36	0.33
CTCF_11265	CTCF_11268	CTCF	156.53	0.2
CTCF_11265	CTCF_11282	CTCF	620.75	0.96
CTCF_11267	CTCF_11268	CTCF	130.96	0.81
CTCF_11268	CTCF_11271	CTCF	81.68	0.34
CTCF_11270	CTCF_11272	CTCF	24.86	1
CTCF_11270	M120	CTCF	153.43	0.07
CTCF_11272	CTCF_11279	CTCF	254.61	0.01
CTCF_11273	M120	CTCF	103.59	0.04
CTCF_11273	CTCF_11279	CTCF	229.63	0.97
CTCF_11274	M120	CTCF	72.86	0.89
CTCF_11274	CTCF_11276	CTCF	97.54	0.25
M120	CTCF_11290	CTCF	1172.34	0.63
M120	CTCF_11277	CTCF	41.66	0.33
M120	CTCF_11279	CTCF	126.03	0.02
M120	CTCF_11280	CTCF	130.26	0.81
CTCF_11276	CTCF_11277	CTCF	16.98	0.12
POLR2A_3698	POLR2A_3701	POLR2A	802.63	0.79
M189	POLR2A_3702	POLR2A	165.76	0.34
POLR2A_3701	POLR2A_3702	POLR2A	94.91	1
POLR2A_3701	POLR2A_3707	POLR2A	223.15	0.8
POLR2A_3704	POLR2A_3705	POLR2A	15.04	0.7
POLR2A_3704	POLR2A_3707	POLR2A	38.69	0.43
POLR2A_3705	POLR2A_3706	POLR2A	10.85	0.36
POLR2A_3705	POLR2A_3707	POLR2A	23.65	0.91
POLR2A_3706	POLR2A_3707	POLR2A	12.8	0.49
CTCF_11263	CTCF_11264	PROX2	5.19	0.1
M189	CTCF_11268	PROX2	5.95	0.1
CTCF_11272	CTCF_11271	PROX2	6.63	0.1
CTCF_11277	POLR2A_3704	PROX2	2.72	0.1
CTCF_11280	CTCF_11279	PROX2	4.22	0.1
CTCF_11271	POLR2A_3701	PROX2	4.88	0.1
CTCF_11272	POLR2A_3701	PROX3	11.51	0.1
CTCF_11277	CTCF_11276	PROX3	16.98	0.1
CTCF_11277	POLR2A_3705	PROX3	17.76	0.1
CTCF_11271	CTCF_11270	PROX3	18.23	0.1
CTCF_11270	POLR2A_3701	PROX3	13.35	0.1
CTCF_11276	POLR2A_3704	PROX3	19.7	0.1
POLR2A_3707	POLR2A_3706	PROX3	12.8	0.1
POLR2A_3704	POLR2A_3705	PROX3	15.04	0.1
POLR2A_3705	POLR2A_3706	PROX3	10.85	0.1

**Table C.2:** Full details of the graph links generated from the LOOPER visualisation algorithm.

