Development of New Computational Tools for Analyzing Hi-C Data and Predicting Three-Dimensional Genome Organization

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Kimberly MacKay

Permission to Use

In presenting this dissertation in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this dissertation in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my dissertation work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this dissertation or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my dissertation.

Disclaimer

Reference in this dissertation to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement, recommendation, or favoring by the University of Saskatchewan. The views and opinions of the author expressed herein do not state or reflect those of the University of Saskatchewan, and shall not be used for advertising or product endorsement purposes.

Requests for permission to copy or to make other uses of materials in this dissertation in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building, 110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan S7N 5C9 Canada

OR

Dean

College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9 Canada

Abstract

Background: The development of Hi-C (and related methods) has allowed for unprecedented sequence-level investigations into the structure-function relationship of the genome. There has been extensive effort in developing new tools to analyze this data in order to better understand the relationship between 3D genomic structure and function. While useful, the existing tools are far from maturity and (in some cases) lack the generalizability that would be required for application in a diverse set of organisms. This is problematic since the research community has proposed many cross-species "hallmarks" of 3D genome organization without confirming their existence in a variety of organisms.

Research Objective: Develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction.

Results: Three new computational tools were developed for Hi-C analysis or 3D genome prediction: GrapHi-C (visualization), GeneRHi-C (3D prediction) and StoHi-C (3D prediction). Each tool has the potential to be used for 3D genome analysis in both model and non-model organisms since the underlying algorithms do not rely on any organism-specific constraints. A brief description of each tool follows. GrapHi-C is a graph-based visualization of Hi-C data. Unlike existing visualization methods, GrapHi-C allows for a more intuitive structural visualization of the underlying data. GeneRHi-C and StoHi-C are tools that can be used to predict 3D genome organizations from Hi-C data (the 3D-genome reconstruction problem). GeneRHi-C uses a combination of mixed integer programming and network layout algorithms to generate 3D coordinates from a ploidy-dependent subset of the Hi-C data. Alternatively, StoHi-C uses t-stochastic neighbour embedding with the complete set of Hi-C data to generate 3D coordinates of the genome. Each tool was applied to multiple, independent existing Hi-C datasets from fission yeast to demonstrate their utility. This is the first time 3D genome prediction has been successfully applied to these datasets. Overall, the tools developed here more clearly recapitulated documented features of fission yeast genomic organization when compared to existing techniques. Future work will focus on extending and applying these tools to analyze Hi-C datasets from other organisms.

Additional Information: This thesis contains a collection of papers pertaining to the development of new tools for analyzing Hi-C data and predicting 3D genome organization. Each paper's publication status (as of January 2020) has been provided at the beginning of the corresponding chapter. For published papers, reprint permission was obtained and is available in the appendix.

Acknowledgements

First, I would like to thank my supervisor Dr. Anthony Kusalik as well as my committee members Dr. Matthew Links, Dr. Ian McQuillan and Dr. Steve Robinson. Your advice, guidance and encouragement has been instrumental as I navigated through graduate school. A special thanks to my family, friends and colleagues for their support and advice over the years. Finally, I would like to thank the following organizations for financially supporting my research: SaskCanola, Vanier CGS (the Tri-Council Agencies) and the University of Saskatchewan.

Contents

List of Tables

List of Figures

List of Abbreviations

AK	Anthony Kusalik
CHi-C	Capture Hi-C
CTCF	CCCTC-binding factor
CHE	Christopher H. Eskiw
CHR	Chromosome
3C	Chromosome Conformation Capture
5C	Chromosome Conformation Capture Carbon Copy
4C	Circularized Chromosome Conformation Capture
CP	Constraint Programming
DAAM	Did Not Appear to be Actively Maintained
DamID	DNA adenine methyltransferase Identification
ESC	Embryonic Stem Cell
GR	Genomic Region
GM	Graph Matching
HR-3C	High-Resolution Chromosome Conformation Capture
IP	Integer Programming
IMP	Integrative Modeling Platform
ICE	Iterative Correction and Eigenvector Decomposition
KM	Kimberly MacKay
KR	Knight-Ruiz
LAD	Lamin-Associated Domains
MCMC	Markov Chain Monte Carlo
MC	Mats Carlsson
MLE	Maximum Likelihood Estimation
M	Mitosis
MDS	Multi-Dimensional Scaling
NA	Not Applicable
NE	Nuclear Envelope
RMSD	Root-Mean-Square Deviation
3D	Three-Dimensional
3D-GRP	Three-Dimensional Genome Reconstruction Problem
T2C	Targeted Chromatin Capture
TCC	Tethered Chromatin Capture
TAD	Topologically Associating Domain
t-SNE	t-Stochastic Neighbourhood Embedding
URLs	Uniform Resource Locators
LOF	List of Figures
LOT	List of Tables

Chapter 1

Introduction

An object's function is closely linked to its structure. This relationship is so natural that it often does not become apparent until we are confronted with an abnormal structure that changes an object's function. Athens-based architect Katerina Kamprani has highlighted this ubiquitous relationship in her Uncomfortable Objects collection. Figure 1.1 contains a selection of three such "uncomfortable objects" that demonstrate how small changes in structure can greatly impact an object's utility and ease of use. These types of structure-function relationships are ubiquitous and extend beyond everyday objects into areas like cellular biology. Since cellular components like the genome cannot be seen by the naked eye, biological assays must be used to help infer their 3D structure and associated function.

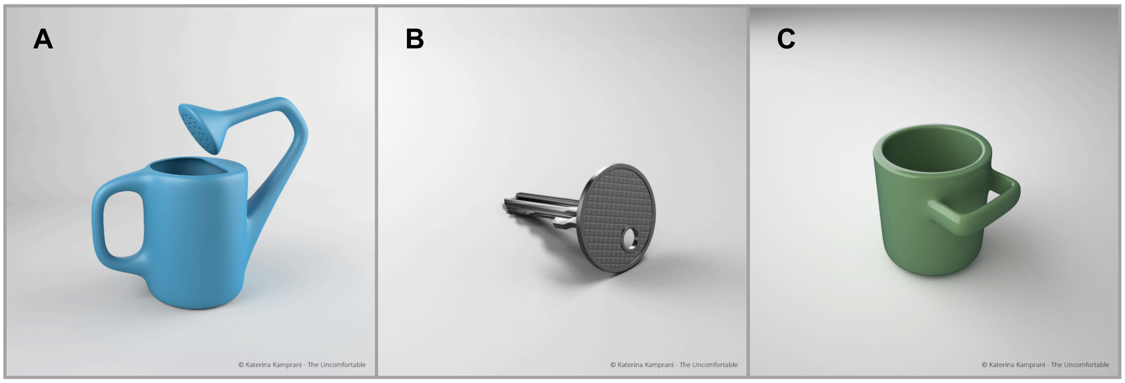

A
B
C
© Katerina Kamprani - The Uncomfortable
© Katerina Kamprani - The Uncomfortable
© Katerina Kamprani - The Uncomfortable

Figure 1.1: A selection of three images from the Uncomfortable Collection designed by Katerina Kamprani © 2017. Each panel contains a representation of an object with an abnormal structure (A: watering can, B: key, C: mug). Images were re-printed with permission (e-mail correspondence, April 8, 2019).

Previously, investigations into the structure-function relationship of the genome were limited since they relied on imaging assays to ascertain 3D genomic organization and structure. While useful, imaging assays have limited resolution leaving many questions unanswered. The recent development of Hi-C (and related methods) has allowed for unprecedented sequence-level investigations into the structure-function relationship of the genome. Briefly, Hi-C is a biological assay that is able to detect regions of the genome that are in close 3D spatial proximity (or "interacting"). A visual depiction of the experimental protocol is provided in Figure 1.2. More specifically, (1) cells are fixed with formaldehyde in order to covalently cross-link genomic regions that are in close 3D proximity. (2) The cross-linked fragments are then digested with a restriction enzyme to remove the potentially large interconnecting segments of DNA and labelled with biotin (biotinylated). (3) Digested fragments are ligated together. (4) The initial cross-linking is removed resulting in DNA fragments that represent the two genomic regions that form an interaction. (5) Biotinylated products are then purified using streptavidin beads allowing for the detection of fragments that were cut by restriction enzymes. Finally, (6) sequencing primers are then ligated to the ends of the purified fragments and high-throughput sequencing is performed.

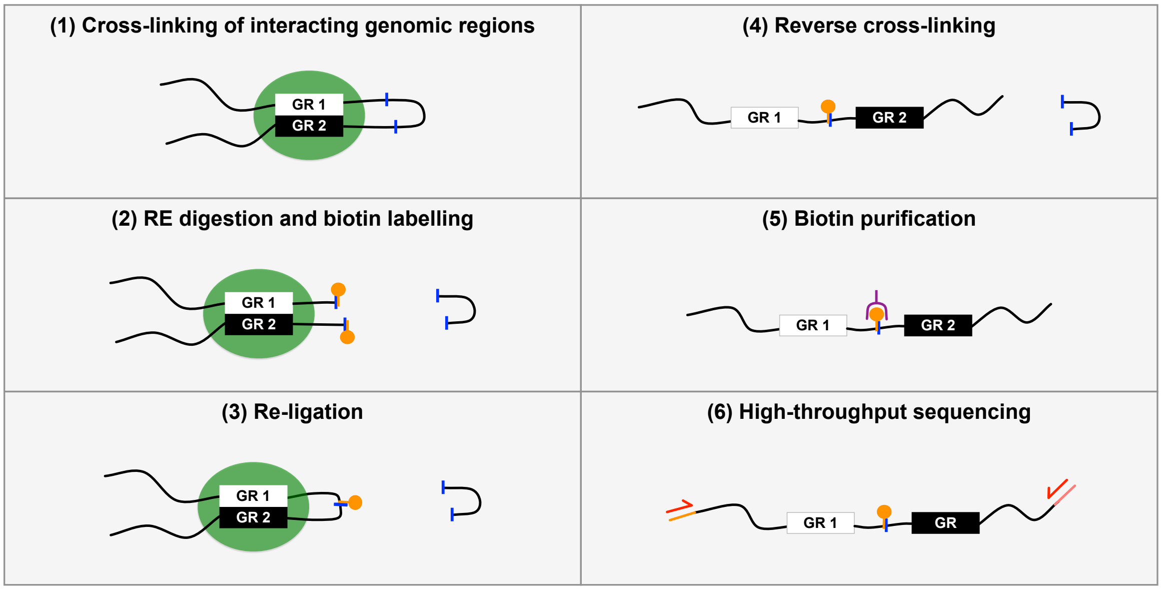

(1) Cross-linking of interacting genomic regions
GR 1
GR 2
(4) Reverse cross-linking
GR 1
GR 2
(2) RE digestion and biotin labelling
GR 1
GR 2
(5) Biotin purification
GR 1
GR 2
(3) Re-ligation
GR 1
GR 2
(6) High-throughput sequencing
GR 1
GR

Figure 1.2: An overview of the typical experimental procedure for a Hi-C assay (adapted from [92, 98]). Coloured boxes differentiate the general steps and are numbered accordingly. GR stands for Genomic Region. The blue lines represent the location of a restriction enzyme cut site. The green circles represent a pair of genomic regions being chemically cross-linked together. The orange circles represent biotin. The purple symbol represents a streptavidin bead that can be used to purify molecules with a biotin label. The red arrows represent the primers that are required for high-throughput sequencing.

The Hi-C experimental protocol results in a collection of paired-end sequencing reads. A variety of computational tools are then used to generate a whole-genome contact map. Lajoie *et al.* [86] provides an in-depth discussion of the computational steps and specific tools required for this process. A general overview of this process is provided in Figure 1.3 in the panel titled "1. CONTACT MAP GENERATION". As described by MacKay [99],

> "a whole-genome contact map is a $N \times N$ matrix, where N is a genomic 'bin' representing a contiguous sequence of linear DNA [86, 101, 177]. In general, the size of the whole-genome contact map (the number of genomic bins) is approximately equal to the total genome size divided by the Hi-C experimental resolution. Each cell ($A_{i,j}$) of a whole-genome contact map ($\mathbf{A}$) indicates the count of how many times the genomic bin i was found to interact with the genomic bin j. These counts are symmetric along the diagonal (i.e. $A_{i,j} = A_{j,i}$) and are often referred to as the frequency of the interaction between A_i and A_j (or interaction frequency)."

Once generated, a whole-genome contact map will serve as the starting point for downstream analysis. For instance, whole-genome contact maps can then be used to predict 3D genome structure and ultimately help better understand the role of the 3D genome in various biological functions like gene regulation [30, 34, 115, 151] and cellular differentiation [3, 4, 65].

The process of predicting 3D genomic structure from Hi-C data is computationally intensive and requires a number of different methodologies. In general, the computational analysis of Hi-C data can be broken down into the following four steps: (1) contact map generation, (2) visualization, (3) pattern detection and (4) 3D genome prediction. A visual depiction of this workflow is presented in Figure 1.3. There has been extensive effort in developing computational tools and techniques in each of the four steps but it has been noted that these tools are far from maturity [99, 176]. Furthermore, the current implementations of existing tools for predicting 3D genome structure from Hi-C data lack the generalizability that would be required for application in non-model organisms. This is due to their reliance on additional datasets or genomic substructures like topologically associating domains (TADs) [99]. **The main objective of this thesis is to address this problem by developing new, generalizable computational tools for Hi-C analysis.** These new tools are imperative as the research community begins to investigate and characterize 3D genome structure in a more diverse set of organisms.

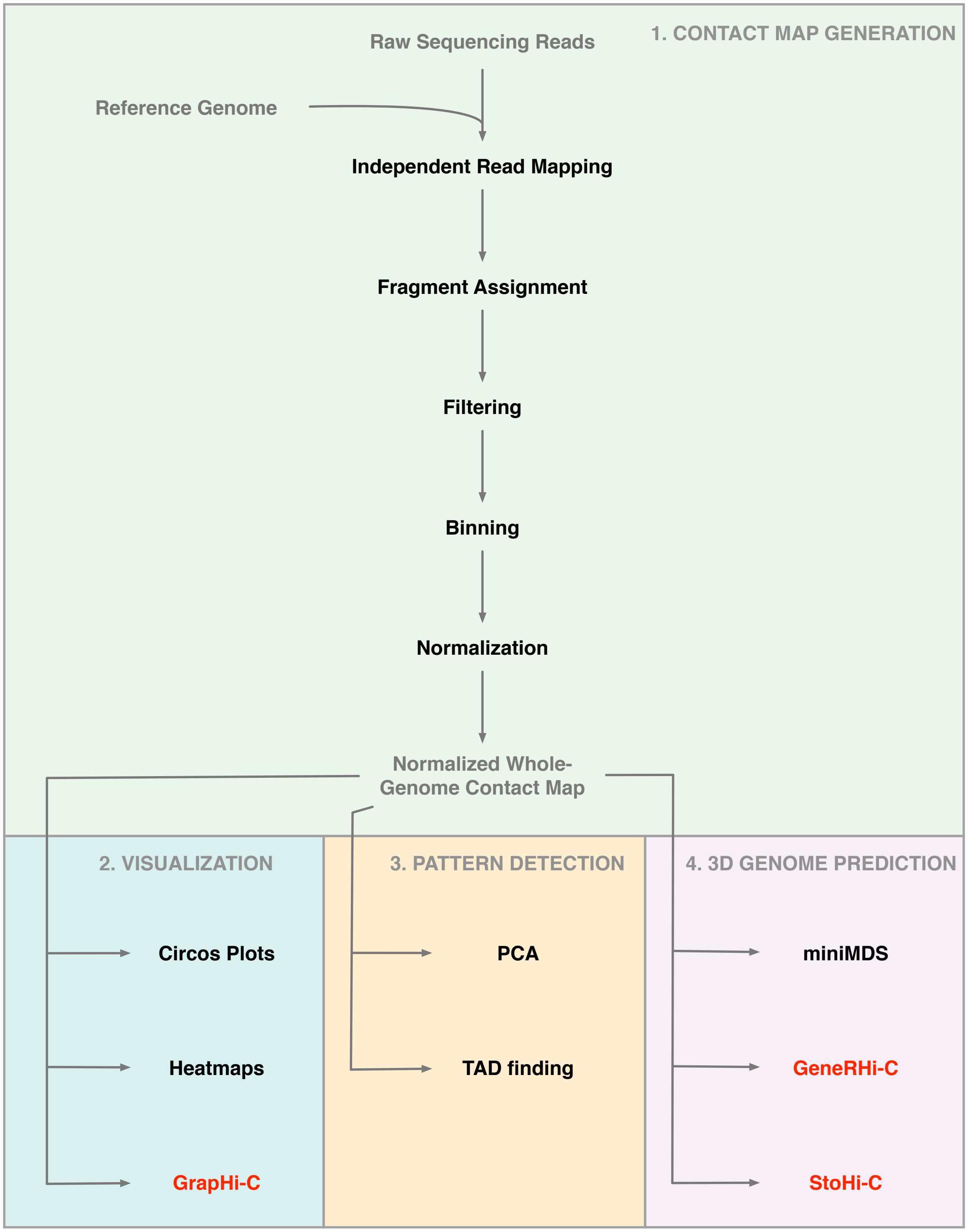


Figure 1.3: An overview of the typical workflow for Hi-C analysis. Arrows depict the flow of information throughout the analysis. Coloured boxes differentiate the general steps. Grey text in the flow chart represents important input/output data or panel labels (numbered and in capital letters). New computational tools that have been developed as a part of this thesis are indicated in red text. Existing computational tools or methodologies are represented in black text.

The vast majority of research in 3D genomics has been conducted in model organisms. For instance, Hi-C datasets from *Homo sapiens* (1675 datasets), *Mus musculus* (1045 datasets) and *Drosophila melanogaster* (156 datasets) account for 89% of all the existing Hi-C datasets in the Gene Expression Omnibus repository (as of January 30, 2020). This is problematic since the research community has proposed many cross-species "hallmarks" of 3D genome organization (such as chromosome territories, distinct regions of the nucleus occupied by a single chromosome, and TADs, linear regions of self-interacting DNA) without confirming their existence in a diverse set of organisms. Even more concerning is how some of the existing tools rely on these "hallmarks" for pattern detection as well as predicting 3D genomic structure. Additionally, it was recently highlighted that 3D genome prediction has only been applied to an even smaller subset of the existing Hi-C data (less than 0.3%) [99]. The tools presented in this thesis are a step towards correcting this bias by providing new options for Hi-C data analysis that are generalizable to a diverse set of organisms.

The following chapters of this thesis consist of papers pertaining to the computational analysis of the 3D genome. The citation and publication status of each paper is given at the beginning of the corresponding chapter. Permission to reprint each paper was obtained where required from the individual publishers and is available in Appendix A. Minor formatting modifications have be made to the manuscripts so they align with the University of Saskatchewan's thesis standards. Appendix B lists any major modifications or additions to the published, in press or submitted manuscripts. I am the first author for each paper and was responsible for writing the manuscripts and performing the bulk of the research. I authored all of the software presented in this thesis except for step 1 of GeneRHi-C which was done by Mats Carlsson. All manuscripts were extensively edited by the co-authors.

A brief overview of the contents of each chapter follows. Chapter 2 describes a new, generalizable tool for Hi-C data visualization (GrapHi-C). Chapter 3 provides an overview of the existing tools for predicting 3D genome structure from Hi-C data. Chapters 4 (GeneRHi-C) and 5 (StoHi-C) describe two new tools for 3D genome prediction that overcome some of the problems identified in Chapter 3. Additionally, Chapters 2, 4 and 5 apply multiple, independent (in terms of strains and in some cases research groups) existing fission yeast datasets to each tool in order to demonstrate their utility. Figure 1.3 indicates how these new tools fit into existing data analysis workflows. Finally, Chapter 6 provides a discussion on the main contributions and limitations of each paper as well as potential future directions stemming from this research.

Chapter 2

GrapHi-C: Graph-Based Visualization of Hi-C Datasets

Kimberly MacKay, Anthony Kusalik and Christopher Eskiw

Citation [101]: K MacKay, A Kusalik, CH Eskiw. GrapHi-C: graph-based visualization of Hi-C datasets. *BMC Research Notes.* (2018) 11(1): 418. doi: `https://doi.org/10.1186/s13104-018-3507-2`.

Detailed Contributions: Kimberly MacKay was responsible for writing the manuscript, performing the research and authoring the software. A brief description of the contributions for all listed authors can be found in Section 2.6.2.

As a reminder, the overarching goal of this thesis is to develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction. GrapHi-C is a tool for visualizing Hi-C datasets which is an important step in Hi-C analysis. Figure 1.3 indicates how GrapHi-C fits into existing Hi-C data analysis workflows.

2.1 Abstract

Objectives: Hi-C is a proximity-based ligation reaction used to detect regions of the genome that are close in 3D space (or “interacting”). Typically, results from Hi-C experiments (contact maps) are visualized as heatmaps or Circos plots. While informative, these visualizations do not directly represent genomic structure and folding, making the interpretation of the underlying 3D genomic organization obscured. Our objective was to generate a graph-based contact map representation that leads to a more intuitive structural visualization.

Results: Normalized contact maps were converted into undirected graphs where each vertex represented a genomic region and each edge represented a detected (intra- and inter-chromosomal) or known (linear) interaction between two regions. Each edge was weighted by the inverse of the linear distance (Hi-C experimental resolution) or the interaction frequency from the contact map. Graphs were generated based on this representation scheme for contact maps from existing fission yeast datasets. Originally, these datasets were used to (1) identify specific principles influencing fission yeast genome organization and (2) uncover changes in fission yeast genome organization during the cell cycle. When compared to the equivalent heatmaps and/or Circos plots, the graph-based visualizations more intuitively depicted the changes in genome organization described in the original studies.

2.2 Introduction

One of the major problems in the genomic era is understanding how genomes are organized and chromosomes are folded within cells. Genomic organization, specifically the close physical proximity of genetic elements located either distally on the same chromosome or located on different chromosomes, greatly impacts cellular processes such as transcription, replication and recombination [9]. The close physical proximity of two genetic elements is often referred to as an "interaction". Knowledge of what interactions are occurring and how they are mediated is essential to understanding genome functions such as gene expression regulation. The biological assay Hi-C [16, 93] (or one of its derivatives [28, 36, 44, 179]) can be used to detect interactions between regions of the genome on the same chromosome (intra-chromosomal or *cis*-interactions) or different chromosomes (inter-chromosomal or *trans*-interactions).

Briefly, Hi-C involves chemically cross-linking regions of the genome that are in close spatial proximity. Restriction enzyme digestion and ligation is then preformed on the cross-linked regions to generate chromatin/DNA complexes which can be identified by high-throughput sequencing. The resultant sequence reads are mapped to a reference genome [7] to determine the frequency with which each interaction occurs within the population of cells. The results of a Hi-C experiment are often encoded as a symmetric $N \times N$ matrix (contact map) where N is the number of genomic "bins" into which the genome is partitioned. Each genomic bin represents a linear region of genomic DNA, where the number of bins is approximately equal to the total genome size divided by the experimental resolution. For instance, a Hi-C experiment in fission yeast that is able to attain 10 kB resolution will generate 1258 genomic bins, each representing roughly 10 kB of linear DNA sequence. Each cell ($CM_{i,j}$) of the contact map records the interaction frequency between genomic bins i and j. Inherent systematic biases within the whole-genome contact map are dampened by normalizing the interaction frequencies. Typically, an ICE [68] or Knight-Ruiz [77, 90] normalization is applied to the raw data resulting in fractional interaction frequencies.

In a typical workflow, normalized contact maps are initially visualized as heatmaps or Circos [82] plots before further downstream analysis and 3D modelling [174]. While informative, these visualizations do not intuitively represent the complex organization and folding of the genome in 3D space. This makes it difficult to quickly understand the underlying 3D genome organization represented by the contact map. Our hypothesis is that representing and visualizing contact maps as a graph will lead to a more intuitive structural visualization of Hi-C data when compared to typical methods. We have developed a protocol called GrapHi-C (pronounced "graphic") for visualizing Hi-C data as a graph. GrapHi-C utilizes a graph-based representation of a contact map and existing interactive tools for a more intuitive structural visualization of Hi-C data. We applied GrapHi-C to two existing datasets to demonstrate the improvements it can bring to interpreting Hi-C data.

2.3 Results and Discussion

2.3.1 Graph-Based Representation

In GrapHi-C visualizations, a contact map is translated into an undirected graph where each genomic bin is represented as a vertex and the detected or known interactions between bins are represented as undirected weighted edges. Specifically, edges represent linear, *cis*- and *trans*-interactions. Each edge is weighted with the inverse of the experimental resolution (for linear interactions) or interaction frequency (for *cis*- and *trans*- interactions). The edges representing *bonafide in vivo* linear connections between bins (i.e. the linear extent of the chromosome) add additional biological constraints. A formal description of the graphical representation used in GrapHi-C is presented in Figure 2.1A.

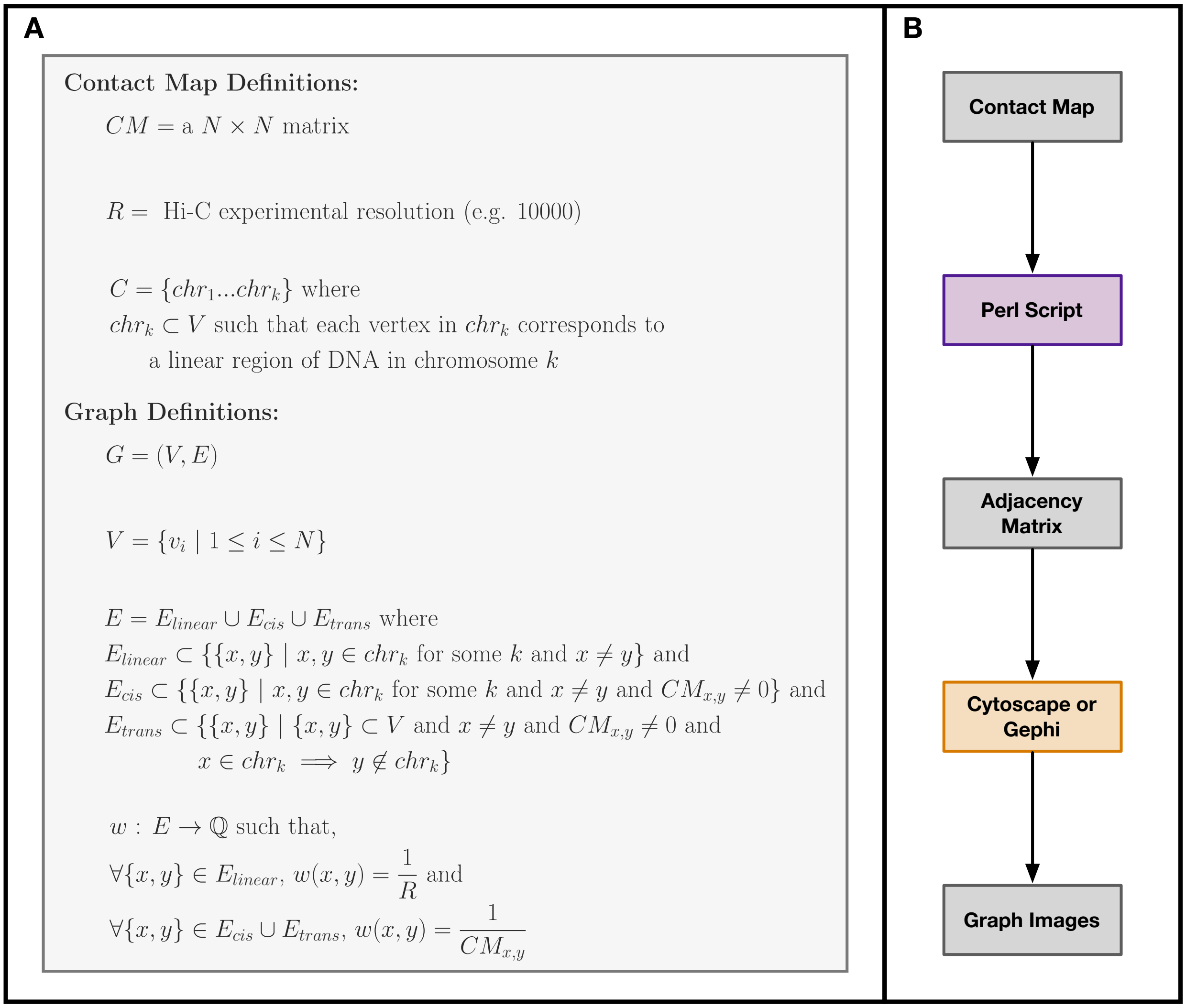
2.3.2 Visualization Protocol

A Perl script was developed that is able to convert a normalized contact map into an adjacency matrix based on the graph representation described above (available at: `https://github.com/kimmackay/GrapHi-C`, or Additional File 1). The output of this script can then be input into a tool like Cytoscape [139] or Gephi [12] to generate a structural visualization. Utilizing existing network visualization tools is advantageous since there are multiple plug-ins and layouts available which allow for flexibility in visualization and subsequent analysis.

It should be noted that Hi-C data cannot be directly input into tools like Cytoscape or Gephi. CytoHiC is the only existing Cytoscape plug-in for Hi-C data. It is used for pairwise comparisons of contact maps based on genetic landmarks such as methylation [140] and would provide a complementary analysis to GrapHi-C. The current version of CytoHiC is not compatible with the latest major release of Cytoscape (released February 2013) and the plug-in does not appear to be actively maintained. Unlike GrapHi-C, CytoHiC does not include edges representing linear interactions. It also utilizes a different edge weight equation to incorporate genetic landmarks into its comparison.

2.3.3 Applications

To demonstrate the value of the GrapHi-C visualization protocol, it was utilized to visualize contact maps from existing fission yeast datasets where (1) fission yeast mutants were studied to determine principles of genomic organization [108] and (2) synchronized fission yeast cells were used to track genomic organization throughout the cell cycle [152]. In each case, normalized fission yeast contact maps (10 kB resolution) were downloaded from the Gene Expression Omnibus database. The specific accession numbers are listed in the "Availability of data and materials" section below. These contact maps were transformed into adjacency matrices using the Perl script described above. The matrices were then input to the developed GrapHi-C protocol depicted in Figure 2.1B. For comparison, the normalized fission



Contact Map Definitions:

$CM =$ a $N \times N$ matrix

$R =$ Hi-C experimental resolution (e.g. 10000)

$C = \{chr_1 ... chr_k\}$ where
$chr_k \subset V$ such that each vertex in chr_k corresponds to
a linear region of DNA in chromosome k

Graph Definitions:

$G = (V, E)$

$V = \{v_i \mid 1 \leq i \leq N\}$

$E = E_{linear} \cup E_{cis} \cup E_{trans}$ where
$E_{linear} \subset \{\{x, y\} \mid x, y \in chr_k$ for some k and $x \neq y\}$ and
$E_{cis} \subset \{\{x, y\} \mid x, y \in chr_k$ for some k and $x \neq y$ and $CM_{x,y} \neq 0\}$ and
$E_{trans} \subset \{\{x, y\} \mid \{x, y\} \subset V$ and $x \neq y$ and $CM_{x,y} \neq 0$ and
$x \in chr_k \implies y \notin chr_k\}$

$w : E \to \mathbb{Q}$ such that,

$\forall \{x, y\} \in E_{linear},\ w(x, y) = \dfrac{1}{R}$ and

$\forall \{x, y\} \in E_{cis} \cup E_{trans},\ w(x, y) = \dfrac{1}{CM_{x,y}}$

Figure 2.1: A formal description of the graph representation and workflow used by GraphHi-C. a The mathematical model used to represent a contact map as an undirected graph in the GraphHi-C protocol. **b** Overview of the GraphHi-C protocol. Each step of the workflow is indicated in a box where the different colours correspond to: data input and output (grey), developed Perl script (purple), and an existing tool (orange). An option for scaling the interaction frequencies is available in the developed Perl script if future studies wish to use it.

yeast contact maps were also visualized as heatmaps and Circos plots. The heatmaps were generated using Java Treeview [129] which is the recommended visualization tool for contact maps generated from Hi-C data analysis pipelines [59]. The Circos plots were visualized in Cytoscape [139].

Application 1: To determine if GrapHi-C can recapitulate the differences between wild-type and mutant fission yeast genome organization identified by Mizuguchi *et. al* [108], the *999a* wild-type and the *rad21* mutant adjacency matrices were visualized in Cytoscape using an edge-weighted spring embedded layout with the default parameters (Figure 2.2A-D). Vertices along the periphery of the graph images correspond to genomic bins that represent centromere and telomere regions. Since these regions are highly condensed and repetitive (making the DNA difficult to assay and map), no interaction data was reported for them. All edges (corresponding to *cis*-, *trans*- and linear interactions) were used to generate the GrapHi-C images. To create the images in Figure 2.2, nodes were manually coloured according to their corresponding chromosome and edges were hidden or revealed to highlight the *cis*- and *trans*-interactions.

