Diachromatic Documentation

Release 0.5.2

Peter Hansen, Peter Robinson

Jun 29, 2019

Contents:

CHAPTER 1

Program setup

Diachromatic requires Java 8 or higher to run. Diachromatic can be obtained from the Diachromatic GitHub page. We recommend to download the `Diachromatic.jar` file of the latest release on the release page of the project.

You can run the program using this command:

```
$ java -jar Diachromatic.jar
```

You should see a help message in the shell.

1.1 Building Diachromatic from source

To build the application on your own, clone the repository and create the Java app with maven:

```
$ git clone https://github.com/TheJacksonLaboratory/diachromatic.git
$ cd diachromatic
$ mvn package
```

To test whether the build process was successful, enter the following command:

```
$ java -jar target/Diachromatic.jar
```

1.2 Preparation for bowtie2

The mapping step of the diachromatic pipeline relies on bowtie2. If needed, install bowtie2 on your system. For instance, on Debian linux systems bowtie2 can be installed with the following command.

```
$ sudo apt-get install bowtie2
```

The prebuilt `bowtie2` indices for human hg19 (3.5 GB) and other genome builds can be downloaded from the bowtie2 website. After downloading the correct archived file to your computer, unpack it with:

```
$ unzip hg19.zip
```

In the following pages, we will call the path to the directory where the index was unpacked **/path/to/bowtie2index/**. Substitute this with the actual path on your computer.

CHAPTER 2

Tutorial

This tutorial shows how to use Diachromatic for processing and quality control of Capture Hi-C reads. Before proceeding with the tutorial, please follow the program setup instructions to build Diachromatic and get bowtie2 as well as the hg19 prebuilt index.

2.1 Test dataset

To get the data, visit this ftp server or use:

```
wget ftp://ftp.jax.org/robinp/Diachromatic/test_dataset/test_1.fastq
wget ftp://ftp.jax.org/robinp/Diachromatic/test_dataset/test_2.fastq
wget ftp://ftp.jax.org/robinp/Diachromatic/test_dataset/hg19_HinDIII_DigestedGenome.
↪txt.gz
```

Then decompress the digest file:

```
gunzip hg19_HinDIII_DigestedGenome.txt.gz
```

2.2 Truncation

The first step of processing raw FASTQ files with Diachromatic is to recognize and truncate reads with filled-in ligation juctions, which indicate reads that include the junction of the chimeric CHC fragment. This is performed with the truncate subcommand:

```
$ java -jar Diachromatic.jar truncate \
    -q test_1.fastq \
    -r test_2.fastq \
    -e HinDIII \
    -x prefix \
    -o outdir
```

2.3 Mapping

The second step of the pipeline is to map the truncated read pairs to the target genome. You also need a file that shows the locations of restriction digests across the genome. This file is included in the test dataset. You can use GOPHER to create probes and the digest file. Diachromatic uses bowtie2 to perform the mapping, and then creates a BAM file containing the valid read pairs.

Use the following command to run the alignment step:

```
$ java -jar Diachromatic.jar align \
    -b /usr/bin/bowtie2 \
    -i /path/to/bowtie2index/hg19 \
    -q prefix.truncated_R1.fastq.gz \
    -r prefix.truncated_R2.fastq.gz \
    -d hg19_HinDIII_DigestedGenome.txt \
    -x prefix \
    -o outdir
```

2.4 Counting

Use the following command to run the counting step:

```
$ java -jar Diachromatic.jar count \
    -v prefix.valid_pairs.aligned.bam \
    -d hg19_HinDIII_DigestedGenome.txt \
    -x prefix \
    -o outdir
```

2.5 Summarize

To run the summarize command with the truncate data, use the following command.

```
$ java -jar Diachromatic.jar summarize \
    -t outdir/prefix.truncation.stats.txt \
    -a outdir/prefix.align.stats.txt \
    -c outdir/prefix.count.stats.txt \
    -x prefix \
    -o outdir
```

This will generate an HTML file called `outdir/prefix.summary.stats.html`.

