

---

# HiCExplorer Documentation

*Release 2.0*

**Fidel Ramírez**

**Mar 12, 2018**



---

## Contents

---

<b>1</b>	<b>Set of programs to process, normalize, analyze and visualize Hi-C data</b>	<b>1</b>
<b>2</b>	<b>The following is the list of tools available in HiCExplorer</b>	<b>3</b>
<b>3</b>	<b>Getting Help</b>	<b>5</b>
<b>4</b>	<b>Contents:</b>	<b>7</b>
4.1	Installation . . . . .	7
4.2	HiCExplorer tools . . . . .	9
4.3	Example usage . . . . .	13
4.4	News and Developments . . . . .	30
<b>5</b>	<b>Citation</b>	<b>35</b>



---

Set of programs to process, normalize, analyze and visualize Hi-C data

---

HiCexplorer addresses the common tasks of Hi-C analysis from processing to visualization.







## CHAPTER 2

The following is the list of tools available in HiCExplorer

tool	description
findRestSite	Identifies the genomic locations of restriction sites
hicBuildMatrix	Creates a Hi-C matrix using the aligned BAM files of the Hi-C sequencing reads
hicQC	Plots QC measures from the output of hicBuildMatrix
hicCorrectMatrix	Uses iterative correction to remove biases from a Hi-C matrix
hicFindEnrichedContacts	Identifies enriched Hi-C contacts
hicCorrelate	Computes and visualises the correlation of Hi-C matrices
hicFindTADs	Identifies Topologically Associating Domains (TADs)
hicPCA	Computes for A / B compartments the eigenvectors
hicTransform	Computes a obs_exp matrix like Lieberman-Aiden (2009), a pearson correlation matrix and or a covariance matrix. These matrices can be used for plotting.
hicMergeMatrixBins	Merges consecutive bins on a Hi-C matrix to reduce resolution
hicMergeTADbins	Uses a BED file of domains or TAD boundaries to merge the bin counts of a Hi-C matrix.
hicPlotDistVsCounts	Plot the decay in interaction frequency with distance
hicPlotMatrix	Plots a Hi-C matrix as a heatmap
hicPlotTADs	Plots TADs as a track that can be combined with other tracks (genes, signal, interactions)
hicPlotViewpoint	A plot with the interactions around a reference point or region.
hicAggreagteContacts	A tool that allows plotting of aggregated Hi-C sub-matrices of a specified list of positions.
hicSumMatrices	Adds Hi-C matrices of the same size
hicPlotDistVsCounts	Plots distance vs. Hi-C counts of corrected data
hicExport	Export matrix to text formats
hicInfo	Shows information about a Hi-C matrix file (no. of bins, bin length, sum, max, min, etc)
hicCompareMatrices	Computes difference or ratio between two matrices
hicLog2Ratio	Computes the log <sub>2</sub> ratio between two matrices.



## CHAPTER 3

---

### Getting Help

---

- For general questions, please use Biostars with Tag *hicexplorer* : [Biostars](#)
- For specific questions and feature requests, use the [deepTools mailing list](#)
- For suggesting changes/enhancements and to report bugs, please create an issue on our [GitHub repository](#)



---

## Contents:

---

## Installation

- *Requirements*
- *Command line installation using conda*
- *Command line installation using pip*
- *Command line installation without pip*
- *Galaxy installation*
  - *Installation with Docker*

## Requirements

- Python 2.7 or 3.6
- numpy >= 1.12.1
- scipy >= 0.19.0
- matplotlib == 2.1.1
- pysam >= 0.11.2
- intervaltree >= 2.1.0
- biopython >= 1.68
- pytables >= 3.3.0
- pyBigWig >=0.3.4
- future >= 0.16.0

- cooler >= 0.7.6
- six >= 1.10.0
- jinja2 >= 2.9.6
- pandas >= 0.20.2

### Command line installation using conda

The fastest way to obtain **Python 3.6 together with numpy and scipy** is via the [Anaconda Scientific Python Distribution](#). Just download the version that's suitable for your operating system and follow the directions for its installation. All of the requirements for HiCEXplorer can be installed in Anaconda with:

```
$ conda install hicexplorer -c bioconda -c conda-forge
```

### Command line installation using pip

Install HiCEXplorer using the following command:

```
$ pip install hicexplorer
```

All python requirements should be automatically installed.