For comparison, heatmaps (Figure 2.2E, F) and Circos plots (Figure 2.2G, H) for the *999a* wild-type and the *rad21* mutant contact maps were generated. These images represent the standard, existing approach for the visualization of Hi-C data. They clearly demonstrate how the traditional forms of visualization do not intuitively represent the complex organization and folding of the genome in 3D space (Figure 2.2E-H). This makes it challenging to generate hypotheses about how differences in the wild-type and mutant contact maps are reflected in genome organization. On the contrary, the GrapHi-C visualizations (Figure 2.2A-D) clearly highlight the loss of structural globules (intra-chromosomal structures) and the greater intermingling of chromosomes in the mutant strain that was described in the original study [108]. The resultant visualizations for the *rad21* mutant strain (Figure 2.2B, D) appear to be very similar since the mutant contact map has smaller interaction frequency values (as compared to the wild type) due to a greater intermingling of chromosomes. This results in the nodes being placed closer together in the edge-weighted spring embedded layout. The *rad21* interaction frequency values are not scaled in order to maintain consistency when comparing the wild-type and mutant strain visualizations. Overall, the GrapHi-C visualizations made

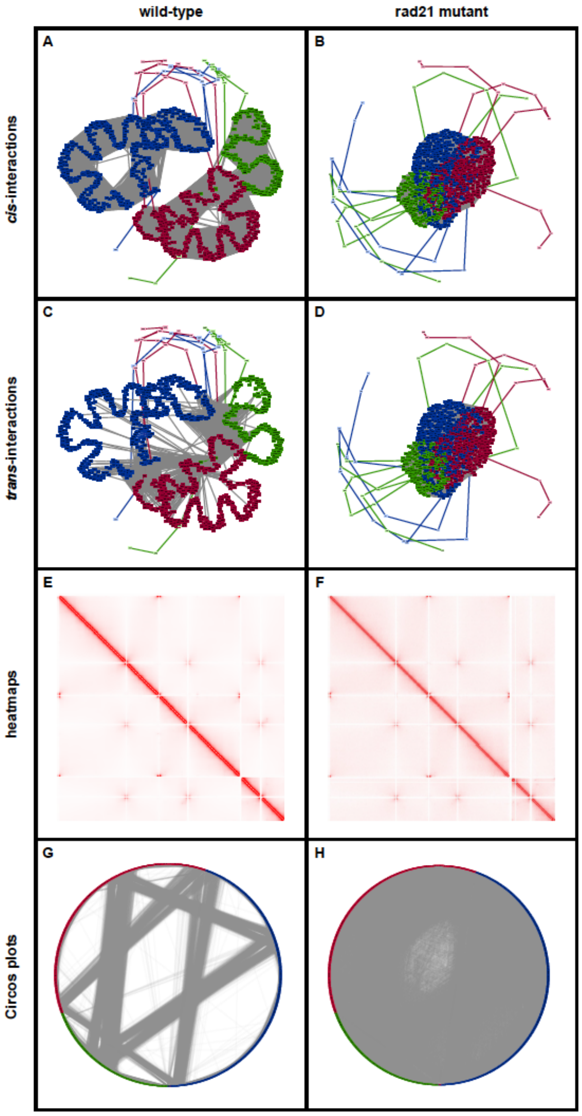


Figure 2.2: Comparison of GrapHi-C Visualizations, Heatmaps and Circos plots. Visualizations of the contact maps for the fission yeast 999a wild-type and rad21 mutant are displayed in the left and right columns, respectively. **a–d** The GrapHi-C visualizations where vertices and linear interactions were coloured according to their corresponding chromosome (chromosome 1: blue, chromosome 2: red, chromosome 3: green). The grey dashed lines represent cis-interactions (**a, b**) and trans-interactions (**c, d**). Graphs were visualized in Cytoscape using an edge-weighted spring embedded layout. **e, f** The heatmaps generated with Java Treeview that correspond to the contact maps. The opacity of a cell is directly related to the frequency of the interaction. **g, h** The Circos plots that correspond to the contact maps. Circos plots were visualized in Cytoscape. Vertices were coloured according to their corresponding chromosome (chromosome 1: blue, chromosome 2: red, chromosome 3: green) and grey lines represent an interaction between two vertices.

it easier to quickly identify the principles of genome organization and associated biological effects of the *rad21* yeast mutant that were discovered in the original study. This suggests the developed graph-based representation and structural visualization is a valid way to represent contact maps.

Application 2: To determine if the GrapHi-C protocol was able to identify the same cell cycle dependent alterations in genome organization described by Tanizawa *et. al* [152]. GrapHi-C images, based on the normalized contact maps for each time point, were visualized in Gephi using the ForceAtlas2 layout [70] (Figure 3.3 – column 1). Similarly to Application 1, all edges (corresponding to *cis*-, *trans*- and linear interactions) were used to generate the images. Nodes were manually coloured according to their corresponding chromosome and edges were hidden in the exported images for simplicity. GrapHi-C images containing all the edges for the 40, 60, 80 and 120 minute time points are provided in Additional Files 2-5. For comparison, the heatmaps (Figure 3.3 – column 2) and Circos plots (Figure 3.3 – column 3) were generated based on the normalized contact maps for each cell cycle time point. As mentioned previously, these images represent the standard, existing approach for Hi-C data visualization.

In the original study, Tanizawa *et. al* [152] established that throughout the cell cycle, small domains approximately 50 kB in size are consistently present. During mitosis (M) the DNA condenses resulting in more *cis*-chromosomal interactions and fewer *trans*-chromosomal interactions. Not only are the GrapHi-C visualizations able to recapitulate these identified cell cycle dependant genomic alterations, they also intuitively highlight established features of fission yeast genomic organization. For instance, during fission yeast interphase (comprised of the G1, S and G2 phases) the chromosomes are organized in a polarized arrangement (*Rab1*-like configuration) where the centromeres of all three chromosomes are clustered at one end of the nucleus and the telomeres of chromosomes 1 and 2 cluster at the opposite end near the nuclear periphery [107]. Additionally, microscopy techniques have established that all three chromosomes are organized into distinct chromosome territories within the nucleus at all phases of the cell cycle [107, 109, 131]. These hallmarks of fission yeast genome organization are distinctly recapitulated in the GrapHi-C visualizations. Furthermore, the GrapHi-C visualizations clearly represent the established condensation of chromosomal DNA

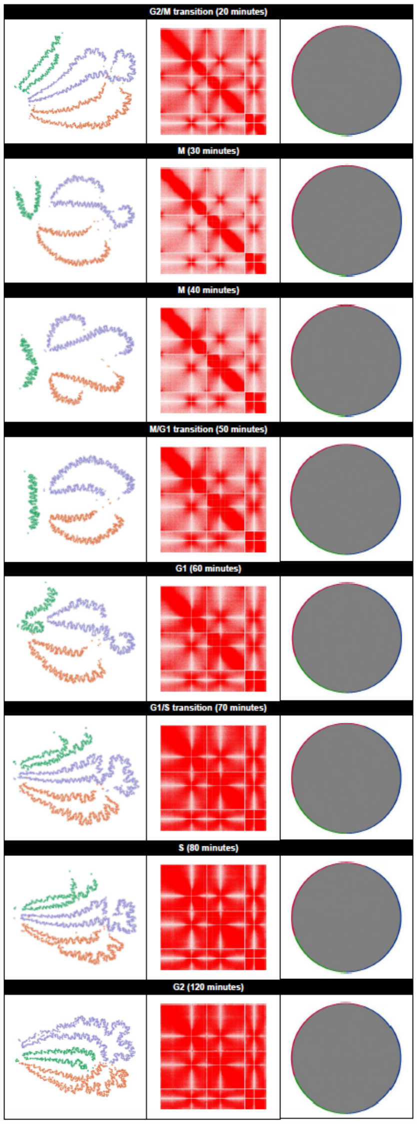


Figure 2.3: GrapHi-C Visualizations for Fission Yeast Contact Maps at Various Stages of the Cell Cycle. Cell cycle labels and the corresponding time points are given on the top of each panel. GrapH-C images are presented in column 1. In these images, vertices were coloured according to their corresponding chromosome (chromosome 1: purple, chromosome 2: orange, chromosome 3: green) and edges were hidden. Graphs were visualized in Gephi using the ForceAtlas2 layout. The corresponding heatmaps generated by Java Treeview are in column 2. In these heatmaps, the opacity of a cell is directly related to the frequency of the interaction. Column 3 contains the Circos plots that correspond to the contact maps. Circos plots were visualized in Cytoscape. Vertices were coloured according to their corresponding chromosome (chromosome 1: blue, chromosome 2: red, chromosome 3: green) and grey lines represent an interaction between two vertices. Note that the area inside the circle appears to be solid grey due to the number (and subsequent density) of interactions in these datasets.

during mitosis and de-condensation during interphase [107]. Overall, GrapHi-C provides a more intuitive representation of how the chromosomes are organized within the nucleus during different phases of the cell cycle. The resultant images are informative additions which support the in-depth analysis performed in the original study.

2.4 Conclusion

In this manuscript, we provide a protocol called GrapHi-C (pronounced “graphic”) for visualizing Hi-C data as a graph and developed a mathematical model for graph-based representations of contact maps. In addition to edges that represent the detected *cis*- and *trans*-interactions, we chose to include edges between each sequential genomic bin within a chromosome to better represent the linear extent of the genome. We developed a Perl script that can be used to convert a contact map into an adjacency matrix related to the developed graph-based representation. This matrix can then be input into a tool like Cytoscape or Gephi for structural visualization. Even though the graph-based representation seems straightforward, it is still novel in the genome structure community.

Overall, the developed GrapHi-C visualizations of the contact maps (compared to the equivalent heatmaps and Circos plots) made it easier to quickly identify the changes in genome organization identified in previous studies. Future work will focus on extending this visualization to allow for the vertices to be coloured according to complementary -omics datasets (such as gene expression, epigenetic markers or transcription factor binding sites) and produce a 3D graph-based visualization. Additionally, we will apply it to organisms with larger genomes to determine how well it scales to larger contact maps. Not only does the graph-based representation of Hi-C data lead to a more intuitive visualization, it also has the potential to lead to new ways of analyzing contact maps by leveraging tools and results from graph theory.

2.5 Limitations

GrapHi-C has only been tested on Hi-C datasets from haploid organisms – it should also be applied to organisms with higher ploidies to establish the robustness of the workflow. Additionally, GrapHi-C needs to be tested on an unfavourable Hi-C dataset that contains a multitude of disparate proximity relationships. Finally, the effect of visualizing a Hi-C dataset with technical problems needs to be established.

2.6 Supplemental Information

2.6.1 Acknowledgements

Not Applicable.

2.6.2 Author's Contributions

KM and AK developed the formal graph model. CHE provided biological insight. KM implemented the Perl script. KM applied the Perl script and visualization protocol to the fission yeast datasets. KM wrote the manuscript. AK and CHE edited the manuscript. All authors have read and approved the manuscript.

2.6.3 Competing Interests

The authors declare that they have no competing interests.

2.6.4 Consent for Publication

Not applicable.

2.6.5 Ethics Approval and Consent to Participate

Not applicable.

2.6.6 Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada [RGPIN 37207 to AK, Vanier Canada Graduate Scholarship to KM].

2.6.7 Availability of Data and Materials

The datasets supporting the conclusions of this article are available in the Gene Expression Omnibus database, [accession number: GSE56849; `https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE56849`, GSE93198; `https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE93198`]. The specific sample numbers for each application follow.

- Application 1:
    - *999a* (GSM1379427)
    - *rad21* (GSM1379430)
- Application 2:
    - 20 minutes (GSM2446256)
    - 30 minutes (GSM2446257)
    - 40 minutes (GSM2446258)
    - 50 minutes (GSM2446259)
    - 60 minutes (GSM2446260)
    - 70 minutes (GSM2446261)
    - 80 minutes (GSM2446262)
    - 120 minutes (GSM2446263)

Software Information
Project Name: GrapHi-C (pronounced “graphic”)
Project Home Page: `https://github.com/kimmackay/GrapHi-C`
Archived Version: v1.0.0

Operating System(s): Platform Independent

Programming Language: Perl

Other Requirements: Not Applicable

2.6.8 Archived Software

Additional File 1

Program 2.1: Perl script used for converting a contact map into an adjacency matrix based on the graph representation in Figure 2.1A.

```
#!/usr/bin/perl
## will generate an adjacency graph that can be input into
## cytoscape to visualize a Hi-C dataset
##
## argument 1: the normalized whole-genome contact map
## argument 2: total number of vertices (number of genomic bins)
## argument 3: number of chromosomes
## argument 4: the value you would like to use to define a linear
##             interaction frequency (experimental resolution)
## argument 5: a value to scale the interaction frequencies from the
##             whole-genome contact map (enter 1 if you wish to not
##             scale the values)
## argument 6: ’C’ or ’G’ to specify cytoscape or gephi visualization
## argument 7: the output file name
##
## Kimberly MacKay
## last updated: March 23, 2017
## license: This work is licensed under the Creative Commons
## Attribution-Non Commercial-ShareAlike 3.0 Unported License.
## To view a  copy of this license, visit
## http://creativecommons.org/licenses/by-nc-sa/3.0/ or send a letter
## to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

use strict;
use warnings;

## check to ensure seven arguments were passed in
die "ERROR: must pass in seven arguments." if @ARGV != 7;

```

```
my $hic_file = $ARGV[0];
my $num_nodes = $ARGV[1];
my $num_chr = $ARGV[2];

## define the value of a linear frequency
my $linear_freq = $ARGV[3];

## value to scale the interaction frequencies by
my $scale = $ARGV[4];

## determine the visualization tool to be used
my $viz_tool = $ARGV[5];

## get the output file name
my $out_file_name =  $ARGV[6];

## get the values for the start and end of each chromosome
## from the user
my @chr_start;
my @chr_stop;

print "enter the starting genomic bin number for each chromosome.
    Enter d when complete: ";

my $input = <STDIN>;
chomp $input;

while(!($input =~ /d/))
{
  # add the input to the array
  push @chr_start, $input;

  # get the next input
  $input = <STDIN>;
  chomp $input;
}

print "enter the ending genomic bin number for each chromosome.
    Enter d when complete: ";

$input = <STDIN>;
chomp $input;

while(!($input =~ /d/))
{
  # add the input to the array
  push @chr_stop, $input;

  # get the next input
  $input = <STDIN>;
  chomp $input;
}

## gut check: ensure the size of the arrays is the same
die "ERROR: the number of chromosome start and end
    positions should be equal." if $#chr_start != $#chr_stop;

#####################################################################
## print out the linear interactions
#####################################################################

## open the output file to print
open(my $out, '>', $out_file_name)
      or die "Could not open $out_file_name";

## print the first line of the output file
## if it is for a cytoscape visualization, print out a different
## header line then the header required for Gephi visualization
if($viz_tool eq "C")
{
  print $out "source_node\tsink_node\tinteraction_type
      \tassociated_freq\tsource_chr\tsink_chr\tlinear_edge_chr\n";
}
elsif($viz_tool eq "G")
{
  print $out "Source\tTarget\tinteraction_type\tWeight\n";
}

## for each chromosome
for(my $chr = 1; $chr <= $num_chr; $chr++)
{
  for(my $j = $chr_start[$chr-1]; $j < $chr_stop[$chr-1]; $j++)
  {
    ## if it is a cytoscape visualization, print out the inverse of
    ## the linear frequency and relevant information
    if($viz_tool eq "C")
    {
      print $out "bin".$j."\tbin".($j+1)."\tlinear\t".1/$linear_freq.
          "\t".$chr."\t".$chr."\t".$chr."\n";
    }
    ## if it is a gephi visualization, just print out the linear
    ## frequency (it will be inverted by the force atlas 2 layout)
    ## and relevant information
    elsif($viz_tool eq "G")
    {
      print $out $j."\t".($j+1)."\tlinear".$chr."
          \t".$linear_freq."\n";
    }
  }
}

close $out;
```

```

#####################################################################
## print out the non-linear interactions
#####################################################################

## open the interaction matrix file
open WGCM, "$hic_file"
     or die "ERROR: $hic_file could not be opened.";
chomp(my @hic_matrix = <WGCM>);
close WGCM;

## convert the decimals to integers and store them in a new array
## note: the 0th row and column of freq will be empty allow for a
## more natural parsing later on
my @frequencies;

## for each line after the header line
for(my $row = 1; $row <= $#hic_matrix; $row++)
{
  ## split the line
  my @matrix_line = split /\t/, $hic_matrix[$row];

  ## loop through the entire file to extract the frequencies
  for(my $col = 1; $col <= $num_nodes; $col++)
  {
    ## adjusts NA's to 0's
    if($matrix_line[$col] =~ "NA")
    {
      $frequencies[$row][$col] = 0;
    }
    else
    {
      ## just store the (potentially scaled) interaction frequency
      $frequencies[$row][$col] = $matrix_line[$col]*$scale;
    }
  }
}

## re-open the output file to append to it
open($out, '>>', $out_file_name)
    or die "Could not open $out_file_name";

## loop through one half of the matrix and print out the edges
## avoid the diagonal to prevent self-self interactions
for(my $row = 1; $row <= $#frequencies; $row++)
{
  for(my $col = $row+1; $col <= $#frequencies; $col++)
  {
    ## if it is a non-zero frequency
    if($frequencies[$row][$col] != 0)
    {
      ## get the source and sink chr numbers
      my $source_chr;
      my $uknown = 1;

      for(my $j = 0; $j <= $#chr_stop && $uknown; $j++)
      {
        if($row <= $chr_stop[$j])
        {
          $source_chr = $j+1;
          $uknown = 0;
        }
      }

      my $sink_chr;
      $uknown = 1;

      for(my $j = 0; $j <= $#chr_stop && $uknown; $j++)
      {
        if($col <= $chr_stop[$j])
        {
          $sink_chr = $j+1;
          $uknown = 0;
        }
      }

      ## check if it is a intra or inter interaction
      if($source_chr == $sink_chr)
      {
        ## print the intra-interaction
        ## if it is a cytoscape visualization
        if($viz_tool eq "C")
        {
          print $out "bin".$row."\tbin".$col."\tintra-interaction\t"
              .$frequencies[$row][$col]."\t".$source_chr."\t"
              .$sink_chr."\t0\n";
        }
        ## if it is a gephi visualization
        elsif($viz_tool eq "G")
        {
          print $out $row."\t".$col."\tintra-interaction\t".
              $frequencies[$row][$col]."\n";
        }
      }
      else
      {
        ## print the inter-interaction
        ## if it is a cytoscape visualization
        if($viz_tool eq "C")
        {
          print $out "bin".$row."\tbin".$col."\tinter-interaction\t"
              .$frequencies[$row][$col]."\t".$source_chr."\t"
              .$sink_chr."\t0\n";
        }
        ## if it is a gephi visualization
        elsif($viz_tool eq "G")
        {
          print $out $row."\t".$col."\tinter-interaction\t".
              $frequencies[$row][$col]."\n";
        }
      }
    }
  }
}
close $out;
```

2.6.9 Supplemental Figures

Additional File 2

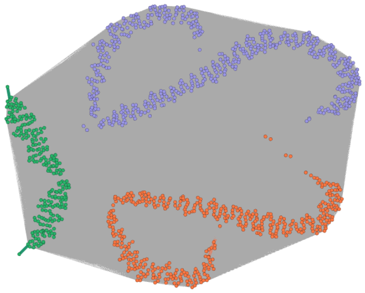


Figure 2.4: GrapHi-C Visualization for Fission Yeast Contact Map During M Phase (40 minutes). In this image, vertices were coloured according to their corresponding chromosome (chromosome 1: purple, chromosome 2: orange, chromosome 3: green). The *cis*- and *trans*-interactions edges are depicted with grey lines. Due to the number (and subsequent density) of these lines, these appear to be a solid grey area. The graph was visualized in Gephi using the ForceAtlas2 layout.

Additional File 3

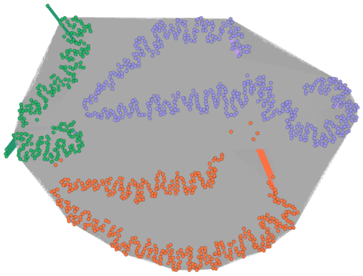


Figure 2.5: GrapHi-C Visualization for Fission Yeast Contact Map During G1 (60 minutes). In this image, vertices were coloured according to their corresponding chromosome (chromosome 1: purple, chromosome 2: orange, chromosome 3: green). The *cis*- and *trans*-interactions edges are depicted with grey lines. Due to the number (and subsequent density) of these lines, these appear to be a solid grey area. The graph was visualized in Gephi using the ForceAtlas2 layout.

Additional File 4

Figure 2.6: GrapHi-C Visualization for Fission Yeast Contact Map During S Phase (80 minutes). In this image, vertices were coloured according to their corresponding chromosome (chromosome 1: purple, chromosome 2: orange, chromosome 3: green). The *cis*- and *trans*-interactions edges are depicted with grey lines. Due to the number (and subsequent density) of these lines, these appear to be a solid grey area. The graph was visualized in Gephi using the ForceAtlas2 layout.

Additional File 5

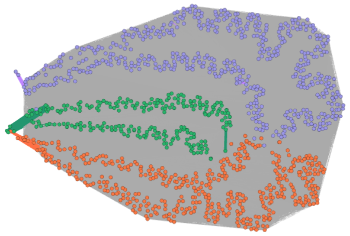


Figure 2.7: GrapHi-C Visualization for Fission Yeast Contact Map During G2 (120 minutes). In this image, vertices were coloured according to their corresponding chromosome (chromosome 1: purple, chromosome 2: orange, chromosome 3: green). The *cis*- and *trans*-interactions edges are depicted with grey lines. Due to the number (and subsequent density) of these lines, these appear to be a solid grey area. The graph was visualized in Gephi using the ForceAtlas2 layout.

Chapter 3

Computational Methods for Predicting 3D Genomic Organization from High-Resolution Chromosome Conformation Capture Data

Kimberly MacKay, and Anthony Kusalik

Citation [99]: K MacKay and A Kusalik. Computational methods for predicting 3D genomic organization from high-resolution chromosome conformation capture data. *Briefings in Functional Genomics*. Accepted as of February 7, 2020. MS #: BFGP-19-0049.R1. The version of the manuscript presented here represents the most recent revision as of January 29, 2020.

Detailed Contributions: Kimberly MacKay was responsible for writing the manuscript and synthesizing the information. A brief description of the contributions for all listed authors can be found in Section 3.13.3.

As a reminder, the overarching goal of this thesis is to develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction. This chapter provides a survey of the existing tools for 3D genome prediction.

3.1 Abstract

The advent of high-resolution chromosome conformation capture assays (like 5C, Hi-C and Pore-C) has allowed for unprecedented sequence-level investigations into the structure-function relationship of the genome. In order to comprehensively understand this relationship, computational tools are required that utilize data generated from these assays to predict 3D genome organization (the 3D genome reconstruction problem). Many computational tools have been developed that answer this need but a comprehensive comparison of their underlying algorithmic approaches has not been conducted. This manuscript provides a comprehensive review of the existing computational tools (from November 2006 to September 2019, inclusive) that can be used to predict 3D genome organizations from high-resolution chromosome conformation capture data. Overall, existing tools were found to use a relatively small set of algorithms from one or more of the following categories: dimensionality reduction, graph/network theory, maximum likelihood estimation and statistical modelling. Solutions in each category are far from maturity and the breadth and depth of various algorithmic categories have not been fully explored. While the tools for predicting 3D structure for a genomic region or single chromosome are diverse, there is a general lack of algorithmic diversity among computational tools for predicting the complete 3D genome organization from high-resolution chromosome conformation capture data.

Key Phrases

Genome Organization, 3D Genome Prediction, 3D Genome Reconstruction Problem, High-Resolution Chromosome Conformation Capture Data, Hi-C, 5C

3.2 Introduction

This manuscript provides a survey of the existing computational tools that can be used for predicting 3D genomic organization from high-resolution chromosome conformation capture data. Relevant biological and computational background is provided in Section 3.3. Section 3.4 describes the 3D genome reconstruction problem (3D-GRP) formalism. Section 3.5 provides an overview of existing tools for solving the 3D-GRP. Two of these existing tools (one consensus and one ensemble) are described in more detail in Section 3.6. Similarly, Section 3.7 provides an overview of existing tools for solving the related, but simpler, problem of predicting 3D organization for a single chromosome or genomic region. An exemplar consensus and ensemble tool for solving this simpler problem are discussed in more detail in Section 3.8. Finally, a discussion of the shortcomings of the existing approaches and future research directions can be found in Section 3.9.

3.3 Background

Like many areas of biology, the relationship between genomic structure and function is closely linked [10]. Alterations in the 3D organization of chromosomes have been demonstrated in a wide variety of nuclear and cellular processes, including DNA translocation [10], differentiation [83], serum response [103], therapeutic response [104] and response to DNA damage [105]. The unique spatial organization of the genome that is seen under these different cellular conditions is hypothesized to be a crucial mechanism driving various nuclear and cellular functions. It has been theorized that this dynamic organization of the genome may be driven by global regulation of gene expression (or vice-versa) [6, 23, 27, 34, 91] since 3D genome organization has been shown to facilitate interactions between genes and their regulatory elements [150, 169].

Traditionally, microscopy techniques have been utilized to visualize the spatial organization of chromosomes within the nucleus. While informative, they do not provide sequence-level information about the observed organizations [43]. Therefore other biological techniques must be used (either in combination or standalone) to allow for the sequence-level inference of 3D genomic organization. Many such biological techniques have been developed to assay the 3D genome organization at various sequence-level resolutions [10, 33, 38, 130]. In general, these techniques are able to determine whether a single (or multiple) pair(s) of genomic regions are in close 3D physical proximity. Genomic regions in close proximity are more commonly referred to as “interacting”.

Table 3.1: Biological techniques that can be used to assay 3D genome organization. Techniques are categorized based on the number of genomic regions they assay.

Category	Biological Technique(s)
one-by-one	chromosome conformation capture (3C) [36] and ChiP-loop [54]
one-by-all	circularized chromosome conformation capture (4C) [141, 170, 179]

many-by-many	chromosome conformation capture carbon copy (5C) [44]
many-by-all	Capture-C [66], Capture Hi-C (CHi-C) [45, 71], HiCap [128], Targeted Chromatin Capture (T2C) [79]
all-by-all	ChIA-PET [54], HiChIP [111], Hi-C [16, 74, 92], Pore-C [85] and Tethered Chromatin Capture (TCC) [74]

The biological techniques used for detecting 3D genomic organization can be broadly classified into the following categories based on the number of genomic regions they assay: *one-by-one* (used to detect an interaction between a single pair of genomic regions); *one-by-all* (used to detect all the interactions between one genomic region and the rest of the genome); *many-by-many* (used to detect interactions between many genomic regions and many other loci, where *many* is the number of loci on a chip or microarray); *many-by-all* (used to detect interactions between many genomic regions and the rest of the genome); and *all-by-all* (used to detect all the interactions occurring between mappable regions of the genome). Table 3.1 provides the specific names and citations for some of the biological techniques in each of these categories. Briefly, these techniques all follow five general steps (with slight modifications): (1) chemical cross-linking, (2) fragmentation, (3) ligation, (4) reverse cross-linking, and (5) technique-specific detection. A visual overview of the general workflow for each technique can be found in the review by Denker and de Laat [38]. Additional information regarding the biological background for these techniques can be found in the review recently published by Han *et al.* [57]. For the purpose of this manuscript, "high-resolution chromosome conformation capture" (HR-3C) will refer to the *many-by-many*, *many-by-all* and *all-by-all* techniques.

Algorithms for predicting 3D genome structure utilize a set of pairwise interactions and associated frequencies as input. Typically, this data is extracted from the results of a *many-by-many*, *many-by-all* or *all-by-all* (HR-3C) assay. The *one-by-one* and *one-by-all* techniques do not generate enough pairwise data points to allow for an accurate prediction of 3D genomic structure on their own. It is possible that the data from a *one-by-one* or *one-by-all* assay could be combined with data from a HR-3C assay and used as input to a 3D prediction algorithm, but this is not common practise in the field. The following paragraphs present a brief overview of how a set of pairwise interactions and associated frequencies can be extracted from a HR-3C assay's raw data (sequencing reads).

In general, HR-3C techniques utilize next-generation sequencing technologies to identify the sequences of interacting regions of the genome. Once these sequencing reads are generated, they are typically processed through a read mapping and filtering pipeline like HiCUP [167]. Briefly, this process involves quality control, read-splitting and independent read-mapping. Mapping sequence reads to a reference genome results in the generation of a matrix called a whole-genome contact map. A whole-genome contact map is a $N \times N$ matrix, where N is the number of genomic "bins" where each bin represents a contiguous sequence of linear DNA [86, 101, 177]. In general, the size of the whole-genome contact map (the number of genomic bins) is approximately equal to the total genome size divided by the assay's experimental resolution. Each cell ($A_{i,j}$) of a whole-genome contact map (A) indicates the count of how many times genomic bin i has been found to interact with genomic bin j. These counts are symmetric along the diagonal (i.e. $A_{i,j} = A_{j,i}$) and are often referred to as the frequency of the interaction between A_i and A_j (or interaction frequency).

After the whole-genome contact map is generated, interaction frequencies are normalized to correct for some of the inherent biases resulting from HR-3C experiments. These biases include (but are not limited to) discrepancies in DNA compaction or "visibility" [68], GC content [64, 171] and copy number variation [137]. Various computational methods have been developed to dampen these biases through normalization [29, 64, 77, 90, 145]. Most commonly, an iterative correction and eigenvector (ICE) decomposition [68] or Knight-Ruiz normalization [77, 90] is applied resulting in fractional interaction frequencies. ICE decomposition aims to achieve equal visibility across all genomic regions and results in relative interaction frequencies. Knight-Ruiz normalization performs matrix balancing resulting in fractional interaction frequencies where the rows and columns sum to 1. A comprehensive comparison of the normalization methods for HR-3C data has been recently published by Lyu *et al.* [96].

Downstream analysis of normalized whole-genome contact maps has uncovered unique genome-level patterns including distance-dependent interaction frequencies and more interactions between genomic regions on the same chromosome (*cis*-chromosomal interactions) than between regions on different chromosomes (*trans*-chromosomal interactions) [86, 92].

Further computational analysis of whole-genome contact maps has revealed the presence of various "hallmarks" of 3D genome organization. For instance, statistical analysis of whole-genome contact maps has revealed the presence of structural subunits called topologically associating domains (TADs) [41]. TADs are linear regions of DNA where interactions occur more frequently within the domain instead of between domains [41]. Originally, TADs were hypothesized to be structural building blocks for 3D genome organization but it has been determined that they serve no structural importance [32, 175].

Normalized whole-genome contact maps can also be used to infer a 3D structure of the genome (or a single genomic region). The process of predicting a model of the 3D genomic organization from a contact map is known as the 3D genome reconstruction problem (3D-GRP) [132] (described in more detail below). Many computational methods have been developed that utilize the data from HR-3C experiments to predict 3D genomic organization. Classically, existing programs have been broadly classified based on the number of genome models the method produces. Ensemble tools generate a collection of structures which represent the different genome organizations that may be present within a population of cells while consensus tools generate one structure which represents the population-averaged genome organization [86]. This manuscript provides a comprehensive review of the existing tools published from November 2006 (the year 5C was first described [44]) to September 2019 that use data extracted from HR-3C techniques to predict a 3D structure of complete genomes or a genomic region (Sections 3.5 and 3.7, respectively). A brief overview of the main chromosome models used by these existing tools is provided in Section 3.4 of this manuscript. The subsequent sections assume that the reader has some familiarity with the following concepts: multi-dimensional scaling [80, 81], shortest path algorithms [40, 51, 72, 166], expectation maximization [37], genetic algorithms [144], gradient descent (or ascent) [11], simulated annealing [76, 149], and Markov chain Monte Carlo sampling [58].

3.4 Problem Formalism

As mentioned above, the process of predicting a 3D genomic organization from HR-3C data is known as the 3D genome reconstruction problem (3D-GRP) [132]. It should be noted that the 3D-GRP has also been referred to as the 3D chromatin structure modelling problem [178] and that these two phrases can be used interchangeably. More formally, the 3D-GRP can be formulated as an optimization problem that tries to optimize the combined distance between multiple pairs of genomic regions. Informally, this is represented as a geometry problem [61] where the genomic bins are encoded as points and the goal is to find each point's (x, y, z) coordinates such that the pairwise distances between points best capture the corresponding interaction frequencies. It is assumed that, on average, a pair of genomic regions with a small interaction frequency will be further away in 3D space than a pair of genomic regions with a higher interaction frequency [6, 13, 14, 46, 52, 63, 87, 127, 163]. This relationship is often modelled through the following inverse function for a given pair of genomic regions (i and j): $dist_{i,j} = \frac{1}{A_{i,j}^{\alpha}}$ where $dist$ is the distance between the two genomic regions, $A_{i,j}$ is the corresponding normalized interaction frequency (a value between 0 and 1) from the whole-genome contact map, and α is an exponential factor with a value typically between 0.1 and 3.0 [156]. Most existing methods focus on finding the optimal value (or a set of values) for α and each point's (x, y, z) coordinates so that the computed distances closely recapitulate the original normalized frequencies from the whole-genome contact map [132]. Formally, the 3D-GRP can be defined in the following way when Euclidean distance is used:

Given a whole-genome contact map A with bins from $1..N$, determine α and each point's (x, y, z) coordinates such that

$$\begin{gathered} \text{for } i = 1..N \text{ and } j = 1..N \\ dist_{i,j} = \frac{1}{A_{i,j}^{\alpha}} \end{gathered} \tag{3.1}$$

and the sum

$$\sum_{i=1,j=1}^{N} \left| dist_{i,j} - \sqrt{\left((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right)} \right| \tag{3.2}$$

is minimized and

$$(x_i, y_i, z_i) \neq (x_j, y_j, z_j) \text{ where } 1 \leq i \leq N \ , \ 1 \leq j \leq N \ , \ i \neq j \tag{3.3}$$

In order to predict a 3D organization for a complete genome or genomic region, individual chromosomes need to be modelled as a set of points that can be assigned 3D coordinates. In general, existing methods use one of the following chromosome models. **(1)** Beads: each individual chromosome is represented as a collection of M beads where M is the number of genomic bins that constitute the linear extent of a chromosome. **(2)** Beads-on-a-String: again, each individual chromosome is represented as a collection of M beads. Unlike the beads model, "strings" of a fixed length are used to connect each pair of adjacent beads. Typically, these represent beads that are linearly adjacent on a chromosome. **(3)** Beads-on-a-Spring: this representation is similar to (2) but beads on an individual chromosome are connected with "springs" to represent the linear extent of a chromosome. Springs typically have a variable length that is based on attractive and repulsive forces of the connected beads. **(4)** Graph/Network: each bin from the whole-genome contact map is represented as a node in a network. Edges between nodes represent interactions from the contact map. Often, edges between bins on the same chromosome that are linearly adjacent are not included. **(5)** Polymer: each chromosome is represented as a line which is composed of consecutive line segments. Each line segment encodes a genomic bin or a genomic region that is delimited by two endonuclease restriction sites. **(6)** Piecewise curve: this is a mathematical formulation where each chromosome is represented as a set of connected 3D curves. Each curve represents an individual genomic bin or region.

3.5 Existing Tools for Solving the 3D-GRP

A comprehensive list of the existing computational tools for predicting 3D genomic organization from HR-3C data is available in Table 3.2. This table represents the majority of tools in the existing literature at the time of manuscript submission. Additional information regarding how these manuscripts were selected can be found in Section 3.13.1.

Table 3.2: Existing computational tools for predicting 3D genome organization from HR-3C data. Tools are categorized as either consensus or ensemble and then listed in alphabetical order. Tools marked with an asterisk (*) did not appear to be actively maintained at the time of manuscript submission. Column headings are as follows: Name, the tool's name or abbreviated reference (Panel labels from Figure 3.1 are provided in parentheses); Technique, the general algorithmic strategy employed; CHR Model, a description of the chromosome model utilized; Additional Data, any additional biological datasets required; *a priori* Constraints, a descriptor denoting whether *a priori* information is required and/or assumed; Language, the programming language used to implement the tool; Availability Mode, a description of how the tool was deployed; Website, a link to the tool. Abbreviations are as follows: MDS, Multi-Dimensional Scaling; MLE, Maximum Likelihood Estimation; LAD, lamin-associated domains; DamID, DNA adenine methyltransferase identification. IPOPT is a software library used for nonlinear optimization. A dash indicates "not applicable", "not available", or "none", as appropriate.