The summary results file for the test dataset can also be downloaded from the ftp server or use:

```
wget ftp://ftp.jax.org/robinp/Diachromatic/test_dataset/test_dataset.summary.stats.
↪html
```

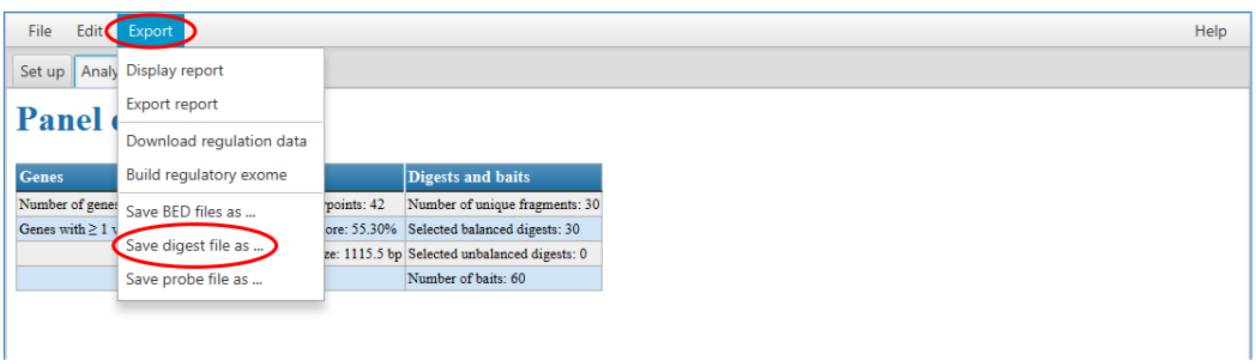
CHAPTER 3

Creating an *in silico* restriction digest map for Diachromatic using GOPHER

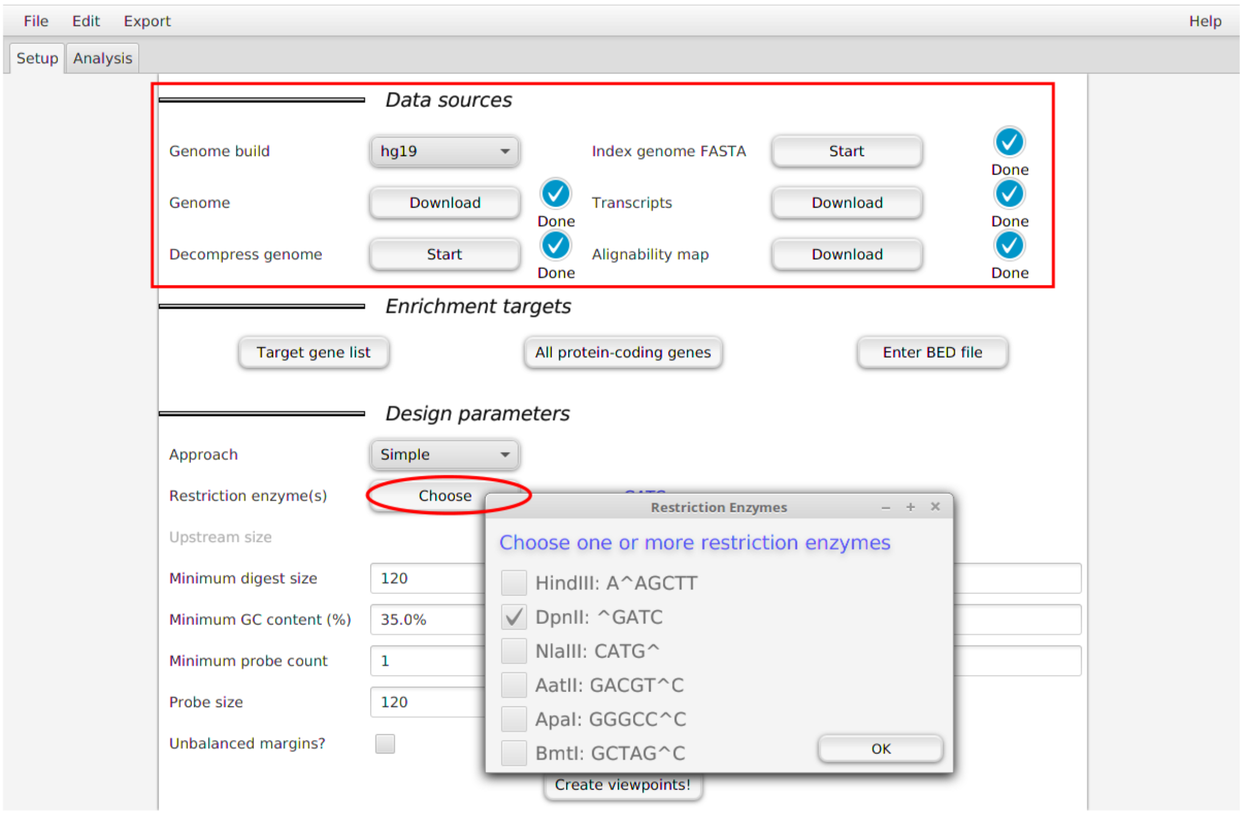
The Capture Hi-C (CHC) protocol involves the restriction digestion of a sample and the downstream analysis assigns reads to pairs of restriction digests. Therefore, the `align` subcommand of Diachromatic requires a list of all restriction digests that result from the *in silico* digestions of a given genome with the chosen enzyme or enzymes. Such lists can be generated using the GOPHER software. The TSV formatted file exported from GOPHER can be passed to Diachromatic using the `-d` or `--digest-file` option.