If you need to specify a specific path for the installation of the tools, make use of *pip install*'s numerous options:

```
$ pip install --install-option="--prefix=/MyPath/Tools/hicexplorer" git+https://  
→github.com/deeptools/HiCEXplorer.git
```

### Command line installation without pip

You are highly recommended to use *pip* rather than these more complicated steps.

1. Install the requirements listed above in the “requirements” section. This is done automatically by *pip*.
2. Download source code

```
$ git clone https://github.com/deeptools/HiCEXplorer.git
```

or if you want a particular release, choose one from <https://github.com/deeptools/HiCEXplorer/releases>:

```
$ wget https://github.com/deeptools/HiCEXplorer/archive/1.5.12.tar.gz  
$ tar -xzf
```

3. To install the source code (if you don't have root permission, you can set a specific folder using the *--prefix* option)

```
$ python setup.py install --prefix /User/Tools/hicexplorer
```

### Galaxy installation

HiCEXplorer can be easily integrated into a local [Galaxy](#).

## Installation with Docker

The HiCEXplorer Galaxy instance is also available as a docker container, for those wishing to use the Galaxy framework but who also prefer a virtualized solution. This container is quite simple to install:

```
$ sudo docker pull quay.io/bgruening/galaxy-hicexplorer
```

To start and otherwise modify this container, please see the instructions on [the docker-galaxy-stable github repository](#). Note that you must use *bgruening/galaxy-hicexplorer* in place of *bgruening/galaxy-stable* in the examples, as the HiCEXplorer Galaxy container is built on top of the galaxy-stable container.

---

**Tip:** For support, or feature requests contact: [deeptools@googlegroups.com](mailto:deeptools@googlegroups.com)

---

## HiCEXplorer tools

- *General principles*
- *Tools for Hi-C data pre-processing*
  - *findRestSite*
  - *hicBuildMatrix*
  - *hicSumMatrices*
  - *hicMergeMatrixBins*
  - *hicCorrectMatrix*
- *Tools for Hi-C QC*
  - *hicQC*
  - *hicCorrelate*
  - *hicPlotDistVsCounts*
  - *hicInfo*
- *Tools for Hi-C data analysis*
  - *hicCompareMatrices*
  - *hicFindEnrichedContacts*
  - *hicPCA*
  - *hicTransform*
- *Tools for TADs processing*
  - *hicFindTADs*
  - *hicMergeTADbins*
- *Tools for Hi-C and TADs visualization*
  - *hicPlotMatrix*
  - *hicPlotTADs*

- *hicPlotViewpoint*
- *hicAggregateContacts*
- *Miscellaneous*
  - *hicExport*

tool	type	input files	main output file(s)	application
find-Rest-Site	pre-processing	1 genome FASTA file	bed file with restriction site coordinates	Identifies the genomic locations of restriction sites
hicBuild-Matrix	pre-processing	2 BAM/SAM files	hicMatrix object	Creates a Hi-C matrix using the aligned BAM files of the Hi-C sequencing reads
hicCorrectMatrix	pre-processing	hicMatrix object	normalized hicMatrix object	Uses iterative correction to remove biases from a Hi-C matrix
hicMerge-MatrixBins	pre-processing	hicMatrix object	hicMatrix object	Merges consecutive bins on a Hi-C matrix to reduce resolution
hicSum-Matrices	pre-processing	2 or more hicMatrix objects	hicMatrix object	Adds Hi-C matrices of the same size
hicFind-Enriched-Contacts	analysis	hicMatrix object	hicMatrix object	Identifies enriched Hi-C contacts
hicCorrelate	analysis	2 or more hicMatrix objects	a heatmap/scatterplot	Computes and visualises the correlation of Hi-C matrices
hicFind-TADs	analysis	hicMatrix object	bedGraph file (TAD score), a boundaries.bed file, a domains.bed file (TADs)	Identifies Topologically Associating Domains (TADs)
hicPlot-Matrix	visualization	hicMatrix object	a heatmap of Hi-C contacts	Plots a Hi-C matrix as a heatmap
hicPlot-TADs	visualization	hicMatrix object, a config file	Hi-C contacts on a given region, along with other provided signal (bigWig) or regions (bed) file	Plots TADs as a track that can be combined with other tracks (genes, signal, interactions)
hicPlot-DistVs-Counts	visualization	hicMatrix object	log log plot of Hi-C contacts per distance	Quality control
hicExport	data integration	multiple Hi-C file formats	Hi-C matrices/outputs in several formats	Export matrix to different formats
hicInfo	information	one or more hicMatrix objects	Screen info	Prints information about matrices, like size, maximum, minimum, bin size, etc.
hicPCA	analysis	one Hi-C matrix	bedgraph or bigwig file(s) for each eigenvector	Computes for A / B compartments the eigenvectors
hicTransform	analysis	one Hi-C matrix	Hi-C matrix	Computes a obs_exp matrix like Lieberman-Aiden (2009), a pearson correlation matrix and or a covariance matrix. These