Name	**Technique**	**CHR Model**	**Additional Data**	***a priori* Constraints**	**Language**	**Availability Mode**	**Website**
				A. Consensus Methods			
Diament and Tuller (Figure 3.1A) [39]	MDS	Beads-on-a-String	Orthologous relationships	organism-specific (nuclear radius, elasticity between adjacent beads, minimum distance between homolgous CHRs, nucleolous position, CHR 12 position	C++ ; requires IPOPT	Source Code	`http://www.cs.tau.ac.il/~tamirtul/reconstruction.zip`
Duan *et al.* (Figure 3.1B) [6, 46, 151]	MDS	Beads-on-a-String	—	organism-specific (CHR12 position, nucleolous size and position); spatial (1000 nm nucleus, distance between beads (30 or 75 nm for cis-beads, or trans-beads, respectively)	—	—	—

Kapilevich *et al.* (Figure 3.1C) [75]	MDS and Genetic Algorithm	Graph	—	—	Python2	Source Code	`https://github.com/skkap/mdsga`
miniMDS (Figure 3.1D) [125]	MDS with Divide-and-Conquer Approach	Beads	—	dataset-specific (must have TADs)	Python2 or Python3	Source Code	`https://github.com/seqcode/miniMDS`
Segal and Bengtsson (Figure 3.1E,F) [132]	MDS	Graph	—	—	—	—	—
Stevens *et al.* (Figure 3.1G) [146]	Simulated Annealing	Beads-on-a-String	—	dataset-specific (must be from a single-cell)	Python2 or Python3	Source Code	`https://github.com/TheLaueLab/nuc_dynamics`
			B. Ensemble Methods				
Chrom3D (Figure 3.1H) [120]	Monte Carlo Optimization	Beads-on-a-String	LAD	dataset-specific (must have TADs)	C++	Source Code	`https://github.com/Chrom3D/Chrom3D`
Kalhor *et al.* (Figure 3.1I) [74]	Simulated Annealing	Beads	—	dataset-specific (must be from a diploid organism, cannot be pre-phased)	—	—	—
Li *et al.* (Figure 3.1J) [89]	Expectation Maximization	Beads	LAD, DamID (only applicable to fruit fly datasets)	organism-specific (nuclear radius, minimum distance between adjacent beads and homolgous CHRs); dataset-specific (must contain TADs)	Python2	Source Code	`https://github.com/alberlab/3DGenome_FruitFly/tree/v1.0.0`
LorDG* (Figure 3.1K) [156]	Gradient Ascent	Beads	—	—	—	—	`https://missouri.app.box.com/v/LorDG`

Tjong *et al.* (Figure 3.1L) [154]	Expectation Maximization	Beads	—	dataset-specific (must be from a diploid organism, cannot be pre-phased)	—	—	—
C. Consensus or Ensemble Methods							
3D-GNOME (Figure 3.1M) [147, 148]	Simulated Annealing or MDS (low-res); Polymer Physics (high-res)	Beads-on-a-Spring	—	dataset-specific (must be from organisms that contain CTCF motif)	—	Web Server	`http://3dgnome.cent.uw.edu.pl/`

In Table 3.2, the existing tools are categorized based on the number of predicted genome organizations they produce (i.e. ensemble vs consensus). Tools marked with an asterisk (*) did not appear to be actively maintained (DAAM) at the time of manuscript submission. This designation was given if the software presented in the original manuscript(s) could no longer be accessed. Typically, this was due to obsolete or nonfunctional website uniform resource locators (URLs). An example of the output produced by each tool can be found in Figure 3.1. In each case, the images were extracted from the corresponding original publication. Permission was obtained to reprint these images where required [1]. All of the existing tools utilize either heuristics or approximations in their solution.

Five of the seven consensus methods and three of the six ensemble methods listed in Table 3.2 provide access to the source code or a web interface. As mentioned in Table 3.2, the method developed by Stevens *et al.* only works with single-cell interaction data while all other methods accept interaction data from a population of cells. Currently, none of the available, actively maintained ensemble methods are usable for solving the 3D-GRP in the general case. This is because they rely on hypothesized “hallmarks” of genome organization like TADs, the presence of binding motifs for proteins often found at TAD boundaries (like CTCF) or require diploid, un-phased datasets to make their predictions. This is problematic since these genomic “hallmarks” have been shown to not exist in some organisms like *Arabidopsis thaliana* [42, 142]. This could pose a major barrier going forward as investigations into the 3D genomic organization of non-model organisms continues.

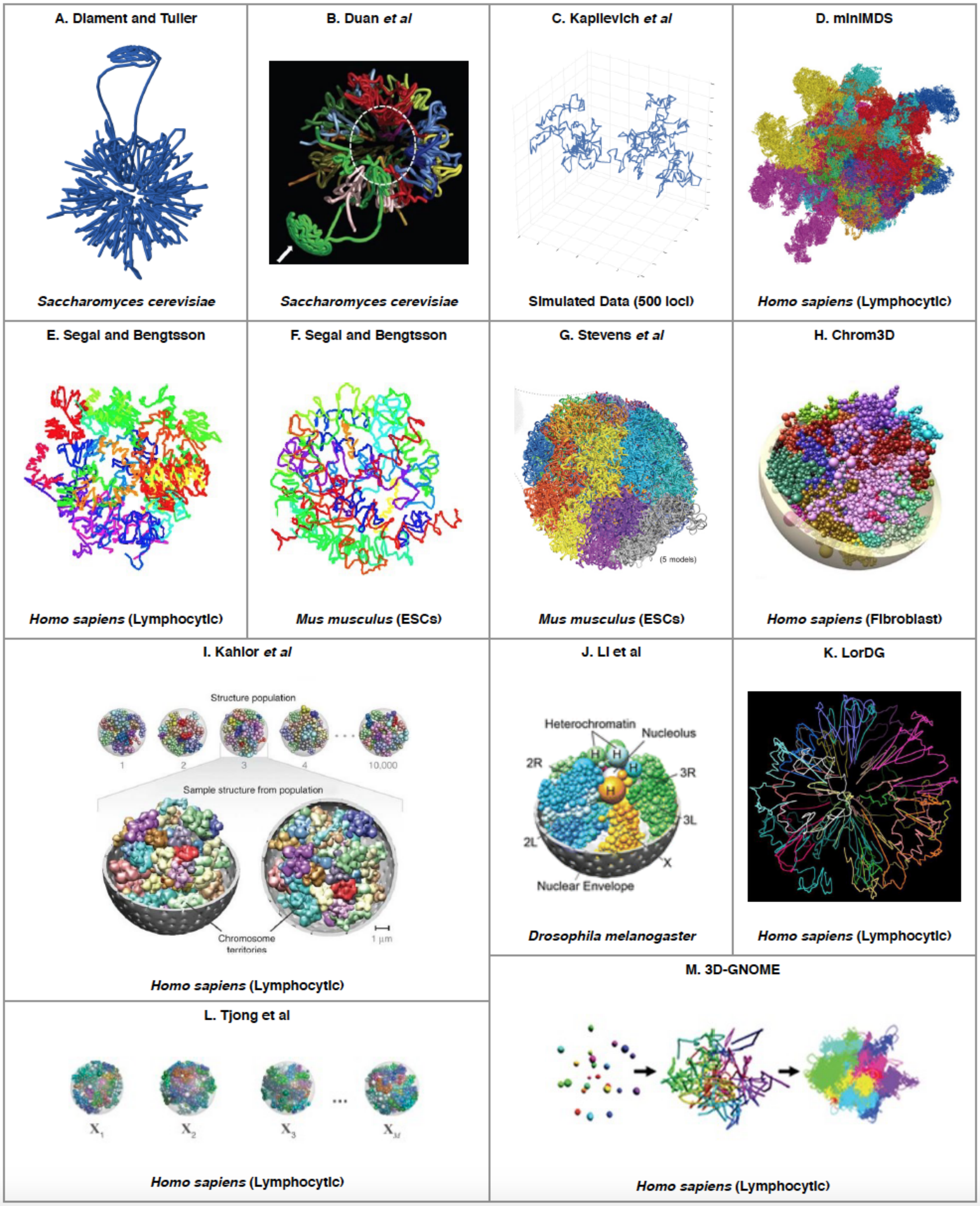


Figure 3.1: An example of a predicted 3D genome organization from each of the existing consensus (A-G) and ensemble (H-M) tools. Tool name or abbreviated reference can be found at the top of each panel and the organisms (and cell type, when applicable) are listed at the bottom of the panel. The abbreviation ESCs stands for embryonic stem cells. Permission was obtained to reprint these images where required [3.12].

3.6 Exemplar 3D-GRP Tools

The following section provides a more detailed discussion of an exemplar consensus and an exemplar ensemble method for solving the 3D-GRP. These methods were chosen since they were the most recent additions to the set of tools presented in Table 3.2 that have been used by the community to predict 3D genome structure based on real (rather than simulated) HR-3C datasets.

3.6.1 Consensus: miniMDS

miniMDS [125] is a consensus method that combines metric multi-dimensional scaling (MDS) with a divide-and-conquer approach to solve the 3D-GRP. Briefly and in general terms, the local structure of each chromosome is solved and then fitted to a low-resolution global genome prediction. First, a hidden Markov model is used to locally partition each chromosome into a set of subproblems. This hidden Markov model is derived from the TAD-finding algorithm developed by Dixon *et al.* [41] to identify local regions of a chromosome where edges of the region preferentially interact with the opposite side of the region. Each subproblem is then converted to a distance matrix based on equation 3.4 and metric MDS is used to solve a high-resolution local structure. It should be noted that the zero-distances (typically unmappable genomic regions) are ignored by MDS. This step is then repeated for each complete chromosome at a lower-resolution. High-resolution local structures are fitted to these lower-resolution chromosome structures using the Kabsch algorithm [73]. Finally, this fitting is repeated at an even lower resolution using the whole dataset to generate a low resolution global 3D structure. This global structure is then used as the final guide to position the chromosome structures resulting in a completed 3D genome prediction. An example of the output produced by miniMDS can be seen in Figure 3.1D. miniMDS should be used with caution in organisms where the existence of TADs or TAD-like structures has not been established since the hidden Markov model used for the initial division relies on the presence of TAD-like structures.

$$dist_{i,j} = \begin{cases} A_{i,j}^{-0.25} \text{ if } A_{i,j} > 0 \\ 0 \text{ if } A_{i,j} = 0 \end{cases} \tag{3.4}$$

3.6.2 Ensemble: Li *et al.*

The method developed by Li *et al.* [89] is an ensemble method that incorporates data from lamina-DamID experiments (which are able to detect interactions between the nuclear lamina and genomic regions) with HR-3C data to predict a 3D genomic organization at TAD-level resolution. The data from lamina-DamID experiments allows for the identification of which TAD regions interact with the nuclear envelope (the periphery of the nucleus; abbreviated NE). Briefly, this method uses maximum likelihood estimation to find a set of 3D genome structures that have statistically consistent TAD-TAD and TAD-NE interactions. Specifically, this method uses a variant of expectation maximization described by Tjong *et al.* [154] to optimize this joint probability. It incorporates additional spatial constraints into the optimization based on known features of the *Drosophila melanogaster* (fruit fly) genome. These Drosophila-specific constraints are based on microscopy imaging and include the nuclear radius, a maximum distance between chromosome copies, a maximum distance between adjacent TADs, links between heterochromatin regions, links between adjacent TADs, and centromere anchoring to the nucleolus. Due to these additional constraints, this method should only be applied to datasets from *Drosophila melanogaster* and would not be suitable for solving the 3D-GRP in the general case. An example of the output produced by this tool can be seen in Figure 3.1J. This method could potentially be applied to other organisms with TADs if the required organism-specific spatial constraints are available.

3.7 Predicting 3D Structures for Genomic Regions or Single Chromosomes

There are significantly more tools available that can be used to predict 3D structure of a single genomic region or chromosome from HR-3C data. For the purpose of this manuscript, we will refer to this as 3D regional prediction. The increased number of available tools for 3D regional prediction is likely because it is a much simpler (and often smaller) problem than the 3D-GRP since it does not have to take *trans*-chromosomal interactions (interactions between genomic regions on different chromosomes) into account. In the majority of cases, it would be computationally infeasible to apply these tools to the 3D-GRP due to their underlying time complexities. It may be possible to overcome this problem by applying a divide-and-conquer approach similar to miniMDS [125].

Table 3.3: Existing computational tools for predicting 3D organization for a genomic region from HR-3C data. Tools are categorized as either consensus or ensemble and then listed in alphabetical order. Tools marked with an asterisk (*) did not appear to be actively maintained at the time of submission. Column headings are as follows: Name, the tool's name or abbreviated reference (Panel labels from Figures 3.2 and 3.3 are provided in parentheses); Technique, the general algorithmic strategy employed; CHR model, a description of the chromosome model utilized; Additional Data, any additional biological datasets required; *a priori* Constraints, a descriptor denoting whether *a priori* information is required and/or assumed; Language, the programming language used to implement the tool; Availability Mode, a description of how the tool was deployed; Website, a link to the tool's source code. Abbreviations are as follows: IMP, Integrative Modeling Platform (`https://integrativemodeling.org/`); MDS, Multi-Dimensional Scaling; MLE, Maximum Likelihood Estimation; MCMC, Markov Chain Monte Carlo. A dash indicates "not applicable", "not available", or "none", as appropriate.

Name	Technique	CHR Model	Additional Data	*a priori* Constraints	Language	Availability Mode	Website
A. Consensus Methods							
AutoChrom3D* [122]	MLE	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	`http://ibi.hzau.edu.cn/3dmodel/`

BACH (Figure 3.2A) [63]	MCMC	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	C++	Source Code	`http://www.fas.harvard.edu/~junliu/BACH/`
Chiariello *et al.* (Figure 3.2B) [24]	Polymer Modelling (String & Binders Switch)	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—
ChromSDE (Figure 3.2C) [178]	Semi-Definite Programming	Graph	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—
5C3D* [50, 53]	Gradient Descent	Beads (Distributed in a Cube)	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	`http://dostielab.biochem.mcgill.ca/`
HSA* [182]	Simulated Annealing with MCMC	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	`http://ouyanglab.jax.org/hsa/`
PASTIS (MDS - Figure 3.2D) [163]	MDS	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	Python2	Source Code	`http://projets.cbio.mines-paristech.fr/~nvaroquaux/pastis/`
PASTIS (NMDS - Figure 3.2D) [163]	Non-Metric MDS	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	Python2	Source Code	`http://projets.cbio.mines-paristech.fr/~nvaroquaux/pastis/`
PASTIS (PM1 - Figure 3.2D) [163]	MLE	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	Python2	Source Code	`http://projets.cbio.mines-paristech.fr/~nvaroquaux/pastis/`

PASTIS (PM2 - Figure 3.2D) [163]	MLE	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	Python2	Source Code	`http://projets.cbio.mines-paristech.fr/~nvaroquaux/pastis/`
RPR (Figure 3.2E) [61]	MDS & Recurrence Plots	Beads	—	dataset-specifc (must be from a single-cell, must be from a haploid organism or pre-phased)	Matlab	Source Code (embedded in a PDF)	`https://media.nature.com/original/nature-assets/srep/2016/161011/srep34982/extref/srep34982-s1.pdf`
ShRec3D (Figure 3.2F) [87, 110]	Shortest Path and MDS	Beads-on-a-Spring	—	organism-specific (nuclear radius), dataset-specific (must be from a haploid organism or pre-phased)	Matlab	Source Code	`https://sites.google.com/site/julienmozziconacci/home/softwares`
ShRec3D+ (Figure 3.2G) [88]	Shortest Path and MDS	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased), user-specific (value of for alpha)	—	—	—
SuperRec (Figure 3.2H) [177]	Shortest Path and MDS	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	Python and Linux Executable	Source Code	`http://www.cs.cityu.edu.hk/~shuaicli/SuperRec/`
3DChrom (Figure 3.2I) [95]	MDS	Beads	—	dataset-specific (must have TADs, must be from a haploid organism or pre-phased)	C++; requires IPOPT	Source Code or Web Server	`http://dna.cs.miami.edu/3DChrom/`

tPAM (Figure 3.2J) [116]	MCMC	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—
tREX* [117]	Truncated Random Effect Expression Model	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	`http://www.stat.osu.edu/~statgen/Software/tRex`
Zhang *et al.* (Figure 3.2K) [176]	MCMC	Piecewise Curve (Helical)	—	dataset-specific (must be from a haploid organism or pre-phased)	C++	Source Code	`https://rsquared1427.github.io/phm/`
B. Ensemble Methods							
BACH-MIX (Figure 3.3A) [63]	MCMC	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	C++	Source Code	`http://www.fas.harvard.edu/~junliu/BACH/`
Caudai *et al.* (Figure 3.3B) [22]	Simulated Annealing & MCMC	Beads	—	dataset-specific (must contain TADs, must be from a haploid organism or pre-phased)	Python2	Source Code	`https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-015-0667-0` (Additional File 2)
Chromsome3D (Figure 3.3C) [1]	Simulated Annealing	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	Perl	Source Code or Web Server	`http://sysbio.rnet.missouri.edu/chromosome3d/`
ChromStruct 4 (Figure 3.3D) [21]	Simulated Annealing & MCMC	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—

GEM (Figure 3.3E) [180]	Manifold Learning & Expectation Maximization	Graph	—	dataset-specific (must be from a haploid organism or pre-phased)	Matlab	Source Code	`https://github.com/mlcb-thu/GEM`
Giorgetti *et al.* (Figure 3.3F) [56]	Polymer Modelling	Beads-on-a-String	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—
IMP (Figure 3.3G) [13, 14, 157]	Monte Carlo Optimization	Beads	—	dataset-specific (must be from a haploid organism or pre-phased)	C++ or Python	Binary or Source Code	`https://integrativemodeling.org/`
InfMod3DGen (Figure 3.3H) [165]	Expectation Maximization	Polymer	—	dataset-specific (must be from a haploid organism or pre-phased)	Matlab	Source Code	`https://github.com/wangsy11/InfMod3DGen`
ISD (Figure 3.3I) [20]	Hamiltonian Monte Carlo	Beads	—	dataset-specifc (must be from a single-cell, must be from a haploid or diploid organism)	Python	Source Code	`https://github.com/michaelhabeck/isdhic`
MBO (Figure 3.3J) [119]	Shortest Path and MDS	Beads	—	dataset-specifc (must be from a single-cell, must be from a haploid or diploid organism)	Matlab	Source Code	`http://folk.uio.no/jonaspau/mbo/`
MCMC5C (Figure 3.3K) [127]	MCMC	Piecewise Curve (Linear)	—	organism-specific (homologous chromosomes must make mutually exclusive contacts)	—	—	—

Meluzzi and Arya (Figure 3.3L) [106]	Polymer Physics & Adaptive Filter Theory	Beads-on-a-Spring	—	dataset-specific (must be from a haploid organism or pre-phased)	—	—	—
Nagano *et al.* (Figure 3.3M) [112]	Simulated Annealing	Beads-on-a-String	—	dataset-specifc (must be from a single-cell, must be from a haploid organism or pre-phased)	—	—	—
TADbit (Figure 3.3O) [135, 157]	IMP	Beads	—	organisms-specific (must have TADs)	Python2	Source Code	`https://github.com/3DGenomes/tadbit`
Trieu and Cheng (Figure 3.3N) [155]	Gradient Descent	Beads	—	organism-specific (only applicable to datasets from Homo sapiens), dataset-specific (must be from a haploid organism or pre-phased)	—	—	—

Table 3.3 provides a list of the computational techniques that utilize HR-3C data to predict a 3D structure for a given genomic region instead of the whole genome. As described above, tools have been categorized in the following ways: DAAM (*), consensus, and/or ensemble. An example of the output produced by each actively maintained consensus and ensemble tool can be found in Figures 3.2 and 3.3, respectively. In each case, the images were extracted from the corresponding original publication. Permission was obtained to reprint these images where required [2] [3]

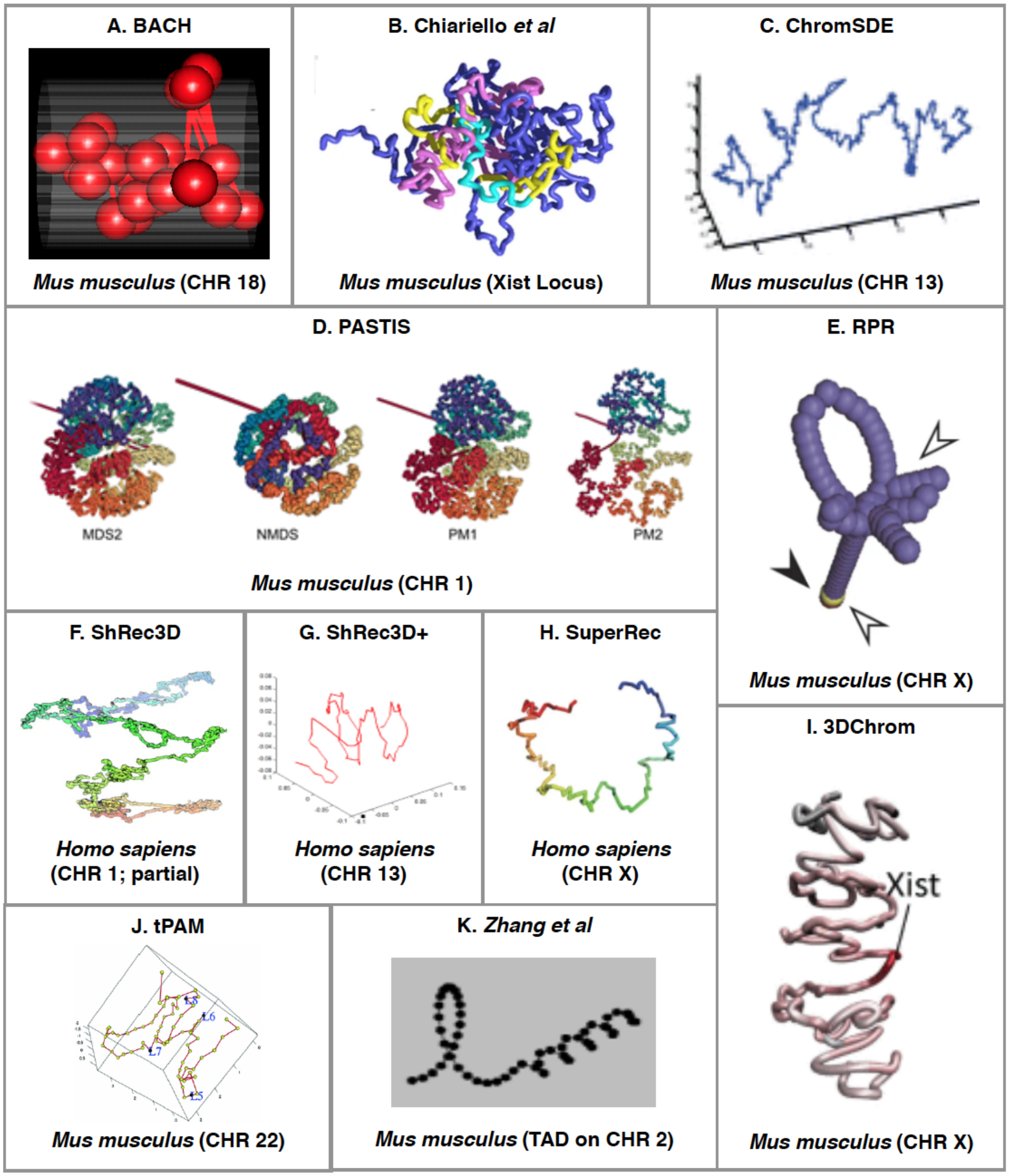


Figure 3.2: An example of a predicted region organization from each of the existing regional consensus tools. Tool name or abbreviated reference can be found at the top of each panel and the organisms (and specific region, when applicable) are listed at the bottom of the panel. The abbreviation CHR stands for chromosome. Permission was obtained to reprint these images where required $^{3.12}$.

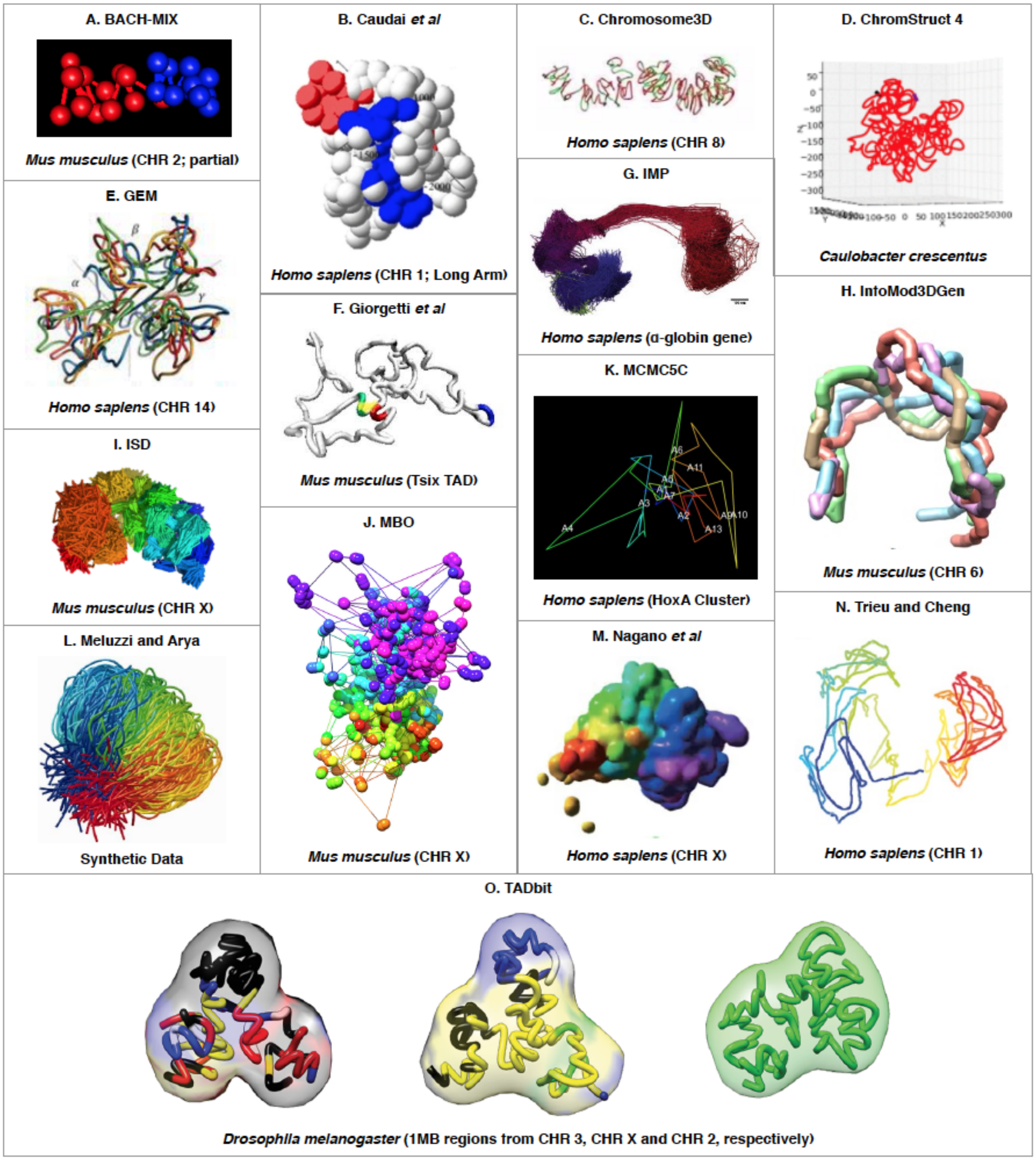


Figure 3.3: An example of a predicted region organization from each of the existing regional ensemble tools. Tool name or abbreviated reference can be found at the top of each panel and the organisms (and specific region, when applicable) are listed at the bottom of the panel. The abbreviation CHR stands for chromosome. Permission was obtained to reprint these images where required [3.12].

3.8 Exemplar Regional 3D Prediction Tools

The following section provides a more detailed discussion of an exemplar consensus and an exemplar ensemble method for predicting 3D structure of a single genomic region or chromosome. ShRec3D+ was chosen as the exemplar consensus method since it is the most recent version of one of the popular and highly cited tools, ShRec3D [87, 110]. Chromosome3D was chosen as the exemplar ensemble method since it is the most recent addition to set of ensemble tools presented in Table 3.3 (Chromosome3D) that has been used by the community to predict 3D regional structures (beyond TAD-level resolution) from real population-based HR-3C data (rather than simulated data).

3.8.1 Consensus: ShRec3D+

ShRec3D+ [88] is a consensus method that is based on ShRec3D [87, 110] and ChromSDE [178]. An overview of the approach taken by ShRec3D+ is as follows. First, interactions are converted into a weighted graph where edge weight (which represents the distance between two vertices) is initially calculated with equation 3.5 where α is a user-selected value between 0.0 and 2.0. Second, the Floyd-Warshall algorithm is applied to optimize the distances so that the vertices satisfy the triangle inequality. Finally, classical MDS is applied to calculate the (x, y, z) coordinates of each vertex in the graph. An example of the output produced by ShRec3D+ can be seen in Figure 3.2G. ShRec3D+ does not optimize the value of α like its predecessor ShRec3D. This was done to improve runtime but adds a significant potential for user error in new applications because the user might unintentionally specify an inappropriate value.

$$w_{i,j} = \begin{cases} A_{i,j}^{-\alpha} \text{ if } A_{i,j} > 0 \\ \infty \text{ if } A_{i,j} = 0 \end{cases} \tag{3.5}$$

3.8.2 Ensemble: Chromosome3D

Chromsome3D [1] is an ensemble method that models a genomic region as a string of beads. Interaction frequencies are converted to distances based on equation 3.6 where K is a scaling constant and α is a tuneable parameter with suggested values of 11 and 1/3, respectively. Simulated annealing is then used to find the $(x,\ y,\ z)$ coordinates for each bead such that the absolute difference between the predicted distances (based on the $(x,\ y,\ z)$ coordinates) and initial calculated distances (based on the interaction frequencies) are minimized. This is repeated twenty times to generate an ensemble of potential 3D genomic structures. This set of structures is ranked using Spearman's rank correlation coefficient to determine which predicted structures best represent the initially distances calculated based on the interaction frequencies. An example of the output produced by Chromosome3D can be seen in Figure 3.3C. Chromsome3D has been shown to outperform ShRec3D when the input data set is noisy [1], which is a characteristic of HR-3C datasets.

$$dist_{i,j} = \frac{K}{A_{i,j}^{\alpha}/average(A_{i,j}^{\alpha})} \tag{3.6}$$

3.9 Future Directions

There is a lack of algorithmically diverse solutions to the 3D-GRP that could be applied to a wide-variety of organisms (we refer to this as generalizable for this manuscript). Five of the six consensus methods use a MDS as a part of their approach for solving the 3D-GRP. MDS presents many potential issues which are described in Section 3.9.2. The remaining method by Stevens *et al.* can only be used with single-cell HR-3C data. Additionally, none of the available ensemble methods are usable for solving the 3D-GRP in the general case. Only two of the five ensemble methods provide source code and are actively maintained. These two methods also require additional biological datasets (DamID and/or LAD) for 3D genome prediction. These types of datasets are not commonly gathered with HR-3C assays; therefore, these solutions are not applicable in the general case. Finally, the web application 3D-GNOME only works with the pre-computed HR-3C datasets hosted on the website. As investigations into the 3D genome organization continue, it is possible that the existing tools can not be utilized for applications in organisms with larger, more complicated genomes (when compared to *Homo sapiens*). The reasons and potential solutions are described in the subsections below.

3.9.1 Computational Limitations

As mentioned previously, the current formulation of the 3D-GRP is a combinatorial optimization problem. Combinatorial optimization problems are known to be demanding in terms of computational resources like memory. This is potentially problematic because it adds an upper bound on the number of genomic bins that can be input into existing tools based on available computational resources. This could render certain 3D-GRP solutions impractical for generating high-resolution predictions and/or predictions from organisms with genomes larger than *Homo sapiens*. For instance, these computational limitations cause polymer models to have a genome size and/or resolution limit (i.e. number of "beads"). For these polymer modelling based solutions, the current upper bound on the number of genomic regions that can be predicted has been reported to be 10,000 [153]. The majority of the existing tools have a $O(N^3)$ time complexity since they rely on MDS and/or the Floyd Warshall algorithm. As the resolution of GR-3C data increases so does the value for N. This will necessitate investigations into more efficient approaches. Fortunately, combinatorial optimization problems have been extensively studied in computer science and many of the existing solutions for solving these types of problems could be leveraged in 3D-GRP solutions. Existing tools like miniMDS have utilized a divide-and-conquer approach to overcome the computational limitations [125]. It is expected that approaches like this (as well as others that take advantage of parallelism or distributed algorithms) will become more common as advances in HR-3C assays continue to allow researchers to obtain finer genomic resolutions. Future research should focus on establishing solutions that are more computationally efficient and/or take advantage of parallel or distributed algorithms to overcome the current computational limitations.

3.9.2 Increasing Algorithmic Diversity

Algorithms for solving the 3D-GRP are far from maturity [176]. While there is some algorithmic diversity in the set of existing tools, the full breadth and depth of solutions in each category have yet to be explored. As mentioned above, five of the six consensus methods use a MDS as a part of their approach for solving the 3D-GRP. Many issues have been noted pertaining to the use of MDS as a part of solutions to the 3D-GRP. For instance, because HR-3C assays represent a heterogeneous population of genome organizations, there is often not a single unique solution. Therefore, the distances calculated by MDS often conflict and cannot be accurately or completely calculated [1]. Furthermore, it is known that standard MDS techniques are inaccurate for sparse high-resolution data [125]. t-Stochastic neighbourhood embedding (tSNE) has been shown to be more accurate than MDS for datasets with these characteristics [159, 160, 161, 162] and is a promising technique for new 3D-GRP solutions.

All of the existing methods utilize Euclidean distances in their solutions to the 3D-GRP but the utility of other distance functions (such as relative Sorensen distances, Canberra distances and cosine (similarity) distances) could and should be investigated going forward to increase the accuracy of predicted models. This is especially pertinent in the case of solutions to the 3D-GRP since it is known that Euclidean distances are often not suitable for sparse, high-dimensional datasets [2] which is the case with many whole-genome contact maps. Finally, most of the existing tools model the chromosome as a set of beads or beads-on-a-string/spring. While this seems like a natural representation, the utility of other chromosome models should be investigated.

In general, there is a lack of algorithmic diversity in the existing set of tools for solving the 3D-GRP. Figure 3.4 provides a visual depiction of the different algorithmic strategies employed by 3D-GRP solutions (purple boxes), 3D regional prediction solutions (orange boxes) and both (green boxes). Additionally, we highlight a few algorithmic strategies that, to the best of our knowledge, have not yet been utilized for predicting 3D structures of the genome or genetic region (grey boxes). In our opinion, these represent promising areas of exploration for new tool development but there are many other algorithms and algorithm types well-suited for combinatorial optimization problems that could also be investigated.