3.1 Performing the in silico digest using GOPHER

If the GOPHER software was used for the design of the given capture Hi-C experiment, you can just open the corresponding GOPHER project and export the required file via the export menu. In such cases the exported file also includes the information about enriched and non-enriched digests, i.e. if there are enrichment probes associated with given digests.



If you did not design CHC probes using GOPHER, you can still use diachromatic, but will need to create file with the same format. It is easy to do so with the GOPHER software. First, setup a new project, and then specify the parameters in the *Data sources* section and the restriction enzyme in the *Design parameters* section. If you prepare the digest map in this way, all digests will be marked as inactive (i.e., not enriched with capture probes). In order to use some features of Diachromatic, you will need to change the digest file to indicate which digests where chosen for enrichment (for instance, with a Python script).



3.2 Format of the digest file

The first line of this file contains the column names, and all other lines correspomd to one restriction digest. Each line consists of 14 fields that are described in the table .

Column	Name	Example	Description
1	Chromosome	chr1	Name of the reference sequence.
2	Digest_Start_Position	18376	1-based start position of the restriction digest.
3	Digest_End_Position	18392	1-based end position of the restriction digest.
4	Digest_Number	42	Consecutive digest number.
5	5'_Restriction_Site	DpnII	Name of the enzyme responsible for the cut at the 5' end of the digest.
6	3'_Restriction_Site	DpnII	Name of the enzyme responsible for the cut at the 3' end of the digest. May be different from field 5 if more than one enzyme is used.
7	Length	1245	Length of the digest.
8	5'_GC_Content	0.500	GC content of the upstream margin (GOPHER's default margin size is 250 bp).
9	3'_GC_Content	0.500	GC content of the downstream margin.
10	5'_Repeat_Content	0.138	Repeat content of the upstream margin.
11	3'_Repeat_Content	0.126	Repeat content of the downstream margin.
12	Enrichment status	T (or F)	Enrichment status of the digest. A digest is considered enriched, if it has at least one probe.
13	5'_Probes	2	Number of probes for the upstream margin.
14	3'_Repeat_Content	0	Number of probes for the downstream margin.

CHAPTER 4

Truncation of chimeric reads

Valid Hi-C read pairs originate from chimeric fragments with DNA from two different loci linked by the ligation junction. Diachromatic searches read sequences in 5'-3' direction and truncates chimeric reads at the location of the ligation site, thereby removing the following sequence.