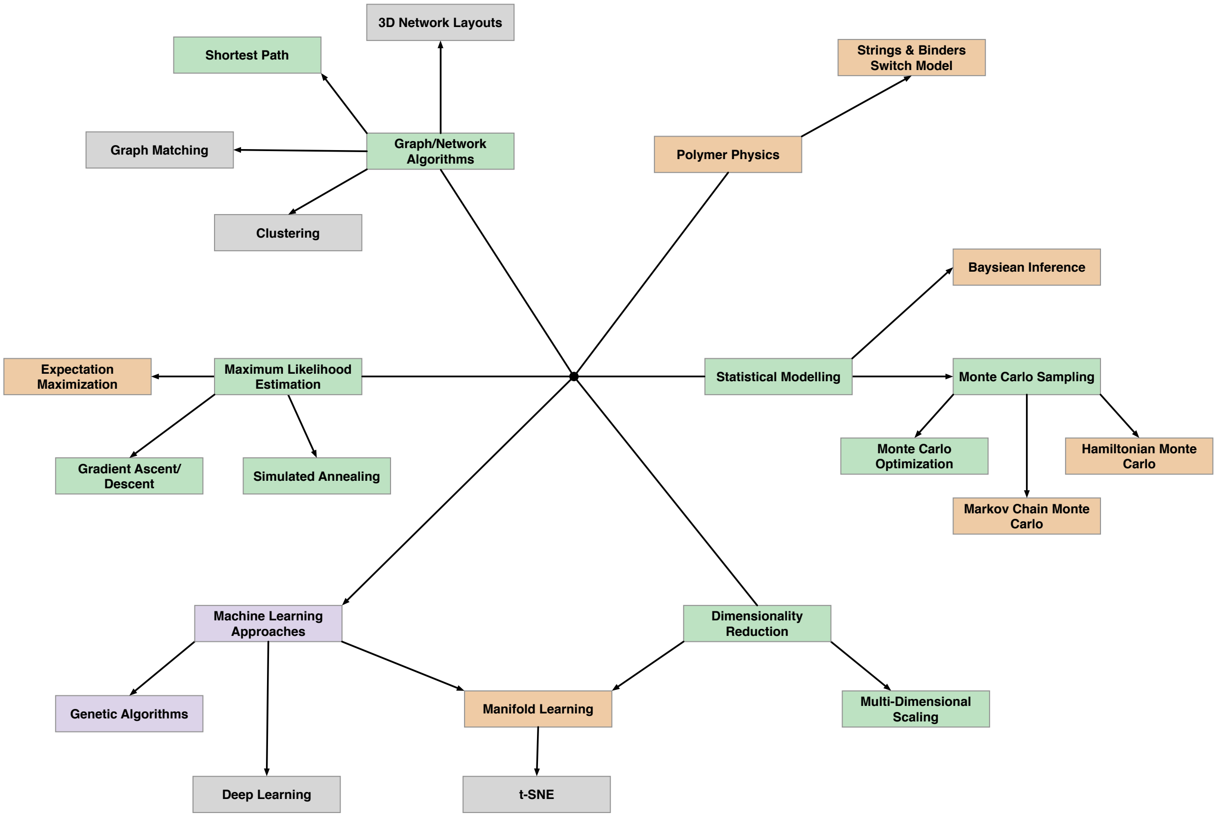


Figure 3.4: An overview of the algorithmic techniques used by existing tools for solving the 3D-GRP (purple boxes), 3D regional prediction (orange boxes) and both (green boxes). A small selection of unexplored algorithmic strategies are indicated with grey boxes. Lines originate at a black dot and represent the hierarchical relationship between each algorithmic approach (more general to more specific).

While the community has made great strides in developing solutions to the 3D-GRP, a lot of work remains to be done as investigations into the 3D genome organization of non-model organisms begins.

3.9.3 Applications to Other Organisms

An increase in algorithmic diversity is necessary to facilitate 3D genome analysis in non-model organisms. As mentioned previously, many methods rely on the presence of previously proposed “hallmarks” of genomic organization like TADs for prediction. This is troubling since the presence of these “hallmarks” has not been verified in a wide variety of organisms. For instance, recently it was found that TADs are not present in certain plant species like *Arabadopsis thaliana* [42] and are therefore not a conserved hallmark of genome organization. Methods like miniMDS that rely on TADs or TAD-like structures for efficient computation would not be applicable to organisms like *Arabadopsis thaliana*.

Many of the existing tools have only been utilized with data generated from standard model organisms such as *Saccharomyces cerevisiae*, *Mus musculus* or *Homo sapiens*. Table 3.4 presents an overview of the datasources that have been used by existing tools for solving the 3D-GRP. They are separated with black outlines into the following groups based on their origin: simulated data, parasite, virus, bacteria, yeast, insect, worm, fish, chicken, mice, primate, human, plant. Data sources used in the original manuscript are represented with a grey box. Applications of the tool were determined by reviewing all of the original publications citing articles. The exact number of articles reviewed for each tool is provided in the second column of Table 3.4. Valid applications of a tool in a different organism and/or dataset than the original paper are indicated with purple (successful) or orange (unsuccessful) boxes. There are many organisms that have Hi-C data available but 3D genomic predictions have not been performed with any of it (Table 3.4; white boxes). Interestingly, at the time of publication there were over 3200 Hi-C datasets deposited in the Gene Expression Omnibus dataset, but complete 3D genome prediction has only been applied to less than 10 unique datasets. This provides an interesting area of future exploration and application in the 3D genomics community.

None of the existing tools have been applied to organisms with a ploidy greater than 2. As such, it is not clear whether these tools can be effectively utilized for predicting 3D genome structure in organisms with higher ploidies such as *Triticum aestivum* (bread wheat; hexaploid) [181]. Additionally, many of the 3D regional tools do not effectively deal with datasets from polyploid organisms and therefore, could not be applied to polyploid datasets (or extended to solve the 3D-GRP irregardless of computational complexity). This can be seen when looking at the applications presented in the original manuscripts of the regional tools where most chose to use either a haploid organism, pre-phased data, or a genomic region from the X chromosome of male cells. How to effectively deconvolute interaction signals from

Table 3.4: An overview of the data sources that have been used by existing tools for solving the 3D-GRP. Tool name is provided in the first column and follows the same ordering presented in Table 3.2. The number of citations that were examined is given in column 2. Data sources are listed in the first row and have been separated (black outlines) into the following groups based on their origin: simulated data, parasite, virus, bacteria, yeast, insect, worm, fish, chicken, mice, primate, human, plant. Grey boxes represent the datasource that was used in the original manuscript. Applications of the tool in other organisms are indicated with purple (successful) or orange (unsuccessful) boxes. Datasets that have not been applied to a tool are indicated with a white box.

	Number of Citations	Simulated Data	*Plasmodium falciparum*	*Trypanosoma brucei brucei*	*Human gammaherpesvirus*	*Human papillomavirus*	*Bluetongue virus*	*Caulobacter vibrioides*	*Schizosaccharomyces pombe* (Fission Yeast)	*Saccharomyces cerevisiae* (Baker's Yeast)	*Culex quinquefasciatus (Common House Mosquito)*	*Aedes aegypti* (Yellow Fever Mosquito)	*Drosophila busckii* (Fruit Fly)	*Drosophila melanogaster* (Fruit Fly)	*Drosophila virilis* (Fruit Fly)	*Caenorhabditis elegans* (Roundworm)	*Danio rerio* (Zebra Fish)	*Gallus gallus* (Chicken)	*Mus musculus x Mus spretus* (Mouse Cross)	*Mus musculus* (Mouse - Haploid)	*Mus musculus* (Mouse - Diploid)	*Callithrix jacchus* (Common marmoset)	*Macaca mulatta* (Rhesus macaque)	*Pan paniscus* (Bonobo)	*Pan troglodytes* (Chimpanzee)	*Homo sapiens* (Human - Diploid, Single-Cell)	*Homo sapiens* (Human - Diploid, GM12878 phased)	*Homo sapiens* (Human - Diploid, GM06990 phased)	*Homo sapiens* (Human - Diploid, Other)	*Arabidopsis thaliana* (Thale Cress)	*Gossypium arboreum* (Tree Cotton)	*Gossypium raimondii* (Cotton sp.)	*Oryza sativa Japonica Group* (Rice, Single-Cell)
Diament and Tuller	9									⬛																							
Duan *et al.*	832		🟪						🟧	⬛																							
Kapilevich *et al.*	1	⬛																															
miniMDS	27																										⬛						
Segal and Bengtsson	17																			⬛							⬛						
Stevens *et al.*	327																			⬛													🟪
Chrom3D	62																											⬛	🟪				
Kalhor *et al.*	472																										⬛		🟪				
Li *et al.*	30													⬛																			
LorDG*	24																										⬛						
Tjong *et al.*	92																										⬛						
3D-GNOME	51																										⬛						

distinct chromosome copies (the ploidy problem) still remains a large, unanswered question in the field. While it may be possible to address this problem during read mapping and/or pre-processing steps, solutions built-in to 3D-GRP tools should also be investigated.

3.10 Conclusion

There has been a great deal of success predicting 3D genome organizations from HR-3C data originating from model organisms like *Saccharomyces cerevisiae* and *Homo sapiens.* Addressing the challenges outlined in Section 3.9 above will be crucial as the field continues to evolve and be extended to non-model organisms (especially ones with larger, non-standard genomes). The set of existing tools for solving the 3D-GRP is far from mature and cannot be applied to analyze 3D genome organization across various species. A tool that can be used to predict 3D genome structure across organisms is urgently needed. Many of the existing solution approaches in computer science for overcoming the difficulties associated with optimization problems like the 3D-GRP have not yet been explored. These types of solutions are likely to be an area of major development in the coming years within the 3D genome community. While a great deal of foundational work has been done, there is a clear lack of generalizable, algorithmically diverse computational tools for predicting the complete 3D genome organization from HR-3C data.

3.11 Key Points

- Many computational solutions exist for predicting 3D genome organizations in a select few model organisms.
- These existing tools cannot necessarily be applied to non-model organisms due to inherent constraints imposed by the underlying techniques.
- New tools are required to facilitate 3D genome organization studies in non-model organisms.
- There are many promising algorithmic areas that have not yet been applied to the 3D-GRP.
- There are many existing Hi-C datasets that have not been used to predict 3D genomic organization with existing tools.

3.12 Endnotes

1. Permission to reprint was obtained for the following panels: B (license number: 4758870758295), C (©2018 IEEE, with permission), G (license number: 4758871194345), H(license number: 4817790396726), I (license number: 4758871306477). All other panels contain images that are allowed to be reprinted under a Creative

Commons License. Additional information pertaining to reprint permissions for each image can be found in Section 3.14.

2. Permission to reprint was obtained for the following panels in Figure 2: C (©2013 Mary Ann Liebert, Inc and Journal of Computational Biology, with permission), F (license number: 4758871501440), G (©2018 IEEE, with permission), I (©2018 IEEE, with permission), J (license number: 4758901288689), K (©2018 IEEE, with permission). All other panels contain images that are allowed to be reprinted under a Creative Commons License. Additional information pertaining to reprint permissions for each image can be found in Section 3.14.

3. Permission to reprint was obtained for the following panels in Figure 3: D (©2018 IEEE, with permission), F (license number: 4759011183449), G (license number: 4759011283587), M (license number 4759011388011). Additional information pertaining to reprint permissions for each image can be found in Section 3.14.

3.13 Supplemental Information

3.13.1 Extended Methodology

- Manuscripts describing existing tools were identified through a google scholar (`https://scholar.google.com/`) literature search using the key phrases: "3D genome reconstruction problem", "3D genome prediction", "3D genome structure" and "3D genome organization". The results were restricted to papers published from 2006 (the year 5C was first described [44]) to September 1, 2019 and filtered to only include peer-reviewed manuscripts describing software for predicting 3D genome organizations or 3D regional organizations.

- Applications of existing tools were identified by examining all manuscripts that cited the original publications (as of January 8, 2020).

- Organisms with available Hi-C datasets (for Table 3.4) were determined by using "Hi-C" as a search term in the Gene Expression Omnibus database (`https://www.ncbi.nlm.nih.gov/geo/`). Results were extracted on January 8, 2020.

3.13.2 Acknowledgements

We would like to thank Daniel Hogan, Morgan W.B. Kirzinger, Dr. Matthew Links, Dr. Ian McQuillan and Dr. Steve Robinson for reading of the manuscript and and providing invaluable feedback.

3.13.3 Author Contributions

KM synthesized the information and wrote the manuscript. AK supervised the research and edited the manuscript.

3.13.4 Author Information

Kimberly MacKay is a Ph.D. candidate in the Department of Computer Science, University of Saskatchewan.

Anthony Kusalik is a Professor in the Department of Computer Science at the University of Saskatchewan. He also is the director of the Bioinformatics Program at the University of Saskatchewan.

3.13.5 Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada [RGPIN 37207 to AK, Vanier Canada Graduate Scholarship to KM].

3.14 Reprint Licenses

- Permission to reprint the images in Figure 3.1.
  - Diament and Tuller [39]: Creative Commons Attribution (CC BY) license.
  - Duan *et al.* [46]: License Number: 4758870758295 ; Issued on January 30, 2020
  - Kapilevich *et al.* [75]: ©2018 IEEE, with permission (The IEEE does not require individuals working on a thesis to obtain a formal reuse license).

  - miniMDS [125]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.
  - Segal and Bengtsson [132]: Creative Commons Attribution (CC BY) license.
  - Stevens *et al.* [146]: License Number: 4758871194345 ; Issued on January 30, 2020
  - Figure 3A from Chrom3D [118]: License Number: 4817790396726 ; Issued on April 28, 2020
  - Kalhor *et al.* [74]: License Number: 4758871306477 ; Issued on January 30, 2020
  - Li *et al.* [89]: Creative Commons Attribution (CC BY) license.
  - LorDG [156]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.
  - Tjong *et al.* [154]: Use of original figures published in PNAS is permitted for noncommercial use (such as dissertations and review articles).
  - 3D-GNOME [147, 148]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.

- Permission to reprint the images in Figure 3.2.
  - BACH [63]: Creative Commons Attribution (CC BY) license.
  - Chiariello *et al.* [24]: Creative Commons Attribution 4.0 International (CC BY 4.0) license.
  - ChromSDE [178]:
  - ©2013 Mary Ann Liebert, Inc and Journal of Computational Biology, with permission. Publisher granted permission via e-mail communication (January 30, 2020)
  - PASTIS [163]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.
  - RPR [61]: Creative Commons Attribution (CC BY) license.
  - ShRec3D [87, 110]: License Number: 4758871501440 ; Issued on January 30, 2020

  - ShRec3D+ [88]: ©2018 IEEE, with permission (The IEEE does not require individuals working on a thesis to obtain a formal reuse license).
  - SuperRec [177]: Creative Commons Attribution 4.0 International (CC BY 4.0) license.
  - 3DChrom [95]: ©2018 IEEE, with permission (The IEEE does not require individuals working on a thesis to obtain a formal reuse license).
  - tPAM [116]: License Number: 4758901288689 ; Issued on January 30, 2020
  - Zhang *et al.* [176]: ©2018 IEEE, with permission (The IEEE does not require individuals working on a thesis to obtain a formal reuse license).

- Permission to reprint the images in Figure 3.3.
  - BACH-MIX [63]: Creative Commons Attribution (CC BY) license.
  - Caudai *et al.* [22]: Creative Commons Attribution (CC BY) license.
  - Chromsome3D [1]: Creative Commons Attribution 4.0 International (CC BY 4.0) license.
  - ChromStruct 4 [21]: ©2018 IEEE, with permission (The IEEE does not require individuals working on a thesis to obtain a formal reuse license).
  - GEM [180]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.
  - Giorgetti *et al.* [56]: License Number: 4759011183449 ; January 30, 2020
  - IMP [13, 14, 157]: License Number: 4759011283587 ; January 30, 2020
  - InfMod3DGen [165]: Creative Commons Attribution (CC BY) license.
  - ISD [20]: Creative Commons Attribution (CC BY) license.
  - MBO [119]: Creative Commons Attribution (CC BY) license.
  - MCMC5C [127]: Creative Commons Attribution (CC BY) license.
  - Meluzzi and Arya [106]: Creative Commons Attribution-NonCommercial (CC BY-NC 4.0) license.

– Nagano *et al.* [112]: License Number: 4759011388011 ; January 30, 2020

– TADbit [135, 157]: Creative Commons Attribution (CC BY) license.

– Trieu and Cheng [155]: Creative Commons Attribution (CC BY) license.

Chapter 4

GeneRHi-C: 3D Genome Reconstruction from Hi-C Data

Kimberly MacKay, Mats Carlsson and Anthony Kusalik

Citation [98]: K MacKay, M Carlsson and A Kusalik. GeneRHi-C: 3D GENomE Reconstruction from Hi-C data. *10th International Conference on Computational Systems-Biology and Bioinformatics*. In Press. Publisher: IEEE.

Detailed Contributions: Kimberly MacKay was responsible for writing the manuscript. In addition, she wrote the software for Step 2 and 3 of GeneRHi-C and performed the visualization. All of the listed authors worked together to develop the Mathematical Models (**CM**, **GM** and **IP**). A brief description of the contributions for all listed authors can be found in Section 4.10.2.

As a reminder, the overarching goal of this thesis is to develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction. GeneRHi-C is a tool for predicting 3D genome organization from Hi-C datasets. Figure 1.3 indicates how GeneRHi-C fits into existing Hi-C data analysis workflows.

4.1 Abstract

Background: Many computational methods have been developed that leverage the results from biological experiments (such as Hi-C) to infer the 3D organization of the genome. Formally, this is referred to as the 3D genome reconstruction problem (3D-GRP). Hi-C data is now being generated at increasingly high resolutions. As this resolution increases, it has become computationally infeasible to predict a 3D genome organization with the majority of existing methods. None of the existing solution methods have utilized a non-procedural programming approach (such as integer programming) despite the established advantages and successful applications of such approaches for predicting high-resolution 3D structures of other biomolecules. Our objective was to develop a new solution to the 3D-GRP that utilizes non-procedural programming to realize the same advantages.

Results: In this paper, we present a three-step consensus method (called GeneRHi-C; pronounced “generic”) for solving the 3D-GRP which utilizes both new and existing techniques. Briefly, (1) the dimensionality of the 3D-GRP is reduced by identifying a biologically plausible, ploidy-dependent subset of interactions from the Hi-C data. This is performed by modelling the task as an optimization problem and solving it efficiently with an implementation in a non-procedural programming language. The second step (2) generates a biological network (graph) that represents the subset of interactions identified in the previous step. Briefly, genomic bins are represented as nodes in the network with weighted-edges representing known and detected interactions. Finally, the third step (3) uses the ForceAtlas 3D network layout algorithm to calculate (x, y, z) coordinates for each genomic region in the contact map. The resultant predicted genome organization represents the interactions of a population-averaged consensus structure. The overall workflow was tested with Hi-C data from *Schizosaccharomyces pombe* (fission yeast). The resulting 3D structure clearly recapitulated previously established features of fission yeast 3D genome organization.

Conclusion: Overall, GeneRHi-C demonstrates the power of non-procedural programming and graph theoretic techniques for providing an efficient, generalizable solution to the 3D-GRP.

Project Homepage: `https://github.com/kimmackay/GeneRHi-C`

4.2 Key Words

3D Genome Reconstruction Problem, Mathematical Modelling, Declarative Programming, Integer Programming, Network Layouts

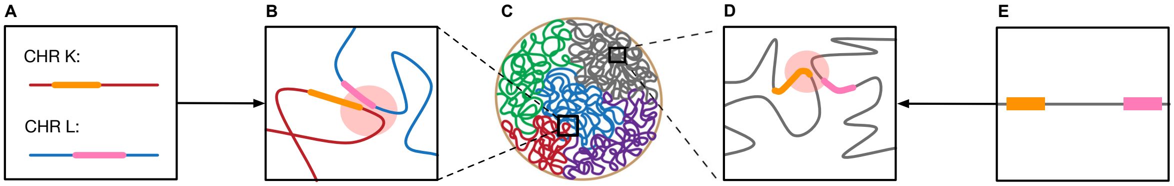


Figure 4.1: A representation of the DNA-DNA interactions that can occur within the 3D genome structure. Panels give the following representations. A: the linear locations of the genes undergoing a *trans*-interaction between two hypothetical chromosomes, K and L. B: a *trans*-interaction. C: a nucleus with the coloured lines representing the separate chromosomes from Babaei et al. [8]. D: a *cis*-interaction. These genes might be linearly "distant" but still have a detectable interaction in 3D space. E: the linear locations of the genes that are undergoing a 3D *cis*-interaction. The orange and pink regions in panels A, B, D and E are examples of possible gene locations. The red circles in panels B and D represent the genomic regions involved in an interaction.

4.3 Introduction

Within the nucleus, a cell's genetic information undergoes extensive folding and reorganization throughout normal physiological processes. Just like in origami where the same piece of paper folded in different ways allows the paper to take on different forms and potential functions, it is possible that different genomic organizations are related to various nuclear functions. Until recently, it has been extremely difficult to comprehensively investigate this relationship due to the lack of high-resolution and high-throughput techniques for identifying genomic organizations. The development of a biological technique called Hi-C (based on chromosome conformation capture) [92] has made it possible to detect the complete set of genomic regions simultaneously in close physical proximity. This proximity is often referred to as an "interaction" between two genomic regions. These interactions can be categorized as either intra-chromosomal (*cis*) interactions or inter-chromosomal (*trans*) interactions (Figure 4.1).

Hi-C [92] is a biological technique that utilizes next generation sequencing technologies to detect regions of the genome that are interacting in 3D space. These regions may be located on different chromosomes or distally on the same chromosome. An overview of the experimental procedure is depicted in Figure 4.2. Briefly, (1) cells are fixed with formaldehyde in order to covalently cross-link genomic regions that are in close 3D proximity. (2) The cross-linked fragments are then digested with a restriction enzyme to remove the potentially large non-interacting interconnecting segments of DNA. (3) The sticky ends generated through the restriction digest are filled in with biotinylated nucleotides. (4) Digested fragments are ligated together. (5) The initial cross-linking is removed, resulting in DNA fragments that represent the two genomic regions that form an interaction. (6) The biotinylated products are purified using streptavidin beads allowing for the detection of fragments that were cut by restriction enzymes. (7) Paired-end sequencing is then performed and the resultant reads are mapped to a reference genome using a Hi-C specific read mapper [7].

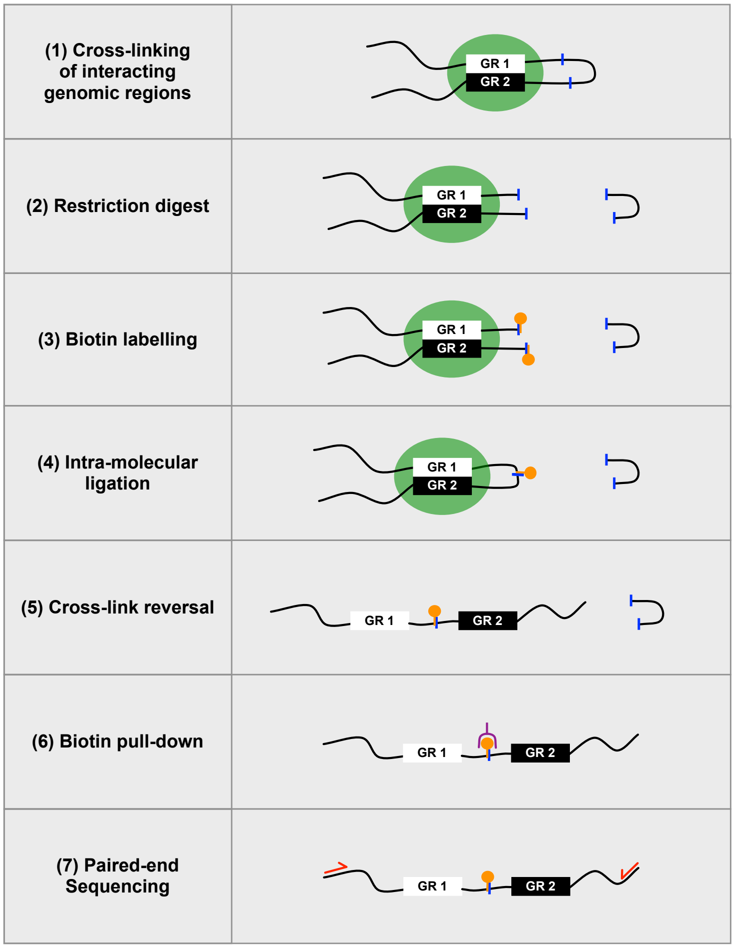


Figure 4.2: A simplified overview of the Hi-C protocol adapted from reference [92]. GR stands for "genomic region". The blue lines represent the location of a restriction enzyme cut site; green circles, a pair of genomic regions being chemically cross-linked together; orange circles, biotin; and red arrows, the primers that are required for paired-end sequencing. The purple symbol in step (6) represents a streptavidin bead that can be used to purify molecules with a biotin label.

Mapping the raw data of a Hi-C experiment to a reference genome results in the generation of a $N \times N$ matrix (a whole-genome contact map) where N is the number of "bins" which represent linear regions of genomic DNA. In general, the number of genomic bins is approximately equal to the total genome size divided by the Hi-C experimental resolution. Whole-genome contact maps are characteristically sparse and symmetric along the diagonal. Each cell ($A_{i,j}$) of a contact map ($\mathbf{A}$) records the count of how many times the genomic bin i was found to interact with the genomic bin j. These counts are often referred to as the frequency of the interaction between A_i and A_j (or interaction frequency). Inherent systematic biases within the whole-genome contact map are dampened by normalizing the interaction frequencies. Typically, an iterative correction and eigenvector decomposition (ICE) [68] or Knight-Ruiz (KR) [77, 90] normalization are/is applied to the raw data resulting in fractional interaction frequencies. It should be noted that during the normalization process interactions involving highly repetitive genomic regions such as centromere and/or telomere regions are often removed from the contact map (represented as 'NA') due to large amounts of noise and/or low signal [86].

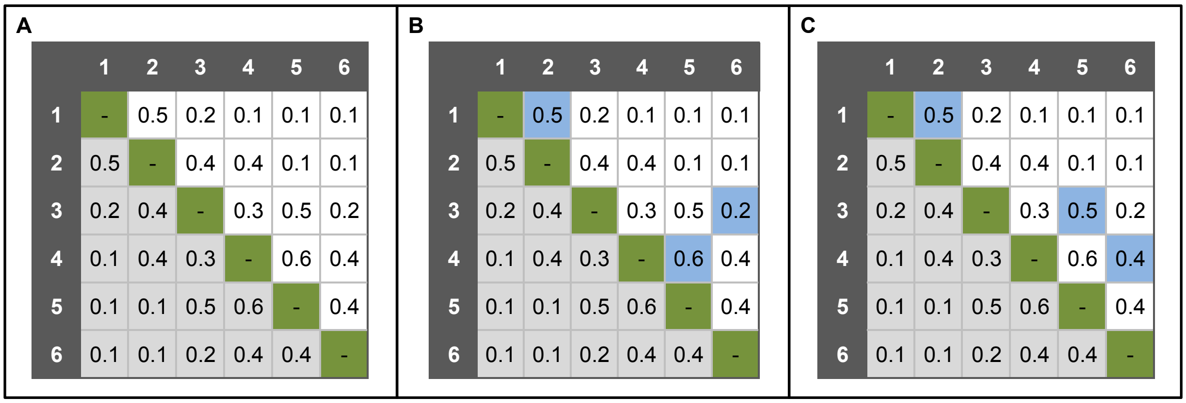
The normalized whole-genome contact maps can be used to infer the 3D organization of the genome. The process of predicting a model of the 3D genomic organization from a contact map is known as the 3D genome reconstruction problem (3D-GRP) [132]. Typically this is done by converting the normalized interaction frequencies into a set of corresponding pairwise Euclidean distances. In general it is assumed that a pair of genomic regions with a higher interaction frequency will often be closer in 3D space than a pair of genomic regions with a lower interaction frequency [46, 63, 87]. Most computational tools for solving the 3D-GRP then take the predicted pairwise Euclidean distances as input and produce a visualization of the 3D genome by modelling the chromatin fibre as a polymer [136]. In general, most existing programs can be broadly classified as either (1) consensus or (2) ensemble methods. Consensus methods generate a single population-averaged genomic model that best represents the whole-genome contact map, while ensemble methods produce a collection of genome models that represent the inherent heterogeneity of genome organizations within a population of cells [86].

As the resolution of whole-genome contact maps increases, it is computationally infeasible to predict a 3D genome organization with many of the existing methods. One notable exception is the method miniMDS [125] which uses a divide-and-conquer approach to overcome this problem. In order to divide the overall problem into subproblems, miniMDS utilizes an algorithm for detecting chromosomal sub-domains called TADs to make the initial division. Unfortunately, this makes it inapplicable to organisms which do not have TADs like *Arabadopsis thaliana* [42, 94]. To overcome this, we have developed a generalizable, three-step consensus method for solving the 3D-GRP called GeneRHi-C (3D **Gen**om**e** **R**econstruction from **Hi-C** data; pronounced "generic"). Unlike other 3D-GRP solutions, GeneRHi-C does not rely on chromosomal sub-domains or organism specific constraints in the prediction process making it generalizable to any organism. Briefly, GeneRHi-C preforms the following three steps: (1) dimensionality reduction, (2) graph representation and (3) calculation of (x, y, z) coordinates. The resulting 3D genome organization represents the interactions of a population-averaged consensus structure. In order to demonstrate its utility GeneRHi-C was used to predict a 3D genome organization from an existing *Schizosaccharomyces pombe* (fission yeast) Hi-C dataset.

4.4 Computational Workflow

4.4.1 Step 1: Dimensionality Reduction

Under normal cellular conditions, a given genomic region can be simultaneously involved in more than one interaction within the genome [48]. In contrast, a single genomic region within an individual cell is only able to participate in one Hi-C mediated interaction due to inherent restrictions within the biochemistry of the Hi-C experimental protocol [163]. In diploid organisms (organisms with two genomic copies) single cell Hi-C reactions are only able to detect two Hi-C mediated interactions per genomic region, one for each genomic copy [112]. An analogous restriction can be assumed in haploid organisms (organisms with only one genomic copy), where a single genomic region can only be actively detected in one Hi-C mediated interaction in a single cell. Using this restriction, a model of the 3D genome organization can be constructed from a whole-genome contact map by selecting a



A

	1	2	3	4	5	6
1	-	0.5	0.2	0.1	0.1	0.1
2	0.5	-	0.4	0.4	0.1	0.1
3	0.2	0.4	-	0.3	0.5	0.2
4	0.1	0.4	0.3	-	0.6	0.4
5	0.1	0.1	0.5	0.6	-	0.4
6	0.1	0.1	0.2	0.4	0.4	-

B

	1	2	3	4	5	6
1	-	0.5	0.2	0.1	0.1	0.1
2	0.5	-	0.4	0.4	0.1	0.1
3	0.2	0.4	-	0.3	0.5	0.2
4	0.1	0.4	0.3	-	0.6	0.4
5	0.1	0.1	0.5	0.6	-	0.4
6	0.1	0.1	0.2	0.4	0.4	-

C

	1	2	3	4	5	6
1	-	0.5	0.2	0.1	0.1	0.1
2	0.5	-	0.4	0.4	0.1	0.1
3	0.2	0.4	-	0.3	0.5	0.2
4	0.1	0.4	0.3	-	0.6	0.4
5	0.1	0.1	0.5	0.6	-	0.4
6	0.1	0.1	0.2	0.4	0.4	-

Figure 4.3: An example of two (of many) possible solutions to a 3D genome reconstruction problem. For all of the panels: the symmetric lower half of the contact map is indicated in light grey, the diagonal that represents "self-self" interactions is indicated in green and the genomic bin labels are represented in dark grey. For panels B and C: the blue boxes represent the subset of frequencies that could be selected as possible solutions (for $m = 1$). Panel B is a representation of a valid, non-optimal solution from the greedy algorithm and panel C is a representation of the valid optimal solution for the contact map where the sum of the selected interaction frequencies are 1.3 and 1.4, respectively.

ploidy-dependent subset of the interactions for each genomic region that maximizes the sum of the corresponding interaction frequencies. The mathematical model and corresponding implementation presented in this paper focus on modelling the 3D organization of haploid genomes but, as outlined in Section 4.7.3, could be extended to organisms with higher ploidies.

Naively, a greedy heuristic could be employed to model the 3D fission yeast genome organization using the strategy described above. Briefly, the subset of interactions representing the solution set can be chosen by sorting and selecting the interactions with the largest corresponding frequency values. This process would then be repeated, rejecting any frequency for a region of the genome that has already been selected. This heuristic will fail to take into account the situation where lower frequencies, which were rejected by selecting a higher frequency interaction, actually result in a greater overall maximum value for the sum of all selected frequencies within the solution. An example of this can be seen in Figure 4.3 where panel A is a hypothetical whole-genome contact map and panels B and C represent two possible solution matrices with different overall frequency sums. Specifically, Figure 4.3B follows the greedy heuristic described above which results in a non-optimal solution where the selected frequencies sum to 1.3. Figure 4.3C shows the optimal solution where the selected frequencies sum to 1.4. This type of optimization problem has been shown to be well-suited for solutions using non-procedural computational paradigms.

We have developed and tested three mathematical models (**CP**, **GM**, **IP**) to reduce the dimensionality of the 3D-GRP which describe the relationships present within the whole-genome contact map. These mathematical models differ in terms of their problem representation, underlying non-procedural implementation and overall generalizability. Briefly, the **CP** mathematical model is encoded as a set of constraints over finite domains with constraint programming [126]. The **GM** mathematical model represents the problem as a maximum-weight matching [47] and is encoded as a logic program that uses Kolmogorov's Blossom V algorithm [78]. The **IP** mathematical model is encoded with integer programming [168] and is described in more detail below. Each model takes a normalized whole-genome contact map as input. As mentioned previously, such a contact map is a $N \times N$ matrix where the genome has been partitioned into N genomic bins. For a hypothetical whole-genome contact map ($\mathbf{A}$), each cell ($A_{i,j}$) records the normalized interaction frequency between genomic bins i and j. By construction, the contact map is symmetric ($A_{i,j} = A_{j,i}$ for all i, j), and its main diagonal elements are all zero ($A_{i,i} = 0$ for all i).

The current formulations of the **CP** and **GM** mathematical models are only valid for haploid organisms whereas the **IP** mathematical model can be applied to organisms with any ploidy through an additional parameter called m. The value of m encodes the maximal number of interactions in which a given genomic bin can be involved based on the source organism's ploidy. For instance, m would be set to the following values based on the number of chromosome copies present: $m = 1$ (haploid), $m = 2$ (diploid; common in mammals), $m = 4$ (tetraploid; common in plants). Since the **IP** mathematical model is the most general, it will be the focus for the rest of this manuscript. Additional details on the **CP** and **GM** mathematical models can be found on the project homepage at GitHub [4] as well as in the first pre-print version of this paper [97]. For the purpose of this thesis, this information was also added to Section 4.10.4.

The mathematical model **IP** uses integer programming [168] and is valid for any value of m. It is based on introducing variables $x_{i,j}$ that assume a value of 1 if genomic bin i interacts with genomic bin j, and 0 otherwise. The goal of this model is to solve $x_{i,j}$ for all i, j. The complete model is given in Mathematical Model 1. It was implemented in SICStus Prolog [5] [19] and solved using the mixed integer programming based Gurobi Optimizer [6] [69]. The implemented program using this representation with the hypothetical whole-genome contact map depicted in Figure 4.3A is shown in Additional File 1 [7]. An example associated data file for this program is given in Additional File 2 [8] and is based on the interaction frequency values from the hypothetical whole-genome contact map depicted in Figure 4.3A. For the fission yeast results, all of this has been automated in a makefile that is available on the project homepage [9].

maximize

$$\sum_{(i,j)\in E} A_{i,j}x_{i,j} \tag{4.1}$$

subject to:

$$\sum_{(i,j)\in E} x_{i,j} + \sum_{(j,i)\in E} x_{j,i} \leq m,\ \forall i \in V \tag{4.2}$$

$$x_{i,j} \in \{0,1\},\ \forall (i,j) \in E \tag{4.3}$$

Mathematical Model 1: The **IP** model, for any m. V is the set $\{1,\ldots,N\}$ representing the genomic bins. E is the set $\{(i,j) \mid i < j \wedge A_{i,j} > 0\}$ representing the interactions whose frequencies (weights) are given by A.

4.4.2 Step 2: Graph Representation

The reduced set of interactions is converted into an undirected graph based on the graphical representation of Hi-C data described in GrapHi-C [101]. Briefly, the nodes in the network represent the individual genomic bins of the whole-genome contact map and the edges represent either selected interactions between bins or known linear interactions between adjacent bins. Linear interactions add additional biological constraints by representing the *bonafide in vivo* linear connections between bins (i.e. the linear extent of the chromosome). Unlike GrapHi-C, each edge is weighted using either: the interaction frequency divided by a dynamics coefficient for *cis*- and *trans*- interactions (described in more detail below) or the experimental resolution for linear interactions.

4.4.3 Step 3: Calculation of (x, y, z) Coordinates

Finally, the ForceAtlas 3D network layout algorithm provided as a Gephi plugin [10] (which is an extension of the ForceAtlas2 layout [70]) is used to calculate (x, y, z) coordinates for the centre of each node in the network. Recall that each node represents a genomic bin from the whole-genome contact map.

4.5 Problem Decomposition

When the **IP** implementation for step 1 was run on a complete fission yeast whole-genome contact map there was only a single *trans*-chromosomal interaction within the solution set making it difficult to infer the organization of the chromosomes in relation to each other. The low number of *trans*-chromosomal interactions is likely due to the fact that *cis*-interactions are known to have higher interaction frequencies than *trans*-interactions within the genome [35, 86]. This makes it more likely for *cis*-interactions to be included in the solution set since the goal of the mathematical models described above is to select a maximal subset of interaction frequencies. Since the disparity between *cis*- and *trans*- interaction frequencies is an inherent characteristic of whole-genome contact maps, all optimization-based solutions to the 3D-GRP must use an additional strategy to overcome this.

There are a number of possible strategies to deal with the disparity between *cis*- and *trans*- interaction frequencies such as data transformation or problem decomposition. Here the original computational problem (the 3D-GRP) has been decomposed into subproblems (described in more detail below) where the *cis*-chromosomal subproblems represent the individual chromosome structure while the *trans*-chromosomal subproblems represent how the chromosomes are organized in relation to each other within the nucleus. Each subproblem has been locally solved and the results are combined to retain the selected interactions from each subproblem. This is similar to the divide-and-conquer strategy employed by miniMDS which aims to first solve local substructures and then fit the results onto a global organization [125].

A single whole-genome contact map can be naturally divided into a finite, organism specific number of subproblems representing its constituent *cis*-interactions and pairwise *trans*-interactions. Each subproblem can be defined within the whole-genome contact map by specifying the range of genomic bins that correspond to the *cis*- or *trans*-interactions for each chromosome. In general, the number of subproblems for a whole-genome contact map with k chromosomes is equal to $\frac{k(k-1)}{2} + k$ where $\frac{k(k-1)}{2}$ represents the number of pairwise *trans*-interaction subproblems and k represents the number of *cis*-interaction subproblems. For example, because fission yeast has three chromosomes, its whole-genome contact map can be naturally partitioned into six subproblems (three *cis*- and three *trans*-interaction subproblems) to be solved in parallel. The location of these subproblems within a fission-yeast whole-genome contact map are depicted in Figure 4.4.

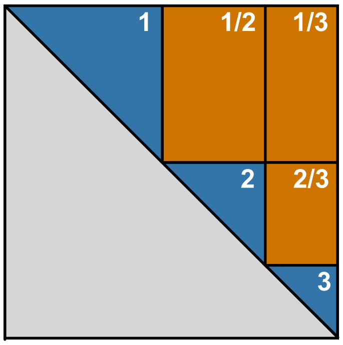


Figure 4.4: Identification of subproblems within the fission yeast contact map. The large grey triangle represents the portion of the contact map that does not need to be processed since all contact maps are mirrored along the diagonal. The blue triangles represent the subsections of the contact map that correspond to intra-chromosomal interactions, while the orange squares represent the subsections that correspond to the inter-chromosomal interactions. The label on the blue and orange areas represent the chromosome(s) involved in the interactions within that subsection of the contact map. In terms of the intra-chromosomal interactions, chromosome 1 contains the largest number of genomic bins while chromosomes 2 and 3 account for 34 and 80 percent fewer bins, respectively.

In order to solve the entire 3D-GRP, programs corresponding to the *cis*-interaction and pairwise *trans*-interaction subproblems can be generated and run independently. The solutions from each subproblem are then combined to reconstruct the entire 3D genomic model. This step is a heuristic which utilizes a novel metric (called the “dynamics coefficient”) to account for the instances when a single genomic region participates in more than m subproblem solutions; i.e. more than m interactions. Instead of discarding interactions from subproblem solutions involving the same genomic region when this region has already been selected in m interactions, each identified interaction is maintained and associated with a region-specific dynamics coefficient to encode the mobility (or lack of mobility) of that genomic region. Briefly, the dynamics coefficient for each genomic region is calculated by scanning all of the resultant files for each subproblem and counting how many times a specific genomic bin is found across the subproblem solution sets. The more interactions a genomic region is involved in, the higher its corresponding dynamics coefficient, and *vice versa*.

In general, the dynamics coefficient is an integer value in the range of 0 to k where k is the number of chromosomes present in the genome. For example, in fission yeast ($k = 3$) if genomic bin 1 was involved in an interaction in the solution sets of the chromosome 1 *cis*-interaction subproblem and the chromosome 1/2 *trans*-interaction subproblem it would have a dynamics coefficient of 2, whereas if it was involved in an interaction in each of the relevant *trans*-interaction subproblems and the *cis*-interaction subproblem it would have an associated dynamics coefficient of 3. A higher dynamics coefficient suggests that the corresponding genomic region has been more mobile within the genome and that there is less certainty about its fixed position within the model. This is similar to the B factor (also known as the temperature factor or the Debye-Waller factor) generated with protein x-ray crystallography experiments [84]. The B factor encodes the degree of uncertainty associated with computed atomic positions in 3D space.

The dynamics coefficient is used to calculate edge weights for *cis*- and *trans*-interactions in step 2 of the GeneRHi-C workflow ($A_{i,j}/d$ where $A_{i.j}$ is the interaction frequency and d is the dynamics coefficient). These weights are then used to visualized the predicted model. Although this use causes violation of the initial ploidy restriction used to constrain each subproblem, it is still biologically valid. As mentioned previously, it is possible for a given genomic bin to be involved in more than one interaction in 3D space [24, 48], even though Hi-C is only able to detect one pairwise interaction per restriction site within a single haploid cell. Additionally, the dynamics coefficient allows the program to encode some of the mobility of genome organization into the predicted model by representing the certainty of whether an interaction is fixed within the population of cells. Overall, this decomposition approach results in a larger number of *trans*-chromosomal interactions being included in the final solution set. It also has been applied to the CP and GM mathematical models and should be utilized in future applications of GeneRHi-C.

4.6 Results

The above workflow was tested with an existing fission yeast Hi-C dataset (GEO accession number: GSM1379427 [108]). All programs were run on a server-grade computer with sufficient main memory to represent the entire problem. All times reported in this and subsequent sections are elapsed times. Results from the **CP** and **GM** mathematical models can be found on the project homepage at GitHub [11] as well as in the first pre-print version of this paper [97]. For the purpose of this thesis, this information was also added to Section 4.10.4.

4.6.1 Step 1: Dimensionality Reduction

The implementation of the **IP** model for the complete whole-genome fission yeast contact map ($m = 1$; $|V| = 1258$, $|E| = 745595$) was able to identify the optimal solution set in 294.44 seconds using the Gurobi optimizer. As mentioned above (and depicted in Figure 4.6A), only one *trans*-chromosomal interaction was represented in the solution set. This outcome made it difficult to infer the organization of the chromosomes in relation to each other. In order to overcome this, the decomposition approach described above was used. Six separate subprograms were generated and run independently. The size of each subproblem in terms of V and E, as well as the time it took to identify the optimal solution, is presented in Table 4.1 (total summed run time of 40.07 seconds).

4.6.2 Step 2: Graph Representation

The results from each subproblem were combined as described above and converted into a GrapHi-C [101] representation using the `generate_gephi_input_subproblems.pl` script [12]. Please note this is an extension of the original GrapHi-C script that allows for the dynamics

Table 4.1: Subproblem sizes and elapsed times (runtime) for the **IP** mathematical model applied to the fission yeast dataset.

Subproblem	**Number of Vertices ($\|V\|$)**	**Number of Edges ($\|E\|$)**	**Run Time (seconds)**
chromosome 1 *cis*-interaction	558	148734	15.75
chromosome 2 *cis*-interaction	454	96562	9.26
chromosome 3 *cis*-interaction	246	27255	2.06
chromosome 1/2 *trans*-interaction	454	241022	4.90
chromosome 1/3 *trans*-interaction	246	128472	4.70
chromosome 2/3 *trans*-interaction	246	103550	3.40

coefficient to be included in the edge-weight calculation. This step took less than 1 second of execution time.

4.6.3 Step 3: Calculation of (x, y, z) Coordinates

The graph generated in Step 2 was used as input to the ForceAtlas 3D network layout algorithm. The resulting layout was exported to a `.gexf` file and (x, y, z) coordinates were extracted using the `gexf2xyz.py` script on the project homepage [13]. This step took less than 10 seconds of execution time.

4.6.4 Visualization

The results were visualized in Gephi (Figure 4.5) [12]. Nodes were coloured according to their chromosome number (Panel A) or genomic feature (Panel B). We would like to stress that this graph-based visualization is not a polymer model of the DNA chain that is often seen in other 3D genome prediction tools. Therefore, the smoothness of the edges is not a result of any bending rigidity constraints. Instead, it is a result of the visualization tool (Gephi) and the network layout applied.

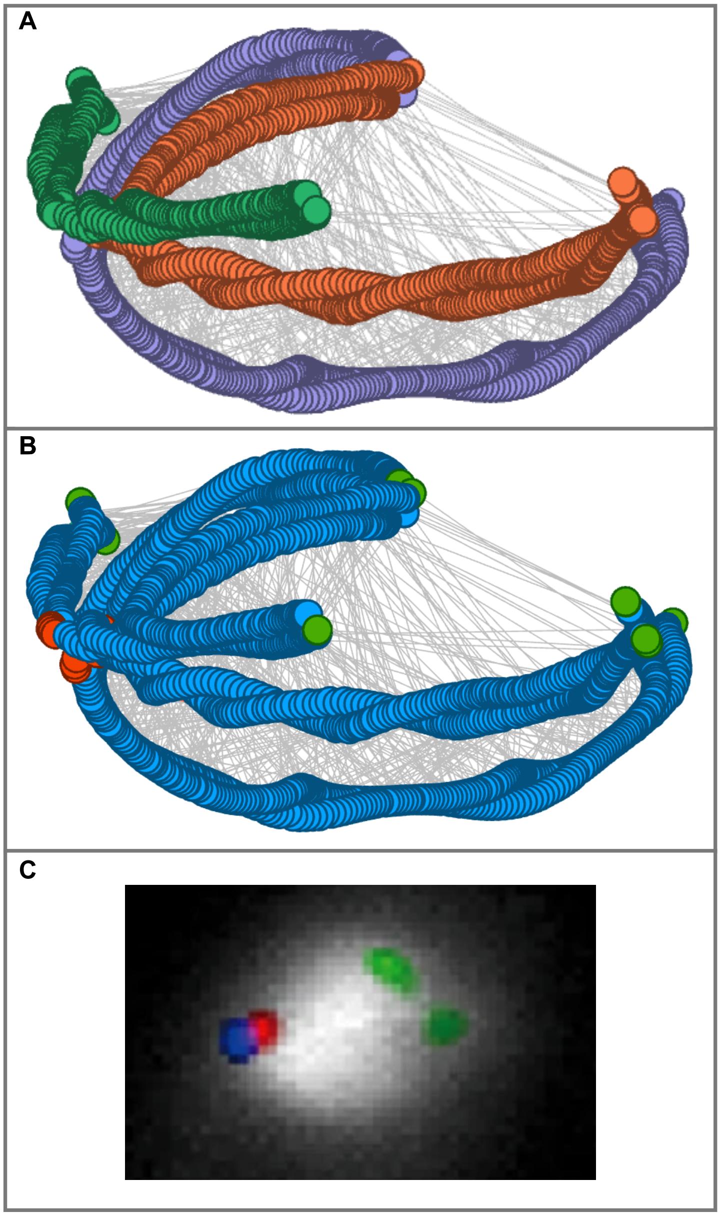


Figure 4.5: Visualization of the Predicted Genome Model Using the **IP** Model and Comparison with Fluorescence In Situ Hybridization. In Panels A and B, circles depict the genomic bins, grey lines represent *cis*- and *trans*-interaction edges selected by the **IP** model, and line lengths are proportional to original edge-weight calculated in step 2. In Panel A, circles are coloured according to their corresponding chromosome (CHR1: purple, CHR2: orange, CHR3: green). In Panel B, the following genomic features are highlighted: telomeres (green), centromeres (red) and nuclear DNA (blue) Panel C is a Fluorescence In Situ Hybridization image that depicts the locations of telomeres (green), and the spindle pole body (red) during fission yeast mitotic interphase (generated by Asakawa *et al.* [5] - reprint license: 4823220956979).

One of the most well–documented features of fission yeast genomic organization is the 3D clustering of centromeres and telomeres within the nucleus [25, 55]. In order to determine whether the yeast model predicted by GeneRHi-C was able to recapitulate these features, the genomic bins corresponding to centromeres and telomeres were coloured in the Gephi visualization (see Figure 4.5B). This figure provides evidence that the predicted genome model is consistent with established principles of fission yeast chromosomal organization including: (1) chromosomal organization into a hemispherical region, (2) a single centromere cluster and (3) the presence of two telomere clusters (chromosome 1/2) located near the nuclear periphery, opposite the centromere cluster [107, 108, 158]. Additionally, the clustering in the GeneRHi-C predicted model is consistent with the clustering seen in previous fission yeast 3D genome predictions [151]. This provides confidence in the accuracy that was achieved using the GeneRHi-C workflow with fission yeast data. This type of evaluation (comparison to known genome and chromosome structural features) is typical within the 3D genome community [114, 155]. Future work (described in Section 4.7.2) will extend this evaluation to provide a more comprehensive snapshot of GeneRHi-C's reconstruction ability.

4.7 Discussion

4.7.1 Effect of m on Genome Organization in Fission Yeast

As mentioned previously, it is possible that each genomic region could be involved with more than one interaction within the genome but is restricted to m Hi-C interactions (where m is organism ploidy). To determine whether or not relaxing this restriction would result in a more comprehensive genomic model without having to use the decomposition approach described in Section 4.5, the GeneRHi-C workflow with the **IP** mathematical model was tested with values of m from 1 to 6 for the same fission yeast Hi-C dataset (GSM1379427 [108]). As mentioned previously, the **IP** mathematical model allows for a single genomic bin to be involved in more than one Hi-C mediated interaction in the predicted genome organization. Recall that the **CP** and **GM** mathematical models could not be used for this since they are only valid for $m = 1$.

For each value of m, the program was able to find an optimal solution in 294.44, 13.20, 104.46, 15.31, 38.79, and 16.94 seconds for $m = 1..6$, respectively. The results were visualized using Gephi (Figure 4.6). Nodes were coloured according to their chromosome number. The nodes in the graph represent the individual genomic bins of the whole-genome contact map and the edges represent either selected interactions between bins or known linear interactions between adjacent bins. The results presented in Figure 4.6 indicate that relaxing the ploidy restriction (by increasing the value of m) did not result in a more comprehensive genomic model. This is consistent with the work of Diament and Tuller [39] which suggested that as little as 5 % of the original Hi-C data is required to generate biologically accurate 3D genome models [132].

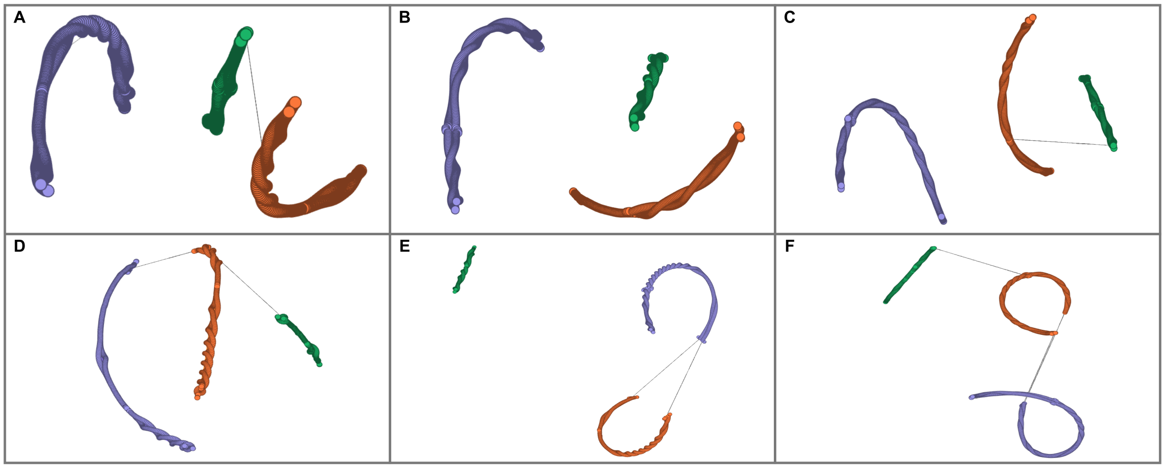


Figure 4.6: Visualization of the Predicted Genome Model Using the **IP** Model with Various m Values. Circles depict the genomic bins, grey lines represent *trans*-interaction edges selected by the **IP** model, and line lengths are proportional to the associated interaction frequency ($A_{i,j}$). Circles are coloured according to their corresponding chromosome (CHR1: purple, CHR2: orange, CHR3: green). The results for each m values are presented in the following panels: A ($m = 1$), B ($m = 2$), C ($m = 3$), D ($m = 4$), E ($m = 5$), F ($m = 6$).

Minimal numbers of *trans*-chromosomal interactions were selected by the model regardless of what the parameter m was set to. Specifically the following number of *trans*-chromosomal interactions were observed in each solution set: 1 ($m = 1$), 0 ($m = 2$), 1 ($m = 3$), 2 ($m = 4$), 2 ($m = 5$), 3 ($m = 6$). As mentioned previously, this is likely due to the fact that *cis*-chromosomal interactions occur more frequently than *trans*-chromosomal interactions within the genome resulting in higher interaction frequency values [35]. The decomposition strategy described in Section 4.5 can again be applied to circumvent this issue.

4.7.2 Evaluation

The 3D genomics community does not have a standardized methodology for evaluating 3D genome reconstruction tools like GeneRHi-C. Ideally, 3D genome reconstruction tools would be evaluated using synthetic Hi-C datasets with known ground-truth structures. The 3D models predicted from these synthetic datasets could then be compared with their ground truth counter-parts using measures like the Spearman correlation coefficient and root-mean-square deviation. Unfortunately, a standardized dataset of synthetic 3D structures and associated Hi-C matrices for tool evaluation does not exist. We are actively working towards the creation of this type of data. Once it has been generated it will be used to evaluate GeneRHi-C's reconstruction ability using the methodology described above. This methodology could also be employed to develop a ranked-list of all existing 3D genome reconstruction tools.

4.7.3 Application to Organisms with Higher Ploidies and/or Larger Genomes

The **IP** mathematical model described above could be applied to organisms with higher ploidies by specifying the value of the m parameter. For instance, m could be modified in the following ways according to the number of chromosome copies present: $m = 2$ (diploid; common in mammals), $m = 4$ (tetraploid; common in plants), and so on. One issue that would need to be addressed in organisms with higher ploidies is phasing the interactions to each chromosome copy. This could potentially be solved using existing phasing tools [26] and additional biological data [17, 134].

Utilizing the decomposition approach described in Section 4.5 allows one to take advantage of coarse-grained parallelism ensuring the mathematical models are scaleable to organisms with larger genomes (and more chromosomes). This type of parallelism is easy to obtain on many types of computational infrastructure. As an example, this decomposition could be easily applied to a whole-genome contact map from *Homo sapiens*. This contact map would result in the generation of 276 subproblems (given $k = 23$ and the number of subproblems $= \frac{k(k-1)}{2} + k$). It should be noted that the size of these subproblems would likely be larger than what was seen in the fission yeast example (depending on the Hi-C resolution).

4.7.4 Future Work

Future work will focus on the validation, modification and extension of the GeneRHi-C. Specifically, an extensive biological validation of the predicted genome models will be performed with targeted chromosome conformation capture assays and advanced microscopy techniques to better characterize the biological accuracy of the developed mathematical model. Different types of data transformations will be explored to address the disproportionate numbers of *cis*- and *trans*-chromosomal interactions in the whole-genome contact map in case they result in a better alternative to the decomposition approach described in Section 4.5. The **IP** mathematical model will be utilized as a computational framework which will be extended and further developed to incorporate a variety of additional genomic datasets and information types into the prediction process. For example, each genomic bin could have an associated list of variables representing the genes found within that bin and their corresponding gene expression values. Constraints could then be applied to favour interactions between regions with similar expression profiles. The **IP** mathematical model will also be utilized as a starting point for predicting the 3D genomic structure of organisms with higher ploidies by applying the modifications suggested in Section 4.7.3.

4.8 Conclusion

This is the first time a non-procedural programming approach has been used to model the 3D genome organization from Hi-C data. Specifically, we developed a three-step consensus method (called GeneRHi-C; pronounced "generic") for solving the 3D-GRP which utilizes both new and existing techniques. Briefly, (1) the **IP** mathematical model is used to reduce the dimensionality of the 3D-GRP by identifying a biologically plausible, ploidy-dependent subset of interactions from the Hi-C data. A decomposition approach is used in this step to generate a more comprehensive 3D genome organization. $\frac{k(k-1)}{2} + k$ separate subproblems (one for each set of *cis*-chromosomal interactions and one for each set of pairwise *trans*-chromosomal interactions) are independently solved and combined. A novel coefficient is defined to aid in combining the results of each subproblem (the dynamics coefficient) which allows a level of positional uncertainty to be encoded into the predicted genomic organization. The second step (2) generates a biological network (graph) that represents the subset of interactions identified in the previous step. Briefly, genomic bins are represented as nodes in the network with weighted-edges representing known and detected interactions. Finally, the third step (3) uses the ForceAtlas 3D network layout algorithm to calculate (x, y, z) coordinates for each genomic region in the contact map. The resultant predicted genome organization represents the interactions of a population-averaged consensus structure. The GeneRHi-C workflow was tested with Hi-C data from *Schizosaccharomyces pombe* (fission yeast). This predicted 3D genome organization was then validated through literature search which verified that the GeneRHi-C prediction recapitulated key documented features of the yeast genome. Overall, GeneRHi-C demonstrates the power of non-procedural programming and graph theoretic techniques for providing an efficient, generalizable solution to the 3D-GRP and is a step towards a better understanding of the relationship between genomic structure and function.

4.9 Endnotes

4. `https://github.com/kimmackay/GeneRHi-C/tree/master/step1`

5. `http://sicstus.sics.se`

6. `http://www.gurobi.com/`

7. `https://github.com/kimmackay/GeneRHi-C/blob/master/step1/IP/supplementary_files/additional_file_1.pl`

8. `https://github.com/kimmackay/GeneRHi-C/blob/master/step1/IP/supplementary_files/additional_file_2.csv`

9. `https://github.com/kimmackay/GeneRHi-C/tree/master/step1/IP`

10. `https://gephi.org/plugins/`

11. `https://github.com/kimmackay/GeneRHi-C/tree/master/step1`

12. `https://github.com/kimmackay/GeneRHi-C/tree/master/step2/scripts`

13. `https://github.com/kimmackay/GeneRHi-C/tree/master/step3`

4.10 Supplemental Information

The following files are available on the project homepage. **Additional file 1:** The implemented program using the **IP** mathematical model. **Additional file 2:** An example data file (`.csv` file) depicting the interaction frequencies from the hypothetical whole-genome contact map depicted in Figure 4.3A utilized by the **IP** model. **Availability of data and material:** The datasets utilized in this article are available in the Gene Expression Omnibus database (accession number: GSM1379427; `https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM1379427`). **Software information: Project Name:** GeneRHi-C (pronounced "generic"); **Project Homepage:** `https://github.com/kimmackay/GeneRHi-C`; **Programming language(s):** SICtus Prolog (Step 1), Perl (Step 2), Python (Step 3); **Other requirements:** A local version of the Gurobi solver is required to run the **IP** model (commercial software that is freely available to academic users).

4.10.1 Acknowledgements

We could like to thank Dr. Christopher Eskiw, Morgan W.B. Kirzinger and Conor Lazarou for their input and advice.

4.10.2 Author's Contributions

KM, MC, AK developed the **CP**, **GM** and **IP** mathematical models. MC implemented and tested the mathematical models. KM visualized the results and verified the biological accuracy. KM wrote the manuscript. MC, AK edited the manuscript. AK supervised the research.

4.10.3 Funding

This work was supported by the Natural Sciences and Engineering Research Council of Canada [RGPIN 37207 to AK, Vanier Canada Graduate Scholarship to KM].

4.10.4 Additional Mathematical Models

Two additional mathematical models (**CP** and **GM**) were developed, implemented and tested as a part of this research. This work is well-documented in the original pre-print [97] and on the project homepage [14] but could not be included in the final manuscript due to space limitations. For the purpose of this thesis, information pertaining to these models has been extracted from the pre-print [97] and is provided below. Please note, some of the following text was written by Mats Carlsson.

CP Mathematical Model

The **CP** mathematical model, is encoded with Constraint Programming [126]. This model is valid for $m = 1$ only and requires integral $A_{i,j}$ values due to the implementation (described below). It is based on introducing variables M_i where $M_i = j$ if genomic bin i interacts with genomic bin j, and $M_i = i$ otherwise. The goal of this model is to solve M_i for all i. The model is given in Mathematical Model 2. Since this model encodes a combinatorial problem, its time complexity is exponential in the worst case.

maximize

$$\sum_{i \in V} A_{i,M_i} \tag{4.4}$$

subject to:

$$M_i = j \leftrightarrow M_j = i, \ \forall i, j \in V \tag{4.5}$$

$$M_i \in \{i\} \cup \{j \mid A_{i,j} > 0\}, \ \forall i \in V \tag{4.6}$$

Mathematical Model 2: The **CP** model, valid for $m = 1$ and integral interaction frequencies ($A_{i,j}$) only. V is the set $\{1, \ldots, N\}$ representing the genomic bins.

The **CP** mathematical model (depicted in Mathematical Model 2) was implemented in MiniZinc [113] with the OR-Tools constraint solver from Google [15]. An archived version of the MiniZinc program based on the hypothetical whole-genome contact map depicted in Figure 4.3A is provided below (Program 4.5). The corresponding datafile for this hypothetical whole-genome contact map [16] as well as the implementation for the fission yeast dataset [17] can be found on the project homepage. This model leverages the fact that the solution will never contain more than $m \times N$ interactions making it scalable to larger genomes in terms of space complexity. It is worth noting that Equation (4.5) can be encoded by the `inverse` global constraint [18], whereas Equation (4.4) is encoded with one `element` constraint per row of A plus one `sum` constraint. These constraints are propagated by efficient algorithms in many constraint programming solvers.

The MiniZinc program corresponding to the complete fission yeast genome could not be solved to optimality after several days of run time on a server-grade computer. In an attempt to overcome this, the divide-and-conquer approach described above was applied. A MiniZinc program for each *cis*- or *trans*- subproblem was generated and run independently. Similarly to the complete whole-genome contact map, not a single *cis*- or *trans*- problem could be solved to optimality in several days.

GM Mathematical Model

Our **GM** mathematical model is an *optimal solution* that works in polynomial time, but it is only valid for the $m = 1$ case. By representing the contact map as an undirected graph with N vertices (genomic bins) and $\frac{N(N-1)}{2}$ edges (interactions) the 3D-GRP can be regarded as the problem of computing a *maximum-weight matching* for the graph $G = (V, E)$. A *matching* in a graph is a set of edges where no two edges share an endpoint. Each edge has an associated weight, and the weight of the matching is simply the sum of the weights of the edges in the matching. In the **GM** model, the vertices V are the set of genomic bins, the edges E are the set $\{(i, j) \mid i < j \wedge A_{i,j} > 0\}$, and the weights are given by A. An $O(|V| \cdot |E| \log |V|)$ implementation of the weighted matching problem was reported by Mehlhorn and Schäfer [102], and is provided in the LEDA algorithm library [19]. However, the LEDA algorithm library was not accessible for this research. In order to overcome this, we utilized an encoding proposed by Mehlhorn [102] to convert our maximum-weight matching problem into an auxiliary *perfect* matching problem [47]. This encoding is used in our model, given in Mathematical Model 3. Additionally, by using this encoding we were able to use Kolmogorov's Blossom V algorithm [78], which was available for this research and is considered the most efficient implementation of Edmonds algorithm.

Solve the maximum-weight perfect matching problem for the graph $G' = (V', E')$ and weight function $f : E' \mapsto \mathbb{R}$, i.e.:

maximize

$$\sum_{(i,j) \in E''} f(i,j) \tag{4.7}$$

subject to:

$$V' = \{i \mid i \in V\} \cup \{i + N \mid i \in V\} \tag{4.8}$$

$$E' = \{(i,j) \mid (i,j) \in E\} \cup \{(i+N, j+N) \mid (i,j) \in E\} \cup \{(i, i+N) \mid i \in V\} \tag{4.9}$$

$$E'' \subseteq E' \text{ is a perfect matching for } G' \tag{4.10}$$

$$f(i,j) = \begin{cases} A_{i,j} & \text{, if } i \leq N \wedge j \leq N \\ A_{i-N,j-N} & \text{, if } i > N \wedge j > N \\ 0 & \text{, otherwise} \end{cases} \tag{4.11}$$

Mathematical Model 3: The **GM** model, for $m = 1$ only. V is the set $\{1, \ldots, N\}$ representing the genomic bins. E is the set $\{(i,j) \mid i < j \wedge A_{i,j} > 0\}$ representing the interactions and the weights are given by A. $f(i,j)$ is the function used to calculate edge weight. $G' = (V', E')$ is an extended graph used to map $G = (V, E)$ to a maximum-weight perfect matching problem. This mapping to maximum-weight perfect matching was given by Mehlhorn [102].

The **GM** mathematical model (depicted in Mathematical Model 3) was implemented in SICStus Prolog [20] [19] using Kolmogorov's Blossom V algorithm [78]. The implemented program using this representation for the hypothetical whole-genome contact map depicted in Figure 4.3A is presented in Program 4.6. An example associated data file for this program can be found on the project homepage [21]. The program is run by: (1) invoking the `compile_adjacency` predicate with a data file and (2) invoking the `match_blossom5` predicate. In this example, this would be done by invoking: `compile_adjacency('testMap.csv',`

Table 4.3: Subproblem sizes and elapsed times (runtime) for the **GM** mathematical model applied to the fission yeast dataset.

Subproblem	Number of Vertices ($\|V\|$)	Number of Edges ($\|E\|$)	Run Time (seconds)
chromosome 1 *cis*-interaction	558	148734	0.164
chromosome 2 *cis*-interaction	454	96562	0.055
chromosome 3 *cis*-interaction	246	27255	0.008
chromosome 1/2 *trans*-interaction	454	241022	9.645
chromosome 1/3 *trans*-interaction	246	128472	7.203
chromosome 2/3 *trans*-interaction	246	103550	1.552

`testMap)`, followed by `match_blossom5(testMap,[1],[1])`. For the fission yeast results, all of this has been automated in a `makefile` that is available on the project homepage [22].

The SICStus Prolog implementation of the **GM** mathematical model was able to predict a fission yeast genomic organization in 1.088 seconds ($m = 1$; for the complete whole-genome contact map where $|V| = 1258$, $|E| = 745595$). In this matching, only one edge representing a *trans*-chromosomal interaction was included while the rest of the edges depicted *cis*-chromosomal interactions. This made it difficult to infer the organization of the chromosomes in relation to each other. In order to overcome this the divide-and-conquer approach described above was applied. Specifically, six separate matchings were identified: one for each chromosome's *cis*-interactions and one for each set of pairwise *trans*-chromosomal interactions. A SICStus Prolog program for each *cis*- or *trans*- subproblem was run independently. For each subproblem, the time it took to identify the optimal solution is presented in Table 4.3. These results were identical to that of the **IP** mathematical model but were identified significantly faster. Therefore, the **GM** model is preferable over the **CP** model when $m = 1$ (i.e. for application in haploid organisms).