For Capture Hi-C, the sticky ends are filled in with biotinylated nucleotides, and the resulting blunt ends are ligated. The corresponding ligation junctions can then be observed as two consecutive copies of the overhang sequence at restriction enzyme cutting sites. For Capture-C, no fill in of the overhangs is performed, and the ligation junctions occur as plain restriction sites.

```
HindIII                    DpnII

Filled blunt ends          Filled blunt ends
5'...AAGCTAGCTT...3'       5'...GATCGATC...3'
3'...TTCGATCGAA...5'       3'...CTAGCTAG...5'

Sticky ends                Sticky ends
5'...AAGCTT...3'           5'...GATC...3'
3'...TTCGAA...5'           3'...CTAG...5'
```

Use the `--sticky-ends` option if no fill in was performed.

4.1 Running the *truncate* subcommand

Use the following command to run the truncation step:

```
$ java -jar Diachromatic.jar truncate \
    -q test_1.fastq \
    -r test_2.fastq \
    -e HinDIII \
    -x prefix \
    -o outdir
```

Available arguments:

Short option	Long option	Example	Required	Description	Default
-q	–fastq-r1	forward.fq.gz	yes	Path to the forward FASTQ file.	–
-r	–fastq-r2	reverse.fq.gz	yes	Path to the reverse FASTQ file.	–
-e	–enzyme	HindIII	yes	Symbol of the restriction enzyme.	null
-s	–sticky-ends	false	no	True, if no fill-in of sticky ends was performed.	false
-o	–out-directory	cd4v2	yes	Directory containing the output of the truncate command.	results
-x	–out-prefix	stim_rep1	yes	Prefix for all generated files in output directory.	prefix

4.2 Output files

The default names of the truncated and gzipped FASTQ files are:

- `prefix.truncated_R1.fastq.gz`
- `prefix.truncated_R2.fastq.gz`

In addition, a file is produced that contains summary statistics about the truncation step.

- `prefix.truncation.stats.txt`

CHAPTER 5

Mapping and categorization of Hi-C reads

The two reads of a valid Hi-C read pair come from two different interacting genomic regions that can be separated by a large number of nucleotides on the same chromosome (cis) or even be located on different chromosomes (trans). The truncated forward (R1) and reverse (R2) reads have to be mapped independently.

5.1 Independent mapping of forward and reverse paired-end reads

Diachromatic separately executes `bowtie2` with the `--very-sensitive` option for the truncated R1 and R2 reads. Read pairs for which at least one read cannot be mapped uniquely are discarded. Diachromatic provides two levels of stringency for the definition of multi-mapped reads:

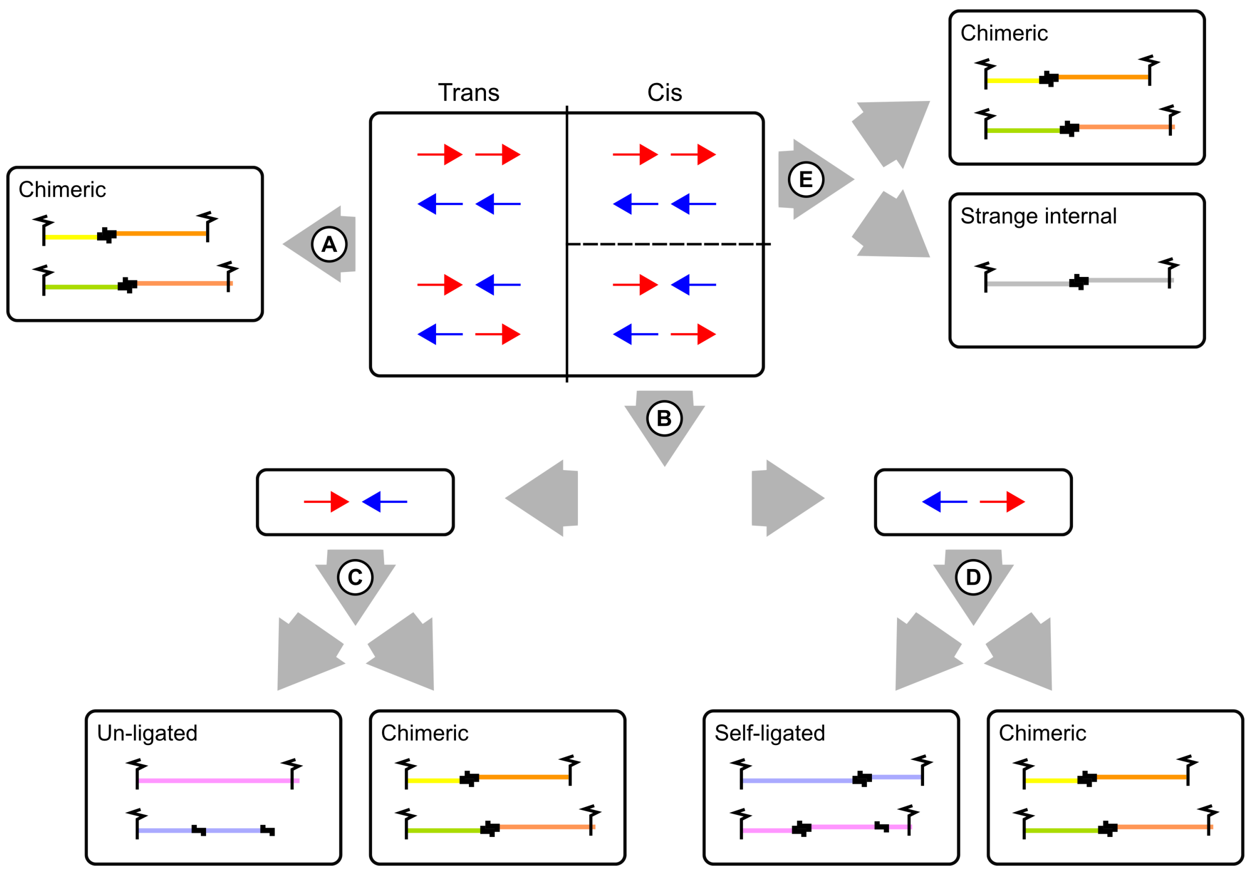
1. **Very stringent mapping:** There is no second best alignment for the given read. In this case the line in the SAM record produced by `bowtie2` contains no `XS` tag. Use Diachromatic's `--bowtie-stringent-unique` or `-bsu` option in order to use this level of stringency.
2. **Less stringent mapping:** There can be a second best alignment, but the score of the alignment (MAPQ) must be at least 30 and the difference of the mapping scores between the best and second best alignment must be at least 10 (following the recommendation of HiCUP). Diachromatic uses this option by default.

5.2 Pairing of properly mapped read pairs

The independently mapped reads are written to two temporary SAM files, whereby the order of read records in the truncated FASTQ files is retained by using bowtie2's option `--reorder`. In the next step, Diachromatic iterates simultaneously over the two SAM files. Read pairs for which both reads can be mapped uniquely are paired, i.e. the two SAM records for single-end reads are combined into one paired-end record with appropriate SAM flags reflecting the relative orientation of the reads.

5.2.1 Categorization of read pairs

Diachromatic distinguishes several read pair categories: (A) Trans reads by definition are chimeric fragments and may represent valid biological interactions or random cross-ligation events. (B) Pairs mapping to different strands of the same chromosome may originate from un-ligated or self-ligated digests. (C) Inward pointing pairs that map to the same digest must have originated from un-ligated fragments. Size thresholds are applied to the remaining fragments to categorize them as valid or artefactual. (D) Outward pointing read pairs that map the same digest must have originated from self-ligated digests. Size thresholds are applied to the remaining fragments to categorize them as valid or artefactual. (E) Read pairs mapping to the same strand can only be chimeric. However, we observe very small proportions of read pairs that are mapped to the same strand and digest. Such read pairs are classified as strange internal.