4.10.5 Archived Software

GeneRHi-C Step 1

Program 4.2: Archived version of the script used for step 1 of GeneRHi-C (authored by Mats Carlsson).

```
:- use_module(library(lists)).
:- use_module(library(csv)).
:- use_module(library(system3)).

chromosome(pombe, 1, 1, 558).
chromosome(pombe, 2, 559, 1012).
chromosome(pombe, 3, 1013, 1258).
chromosome(elegans, 1, 1, 302).
chromosome(elegans, 2, 303, 608).
chromosome(elegans, 3, 609, 884).
chromosome(elegans, 4, 885, 1234).
chromosome(elegans, 5, 1235, 1653).
chromosome(elegans, 6, 1654, 2008).

solve_ip(Species, Set1, Set2, M) :-
  load_files(Species),
  retractall(edge(_,_,_)),
  (   adjacency(B1, B2, F, Chr1, Chr2),
      B1 < B2,
      (   member(Chr1, Set1), member(Chr2, Set2) -> true
      ;   member(Chr1, Set2), member(Chr2, Set1) -> true
      ),
      assertz(edge(B1, B2, F)),
      fail
  ;   true
  ),
  % findall(edge(I,J,W), edge(I,J,W), Edges),
  findall(I-J, (edge(I,J,_); edge(J,I,_)), Pairs0),
  keysort(Pairs0, Pairs1),
  keyclumped(Pairs1, Pairs2),
  tell('/tmp/all.lp'),
  write('Maximize\n  obj:'),
  (   foreach(I1-J1,Pairs1),
      fromto(' ',S1,S2,_)
  do  (   I1>J1 -> S1 = S2
      ;   edge(I1, J1, F1),
    format('~a~w X_~d_~d', [S1,F1,I1,J1]),
    S2 = ' + '
      )
  ), nl,
  write('Subject To\n'),
  (   foreach(I2-J2s,Pairs2),
      param(M)
  do  (   length(J2s,Len), Len =< M -> true
      ;   (   foreach(J2,J2s),
        fromto(' ',S3,' + ',_),
        param(I2)
    do  sort2(I2, J2, I3, J3),
        format('~aX_~d_~d', [S3,I3,J3])
    ),
    format(' <= ~d\n', [M])
      )
  ),
  write('Binary\n'),
  (   foreach(I4-J4,Pairs1)
  do  (   I4>J4 -> true
      ;   format(' X_~d_~d\n', [I4,J4])
      )
  ),
  write('End\n'),
  told,
  system('gurobi_all'),
  parse_solution.

parse_solution :-
  see('/tmp/all.sol'),
  retractall(solution(_,_,_)),
  read_line(_),
  read_line(_),
  repeat,
    read_line(Codes),
    (   Codes = end_of_file -> true
    ;   tok_sol_line(I, J, 1, Codes, []),
        edge(I, J, W),
        assertz(solution(I,J,W)),
        fail
    ), !,
  seen,
  findall(W, solution(_,_,W), Ws),
  length(Ws, Size),
  sumlist(Ws, Weight),
  format('% Gurobi computed a set of ~d edges and weight ~w\n',
        [Size,Weight]),
  findall(0, (solution(I,J,W), print_tab([I,J,W])), _).

tok_sol_line(I, J, Z01) --> "X_",
  tok_int(0, I), "_",
  tok_int(0, J), " ",
  tok_int(0, Z01).

tok_int(Int0, Int) --> [D], {D >= 0'0, D =< 0'9}, !,
  {Int1 is 10*Int0 + D - 0'0},
  tok_int(Int1, Int).
tok_int(Int, Int) --> [].

print_tab(L) :-
  (foreach(X,L) do write(X), write(’ ’)),
  nl.

parse_record(Record, B1, B2, F) :-
  Record = [integer(B1,_), integer(B2,_), float(F,_)].

bin2chr(Bin, Species, Chr) :-
  chromosome(Species, Chr, A, B),
  Bin >= A,
  Bin =< B, !.

sort2(X, Y, X, Y) :- X =< Y, !.
sort2(X, Y, Y, X).
```

GeneRHi-C Step 2

Program 4.3: Archived version of the script used for step 2 of GeneRHi-C.

```
#!/usr/bin/perl
## generates the two files (nodes.tsv and edges.tsv)
## needed for gephi visualization
##
## argument 1: the number of chromsomes
## argument 2: the experimental resolution
## argument 3: the number of subproblems (or result files that
##             need to be integrated)
## agrument 4: the beginning out the output files names
##
## Kimberly MacKay Nov 9, 2017
## license: This work is licensed under the Creative Commons
## Attribution-NonCommercial-ShareAlike 3.0 Unported License.
## To view a copy of this license, visit
## http://creativecommons.org/licenses/by-nc-sa/3.0/ or send
## a letter to Creative Commons, PO Box 1866, Mountain View,
## CA 94042, USA.

use strict;
use warnings;

## check to ensure four arguments were passed in
die "ERROR: must pass in four arguments." if @ARGV != 4;

my $num_chr = $ARGV[0];
my $experimental_resolution = $ARGV[1];
my $num_sub_problems = $ARGV[2];
my $out_file = $ARGV[3];

# scan each of the results files and collect the result in
# this hash table
my %interactions;
my %dynamics_coefficents;

##########################################################################
##  initialize variables
##########################################################################

## get the ending index of each chromosome
my @stop_index;
for(my $chr = 1; $chr <= $num_chr; $chr++)
{
  print STDERR "What is the ending index of CHR".$chr."? ";
  my $input = <STDIN>;
  $stop_index[$chr] = int($input);
}

$stop_index[0] = 0;

## make an array that maps genomic bin to the
## corresponding chromosome
my @chrs;
my $bin = 1;
for(my $j = 1; $j <= $num_chr; $j++)
{
  for(my $i = $bin; $i <= $stop_index[$j]; $i++)
  {
      $chrs[$bin] = $j;
      $bin = $bin +1;
  }
}

##########################################################################
##  print the *_nodes.tsv file
##########################################################################

## open the *_nodes.tsv file for printing
open(my $NODE_FILE, '>', $out_file."_nodes.tsv")
          or die "Could not open file: ".$out_file."_nodes.tsv";

# print the header line
print $NODE_FILE "id \t label \t chromosome\n";

# chromosome counter
my $c = 1;

# for each genomic bin;
```

```
for(my $row = 1; $row <= $stop_index[$num_chr]; $row++)
{
  # print the node information
  print $NODE_FILE $row."\tbin".$row."\t$c\n";

  if($row == $stop_index[$c])
  {
    # increment the chromosome counter
    $c = $c + 1;
  }
}

# close the *_nodes.tsv file
close $NODE_FILE;

#####################################################################
##  print the *_edges.tsv file
#####################################################################

## NOTE: when using the ForceAtlas2 layout - a larger weight will
## result in nodes being closer together. In this layout, edge weight
## represents the strength of the attraction

# open the *_edges.tsv file for printing
open(my $EDGE_FILE, ’>’, $out_file."_edges.tsv")
          or die "Could not open file: ".$out_file."_edges.tsv";

# print the header line
print $EDGE_FILE "Source\tTarget\tType_of_interaction\tWeight\n";

# chromosome counter
$c = 1;

# print out the linear interactions and their "distances" according
#  to the  requency value or experimental resolution
# (for s.pombe was 10 kb)
for(my $row = 1; $row <= $stop_index[$num_chr]; $row++)
{
  if($row < $stop_index[$c])
  {
    # experimental resolution is not inverted since edge weight
    # represents the strength of the attraction
    print $EDGE_FILE $row."\t".($row+1)."\tlinear".$c."\t".
              ($experimental_resolution)."\n";
  }
  else
  {
    # increment the chromosome counter
    $c = $c + 1;
  }

  # initialize the dynamics_coefficients hash for each bin to be = 0
  $dynamics_coefficents{$row} = 0;
}

##  Scan the result files

# for each intra-interaction
for(my $sp = 0; $sp < $num_sub_problems; $sp++)
{
  # read in the result file
  print STDERR "subproblem ".($sp+1).
          ": give the path for the results file: ";
  chomp(my $out_file = <STDIN>);

  ## open the file
  open RESULTS, "$out_file"
            or die "ERROR: $out_file could not be opened.";
  chomp(my @results_file = <RESULTS>);
  close RESULTS;

  # a boolean flag to let us know once the adjacency results start
  my $read_results = 0;

  for(my $i = 0; $i <= $#results_file; $i++)
  {

  if($read_results)
  {
    ## split the line from the clp file
    my @results_line =  split /\s+/, $results_file[$i];

    ## get node1
    my $node1 = $results_line[0];

    ## get node(s)2
    my $node2 = $results_line[1];

    ## get the corresponding frequency
    ## scale by $experimental_resolution
    my $freq = $results_line[2] * $experimental_resolution;

    ## determine if it is a cis- or trans- interaction
    my $chr1 = $chrs[$results_line[0]];
    my $chr2 = $chrs[$results_line[1]];

    ## if it is a cis- interaction
    if($chr1 == $chr2)
    {
      # add it to the hash
      $interactions{"cis_".$node1."_".$node2}{NODE1_BIN} = $node1;

      $interactions{"cis_".$node1."_".$node2}{NODE2_BIN} = $node2;

      $interactions{"cis_".$node1."_".$node2}{INTERACTION_FREQ} =
                $freq;
      $interactions{"cis_".$node1."_".$node2}{INTERACTION_TYPE} =
                "cis";

      $dynamics_coefficents{$node1} =
                $dynamics_coefficents{$node1} + 1;
      $dynamics_coefficents{$node2} =
                $dynamics_coefficents{$node2} + 1;
    }
    ## if it is a trans- interaction
    else
    {
      # add it to the hash
      $interactions{"trans_".$node1."_".$node2}{NODE1_BIN} = $node1;
      $interactions{"trans_".$node1."_".$node2}{NODE1_CHR} = $chr1;

      $interactions{"trans_".$node1."_".$node2}{NODE2_BIN} = $node2;
      $interactions{"trans_".$node1."_".$node2}{NODE2_CHR} = $chr2;

      $interactions{"trans_".$node1."_".$node2}{INTERACTION_FREQ} =
                 $freq;
      $interactions{"trans_".$node1."_".$node2}{INTERACTION_TYPE} =
                "trans";

      $dynamics_coefficents{$node1} =
                $dynamics_coefficents{$node1} + 1;
      $dynamics_coefficents{$node2} =
                $dynamics_coefficents{$node2} + 1;
    }
  }

  ## check to see if we should start reading during the next
  ## iteration; results start after a comment line beginning with %
  if($results_file[$i] =~ /^%/)
  {
    $read_results = 1;
  }
  }
}

#####################################################################
## print out the cis and trans interactions and their "distances"
##  according to the frequency value and dynamics coefficient
#####################################################################

foreach my $selected_interaction (sort keys %interactions)
{
    # calculate the average dynamics coefficient between
    # the two bins involved in the interactions
    my $d_coefficient =
    ($dynamics_coefficents{
        $interactions{
              $selected_interaction}{
                    NODE1_BIN}} +
    $dynamics_coefficents{
        $interactions
              {$selected_interaction}{
                    NODE2_BIN}})/2;

    # print the interaction
    print $EDGE_FILE $interactions{$selected_interaction}{NODE1_BIN}
      ."\t".$interactions{$selected_interaction}{NODE2_BIN}
      ."\t".$interactions{$selected_interaction}{INTERACTION_TYPE}
      ."\t".($interactions{$selected_interaction}{
              INTERACTION_FREQ}/$d_coefficient)."\n";
}

END {
  warn "this script took ", time - $^T, " seconds\n";
}
```

GeneRHi-C Step 3

Program 4.4: Archived version of the script used for step 3 of GeneRHi-C.

```
## script for calculating xyz coordinates in GeneRHi-C
## coordinates will be output in a csv format (Node,X,Y,Z)
## Argument 1: the input file name ( .gexf file)
## Argument 2: the output file name
##
## Kimberly MacKay Oct 16, 2019
## license: This work is licensed under the Creative Commons
## Attribution-NonCommercial-ShareAlike 3.0 Unported License.
## To view a copy of this license, visit
## http://creativecommons.org/licenses/by-nc-sa/3.0/ or send
## a letter to Creative Commons, PO Box 1866, Mountain View,
## CA 94042, USA.

import sys
import csv

# read in the gexf file, and parse appropriately
infilename = sys.argv[1]
infile = open(infilename, 'r')

allNodes = []
```

```
nodeID = ""
xyz = []

for line in infile:
  # if we are at a new node
  if "<node id=" in line:
    # add the old information to the list
    allNodes.append([nodeID] + xyz)

    l = line.split(’"’)

    # resent the variables
    nodeID = l[3]
    xyz = []
  # if we are at a line with position information
  elif "<viz:position" in line:
    # split the line
    l = line.split(’"’)

    # store the position information
    xyz = [l[1], l[3]]

# add the remaining node information
allNodes.append([nodeID] + xyz)

# output the results
outfilename = sys.argv[2]
header = ["Node", "X", "Y", "Z"]

outfile = open(outfilename, ’w’)
csv_wr = csv.writer(outfile)

csv_wr.writerow(header)
csv_wr.writerows(allNodes)

infile.close()
outfile.close()
```

CP Implementation

Program 4.5: Archived version of the implementation for the **CP** mathematical model (authored by Mats Carlsson).

```
%% Load the relevant libraries
include "globals.mzn";

%% Variable Declarations
int: N;
```

```

%% first chromosome of interest
set of 1..N: Chr1;

%% second chromosome of interest
set of 1..N: Chr2;

%% the given frequency map, assumed symmetric, main diagonal = 0
array[1..N,1..N] of int: map;

%% main decision variables
array[1..N] of var 1..N: match;

%% objective per row
array[1..N] of var int: rowobj;

%% total objective, N.B. counting each binding twice
var int: dobj;

%% mask out all entries not connecting Chr1 and Chr2
int: masked_map(1..N: i, 1..N: j) =
  if i in Chr1 /\ j in Chr2 then
    map[i,j]
  else if i in Chr2 /\ j in Chr1 then
    map[i,j]
  else 0 endif endif;

%% assertion: frequency map is symmetric
constraint
  forall(i in 1..N, j in 1..N where i<j)
    (assert(masked_map(i,j) = masked_map(j,i), "Asymmetry!"));

%% constrain the objective, one slice per row
constraint
  forall(i in 1..N)
    (rowobj[i] = [masked_map(i,j) | j in 1..N][match[i]]);

%% constrain the total objective
constraint
  dobj = sum(i in 1..N)(rowobj[i]);

%% domination: prevent zero-frequency edges
constraint
  forall(i in 1..N)
    (match[i] in {i} union {j | j in 1..N where masked_map(i,j)>0});

%% domination: prevent obviously suboptimal solutions
constraint
  forall(i in 1..N, j in 1..N where
    masked_map(i,j)>0)(rowobj[i] + rowobj[j] >= 1);

```

```
%% essential matching constraint
constraint
  inverse(match, match) :: domain;

%% Solve
solve :: int_search(rowobj, max_regret, indomain_max, complete)
  maximize(dobj);

%% output the results
output
 ["edge(\(i),\(match[i])). % benefit = \(rowobj[i])\n" |
   i in 1..N where fix(match[i])>i] ++
 ["objective(\(dobj div 2)).\n"] ++
 [];
```

GM Implementation

Program 4.6: Archived version of the implementation for the **GM** mathematical model (authored by Mats Carlsson).

```
%% Note: this program assumes a local version of BlossumV
%% exists on the computer.  The majority of this program formats the
%% input file in order to pass the data to BlossumV and parses
%% the output generated by that solver.  The program makes use of the
%% temporary files /tmp/all.in and /tmp/all.out.

:- use_module(library(lists)).
:- use_module(library(csv)).
:- use_module(library(system3)).

chromosome(testMap, 1, 1, 6).

genome_size(testMap, 6).

scale_factor(testMap, 1.0E1).

compile_adjacency(Path, Species) :-
  retractall(adjacency(_,_,_,_,_)),
  see(Path),
  read_record(_),
  repeat,
    read_record(Record),
    (   Record = end_of_file -> true
    ;   parse_record(Record, B1, B2, F),
        bin2chr(B1, Species, Chr1),
        bin2chr(B2, Species, Chr2),
        assertz(adjacency(B1, B2, F, Chr1, Chr2)),
        fail
    ), !,
  seen,
  save_predicates([adjacency/5], Species).

match_blossom5(Species, Set1, Set2) :-
  load_files(Species),
  retractall(edge(_,_,_)),
  (   adjacency(B1, B2, F, Chr1, Chr2),
      B1 < B2,
      (   member(Chr1, Set1), member(Chr2, Set2) -> true
      ;   member(Chr1, Set2), member(Chr2, Set1) -> true
      ),
      assertz(edge(B1, B2, F)),
      fail
  ;   true
  ),
  findall(edge(I,J,W), edge(I,J,W), Edges),
  length(Edges, E),
  genome_size(Species, N),
  scale_factor(Species, Scale),
  tell('/tmp/all.in'),
  NN is 2*N,
  EEN is 2*E+N,
  print_tab([NN,EEN]),
  (   foreach(edge(I1,J1,W1),Edges),
      param(Scale)
  do  I11 is I1-1,
      J11 is J1-1,
      W11 is integer(-Scale*W1),
      print_tab([I11,J11,W11])
  ),
  (   foreach(edge(I2,J2,W2),Edges),
      param(N,Scale)
  do  I21 is I2+N-1,
      J21 is J2+N-1,
      W21 is integer(-Scale*W2),
      print_tab([I21,J21,W21])
  ),
  (   for(K,1,N),
      param(N)
  do  K1 is K-1,
      K2 is K+N-1,
      print_tab([K1,K2,0])
  ),
  told,
  system('blossom5 -e /tmp/all.in -w /tmp/all.out '),
  see('/tmp/all.out'),
  retractall(solution(_,_,_)),
  read_line(_),
  repeat,
    read_line(Codes),
    (   Codes = end_of_file -> true
    ; append(Icodes, [0’ |Jcodes], Codes),
    number_codes(I0, Icodes),
    number_codes(J0, Jcodes),
    I is I0+1,
    J is J0+1,
    edge(I, J, W),
    assertz(solution(I,J,W)),
    fail
  ), !,
  seen,
  findall(W, solution(_,_,W), Ws),
  length(Ws, Size),
  sumlist(Ws, Weight),
  format(’% Blossom V computed a matching of size ~d and weight ~w\n’,
      [Size,Weight]),
  findall(0, (solution(I,J,W), print_tab([I,J,W])), _).

print_tab(L) :-
  (foreach(X,L) do write(X), write(’ ’)),
  nl.

parse_record(Record, B1, B2, F) :-
  Record = [integer(B1,_), integer(B2,_), float(F,_)].

bin2chr(Bin, Species, Chr) :-
  chromosome(Species, Chr, A, B),
  Bin >= A,
  Bin =< B, !.
```

4.11 Supplemental Endnotes

14. https://github.com/kimmackay/GeneRHi-C/tree/master/step1

15. https://developers.google.com/optimization/

16. https://github.com/kimmackay/GeneRHi-C/blob/master/step1/CP/supplementary/additional_file_2.dzn

17. https://github.com/kimmackay/GeneRHi-C/tree/master/step1/CP/pombe

18. http://www.minizinc.org/doc-lib/doc-globals-channeling.html

19. http://www.algorithmic-solutions.com/

20. `http://sicstus.sics.se`

21. `https://github.com/kimmackay/GeneRHi-C/blob/master/step1/GM/supplementary/additional_file_2.csv`

22. `https://github.com/kimmackay/GeneRHi-C/blob/master/step1/GM/pombe/Makefile`

Chapter 5

StoHi-C: Using t-Distributed Stochastic Neighbor Embedding (t-SNE) to Predict 3D Genome Organization from Hi-C Data

Kimberly MacKay Anthony Kusalik

Citation [100]: K MacKay and A Kusalik. StoHi-C: Using t-Distributed Stochastic Neighbor Embedding (t-SNE) to predict 3D genome organization from Hi-C Data. This version of the manuscript is available as a pre-print on BioRxiv: doi: `https://doi.org/10.1101/2020.01.28.923615`.

Detailed Contributions: Kimberly MacKay was responsible for writing the manuscript, performing the research and authoring the software. A brief description of the contributions for all listed authors can be found in Section 5.6.1.

As a reminder, the overarching goal of this thesis is to develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction. StoHi-C is a tool for predicting 3D genome organization from Hi-C datasets. Figure 1.3 indicates how StoHi-C fits into existing Hi-C data analysis workflows.

Additional Background: t-Stochastic Neighbourhood Embedding (t-SNE) is a non-linear dimensionality reduction technique that belongs to the class of stochastic neighbourhood embedding methods [60, 159, 160, 161, 162]. These methods use manifold-based learning to determine a low-dimensional representation of a given dataset while preserving local structure. [60, 160]. Briefly, a low-dimensional embedding is selected by minimizing the Kullback-Leibler divergence [124]. In order to find the optimal (or near-optimal) solution, this minimization is repeated multiple times with different stochastically-selected seeds. Ultimately, the embedding that resulted in the lowest Kullback-Leibler divergence is reported [159, 160, 161, 162].

5.1 Abstract

In order to comprehensively understand the structure-function relationship of the genome, 3D genome structures must first be predicted from biological data (like Hi-C) using computational tools. Many of these existing tools rely partially or completely on multi-dimensional scaling (MDS) to embed predicted structures in 3D space. MDS is known to have inherent problems when applied to high-dimensional datasets like Hi-C. Alternatively, t-Distributed Stochastic Neighbor Embedding (t-SNE) is able to overcome these problems but has not been applied to predict 3D genome structures. In this manuscript, we present a new workflow called StoHi-C (pronounced "stoic") that uses t-SNE to predict 3D genome structure from Hi-C data. StoHi-C was used to predict 3D genome structures for multiple, independent existing fission yeast Hi-C datasets. Overall, StoHi-C was able to generate 3D genome structures that more clearly exhibit the established principles of fission yeast 3D genomic organization

5.2 Introduction

Understanding the structure-function relationship of various biomolecules has been the foundation of molecular biology research for many years. Recently, the development of Hi-C (and related methods) has resulted in the generation of unprecedented sequence-level investigations into the structure-function relationship of the genome [15, 16, 92]. Hi-C is able to detect regions of the genome that are "interacting" (*i.e.* in close 3D spatial proximity). Typically, this is done by mapping Hi-C sequence reads to a reference genome [86, 99, 101, 167]. This results in the generation of a whole-genome contact map which is a $N \times N$ matrix where N is the number of "bins" which represent linear regions of genomic DNA. Each cell within a whole-genome contact map records a count of how many times two genomic regions were found in close proximity within a population of cells [86, 99, 101]. This is more commonly referred to as an interaction count. Interaction counts are often normalized using methods like iterative correction and eigenvector (ICE) decomposition [68, 164] to reduce inherent biases within Hi-C datasets [90, 96, 138, 143, 145, 172]. This normalization process results in fractional interaction counts also known as interaction frequencies.

Normalized whole-genome contact maps can be used to infer 3D genomic structure(s). This is known as the 3D genome reconstruction problem (3D-GRP) [99, 132] or the 3D chromatin structure modelling problem [178]. For the purpose of this manuscript, we will be using the term 3D-GRP. A formal representation of the 3D-GRP is provided by MacKay and Kusalik [99]. Briefly, normalized interaction frequencies are converted into a set of pairwise distances (based on the inverse of the interaction frequency). This calculation uses the assumption that a pair of genomic regions with a small interaction frequency will be further away in 3D space than a pair of genomic regions with a higher interaction frequency [6, 13, 14, 46, 52, 63, 87, 99, 127, 133, 163]. Each genomic bin's (x, y, z) coordinates are then calculated using various optimization techniques [99].

Many of the existing tools for solving the 3D-GRP rely on multi-dimensional scaling (MDS) either completely or partially to predict and embed genomic structures in 3D space. MDS is known to have inherent problems when calculating embeddings from population-based, sparse, high-dimensional datasets (which are characteristics of Hi-C datasets) [1, 125]. Alternatively, t-Stochastic Neighbourhood Embedding (t-SNE) has resulted in more accurate embeddings for datasets with these characteristics [159, 160, 161, 162]. Recently, Zhu *et al.* [180] were able to predict 3D structures of individual chromosomes using a manifold-learning approach (similar to t-SNE) combined with multi-conformation optimization. Their tool was shown to outperform many of the existing MDS-based methods but could not be applied to the entire 3D-GRP due to the underlying time complexity of multi-conformation optimization [180]. Based on the results of this regional prediction tool, t-SNE should result in more accurate solutions to the 3D-GRP when compared to existing MDS methods. To test this hypothesis, we developed a new workflow called StoHi-C (pronounced "stoic") that uses t-SNE to predict 3D genome structure from Hi-C data. StoHi-C and MDS were used to predict 3D genome structure for four existing fission yeast datasets (wild-type, G1-arrested, *rad21* deletion and *clr4D* deletion). Overall, StoHi-C was able to more clearly recapitulate well-documented features of fission yeast chromosomal organization (such as the RabI structure) when compared to the MDS method.

5.3 Methods

StoHi-C is a two step workflow that involves (1) 3D embedding and (2) visualization. A more detailed description of each step is provided in the subsections below. Each step can be run independently or users can invoke an automated shell script [23] that runs each step in succession. Complete documentation describing expected inputs, outputs and software requirements can be found on the project homepage [24]. In the subsequent sections, the StoHi-C workflow is described in general, but also provides details regarding the specific illustrative examples presented in this paper.

5.3.1 Step 1: 3D Embedding

The 3D coordinates for each genomic bin are calculated using t-SNE [159, 160, 161, 162]. A python script [25] was developed that accepts a normalized whole-genome contact map as input and outputs the (x, y, z) coordinates for each genomic bin. An example of the required input and expected output can be found on the project homepage. This script uses the `TSNE` method from the `sklearn.manifold` library to embed genomic bins in 3D space. The exact parameter values that were used for the fission yeast datasets as well as a brief description their function follow: `n_components = 3` (embedding dimensionality), `perplexity=5.0` (number of nearest neighbours), `early_exaggeration=3.0` (controls the tightness of clusters), `n_iter=5000` (maximum number of iterations), `method='exact'` (do not use the Barnes-Hut approximation), `init='pca'` (use a principle component analysis to initialize the embedding). These values were selected based on the suggestions provided on t-SNE's homepage [26].

5.3.2 Step 2: Visualization

Once the (x, y, z) coordinates are generated, a multitude of different tools can be used for visualization. Three options are discussed below but any graphing or network visualization tool that accepts 3D coordinates (where x, y, z values are space-delimited with each point on a separate row) could be used.

1. `plot.ly`: A python script `plotly_viz.py` [27] was developed that accepts the (x, y, z) coordinates generated in Step 1 and produces a static `PNG` image and an interactive 3D graph (`HTML`) using the `plot.ly` library [123]. The interactive graph can be opened in any web browser. *This option was used to generate the figures for the illustrative examples in this manuscript.*

2. `matplotlib`: A python script `matplotlib_viz.py` [28] was developed that accepts the (x, y, z) coordinates generated in Step 1 and produces a static `PNG` image of the corresponding 3D graph as well as a simple `MP4` animation that rotates around the y-axis. This script uses the `py.plot` and `animation` modules from the `matplotlib` library [67] as well as the `mpl_toolkits.mplot3d` toolkit [29].

3. `Chart Studio` : Alternatively, `plot.ly` has a web-based, interactive version available online called Chart Studio [30]. The 3D coordinates can be directly uploaded to the website to generate an interactive graph. Chart Studio has provided a detailed tutorial on generating this type of visualization [31]. Customized styles such as colours, labels, size, transparency, etc. for nodes and/or edges can then be set directly by the user through the graphical user interface. Additional node and/or edge attributes can be added to the 3D graph to incorporate complementary biological datasets (if available) with the visualization.

Options 1 and 2 have been automated for the visualization of fission yeast datasets with 10kb resolution. Applying them to datasets from other organisms or datasets with different resolutions would require slight adjustments to the scripts. Documentation of how to make these changes is provided on the project homepage [32]. Option 3 can not be automated since it is a graphical user interface.

5.3.3 Comparison with MDS

In order to compare the results of StoHi-C with MDS, the generation of (x, y, z) coordinates in step 1 was also done with metric-MDS. The use of metric-MDS for 3D genome prediction has been widely used since 2010 [46, 151]. Similarly to Step 1 of the StoHi-C workflow, a python script was developed [33] that accepts a normalized whole-genome contact map as input and outputs the (x, y, z) coordinates for each genomic bin. This script uses the `MDS` method from the `sklearn.manifold` library [121] to embed genomic bins in 3D space. The exact parameter values that were used for the fission yeast datasets as well as a brief description of their function follow: `n_components = 3` (embedding dimensionality), `metric=True` (use metric MDS), `max_iter=5000` (maximum number of iterations), `dissimilarity='precomputed'` (use a custom dissimilarity matrix). To be consistent with the StoHi-C workflow, the `plot.ly` script used for Step 2 (described above) was used to visualize the results for the illustrative examples in this manuscript.

5.3.4 Data Availability

The datasets supporting the conclusions of this article were originally generated by Mizuguchi *et al.* [108] and are available in the Gene Expression Omnibus database (accession number: GSE56849 [34]). The specific sample numbers are *999a* wild-type (GSM1379427), G1-arrested (GSM1379429), *rad21-K1* mutation (GSM1379430) and *clr4* deletion (GSM1379431).

5.3.5 Web Resources

StoHi-C is freely available at `https://github.com/kimmackay/StoHi-C` and is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0. It requires Python3 and local copies the following libraries: `numpy`, `sklearn` and `plot.ly`. These libraries are open access and can be downloaded through a package manager like `pip` or `conda`. Archived versions of the scripts used to generate the results in this manuscript are available as supplemental data.

5.4 Results & Discussion

The StoHi-C workflow and MDS method described above were used to generate 3D genome predictions for four existing Hi-C fission yeast datasets (*999a* wild type, G1-arrested, *rad21-K1* mutation, and *clr4* deletion) [108]. Depending on the method, either t-SNE or MDS was used to generate (x, y, z) coordinates. These results were then visualized with `plot.ly` which generates both static images and interactive graphs. Images representing the genomic predictions for each dataset with the StoHi-C workflow (Panels A,C,E,G) and MDS method (Panels B,D,F,H) are presented in Figure 5.1. Interactive versions of each `plot.ly` graph can be found on the project homepage [35]. Additionally, the project homepage contains the resultant images and animations generated with the `matplotlib` visualization for the *999a* wild-type dataset [36] [37].

Based on the results presented in Figure 5.1, the 3D genomic predictions generated with StoHi-C more clearly represent known features of fission yeast chromosomal organization when compared to the MDS method. The StoHi-C predictions (Figure 5.1A,C,E,G) all clearly depict universal hallmarks of genome organization (e.g. chromosome territories [31]) as well as fission yeast specific features (e.g. RabI configuration [49, 107]). Meanwhile, the MDS predictions all resulted in a hairball-like configuration with no apparent biological significance (Figure 5.1B,D,F,H). This is likely due to a fundamental difference in the algorithms underlying t-SNE and MDS. One of the goals of t-SNE is to preserve the local structure of high-dimensional datasets by placing similar features close together in the final embedding [159, 160, 161, 162]. MDS does the opposite, focusing on placing dissimilar features further away in the embedding [62].

StoHi-C has a worst-case time complexity of $O(N^2)$ (the time complexity of t-SNE [161]) where N is the number of genomic bins. This can be improved to $O(N\text{log}N)$ by using the Barnes-Hut approximation [160] which may be necessary for larger datasets. Classical metric-MDS has a worst-case time complexity of $O(N^3)$ [173] suggesting StoHi-C's runtime would be better than MDS-based methods in the extreme worst case. Table 5.1 lists the elapsed runtime required to embed and visualize each dataset using both the StoHi-C workflow and MDS method. These timings do not represent a comprehensive complexity analysis and

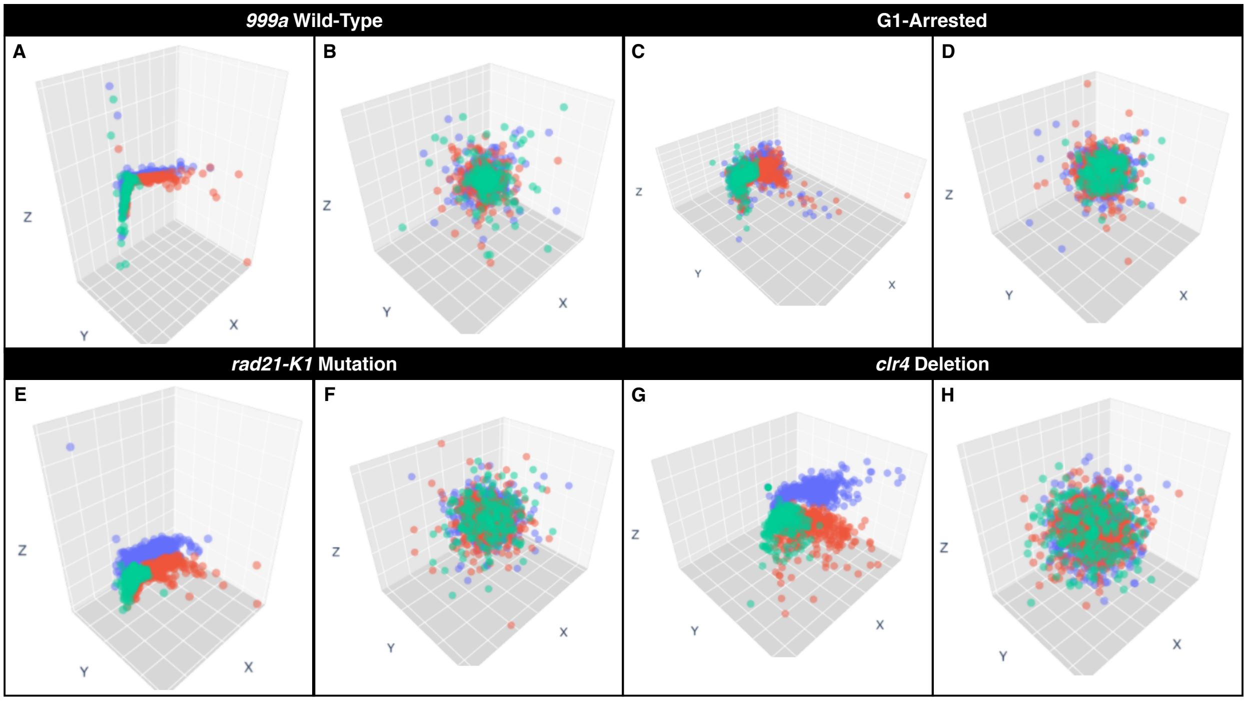


Figure 5.1: Visualizations of 3D genome predictions for four fission yeast datasets using StoHi-C and MDS. Panels A,C,E and G show 3D genome predictions produced with the StoHi-C workflow, while Panels B,D,F, and H depict the 3D genome predictions generated with MDS. In all panels, chromosomes are indicated with the following colours: purple (chromosome 1), orange (chromosome 2), green (chromosome 3). Corresponding datasets are indicated in the black box directly above the panels (*999a* wild-type: Panels A and B; G1-arrested: Panels C and G; *rad21-K1* mutation: Panels E and F; *clr4* deletion: Panels G and H). In each panel, the X, Y and Z axes are indicated with a corresponding label.

instead are presented to provide context as to whether or not these methods are practical for Hi-C-sized datasets. Interestingly, the MDS embedding is much faster than the t-SNE embedding with average elapsed times of 0.53 seconds and 11.0 seconds, respectively. This could be due to efficiencies in the implementations of the two algorithms.

To the best of our knowledge, none of the existing methods for predicting 3D genomic organization have been successfully applied to the datasets used in this paper. Previously, Tanizawa *et al.* [151] applied MDS to chromosome conformation capture data from fission yeast but the results were not able to recapitulate the RabI configuration of fission yeast chromosomes. StoHi-C was able to produce 3D genomic predictions that are consistent with the large body of work depicting fission yeast genomic organization including the RabI

Table 5.1: Elapsed Runtimes for 3D genome prediction with the StoHi-C workflow and MDS method. Elapsed runtimes are shown in seconds for the embedding (step 1), visualization (step 2) and complete workflow (total).

	StoHi-C Elapsed Time			**MDS Elapsed Time**		
Dataset	**Step 1**	**Step 2**	**Total**	**Step 1**	**Step 2**	**Total**
999a Wildtype	10.6	5.7	16.2	0.54	5.7	6.2
G1-Arrested	11.7	6.3	18.0	0.52	5.9	6.4
rad21-K1 Mutation	10.8	6.3	17.1	0.54	5.8	6.3
clr4 Deletion	10.7	5.8	16.5	0.51	5.6	6.1

configuration. This is the first time the RabI configuration has been successfully predicted from fission yeast Hi-C data. This is surprising when considering the relative simplicity of the fission yeast genome, but more understandable due to existing tools heavy reliance on MDS. It should be noted that polymer modelling of the same datasets was not successful [108].

While StoHi-C appears to be working well with data from the haploid organism fission yeast, additional step(s) may be required to apply it to organisms with higher ploidy (diploid, hexaploid, etc.) if the data is not pre-phased. This is because StoHi-C will have to determine which chromosome copy (or copies) contribute to the detected interactions (the ploidy problem). For now, users should preprocess polyploid Hi-C data with existing phasing tools (see review by Browning and Browning [18]) prior to using StoHi-C. To solve this problem more permanently, future work will focus on extending StoHi-C to include a step that performs phasing. This is something we are actively working toward in the hopes of applying StoHi-C to polyploid organisms. Once this has been completed, it will be deployed as a new version on the project homepage.

In this manuscript, we present a new workflow called StoHi-C (pronounced "stoic") that uses t-SNE to predict 3D genome structure from Hi-C data. Unlike MDS, t-SNE is well-suited for embedding population-based, sparse, high-dimensional data in 3D space. StoHi-C was used to predict 3D genome structures for four fission yeast Hi-C datasets. The results were compared to the 3D genomic structures predicted from the same datasets using a MDS approach. The 3D genomic predictions generated with StoHi-C more clearly represent known features of fission yeast chromosomal organization when compared to the MDS method. Additionally, this is the first time the RabI 3D genomic organization was successfully predicted from fission yeast Hi-C data. Overall, StoHi-C was able to generate 3D genome structures that more clearly exhibit the established principles of fission yeast 3D genomic organization when compared to the MDS results.

5.5 Endnotes

23. https://github.com/kimmackay/StoHi-C/blob/master/stohic.sh

24. https://github.com/kimmackay/StoHi-C/

25. https://github.com/kimmackay/StoHi-C/blob/master/step1/tSNE/run_tSNE.py

26. https://lvdmaaten.github.io/tsne/

27. https://github.com/kimmackay/StoHi-C/blob/master/step2/plotly_viz.py

28. https://github.com/kimmackay/StoHi-C/blob/master/step2/matplotlib_viz.py

29. https://matplotlib.org/mpl_toolkits/mplot3d/index.html#matplotlib-mplot3d-toolkit

30. https://chart-studio.plot.ly/create/#/

31. https://plotly.github.io/make-a-3d-scatter-plot/

32. https://github.com/kimmackay/StoHi-C/issues

33. https://github.com/kimmackay/StoHi-C/blob/master/step1/MDS/run_MDS.py

34. https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE56849

35. https://github.com/kimmackay/StoHi-C/tree/master/interactive_visualizations/

36. https://github.com/kimmackay/StoHi-C/tree/master/step2/tSNE_results/matplotlib/999a_WT

37. https://github.com/kimmackay/StoHi-C/tree/master/step2/MDS_results/matplotlib/999a_WT

5.6 Supplemental Information

5.6.1 Author Contributions

KM preformed the research and wrote the manuscript. AK supervised the research and edited the manuscript.

5.6.2 Funding Information

This work was supported by the Natural Sciences and Engineering Research Council of Canada [RGPIN 37207 to AK, Vanier Canada Graduate Scholarship to KM].

5.6.3 Archived Software

The following scripts are archived versions of the scripts used to generate the results presented in this manuscript. For the most recent version and/or to report any software problems please see the project homepage at `https://github.com/kimmackay/StoHi-C/`

Supplementary Script 1: StoHi-C

Program 5.7: Archived version of the shell script used to run each step of StoHi-C in succession.

```
#!/bin/bash
## Simple shell script that runs both steps of the StoHi-C workflow
## Author: Kimberly MacKay
## Date: December 16, 2019

## the script requires the following command-line inputs:
## Argument 1: should be the name of the whole-genome contact map
## Argument 2: is the name of the output file for the XYZ coordinates
## Argument 3: is the name of the output file for the distance matrix
## Argument 4: is the filename for the resultant image
## Argument 5: is the filename for the resultant interactive graph
##          (html)

# LISCENSE INFORMATION
# This work is licensed under the Creative Commons Attribution-Non
# Commercial-ShareAlike 3.0 Unported License. To view a copy of this
# license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or
# send a letter to Creative Commons, PO Box 1866, Mountain View, CA
# 94042, USA.

echo "running step 1..."
python ./step1/tSNE/run_tSNE.py $1 $2 $3

echo "running step 2..."
python ./step2/plotly_viz.py $2 $4 $5
```

Supplementary Script 2: StoHi-C Step 1

Program 5.8: Archived version of the the Python script used for Step 1 of StoHi-C.

```
# StoHi-C Step 1: 3D embedding using tSNE
# This script takes a normalized whole-genome contact map as input
# and embeds the genomic bins in 3D space using TSNE from
# sklearn.manifold

# Argument 1: the file name of the normalized whole-genome
#             contact map
# Argument 2: the output file name for the XYZ coordinates
# Argument 3: the output file name for the distance matrix
#             generated by tSNE

# AUTHOR INFORMATION:
# Kimberly MacKay
# kimberly.mackay@usask.ca
# @mackayka (twitter)

# Authored on April 30, 2019

# LISCENSE INFORMATION
# This work is licensed under the Creative Commons Attribution-Non
# Commercial-ShareAlike 3.0 Unported License. To view a copy of this
# license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or
# send a letter to Creative Commons, PO Box 1866, Mountain View, CA
# 94042, USA.

# import relevant libraries
import numpy as np
from sklearn.manifold import TSNE
import time
import sys

# define function for reading in data
def populate_matrix(filename, matrix):
  infile = open(filename, "r")

  row = 0

  for line in infile:
    line = line.rstrip()
    data = line.split("\t")

    col = 0

    # loop through all the elements in the line
    for val in data:
      if val == ’NA’:
        dist = 0.0
      elif float(val) == 0.0:
        dist = 0.0
      else:
        dist = 1.0/(float(val)**2)

      matrix[row][col] = dist

      # enforce that matrix[i][j] == matrix[j][i]
      matrix[col][row] = dist

      col = col + 1
    row = row + 1

  infile.close()
  return matrix

# grab command line arguments
input_file = sys.argv[1]
coord_file = sys.argv[2]
dist_file = sys.argv[3]

# initialize the distance matrix
dist_matrix = np.zeros((1258,1258))
dist_matrix = populate_matrix(input_file, dist_matrix)

# gut check that all the self-self interactions are zero
if sum(dist_matrix.diagonal()) != 0:
  print("WARNING: non-zero elements present in the diagonal")

#run TSNE
start_time = time.time()

# parameters
# n_components = dimensionality
# perplexity = # of nearest neighbours
# early_exaggeration = determines how "close" nodes will be in the
#                     final embedding
# n_iter = maximum number of iterations for the optimization
# method = exact (alternative would be an approximation)
# init = run PCA and use those results as input to tSNE
data_embedded = TSNE(n_components = 3, perplexity=5.0,
        early_exaggeration=3.0, n_iter=5000, method='exact',
        init='pca').fit_transform(dist_matrix)

stop_time = time.time()

print("tSNE runtime: " + str(stop_time - start_time) + " seconds")

# output embedded data
np.savetxt(coord_file, data_embedded)

# output distance matrix
np.savetxt(dist_file, dist_matrix)
```

Supplementary Script 3: StoHi-C Step 2

Program 5.9: Archived version of the the Python script used for Step 2 of StoHi-C.

```
## 3D visualization and basic animation of XYZ coordinates from step
## 1 of StoHi-C. This script uses plotly to generate a 3D scatter
## plot currently all the parameters are hardcoded for s. pombe data

## Argument 1: the XYZ co-ordinates for each genomic bin generated
##         from step 1 this file should have the XYZ coords for each
##         bin on a separate lineeach coord should be separated by
##         white space, bins should be in sorted numerical order,
##         there shouldn't be any column or row labels
## Argument 2: the name of the file for the output image
## Argument 3: the name of the file for the output html
##         (interactive graph)

# AUTHOR INFORMATION:
# Kimberly MacKay
# kimberly.mackay@usask.ca
# @mackayka (twitter)

# Authored on Dec. 12, 2019

# LISCENSE INFORMATION
# This work is licensed under the Creative Commons Attribution-Non
# Commercial-ShareAlike 3.0 Unported License. To view a copy of this
# license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or
# send a letter to Creative Commons, PO Box 1866, Mountain View, CA
# 94042, USA.

# import relavent libraries
```

```
import sys
import time
import numpy as np
from plotly import graph_objs as go
import chart_studio.plotly as py
import plotly.io as pio

start_time = time.time()

outimagename = sys.argv[2]
outfilename = sys.argv[3]

# read in the data
filename = sys.argv[1]
data_embedded = np.loadtxt(filename)


# generate a figure of the results
fig = go.Figure()


# chr1 - Fission Yeast
fig.add_trace(go.Scatter3d( x=data_embedded[0:558,0],
        y=data_embedded[0:558,1],
        z=data_embedded[0:558,2],
                    mode=’markers’,
                    opacity=0.5,
                    name="CHR1"))

# chr2 - Fission Yeast
fig.add_trace(go.Scatter3d( x=data_embedded[558:1012,0],
        y=data_embedded[558:1012,1],
        z=data_embedded[558:1012,2],
                    mode=’markers’,
                    opacity=0.5,
                    name="CHR2"))

# chr3 - Fission Yeast
fig.add_trace(go.Scatter3d( x=data_embedded[1012:1258,0],
        y=data_embedded[1012:1258,1],
        z=data_embedded[1012:1258,2],
                    mode=’markers’,
                    opacity=0.5,
                    name="CHR3"))


# set marker size
fig.update_traces(marker=dict(size=5))

# set axis titles
fig.update_layout(scene = dict(
                    xaxis_title='X',
                    yaxis_title='Y',
                    zaxis_title='Z'))

# set background colours, remove tick labels
fig.update_layout(scene = dict(
                    xaxis = dict(
                         backgroundcolor="rgb(245,245,245)",
                         gridcolor="white",
                         showbackground=True,
                         zerolinecolor="white",
                         showticklabels=False,),
                    yaxis = dict(
                        backgroundcolor="rgb(230,230,230)",
                        gridcolor="white",
                        showbackground=True,
                        zerolinecolor="white",
                        showticklabels=False,),
                    zaxis = dict(
                        backgroundcolor="rgb(215,215,215)",
                        gridcolor="white",
                        showbackground=True,
                        zerolinecolor="white",
                        showticklabels=False,),))

# output results
fig.write_image(outimagename)
plot_url = pio.write_html(fig, file=outfilename, auto_open=False)

stop_time = time.time()
print("Step 2 runtime: " + str(stop_time - start_time) + " seconds")
```

Supplementary Script 5: MDS

Program 5.10: Archived version of the shell script used to generate MDS embedding and visualize results.

```
#!/bin/bash
## Simple shell script that runs and visualizes the MDS prediction
## Author: Kimberly MacKay
## Date: December 16, 2019

## the script requires the following command-line inputs:
## Argument 1: should be the name of the whole-genome contact map
## Argument 2: is the name of the output file for the XYZ coordinates
## Argument 3: is the name of the output file for the distance matrix
## Argument 4: is the filename for the resultant image
## Argument 5: is the filename for the resultant interactive graph
##             (html)

# LISCENSE INFORMATION
# This work is licensed under the Creative Commons Attribution-Non
# Commercial-ShareAlike 3.0 Unported License. To view a copy of this
# license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or
# send a letter to Creative Commons, PO Box 1866, Mountain View, CA
# 94042, USA.

echo "running step 1..."
python ./step1/MDS/run_MDS.py $1 $2 $3

echo "running step 2..."
python ./step2/plotly_viz.py $2 $4 $5
```

Supplementary Script 5: MDS Step 1

Program 5.11: Archived version of the the Python script used to generate 3D coordinates with MDS.

```
# MDS Step 1: 3D embedding using MDS
# This script takes a normalized whole-genome contact map as input
# and embeds the genomic bins in 3D space using MDS from
# sklearn.manifold
# Argument 1: the file name of the normalized whole-genome
#             contact map
# Argument 2: the output file name for the XYZ coordinates
# Argument 3: the output file name for the distance matrix
#             generated

# AUTHOR INFORMATION:
# Kimberly MacKay
# kimberly.mackay@usask.ca
# @mackayka (twitter)

# Authored on April 30, 2019

# LISCENSE INFORMATION
# This work is licensed under the Creative Commons Attribution-Non
# Commercial-ShareAlike 3.0 Unported License. To view a copy of this
# license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/ or
# send a letter to Creative Commons, PO Box 1866, Mountain View, CA
# 94042, USA.

# import relevant libraries
import numpy as np
from sklearn.manifold import MDS
from sklearn.decomposition import PCA
import time
import sys

# define function for reading in data
# note for MDS it takes a dissimilarity matrix
def populate_matrix(filename, matrix):
  infile = open(filename, "r")

  row = 0

  for line in infile:
    line = line.rstrip()
    data = line.split("\t")

    col = 0

    # loop through all the elements in the line
    for val in data:
      # need to do a smarter imputation of missing data
      if val == ’NA’:
        dist = 0.0
      elif float(val) == 0.0:
        dist = 0.0
      else:
        dist = (1.0/(float(val)**2))

      matrix[row][col] = dist
      # enforce that matrix[i][j] == matrix[j][i]
      matrix[col][row] = dist

      col = col + 1
    row = row + 1

  infile.close()
  return matrix


# grab command line arguments
input_file = sys.argv[1]
coord_file = sys.argv[2]
dist_file = sys.argv[3]

# initialize the distance matrix
dist_matrix = np.zeros((1258,1258))
dist_matrix = populate_matrix(input_file, dist_matrix)

# gut check that all the self-self interactions are zero
if sum(dist_matrix.diagonal()) != 0:
  print("WARNING: non-zero elements present in the diagonal")

```

```
# compute the dissimilarity matrix
dissimilarity_matrix = 1 - dist_matrix

#run MDS
#print("running MDS...")
start_time = time.time()

data_embedded = MDS(n_components=3, metric=True, max_iter=5000,
        dissimilarity='precomputed').
        fit_transform(dissimilarity_matrix)

stop_time = time.time()

print("MDS runtime: " + str(stop_time - start_time) + " seconds")

# output embedded data
np.savetxt(coord_file, data_embedded)

# output distance matrix
np.savetxt(dist_file, dist_matrix)
```

Chapter 6
Discussion & Conclusion

6.1 Main Contributions

The overarching goal of this thesis was to develop new computational tools for Hi-C analysis and 3D genome prediction. To answer this objective, we developed new tools for Hi-C analysis (GrapHi-C – Chapter 2) and 3D genome prediction (GeneRHi-C and StoHi-C – Chapters 4 and 5, respectively). Figure 1.3 highlights how these new tools fit into existing Hi-C analysis workflows. All of the tools developed in this thesis were applied to existing fission yeast dataset(s) and were shown to more clearly recapitulate documented features of fission yeast genomic organization when compared to existing methods. We also performed a comprehensive survey of the existing tools for predicting 3D genome structure from Hi-C data (Chapter 3). The following provides a brief discussion on the main findings and conclusions from each paper.

In Chapter 2, we described a new tool called GrapHi-C that can be used for visualizing Hi-C datasets. Briefly, we developed a mathematical model for graph-based representations of contact maps that explicitly encodes linearly adjacent regions (i.e. regions on the same chromosome) as well as *cis-* and *trans-* interactions. GrapHi-C was applied to multiple existing fission yeast datasets and the resulting images were compared to the corresponding heatmaps and Circos plots. Overall, it was shown that GrapHi-C generates a more intuitive, structural visualization of Hi-C data. Even though the graph-based representation seems straightforward, this representation was still novel in the genome structure community. Overall, the developed GrapHi-C visualizations of the contact maps (compared to the equivalent heatmaps and Circos plots) made it easier to quickly identify the changes in genome organization identified in previous studies.

As depicted in Figure 1.3, 3D genomic prediction can be a part of Hi-C analysis workflows. More formally, 3D genome prediction from Hi-C (or related) data is known as the 3D-GRP. Many existing tools for solving the 3D-GRP have been published. The manuscript presented in Chapter 3, provides a comprehensive review of these tools (from November 2006 to September 2019, inclusive). Over 40 tools were examined and characterized. Overall, we determined that many of these existing tools could not be applied to non-model organisms due to inherent constraints imposed by the underlying techniques. Additionally, existing tools relied on a relatively small set of algorithmic strategies. While this is not inherently problematic, we were able to identify some of the unexplored algorithmic areas that will be promising as the community moves towards the development of more general tools. Finally, we investigated which organisms and/or datasets each tool had successfully used for predicting 3D genomic structure. This analysis showed that while a diverse set of Hi-C data exists, the vast majority has not been applied to 3D genome prediction. Interestingly, none of the existing tools have been able to successfully predict 3D genomic structure from the simple model organism fission yeast [99]. This was surprising since simpler model organisms like fission yeast are often used to demonstrate a tools utility before moving on towards more complex organisms.

In order to address the problems presented in Chapter 3, two new tools for solving the 3D-GRP were developed (GeneRHi-C – Chapter 4 and StoHi-C – Chapter 5). Both tools use algorithmic strategies that had not been employed by existing 3D-GRP solutions and result in consensus predictions of 3D genome organization. Specifically, GeneRHi-C predicts 3D genomic structure by using integer programming combined with 3D network layouts while StoHi-C uses t-SNE. GeneRHi-C utilizes a ploidy dependent subset of interactions while StoHi-C uses the complete contact map. To demonstrate their utility, they were used to predict 3D genome organization from an existing fission yeast dataset. As mentioned previously, none of the existing tools have been successful in reconstructing genome organization from this dataset. Alternatively, both GeneRHi-C and StoHi-C were able to successfully reconstruct known hallmarks of fission yeast genomic organization like the RabI chromosomal configuration. While they do not solve all of the problems identified in Chapter 3, they are a step towards more generalizable solutions to the 3D-GRP.

6.2 Main Limitations

All of the tools presented in this thesis have only been tested with Hi-C datasets from fission yeast (a haploid organism with a relatively small genome). In order to gain a more comprehensive understanding of their utility, these tools should be applied to a more diverse collection of datasets, with a variety of qualities and ploidies. The current formulations of GeneRHi-C and StoHi-C can only be applied to datasets from haploid organisms or datasets that have been pre-phased. This restriction is quite common in the set of existing tools [99] but it is not desirable since it limits investigations of 3D genomic structure to a small set of organisms and/or datasets.

6.3 Future Work

While a great deal of foundational research has been done by the 3D genomics community, many areas remain unexplored. For instance, 3D genomic prediction has not been applied to over 99% of the existing Hi-C datasets in the Gene Expression Omnibus database [99]. Additionally, tools for predicting 3D genome organization have not been used with Hi-C datasets from organisms with triploid or higher ploidy [99]. Future work will focus on addressing this imbalance by extending and applying the tools presented in this thesis to underrepresented organisms. These genomic predictions could then be integrated with other types of biological data to better understand underlying mechanisms of phenomena like gene regulation and complex traits. Currently, we are using the tools presented here to help characterize and predict 3D genome organization in *Brassica napus* (canola).

For the tools presented in this thesis, future work will focus on extending them to address the limitations described above as well as providing added functionality. For instance, GrapHi-C will be extended to allow for the co-visualization of complementary -omics datasets (such as gene expression, epigenetic markers or transcription factor binding sites). We will also establish how well GrapHi-C performs with unfavourable Hi-C datasets or datasets with technical problems. GeneRHi-C and StoHi-C will be extended to allow for additional -omics datasets to be incorporated into the prediction and/or final visualization. Additionally, GeneRHi-C and StoHi-C will be modified so that they can be used with polyploid datasets. Finally, feature(s) will be added to these tools that allow for the selection of substructures surrounding genes or genomic regions of interest.

Interestingly, a formal proof of the complexity associated with the 3D-GRP has not been published. In more accessible terms, a formal proof of 3D-GRP complexity would give the community a better understanding of how hard the problem is and allow for the easy identification of computational strategies (or approximations) that could be used to generate efficient solutions. Future work will focus on generating this proof. Initial analysis has suggested that the current formulation of the 3D-GRP (given in Chapter 3) might be NP-hard but faster approximations exist that could exist. Once this proof has been finalized, a new tool for solving the 3D-GRP (or an approximation) will be developed that is more efficient then existing methods. This is important as the size of 3D-GRP problems is increasing due to higher resolution datasets and/or applications to organisms with larger genomes.

As mentioned in Section 4.7.2, one of the major bottlenecks affecting the 3D genomics community is a lack of "ground-truth" structures and associated datasets that could be used to evaluate the accuracy of 3D-GRP tools. This is a problem since it is difficult to assess which of the existing methods does the "best" job of reconstructing genomic structure. Typically, tools are benchmarked based on other factors like runtime and visual comparison with microscopy images. While useful, these metrics do not give a complete representation of a given tool's reconstruction accuracy. Future work will focus on generating synthetic datasets (that maintain the characteristics of Hi-C data) with known ground-truth structures. Once these are generated, we will preform a comprehensive evaluation of existing tools for solving the 3D-GRP using this dataset as well as other metrics currently used by the community. This type of evaluation is imperative in-order to gain a sequence-level understanding of the structure-function relationship of the genome.

Future work will also include a comprehensive analysis to characterize the effect(s) of reference genome selection in Hi-C analysis and 3D genomic predictions. As indicated in Figure 1.3 (Panel 1), all of the existing Hi-C analysis tools require a reference genome to generate a whole-genome contact map. While useful, the choice of reference genome could introduce various biases or inaccuracies into the analysis pipeline. We will assess whether different reference genomes (i.e. ones generated from short versus long read sequencing) result in different 3D predictions. Finally, we will characterize the potential biological implications of reference genome selection in regards to 3D genome predictions.

6.4 Conclusion

As mentioned previously, the overarching goal of this thesis was to develop new, generalizable computational tools for Hi-C analysis and 3D genome prediction. To answer this objective, we developed new tools for Hi-C analysis (GrapHi-C – Chapter 2) and 3D genome prediction (GeneRHi-C and StoHi-C – Chapters 4 and 5, respectively). Figure 1.3 highlights how these new tools fit into existing Hi-C analysis workflows. All the tools were applied to multiple existing fission yeast datasets and were shown to more clearly recapitulate documented features of fission yeast genomic organization when compared to existing techniques. We also performed a comprehensive survey of the existing tools for predicting 3D genome structure from Hi-C data (Chapter 3). Overall, the manuscripts presented in this thesis provide new options for Hi-C data analysis and are a step towards a more comprehensive understanding of the structure-function relationship of the genome.

References

[1] Badri Adhikari, Tuan Trieu, and Jianlin Cheng. Chromosome3D: reconstructing three-dimensional chromosomal structures from Hi-C interaction frequency data using distance geometry simulated annealing. *BMC Genomics*, 17:3210–3214, 2016.

[2] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Proceedings, Database Theory — International Conference on Database Theory (London, United Kingdom)*, pages 420–434. Springer Berlin Heidelberg, 2001.

[3] Guillaume Andrey and Stefan Mundlos. The three-dimensional genome: regulating gene expression during pluripotency and development. *Development*, 144:3646–3658, 2017.

[4] Rodrigo G. Arzate-Mejıá, Félix Recillas-Targa, and Victor G. Corces. Developing in 3D: the role of CTCF in cell differentiation. *Development*, 145:dev137729, 2018.

[5] Haruhiko Asakawa, Tokuko Haraguchi, and Yasushi Hiraoka. Reconstruction of the kinetochore: a prelude to meiosis. *Cell Division*, 2:17, 2007.

[6] Ferhat Ay, Evelien M. Bunnik, Nelle Varoquaux, Sebastiaan M. Bol, Jacques Prudhomme, Jean-Philippe Vert, William Stafford Noble, and Karine G. Le Roch. Three-dimensional modeling of the *P. falciparum* genome during the erythrocytic cycle reveals a strong connection between genome architecture and gene expression. *Genome Research*, 24:974–988, March 2014.

[7] Ferhat Ay and William S. Noble. Analysis methods for studying the 3D architecture of the genome. *Genome Biology*, 16(1):1–15, September 2015.

[8] Sepideh Babaei, Waseem Akhtar, Johann de Jong, Marcel Reinders, and Jeroen de Ridder. 3D hotspots of recurrent retroviral insertions reveal long-range interactions with cancer genes. *Nature Communications*, 6:6381, 2015.

[9] Deepak Babu and Melissa J. Fullwood. 3D genome organization in health and disease: emerging opportunities in cancer translational medicine. *Nucleus*, 6(5):382–393, 2015.

[10] A. Rasim Barutcu, Andrew J. Fritz, Sayyed K. Zaidi, André J. van Wijnen, Jane B. Lian, Janet L. Stein, Jeffrey A. Nickerson, Anthony N. Imbalzano, and Gary S. Stein. C-ing the genome: A compendium of chromosome conformation capture methods to study higher-order chromatin organization. *Journal of Cellular Physiology*, 231(1):31–35, 2016.

[11] Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[12] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: an open source software for exploring and manipulating networks. In *Proceedings, International AAAI Conference on Weblogs and Social Media (Palo Alto, California, US)*. AAAI Press, 2009.

[13] Davide Baù and Marc A. Marti-Renom. Structure determination of genomic domains by satisfaction of spatial restraints. *Chromosome Research*, 19:25–35, 2011.

[14] Davide Baù and Marc A. Marti-Renom. Genome structure determination via 3C-based data integration by the Integrative Modeling Platform. *Methods*, 58:300–306, 2012.

[15] Houda Belaghzala, Job Dekker, and Johan H. Gibcusa. Hi-C 2.0: An optimized Hi-C procedure for high-resolution genome-wide mapping of chromosome conformation. *Methods*, 123:56–65, 2017.

[16] Jon-Matthew Belton, Rachel P. McCord, Johan Harmen Gibcus, Natalia Naumova, Ye Zhan, and Job Dekker. Hi–C: A comprehensive technique to capture the conformation of genomes. *Methods*, 58:268–276, 2012.

[17] Shay Ben-Elazar, Benny Chor, and Zohar Yakhini. Extending partial haplotypes to full genome haplotypes using chromosome conformation capture data. *Bioinformatics*, 32:i559–i566, 2016.

[18] Sharon R. Browning and Brian L. Browning. Haplotype phasing: existing methods and new developments. *Nature Reviews Genetics*, 12:703–714, 2011.

[19] Mats Carlsson and Per Mildner. SICStus Prolog - the first 25 years. *Theory and Practice of Logic Programming*, 12(1-2):35–66, 2012.

[20] Simeon Carstens, Michael Nilges, and Michael Habeck. Inferential structure determination of chromosomes from single-cell Hi-C data. *PLOS Computational Biology*, 12(12):e1005292, 2016.

[21] Claudia Caudai, Emanuele Salerno, Monica Zoppe, Ivan Merelli, and Anna Tonazzini. ChromStruct 4: A Python code to estimate the chromatin structure from Hi-C data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(6):1867–1878, 2018.

[22] Claudia Caudai, Emanuele Salerno, Monica Zoppè, and Anna Tonazzini. Inferring 3D chromatin structure using a multiscale approach based on quaternions. *BMC Bioinformatics*, 16:234, 2015.

[23] Lyubomira Chakalova, Emmanuel Debrand, Jennifer A. Mitchell, Cameron S. Osborne, and Peter Fraser. Replication and transcription: shaping the landscape of the genome. *Nature Reviews Genetics*, 6(9):669–677, 2005.

[24] Andrea M. Chiariello, Carlo Annunziatella, Simona Bianco, Andrea Esposito, and Mario Nicodemia. Polymer physics of chromosome large-scale 3D organisation. *Scientific Reports*, 6:29775, July 2016.

[25] Yuji Chikashige, Da-Qiao Ding, Yoshiyuki Imai, Masayuki Yamamoto, Tokuko Haraguchi, and Yasushi Hiraoka. Meiotic nuclear reorganization: switching the position of centromeres and telomeres in the fission yeast Schizosaccharomyces pombe. *The EMBO Journal*, 16(1):193–202, 1997.

[26] Yongwook Choi, Agnes P. Chan, Ewen Kirkness, Amalio Telenti, and Nicholas J. Schork. Comparison of phasing strategies for whole human genomes. *PLOS Genetics*, 14(4):e1007308, 2018.

[27] Peter R. Cook and Davide Marenduzzo. Transcription-driven genome organization: a model for chromosome structure and the regulation of gene expression tested through simulations. *Nucleic Acids Research*, 46(19):9896–9906, 2018.

[28] Nathan F. Cope and Peter Fraser. Chromosome conformation capture. *Cold Spring Harbor Protocols*, 2009(2):pdb.prot5137, 2009.

[29] Axel Cournac, Hervé Marie-Nelly, Martial Marbouty, Romain Koszul, and Julien Mozziconacci. Normalization of a chromosomal contact map. *BMC Genomics*, 13:436, 2012.

[30] Thomas Cremer and Christoph Cremer. Chromosome territories, nuclear architecture and gene regulation in mammalian cells. *Nature Reviews Genetics*, 2:292–301, 2001.

[31] Thomas Cremer and Marion Cremer. Chromosome territories. *Cold Spring Harbor Perspectives in Biology*, 2(3):a003889, 2010.

[32] Elzo de Wit. TADs as the caller calls them. *Journal of Molecular Biology*, 432(3):638–642, 2020.

[33] Elzo de Wit and Wouter de Laat. A decade of 3C technologies: insights into nuclear organization. *Genes & Development*, 26:11–24, 2017.

[34] Job Dekker. Regulation of gene expression through chromatin interaction networks. *Blood Cells, Molecules, and Diseases*, 38(2):135, 2007.

[35] Job Dekker and Leonid Mirny. The 3D genome as moderator of chromosomal communication. *Cell*, 164(6):1110–1121, 2016.

[36] Job Dekker, Karsten Rippe, Martijn Dekker, and Nancy Kleckner. Capturing chromosome conformation. *Science*, 295(5558):1306–1311, 2002.

[37] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[38] Annette Denker and Wouter de Laat. The second decade of 3C technologies: detailed insights into nuclear organization. *Genes & Development*, 30(12):1357–1382, 2016.

[39] Alon Diament and Tamir Tuller. Improving 3D genome reconstructions using orthologous and functional constraints. *PLOS Computational Biology*, 11(5):e1004298, 2015.

[40] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[41] Jesse R Dixon, Siddarth Selvaraj, Feng Yue, Audrey Kim, Yan Li, Yin Shen, Ming Hu, Jun S Liu, and Bing Ren. Topological domains in mammalian genomes identified by analysis of chromatin interactions. *Nature*, 485(7398):376–380, 2012.

[42] Pengfei Dong, Xiaoyu Tu, Po-Yu Chu, Peitao Lü, Ning Zhu, Donald Grierson, Baijuan Du, Pinghua Li, and Silin Zhong. 3D chromatin architecture of large plant genomes determined by local A/B compartments. *Molecular Plant*, 10(12):1497–1509, 2017.

[43] Qianli Dong, Ning Li, Xiaochong Li, Zan Yuan, Dejian Xie, Xiaofei Wang, Jianing Li, Yanan Yu, Jinbin Wang, Baoxu Ding, Zhibin Zhang, Changping Li, Yao Bian, Ai Zhang, Ying Wu, Bao Liu, and Lei Gong. Genome-wide Hi-C analysis reveals extensive hierarchical chromatin interactions in rice. *The Plant Journal*, 94(6):1141–1156, 2018.

[44] Josée Dostie, Todd A. Richmond, Ramy A. Arnaout, Rebecca R. Selzer, William L. Lee, Tracey A. Honan, Eric D. Rubio, Anton Krumm, Justin Lamb, Chad Nusbaum, Roland D. Green, and Job Dekker. Chromosome conformation capture carbon copy (5C): A massively parallel solution for mapping interactions between genomic elements. *Genome Research*, 16(10):1299–1309, 2006.

[45] Nicola H. Dryden, Laura R. Broome, Frank Dudbridge, Nichola Johnson, Nick Orr, Stefan Schoenfelder, Takashi Nagano, Simon Andrews, Steven Wingett, Iwanka Kozarewa, Ioannis Assiotis, Kerry Fenwick, Sarah L. Maguire, James Campbell, Rachael Natrajan, Maryou Lambros, Eleni Perrakis, Alan Ashworth, Peter Fraser, and Olivia Fletcher. Unbiased analysis of potential targets of breast cancer susceptibility loci by Capture Hi-C. *Genome Research*, 24(11):1854–1868, 2014.

[46] Zhijun Duan, Mirela Andronescu, Kevin Schutz, Sean McIlwain, Yoo Jung Kim, Choli Lee, Jay Shendure, Stanley Fields, C. Anthony Blau, and William S. Noble. A three-dimensional model of the yeast genome. *Nature*, 465:363–367, 2010.

[47] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17(3):449–467, 1965.

[48] Stephanie Fanucchi, Youtaro Shibayama, Shaun Burd, Marc S. Weinberg, and Musa M. Mhlanga. Chromosomal contact permits transcription between coregulated genes. *Cell*, 115(3):606–620, 2013.

[49] Alfonso Fernández-Álvarez and Julia Promisel Cooper. The functionally elusive rabi chromosome configuration directly regulates nuclear membrane remodeling at mitotic onset. *Cell Cycle*, 16(15):1392–1396, 2017.

[50] Maria A. Ferraiuolo, Mathieu Rousseau, Carol Miyamoto, Solomon Shenker, Xue Qing David Wang, Michelle Nadler, Mathieu Blanchette, and Josée Dostie. The three-dimensional architecture of hox cluster silencing. *Nucleic Acids Research*, 38(21):7472–7484, 2010.

[51] Robert W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.

[52] James Fraser, Mathieu Rousseau, Mathieu Blanchette, and Josée Dostie. Computing chromosome conformation. *Methods in Molecular Biology*, 674:251–268, 2010.