5.3 Running the *align* subcommand

Use the following command to run the alignment step:

```
$ java -jar target/Diachromatic.jar align \
    -b /usr/bin/bowtie2 \
    -i /path/to/bowtie2index/hg19 \
    -q prefix.truncated_R1.fastq.gz \
    -r prefix.truncated_R2.fastq.gz \
    -d hg19_HinDIII_DigestedGenome.txt \
    -x prefix \
    -o outdir
```

The table lists all possible arguments:

Short option	Long option	Example	Required	Description	Default
-q	–fastq-r1	prefix.truncated_R1.fq.gz	yes	Path to the truncated forward FASTQ file.	–
-r	–fastq-r2	prefix.truncated_R2.fq.gz	yes	Path to the truncated forward FASTQ file.	–
-b	–bowtie2	/tools/bowtie2-2.3.4.1-linux-x86_64/bowtie2	yes	Path to bowtie2 executable.	–
-i	–bowtie2-index	/data/indices/bowtie2/hg38/hg38	yes	Path to bowtie2 index of the corresponding genome.	–
-d	–digest-file	/data/GOPHER/hg38_DpnII_DigestedGenome.txt	yes	Path to the digest file produced with GOPHER.	–
-o	–out-directory	cd4v2	yes	Directory containing the output of the align subcommand.	results
-x	–out-prefix	stim_rep1	yes	Prefix for all generated files in output directory.	prefix
-p	–thread-num	15	no	Number of threads used by bowtie2.	1
-j	–output-rejected	–	no	If set, a BAM file containing the reject read pairs will be created.	false
-l	–lower-frag-size-limit	50	no	Lower threshold for the size of sheared fragments.	50
-u	–upper-frag-size-limit	1000	no	Upper threshold for the size of sheared fragments.	1000
-s	–self-ligtion-threshold	3000	no	Upper threshold for the size of self-ligating fragments.	3000
-k	–keep-sam	–	no	Do not delete temporary SAM files.	false

5.4 Output files

The default name of the BAM file containing all unique valid pairs that can be used for downstream analysis is:

- `prefix.valid_pairs.aligned.bam`

If `--output-rejected` is set, Diachromatic will output a second BAM file cointaing all rejected pairs:

- `prefix.rejected_pairs.aligned.bam`

Diachromatic uses optional fields of the SAM records to indicate the read pair category:

- Un-ligated due to size (Tag: `UL`)
- Un-ligated due to same digest (Tag: `ULSI`)
- Self-ligated due to size (Tag: `SL`)
- Self-ligated due to same digest (Tag: `SLSI`)
- Too short chimeric (Tag: `TS`)
- Too long chimeric (Tag: `TL`)
- Valid pair (Tag: `VP`)

In addition, a file `prefix.align.stats.txt` is produced that contains summary statistics about the alignment step.

Finally, an R script `prefix.frag.sizes.counts.script.R` is generated that contains fragment size counts and can be used to generate a plot as shown above. In order to produce a PDF file, execute the script as follows:

```
$ Rscript prefix.frag.sizes.counts.script.R
```

Or source the script from the R environment:

```
> source("prefix.frag.sizes.counts.script.R")
```

CHAPTER 6

Counting of valid read pairs between pairs of restriction fragments

Mapped Hi-C read pairs are typically transformed into contact matrices, whereby the pairs are counted between windows of fixed size, typically 5 kbp (Forcato et al., 2017) provide a review of the methodology). Diachromatic was developed for *capture Hi-C*, which achieves a much higher resolution than Hi-C. Therefore, for Diachromatic the read counts are determined for each restriction digest.

6.1 Required input files

6.1.1 GOPHER digest file

Due to the fact that the counts are determined on the restriction fragment level, the *digest file* needs to be passed to `Diachromatic count`. If the captured viewpoints were designed with GOPHER, this file also includes information about active and inactive restriction fragments.

6.1.2 BAM file with unique valid pairs

The second required input file contains the unique valid mapped read pairs in BAM format. If this file was generated using Diachromatic with the align subcommand, nothing has to be done or taken care of. If the BAM file was produced in a different way, make sure that the two reads of any given pair occur consecutively. Furthermore, make sure that duplicates were previously removed.

6.2 Running the *count* subcommand

Use the following command to run the counting step:

```
$ java -jar Diachromatic.jar count \
    -v prefix.valid_pairs.aligned.bam \
    -d hg19_HinDIII_DigestedGenome.txt\
```

(continues on next page)

(continued from previous page)

```
    -x prefix \
    -o outdir
```

Short option	Long option	Example	Required	Description	Default
-v	–valid-pairs-bam	prefix.valid_pairs.aligned.bam	yes	Path to BAM file containing unique valid pairs.	–
-d	–digest-file	/data/GOPHER/hg38_DpnII_DigestedGenome.txt	yes	Path to the digest file produced with GOPHER.	–
-o	–out-directory	cd4v2	yes	Directory containing the output of the align subcommand.	results
-x	–out-prefix	stim_rep1	yes	Prefix for all generated files in output directory.	prefix

6.3 Output files

The default name of the output file with statistics is:

- `prefix.count.stats.txt`

6.3.1 Interaction counts

The interactions are written to a tab separated text file that has the following name by default:

- `prefix.interaction.counts.table.tsv`

The structure of this file is similar to that of (iBED) files. Each line stands for one pair of interacting fragments. Consider the following example:

```
chr7    42304777        42314850        A       chr7    152941166       152943990       I       14
chr7    42304777        42314850        A       chr7    38624777        38625305        I       11
```

The first three columns contain the coordinates of a restriction fragment on chromosome 7. The `A` in column 4 indicates that this fragment is defined to be active, i.e. it is part of a viewpoint that was enriched using capture technology. The information about active states of fragments originates from the GOPHER digest file passed to Diachromatic using the `-d` option.

In addition, interactions are written to a simple pairwise interaction file format for long-range interactions established by WashU:

```
chr13:84250549-84256429    chr13:105017710-105020949       1
chr3:74550953-74553110     chr3:83489595-83490326          1
```

Trans and short-range (<10,000) interactions are discarded.

6.3.2 Read counts at interacting fragments

Another file that is created contains the counts of reads at interacting fragments. By default the name of this file is:

- `prefix.interacting.fragments.counts.table.tsv`

The structure is again similar to that of BED files. Consider the following example:

```
chr7    42304777       42314850        A       25
chr7    152941166      152943990       I       14
chr7    38624777       38625305        I       11
```

The first three columns contain the coordinates of interacting restriction fragments. This is again followed by either an A or I in column 4, whereby A means active and I inactive. The fifth column contains the read counts aggregated from all interactions that end in the corresponding fragment. For better understanding, compare these counts to the two interactions given above.

CHAPTER 7

Summarize results

The summarize subcommand outputs an HTML file with a summary of the analysis.

7.1 Running the *summarize* subcommand

To run the summarize subcommand with the truncate data, run the following command.

```
$ java -jar Diachromatic.jar summarize \
    -t outdir/prefix.truncation.stats.txt \
    -a outdir/prefix.align.stats.txt \
    -c outdir/prefix.count.stats.txt \
    -x prefix \
    -o outdir
```

Available arguments:

Short option	Long option	Example	Required	Description	Default
-o	–out-dir	outdir	yes	Directory for output of the summarize subcommand.	results
-x	–out-prefix	prefix	yes	Prefix for generated file in output directory.	prefix
-t	–truncate	prefix.truncation.stats.txt	yes	Path to truncate statistics file.	null
-a	–align	prefix.align.stats.txt	yes	Path to align statistics file.	null
-c	–count	prefix.count.stats.txt	yes	Path to count statistics file.	null

7.2 Output files

This will generate an HTML file called `outdir/prefix.summary.stats.html`.

CHAPTER 8

Differential Analysis of Chromatin Interactions by Capture (Diachromatic)

Diachromatic is a Java application that implements a capture Hi-C preprocessing pipeline followed by analysis of differential chromatin interactions ("loopings"). Diachromatic is designed to work with the capture probes as designed by GOPHER.

CHAPTER 9

Generator Of Probes for capture Hi-C Expriments at high Resolution (GOPHER)

GOPHER is a Java application designed to help design capture probes for capture Hi-C and related protocols. Capture Hi-C (CHC) is based on the Hi-C protocol but uses capture baits (similar to whole-exome sequencing) to enrich a set of viewpoints. Commonly, the viewpoints represent proximal promoter regions (surrounding the transcription start site [TSS]) of genes of interest or of all protein-coding genes. CHC detects interactions between viewpoint regions and distal enhancers (or other genomic regions) and has been most commonly performed with the 4-cutter DpnII or with the 6-cutter HindIII.