[53] James Fraser, Mathieu Rousseau, Solomon Shenker, Maria A. Ferraiuolo, Yoshihide Hayashizaki, Mathieu Blanchette, and Josée Dostie. Chromatin conformation signatures of cellular differentiation. *Genome Biology*, 10:R37, 2009.

[54] Melissa J. Fullwood and Yijun Ruan. ChIP-based methods for the identification of long-range chromatin interactions. *Journal of Cellular Biochemistry*, 107(1):30–39, 2009.

[55] Hironori Funabiki, Iain Hagan, Satoru Uzawa, and Mitsuhiro Yanagida. Cell cycle-dependent specific positioning and clustering of centromeres and telomeres in fission yeast. *Journal of Cell Biology*, 121:961–976, 1993.

[56] Luca Giorgetti, Rafael Galupa, Elphège P. Nora, Tristan Piolot, France Lam, Job Dekker, Guido Tiana, and Edith Heard. Predictive polymer modeling reveals coupled fluctuations in chromosome conformation and transcription. *Cell*, 157:950–963, 2014.

[57] Jinlei Han, Zhiliang Zhang, and Kai Wang. 3C and 3C-based techniques: the powerful tools for spatial genome organization deciphering. *Molecular Cytogenetics*, 11:21, 2018.

[58] Wilfred K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[59] Sven Heinz, Christopher Benner, Nathanael Spann, Eric Bertolino, Yin C. Lin, Peter Laslo, Jason X. Cheng, Cornelis Murre, Harinder Singh, and Christopher K. Glass. Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Molecular Cell*, 38(4):576–589, 2010.

[60] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. In *Proceedings, Fifteenth International Confrence on Neural Information Processing Systems (Vancouver, BC, Canada)*, volume 15, pages 857–864. MIT Press, 2002.

[61] Yoshito Hirata, Arisa Oda, Kunihiro Ohta, and Kazuyuki Aihara. Three-dimensional reconstruction of single-cell chromosome structure using recurrence plots. *Scientific Reports*, 6:34982, 2016.

[62] Michael C. Hout, Megan H. Papesh, and Stephen D. Goldinger. Multidimensional scaling. *WIREs Cognitive Science*, 4(1):93–103, 2013.

[63] Ming Hu, Ke Deng, Zhaohui Qin, Jesse Dixon, Siddarth Selvaraj, Jennifer Fang, Bing Ren, and Jun S. Liu. Bayesian inference of spatial organizations of chromosomes. *PLOS Computational Biology*, 9(1):e1002893, 2013.

[64] Ming Hu, Ke Deng, Siddarth Selvaraj, Zhaohui Qin, Bing Ren, and Jun S. Liu. HiCNorm: removing biases in Hi-C data via Poisson regression. *Bioinformatics*, 28(23):3131–3133, 2012.

[65] Clemens B. Hug and Juan M. Vaquerizas. The birth of the 3D genome during early embryonic development. *Trends in Genetics*, 34(12):903–914, 2018.

[66] Jim R. Hughes, Nigel Roberts, Simon McGowan, Deborah Hay, Eleni Giannoulatou, Magnus Lynch, Marco De Gobbi, Stephen Taylor, Richard Gibbons, and Douglas R. Higg. Analysis of hundreds of cis-regulatory landscapes at high resolution in a single, high-throughput experiment. *Nature Genetics*, 46(2):205–212, 2014.

[67] John D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[68] Maxim Imakaev, Geoffrey Fudenberg, Rachel Patton McCord, Natalia Naumova, Anton Goloborodko, Bryan R. Lajoie, Job Dekker, and Leonid A. Mirny. Iterative correction of Hi-C data reveals hallmarks of chromosome organization. *Nature Methods*, 9(10):999–1003, 2012.

[69] Gurobi Optimization Inc. Gurobi Optimizer Reference Manual, 2014. http://www.gurobi.com.

[70] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLOS ONE*, 9(6):e98679, 2014.

[71] Roland Jäger, Gabriele Migliorini, Marc Henrion, Radhika Kandaswamy, Helen E. Speedy, Andreas Heindl, Nicola Whiffin, Maria J. Carnicer, Laura Broome, Nicola Dryden, Takashi Nagano, Stefan Schoenfelder, Martin Enge, Yinyin Yuan, Jussi Taipale, Peter Fraser, Olivia Fletcher, and Richard S. Houlston. Capture Hi-C identifies the chromatin interactome of colorectal cancer risk loci. *Nature Communications*, 6:6178, 2015.

[72] Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM*, 24(1):1–13, 1977.

[73] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Foundations and Advances*, 32(5):922–923, 1976.

[74] Reza Kalhor, Harianto Tjong, Nimanthi Jayathilaka, Frank Alber, and Lin Chen. Genome architectures revealed by tethered chromosome conformation capture and population-based modeling. *Nature Biotechnology*, 30(1):90–98, 2012.

[75] Viacheslav Kapilevich, Shigeto Seno, Hideo Matsuda, and Yoichi Takenaka. Chromatin 3D reconstruction from chromosomal contacts using a genetic algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(5):1620–1626, 2018.

[76] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[77] Philip A. Knight and Daniel Ruiz. A fast algorithm for matrix balancing. *Journal of Numerical Analysis*, 33(3):1029–1047, 2012.

[78] Vladimir Kolmogorov. Blossom V: a new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1):43–67, 2009.

[79] Petros Kolovos, Harmen J.G. van de Werken, Nick Kepper, Jessica Zuin, Rutger W.W. Brouwer, Christel E.M. Kockx, Kerstin S. Wendt, Wilfred F.J. van IJcken, Frank Grosveld, and Tobias A. Knoch. Targeted chromatin capture (T2C): a novel high resolution high throughput method to detect genomic interactions and regulatory elements.): a novel high resolution high throughput method to detect genomic interactions and regulatory elements. *Epigenetics & Chromatin*, 7:10, 2014.

[80] Joseph B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[81] Joseph B. Kruskal. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29(2):115–129, 1964.

[82] Martin I. Krzywinski, Jacqueline E. Schein, Inanc Birol, Joseph Connors, Randy Gascoyne, Doug Horsman, Steven J. Jones, and Marco A. Marra. Circos: An information aesthetic for comparative genomics. *Genome Research*, 19:1639–1645, 2009.

[83] Masahiko Kuroda, Hideyuki Tanabe, Keiichi Yoshida, Kosuke Oikawa, Akira Saito, Tomoharu Kiyuna, Hiroshi Mizusawa, and Kiyoshi Mukai. Alteration of chromosome positioning during adipocyte differentiation. *Journal of Cell Science*, 117:5897–5903, November 2004.

[84] Antonija Kuzmanic, Navraj S. Pannu, and Bojan Zagrovic. X-ray refinement significantly underestimates the level of microscopic heterogeneity in biomolecular crystals. *Nature Communications*, 5:3220, 2014.

[85] Imielinski lab (New York Genome Center) collaboration. Pore-C: using nanopore reads to delineate long-range interactions between genomic loci in the human genome. Technical report, Oxford Nanoproe Technologies, (Poster) 2018.

[86] Bryan R. Lajoie, Job Dekker, and Noam Kaplan. The hitchhiker's guide to Hi-C analysis: Practical guidelines. *Methods*, 72:65–75, 2015.

[87] Annick Lesne, Julien Riposo, Paul Roger, Axel Cournac, and Julien Mozziconacci. 3D genome reconstruction from chromosomal contacts. *Nature Methods*, 11:1141–1143, 2014.

[88] Jiangeng Li, Wei Zhang, and Xiaodan Li. 3D genome reconstruction with ShRec3D+ and Hi-C data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 15(2):460–467, 2018.

[89] Qingjiao Li, Harianto Tjong, Xiao Li, Ke Gong, Xianghong Jasmine Zhou, Irene Chiolo, and Frank Alber. The three-dimensional genome organization of *Drosophila melanogaster* through data integration. *Genome Biology*, 18:145, 2017.

[90] Wenyuan Li, Ke Gong, Qingjiao Li, Frank Alber, and Xianghong Jasmine Zhou. Hi-Corrector: a fast, scalable and memory-efficient package for normalizing large-scale Hi-C data. *Bioinformatics*, 31(6):960–962, 2015.

[91] Yun Li, Ming Hu, and Yin Shen. Gene regulation in the 3D genome. *Human Molecular Genetics*, R2:R228–R233, 2018.

[92] Erez Lieberman-Aiden, Nynke L. van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R. Lajoie, Peter J. Sabo, Michael O. Dorschner, Richard Sandstrom, Bradley Bernstein, M A. Bender, Mark Groudine, Andreas Gnirke, John Stamatoyannopoulos, Leonid A. Mirny, Eric S. Lander, and Job Dekker. Comprehensive mapping of long range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, 2009.

[93] Erez Lieberman-Aiden, Nynke L. van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R. Lajoie, Peter J. Sabo, Michael O. Dorschner, Richard Sandstrom, Bradley Bernstein, M. A. Bender, Mark Groudine, Andreas Gnirke, John Stamatoyannopoulos, Leonid A. Mirny, Eric S. Lander, and Job Dekker. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science*, 326:289–293, 2009.

[94] Chang Liu, Ying-Juan Cheng, Jia-Wei Wang, and Detlef Weigel. Prominent topologically associated domains differentiate global chromatin packing in rice from arabidopsis. *Nature Plants*, 3(9):742–748, 2017.

[95] Tong Liu and Zheng Wang. Measuring the three-dimensional structural properties of topologically associating domains. In *Proceedings, IEEE International Conference on Bioinformatics and Biomedicine (Madrid, Spain)*, pages 21–28. IEEE, 2018.

[96] Hongqiang Lyu, Erhu Liu, and Zhifang Wu. Comparison of normalization methods for Hi-C data. *BioTechniques*, 68(2):56–64, 2020.

[97] Kimberly MacKay, Mats Carlsson, and Anthony Kusalik. SonHi-C: a set of non-procedural approaches for predicting 3D genome organization from Hi-C data. Technical report, bioRxiv 392407, 2018.

[98] Kimberly MacKay, Mats Carlsson, and Anthony Kusalik. GeneRHi-C: 3D GENomE Reconstruction from Hi-C data. In *Proceedings, 10th International Conference on Computational Systems-Biology and Bioinformatics (Nice, France)*, number 9, pages 1–9. Association for Computing Machinery, 2019.

[99] Kimberly MacKay and Anthony Kusalik. Computational methods for predicting 3D genomic organization from high-resolution chromosome conformation capture data. *Briefings in Functional Genomics*, page elaa004, 2020.

[100] Kimberly MacKay and Anthony Kusalik. StoHi-C: Using t-Distributed Stochastic Neighbor Embedding (t-SNE) to predict 3D genome organization from Hi-C Data. Technical report, bioRxiv 2020.01.28.923615, 2020.

[101] Kimberly MacKay, Anthony Kusalik, and Christopher H. Eskiw. GrapHi-C: graph-based visualization of Hi-C datasets. *BMC Research Notes*, 11(1):418, 2018.

[102] Kurt Mehlhorn and Guido Schäfer. Implementation of $O(nm \log n)$ weighted matchings in general graphs: the power of data structures. *Journal of Experimental Algorithmics (JEA)*, 7:4, 2002.

[103] Ishita S. Mehta, Manelle Amira, Amanda J. Harvey, and Joanna M. Bridger. Rapid chromosome territory relocation by nuclear motor activity in response to serum removal in primary human fibroblasts. *Genome Biology*, 11(1):R5, 2010.

[104] Ishita S. Mehta, Christopher H. Eskiw, Halime D. Arican, Ian R. Kill, and Joanna M. Bridger. Farnesyltransferase inhibitor treatment restores chromosome territory positions and active chromosome dynamics in Hutchinson-Gilford progeria syndrome cells. *Genome Biology*, 12(8):R74, 2011.

[105] Ishita S. Mehta, Mugdha Kulashreshtha, Sandeep Chakraborty, Ullas Kolthur-Seetharam, and Basuthkar J. Rao. Chromosome territories reposition during DNA damage-repair response. *Genome Biology*, 14(12):R135, 2013.

[106] Dario Meluzzi and Gaurav Arya. Recovering ensembles of chromatin conformations from contact probabilities. *Nucleic Acids Research*, 41(1):63–75, 2012.

[107] Takeshi Mizuguchi, Jemima Barrowman, and Shiv I.S. Grewal. Chromosome domain architecture and dynamic organization of the fission yeast genome. *FEBS Letters*, 589(20 part A):2975–2986, 2015.

[108] Takeshi Mizuguchi, Geoffrey Fudenberg, Sameet Mehta, Jon-Matthew Belton, Nitika Taneja, Hernan Diego Folco, Peter FitzGerald, Job Dekker, Leonid Mirny, Jemima Barrowman, and Shiv I.S. Grewal. Cohesin-dependent globules and heterochromatin shape 3D genome architecture in S. pombe. *Nature*, 516(7531):432–435, 2014.

[109] Monika Molnar and Nancy Kleckner. Examination of interchromosomal interactions in vegetatively growing diploid Schizosaccharomyces pombe cells by Cre/loxP site-specific recombination. *Genetics*, 178(1):99–112, 2008.

[110] Jean-Baptiste Morlot, Julien Mozziconacci, and Annick Lesne. Network concepts for analyzing 3D genome structure from chromosomal contact maps. *EPJ Nonlinear Biomedical Physics*, 4:2, 2016.

[111] Maxwell R. Mumbach, Adam J. Rubin, Ryan A. Flynn, Chao Dai, Paul A. Khavari, William J. Greenleaf, and Howard Y. Chang. HiChIP: efficient and sensitive analysis of protein-directed genome architecture. *Nature Methods*, 13:919–922, 2016.

[112] Takashi Nagano, Yaniv Lubling, Tim J. Stevens, Stefan Schoenfelder, Eitan Yaffe, Wendy Dean, Ernest D. Laue, Amos Tanay, and Peter Fraser. Single-cell Hi-C reveals cell-to-cell variability in chromosome structure. *Nature*, 502:59–64, 2013.

[113] Nicholas Nethercote, Peter J. Stuckey, Ralph Becket, Sebastian Brand, Gregory J. Duck, and Guido Tack. MiniZinc: Towards a standard CP modelling language. In *Proceedings, Principles and Practice of Constraint Programming (Providence, RI, USA)*, LNCS, pages 529–543. Springer Berlin Heidelberg, 2007.

[114] Oluwatosin Oluwadare, Max Highsmith, and Jianlin Cheng. An overview of methods for reconstructing 3-D chromosome and genome structures from Hi-C data. *Biological Procedures Online*, 21:7, 2019.

[115] Argyris Papantonis and Peter R Cook. Transcription factories: genome organization and gene regulation. *Chemical Reviews*, 113(11):8683–8705, 2013.

[116] Jincheol Park and Shili Lin. Statistical inference on three-dimensional structure of genome by truncated Poisson architecture model. *Ordered Data Analysis, Modeling and Health Research Methods*, 149:245–261, 2015.

[117] Jincheol Park and Shili Lin. Impact of data resolution on three-dimensional structure inference methods. *BMC Bioinformatics*, 17:70, 2016.

[118] Jonas Paulsen, Tharvesh Moideen Liyakat Ali, and Philippe Collas. Computational 3d genome modeling using Chrom3D. *Nature Protocols*, 13:1137–1152, 2018.

[119] Jonas Paulsen, Odin Gramstad, and Philippe Collas. Manifold based optimization for single-cell 3D genome reconstruction. *PLOS Computational Biology*, 11(8):e1004396, 2015.

[120] Jonas Paulsen, Monika Sekelja, Anja R. Oldenburg, Alice Barateau, Nolwenn Briand, Erwan Delbarre, Akshay Shah, Anita L. Sørensen, Corinne Vigouroux, Brigitte Buendia, and Philippe Collas. Chrom3D: three-dimensional genome modeling from Hi-C and nuclear lamin-genome contacts. *Genome Biology*, 18:21, 2017.

[121] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[122] Cheng Peng, Liang-Yu Fu, Peng-Fei Dong, Zhi-Luo Deng, Jian-Xin Li, Xiao-Tao Wang, and Hong-Yu Zhang. The sequencing bias relaxed characteristics of Hi-C derived data and implications for chromatin 3D modeling. *Nucleic Acids Research*, 41(19):e183, 2013.

[123] Plotly Technologies Inc. Collaborative data science. Technical report, Plotly Technologies Inc., 2015.

[124] Search Results Web results Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 951.

[125] Lila Rieber and Shaun Mahony. miniMDS: 3D structural inference from high-resolution Hi-C data. *Bioinformatics*, 33(14):i261–i266, 2017.

[126] Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *Handbook of Constraint Programming*. Elsevier, New York, NY, USA, 2006.

[127] Mathieu Rousseau, James Fraser, Maria A Ferraiuolo, Josée Dostie, and Mathieu Blanchette. Three-dimensional modeling of chromatin structure from interaction frequency data using Markov chain Monte Carlo sampling. *BMC Bioinformatics*, 12:414, 2011.

[128] Pelin Sahlén, Ilgar Abdullayev, Daniel Ramsköld, Liudmila Matskova, Nemanja Rilakovic, Britta Lötstedt, Thomas J. Albert, Joakim Lundeberg, and Rickard Sandberg. Genome-wide mapping of promoter-anchored interactions with close to single-enhancer resolution. *Genome Biology*, 16:156, 2015.

[129] Alok J. Saldanha. Java treeview—extensible visualization of microarray data. *Bioinformatics*, 20(17):3246–3248, 2004.

[130] Satish Sati and Giacomo Cavalli. Chromosome conformation capture technologies and their impact in understanding genome function. *Chromosoma*, 126(1):33–44, 2017.

[131] Harry Scherthan, Jürg Bähler, and J. Kohli. Dynamics of chromosome organization and pairing during meiotic prophase in fission yeast. *Journal of Cell Biology*, 127(2):273–285, 1994.

[132] Mark R. Segal and Henrik L. Bengtsson. Reconstruction of 3D genome architecture via a two-stage algorithm. *BMC Bioinformatics*, 16:373, 2015.

[133] Monika Sekelja, Jonas Paulsen, and Philippe Collas. 4D nucleomes in single cells: what can computational modeling reveal about spatial chromatin conformation? *Genome Biology*, 17:54, 2016.

[134] Siddarth Selvaraj, Jesse R. Dixon, Vikas Bansal, and Bing Ren. Whole-genome haplotype reconstruction using proximity-ligation and shotgun sequencing. *Nature Biotechnology*, 31(12):1111–1118, 2013.

[135] François Serra, Davide Baù, Mike Goodstadt, David Castillo, Guillaume Filion, and Marc A. Marti-Renom. Automatic analysis and 3D-modelling of Hi-C data using TADbit reveals structural features of the fly chromatin colors reveals structural features of the fly chromatin colors. *PLOS Computational Biology*, 13(7):e1005665, 2017.

[136] François Serra, Marco Di Stefano, Yannick G. Spill, Yasmina Cuartero, Michael Goodstadt, Davide Baù, and Marc A. Marti-Renom. Restraint-based three-dimensional modeling of genomes and genomic domains. *FEBS Letters*, 589(20):2987–2995, May 2015.

[137] Nicolas Servant, Nelle Varoquaux, Edith Heard, Emmanuel Barillot, and Jean-Philippe Vert. Effective normalization for copy number variation in Hi-C data. *BMC Bioinformatics*, 19:313, 2018.

[138] Nicolas Servant, Nelle Varoquaux, Bryan R. Lajoie, Eric Viara, Chong-Jian Chen, Jean-Philippe Vert, Edith Heard, Job Dekker, and Emmanuel Barillot. HiC-Pro: an optimized and flexible pipeline for Hi-C data processing. *Genome Biology*, 16(259), 2015.

[139] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.

[140] Yoli Shavit and Pietro Lio. CytoHiC: a cytoscape plugin for visual comparison of Hi-C networks. *Bioinformatics*, 29(9):1206–1207, 2013.

[141] Marieke Simonis, Petra Klous, Erik Splinter, Yuri Moshkin, Rob Willemsen, Elzo de Wit, Bas van Steensel, and Wouter de Laat. Nuclear organization of active and inactive chromatin domains uncovered by chromosome conformation capture-on-chip (4C). *Nature Genetics*, 38(11):1348–1354, 2006.

[142] Mariana Sotelo-Silveira, Ricardo A. Chávez Montes, Jose R. Sotelo-Silveira, Nayelli Marsch-Martínez, and Stefan de Folter. Entering the next dimension: Plant genomes in 3D. *Trends in Plant Science*, 23(7):598–612, 2018.

[143] Yannick G. Spill, David Castillo, Enrique Vidal, and Marc A. Marti-Renom. Binless normalization of Hi-C data provides significant interaction and difference detection independent of resolution. *Nature Communications*, 10:1938, 2019.

[144] M. Srinivas and Lalit M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6):17–26, 1994.

[145] John C. Stansfield, Kellen G. Cresswell, Vladimir I. Vladimirov, and Mikhail G. Dozmorov. HiCcompare: an R-package for joint normalization and comparison of Hi-C datasets. *BMC Bioinformatics*, 19(279), 2018.

[146] Tim J. Stevens, David Lando, Srinjan Basu, Liam P. Atkinson, Yang Cao, Steven F. Lee, Martin Leeb, Kai J. Wohlfahrt, Wayne Boucher, Aoife O'Shaughnessy-Kirwan, Julie Cramard, Andre J. Faure, Meryem Ralser, Lluis Morey Enrique Blanco, Miriam Sansó, Matthieu G. S. Palayret, Ben Lehner, Luciano Di Croce, Anton Wutz, Brian Hendrich, Dave Klenerman, and Ernest D. Laue. 3D structures of individual mammalian genomes studied by single-cell Hi-C. *Nature*, 544:59–64, 2017.

[147] Przemyslaw Szalaj, Paul J. Michalski, Przemystaw Wroblewski, Zhonghui Tang, Michal Kadlof, Giovanni Mazzocco, Yijun Ruan, and Dariusz Plewczynski. 3D-GNOME: an integrated web service for structural modeling of the 3D genome. *Nucleic Acids Research*, 44:W288–W293, 2016.

[148] Przemysław Szałaj, Zhonghui Tang, Paul Michalski, Michal J. Pietal, Oscar J. Luo, Michał Sadowski, Xingwang Li, Kamen Radew, Yijun Ruan, , and Dariusz Plewczynski. An integrated 3-dimensional genome modeling engine for data-driven simulation of spatial genome organization. *Genome Research*, 26:1697–1709, 2016.

[149] Harold Szu and Ralph Hartley. Fast simulated annealing. *Physics Letters A*, 122(3-4):157–162, 1987.

[150] Phillippa C. Taberlay, Joanna Achinger-Kawecka, Aaron T.L. Lun, Fabian A. Buske, Kenneth Sabir, Cathryn M. Gould, Elena Zotenko, Saul A. Bert, Katherine A. Giles, Denis C. Bauer, Gordon K. Smyth, Clare Stirzaker, Sean I. O'Donoghue, and Susan J. Clark. Three-dimensional disorganisation of the cancer genome occurs coincident with long range genetic and epigenetic alterations. *Genome Research*, 26(6):719–731, 2016.

[151] Hideki Tanizawa, Osamu Iwasaki, Atsunari Tanaka, Joseph R. Capizzi, Priyankara Wickramasinghe, Mihee Lee, Zhiyan Fu, and Ken ichi Noma. Mapping of long-range associations throughout the fission yeast genome reveals global genome organization linked to transcriptional regulation. *Nucleic Acids Research*, 38(22):8164–8177, 2010.

[152] Hideki Tanizawa, Kyoung-Dong Kim, Osamu Iwasaki, and Ken ichi Noma. Architectural alterations of the fission yeast genome during the cell cycle. *Nature Structural & Molecular Biology*, 24(11):965–976, 2017.

[153] Mariliis Tark-Dame, Roel van Driel, and Dieter W. Heermann. Chromatin folding – from biology to polymer models and back. *Journal of Cell Science*, 124:839–845, 2011.

[154] Harianto Tjong, Wenyuan Li, Reza Kalhor, Chao Dai, Shengli Hao, Ke Gong, Yonggang Zhou, Haochen Li, Xianghong Jasmine Zhou, Mark A. Le Gros, Carolyn A. Larabell, Lin Chen, and Frank Albera. Population-based 3D genome structure analysis reveals driving forces in spatial genome organization. *PNAS*, 113(12):E1663–E1672, 2016.

[155] Tuan Trieu and Jianlin Cheng. Large-scale reconstruction of 3D structures of human chromosomes from chromosomal contact data. *Nucleic Acids Research*, 42(7):e52, 2014.

[156] Tuan Trieu and Jianlin Cheng. 3D genome structure modeling by Lorentzian objective function. *Nucleic Acids Research*, 45(3):1049–1058, 2017.

[157] Marie Trussart, François Serra, Davide Baù, Ivan Junier, Luís Serrano, and Marc A. Marti-Renom. Assessing the limits of restraint-based 3D modeling of genomes and genomic domains. *Nucleic Acids Research*, 43(7):3465–3477, 2015.

[158] S Uzawa and M Yanagida. Visualization of centromeric and nucleolar DNA in fission yeast by fluorescence in situ hybridization. *Journal of Cell Science*, 101(Pt 2):267–275, 1992.

[159] Laurens van der Maaten. Learning a parametric embedding by preserving local structure. In *Proceedings, Twelfth International Conference on Artificial Intelligence & Statistics (Clearwater Beach, Florida, USA)*, pages 384–391. PMLR, 2009.

[160] Laurens van der Maaten. Accelerating t-SNE using tree-based algorithms. *Journal of Machine Learning Research*, 15(Oct):3321–3245, 2014.

[161] Laurens van der Maaten and Geoffrey Hinton. Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[162] Laurens van der Maaten and Geoffrey Hinton. Visualizing non-metric similarities in multiple maps. *Machine Learning*, 87(1):33–55, 2012.

[163] Nelle Varoquaux, Ferhat Ay, William Stafford Noble, and Jean-Philippe Vert. A statistical approach for inferring the 3D structure of the genome. *Bioinformatics*, 30(12):i26–i33, 2014.

[164] Nelle Varoquaux and Nicolas Servant. Iced: fast and memory efficient normalization of contact maps. *Journal of Open Source Software*, 4(36):1286, 2019.

[165] Siyu Wang, Jinbo Xu, and Jianyang Zeng. Inferential modeling of 3D chromatin structure. *Nucleic Acids Research*, 43(8):e54, 2015.

[166] Stephen Warshall. A theorem on boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.

[167] Steven Wingett, Philip Ewels, Mayra Furlan-Magaril, Takashi Nagano, Stefan Schoenfelder, Peter Fraser, and Simon Andrews. HiCUP: pipeline for mapping and processing Hi-C data. *F1000Research*, 4:1310, 2015.

[168] Laurence A Wolsey. *Integer Programming.* Wiley, 1998.

[169] Hyejung Won, Luis de la Torre-Ubieta, Jason L. Stein, Neelroop N. Parikshak, Jerry Huang, Carli K. Opland, Michael J. Gandal, Gavin J. Sutton, Farhad Hormozdiari, Daning Lu, Changhoon Lee, Eleazar Eskin, Irina Voineagu, Jason Ernst, and Daniel H. Geschwind. Chromosome conformation elucidates regulatory relationships in developing human brain. *Nature*, 538(7626):523–527, 2016.

[170] Hugo Würtele and Pierre Chartrand. Genome-wide scanning of HoxB1-associated loci in mouse ES cells using an open-ended Chromosome Conformation Capture methodology. *Chromosome Research*, 14(5):477–495, 2006.

[171] Eitan Yaffe and Amos Tanay. Probabilistic modeling of Hi-C contact maps eliminates systematic biases to characterize global chromosomal architecture. *Nature Genetics*, 43:1059–1065, 2011.

[172] Ei-Wen Yang and Tao Jiang. GDNorm: an improved Poisson regression model for reducing biases in Hi-C data. In *Proceedings, International Workshop on Algorithms in Bioinformatics (Wroclaw, Poland)*, pages 263–280. Springer Berlin Heidelberg, 2014.

[173] Tynia Yang, Jinze Liu, Leonard Mcmillan, and Wei Wang. A fast approximation to multidimensional scaling. In *Proceedings, ECCV Workshop on Computation Intensive Methods for Computer Vision (Graz, Austria)*. IEEE, 2006.

[174] Galip G. Yardımcı and William S. Noble. Software tools for visualizing Hi-C data. *Genome Biology*, 18(1):26, 2017.

[175] Yinxiu Zhan, Luca Mariani, Iros Barozzi, Edda G. Schulz, Nils Blüthgen, Michael Stadler, Guido Tiana, and Luca Giorgetti. Reciprocal insulation analysis of hi-c data shows that TADs represent a functionally but not structurally privileged scale in the hierarchical folding of chromosomes. *Genome Research*, 27(3):479–490, 2017.

[176] Rongrong Zhang, Ming Hu, Yu Zhu, Zhaohui Qin, Ke Deng, and Jun S. Liu. Inferring spatial organization of individual topologically associated domains via piecewise helical model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(2):647–656, 2020.

[177] Yanlin Zhang, Weiwei Liu, Yu Lin, Yen Kaow Ng, and Shuaicheng Li. Large-scale 3D chromatin reconstruction from chromosomal contacts. *BMC Genomics*, 20(Suppl 2):186, 2019.

[178] Zhizhuo Zhang, Guoliang Li, Kim-Chuan Toh, and Wing-Kin Sung. 3D chromosome modeling with semi-definite programming and Hi-C data. *Journal of Computational Biology*, 20(11):831–846, 2013.

[179] Zhihu Zhao, Gholamreza Tavoosidana, Mikael Sjölinder, Anita Göndör, Piero Mariano, Sha Wang, Chandrasekhar Kanduri, Magda Lezcano, Kuljeet S. Sandhu, Umashankar Singh, Vinod Pant, Vijay Tiwari, Sreenivasulu Kurukuti, and Rolf Ohlsson. Circular chromosome conformation capture (4C) uncovers extensive networks of epigenetically regulated intra- and interchromosomal interactions. *Nature Genetics*, 38:1341–1347, 2006.

[180] Guangxiang Zhu, Wenxuan Deng, Hailin Hu, Rui Ma, Sai Zhang, Jinglin Yang, Jian Peng, Tommy Kaplan, and Jianyang Zeng. Reconstructing spatial organizations of chromosomes through manifold learning. *Nucleic Acids Research*, 46(8):e50, 2018.

[181] Aleksey V. Zimin, Daniela Puiu, Richard Hall, Sarah Kingan, Bernardo J. Clavijo, and Steven L. Salzberg. The first near-complete assembly of the hexaploid bread wheat genome, triticum aestivum. *GigaScience*, 6(11):1–7, 2017.

[182] Chenchen Zou, Yuping Zhang, and Zhengqing Ouyang. HSA: integrating multi-track Hi-C data for genome-scale reconstruction of 3D chromatin structure. *Genome Biology*, 14:40, 2016.

Appendix A

Permission to Reprint Manuscripts

A.1 GrapHi-C: Graph-Based Visualization of Hi-C Datasets

Citation [101]: **K MacKay**, A Kusalik, CH Eskiw. GrapHi-C: graph-based visualization of Hi-C datasets. *BMC Research Notes.* (2018) 11(1): 418. doi: `https://doi.org/10.1186/s13104-018-3507-2`. This article is available under a Creative Commons CC-BY license.

A.2 Computational Methods for Predicting 3D Genomic Organization from High-Resolution Chromosome Conformation Capture Data

Citation [99]: **K MacKay** and A Kusalik. Computational methods for predicting 3D genomic organization from high-resolution chromosome conformation capture data. *Briefings in Functional Genomics.* (2020) elaa004. Reprint licenses: 4824380767094 (text), 4824381317794 (figures and tables), 4824380961544 (abstract).

A.3 GeneRHi-C: 3D Genome Reconstruction from Hi-C Data

Citation [98]: **K MacKay**, M Carlsson and A Kusalik. GeneRHi-C: 3D GENomE Reconstruction from Hi-C data. In proceedings of the *10th International Conference on Computational Systems-Biology and Bioinformatics.* (2019) 9:1-9. Publisher: Association for Computing Machinery (ACM). doi: `https://doi.org/10.1145/3365953.3365962`. ACM allows authors to include partial or complete papers of their own in a dissertation (`https://authors.acm.org/author-services/author-rights`)

A.4 StoHi-C: Using t-Distributed Stochastic Neighbor Embedding (t-SNE) to Predict 3D Genome Organization from Hi-C Data

Citation [100]: **K MacKay** and A Kusalik. StoHi-C: Using t-Distributed Stochastic Neighbor Embedding (t-SNE) to predict 3D genome organization from Hi-C Data. This version of the manuscript is available on BioRxiv (`https://doi.org/10.1101/2020.01.28.923615`) and is available under a Creative Commons CC-BY-NC license.

Appendix B

Modifications to Published Manuscripts

For all manuscripts: Minor formatting modifications have be made so the manuscripts align with the University of Saskatchewan's standards for thesis documents. To be consistent across chapters, existing information regarding acknowledgements, author contributions, funding and archived software has been moved to a section titled "Supplemental Information".

Additionally the following modifications have been made:

- Chapter 2 (GrapHi-C)
  - The section header "Main Text" was removed for consistency.
  - The section titled "Declarations" was renamed to "Supplemental Information" and subsections were re-ordered within this section to maintain consistency across chapters.
  - The section titled "Additional Files" was separated into two new subsections under "Supplemental Information" named "Archived Software" and "Supplemental Figures".
- Chapter 3
  - The section titled "Supplementary Information" was renamed to "Supplemental Information" and now appears before Appendix A.
  - Sections titled "Acknowledgements", "Author Information" and "Funding" were moved to corresponding subsections in Supplemental Information.
  - Section 3.13.3 was added to provide a brief description of the author contributions.
  - Appendix A was renamed to Section 3.14.
  - Alternative reprint licenses are listed in the footnotes and Section 3.14 for use of images in a thesis (as opposed to a journal article).
  - Footnotes were converted to endnotes and are listed in a separate section.
- Chapter 4 (GeneRHi-C)
  - Figure 4.5 was updated to include a FISH image of fission yeast genome organization.
  - Subsections Under Section 4.10 were reordered to maintain consistency with other manuscripts.
  - Archived versions of the programs used to generate the results of GeneRHi-C have been provided as supplementary material (Section 4.10.5). These were not included in the published version of the manuscript as per the submission guidelines for the conference.

  - Footnotes were converted to endnotes and are listed in a separate section.
  - The following sentence was added to Section 4.4.1 and 4.6 “For the purpose of this thesis, this information was also added to Section 4.10.4.”
  - Section 4.10.4 was added to describe the **CP** and **GM** mathematical models that were also developed.

- Chapter 5 (StoHi-C)
  - Additional background on the t-SNE method was added to the preamble for this chapter.
  - Section 5.6.1 was added to provide a brief description of the author contributions.
  - Funding information was moved from “Acknowledgments” to a new section titled “Funding Information” (Section 5.6.2) for consistency.
  - Reference labels were changed from author/date to numeric.