#### 4.2. HiCEXplorer tools

## General principles

A typical HiCEXplorer command could look like this:

```
$ hicPlotMatrix -m myHiCmatrix.h5 \  
-o myHiCmatrix.pdf \  
--clearMaskedBins \  
--region chrX:10,000,000-15,000,000 \  
--vMin -4 --vMax 4 \  

```

You can always see all available command-line options via `--help`:

```
$ hicPlotMatrix --help
```

- Output format of plots should be indicated by the file ending, e.g. `MyPlot.pdf` will return a pdf file, `MyPlot.png` a png-file.
- Most of the tools that produce plots can also output the underlying data - this can be useful in cases where you don't like the HiCEXplorer visualization, as you can then use the data matrices produced by deepTools with your favorite plotting tool, such as R.
- The vast majority of command line options are also available in Galaxy (in a few cases with minor changes to their naming).

## Tools for Hi-C data pre-processing

**findRestSite**

**hicBuildMatrix**

**hicSumMatrices**

**hicMergeMatrixBins**

**hicCorrectMatrix**

## Tools for Hi-C QC

**hicQC**

**hicCorrelate**

**hicPlotDistVsCounts**

**hicInfo**

## Tools for Hi-C data analysis

**hicCompareMatrices**

**hicFindEnrichedContacts**

**hicPCA**



## hicTransform

## Tools for TADs processing

### hicFindTADs

### hicMergeTADbins

## Tools for Hi-C and TADs visualization

### hicPlotMatrix

### hicPlotTADs

### hicPlotViewpoint

### hicAggregateContacts

## Miscellaneous

### hicExport

## Example usage

- *How we use HiCEXplorer*
  - *Reads mapping*
  - *Creation of a Hi-C matrix*
  - *Correction of Hi-C matrix*
  - *Visualization of results*
  - *Quality control of Hi-C data and biological replicates comparison*
  - *TAD calling*
  - *A / B compartment analysis*

## Hi-C analysis of mouse ESCs using HiCEXplorer

The following example shows how we can use HiCEXplorer to analyze a published dataset. Here we are using a Hi-C dataset from [Marks et. al. 2015](#), on mouse ESCs.

### Protocol

The collection of the cells for Hi-C and the Hi-C sample preparation procedure was performed as previously described [Lieberman-Aiden et al.](#), with the slight modification that *DpnII* was used as restriction enzyme during initial digestion. Paired-end libraries were prepared according to Lieberman-Aiden et al. and sequenced on the NextSeq 500 platform using  $2 \times 75$  bp sequencing.

## Prepare for analysis

### Download Raw fastq files

The fastq files can be downloaded from the EBI archive (or NCBI archive). We will store the files in the directory *original\_data*.

```
mkdir original_data

wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/007/SRR1956527/SRR1956527_1.fastq.gz -
↪O original_data/SRR1956527_1.fastq.gz
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/007/SRR1956527/SRR1956527_2.fastq.gz -
↪O original_data/SRR1956527_2.fastq.gz

wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/008/SRR1956528/SRR1956528_1.fastq.gz -
↪O original_data/SRR1956528_1.fastq.gz
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/008/SRR1956528/SRR1956528_2.fastq.gz -
↪O original_data/SRR1956528_2.fastq.gz

wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/009/SRR1956529/SRR1956529_1.fastq.gz -
↪O original_data/SRR1956529_1.fastq.gz
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR195/009/SRR1956529/SRR1956529_2.fastq.gz -
↪O original_data/SRR1956529_2.fastq.gz
```

### Create an index

We start with creating an index for our alignment software for the *GRCm38/mm10* genome. As a source we use the mm10 genome from UCSC

```
mkdir genome_mm10
wget http://hgdownload-test.cse.ucsc.edu/goldenPath/mm10/bigZips/chromFa.tar.gz -O_
↪genome_mm10/chromFa.tar.gz
tar -xvzf genome_mm10/chromFa.tar.gz
cat genome_mm10/*.fa > genome_mm10/mm10.fa
```

We have the mm10 genome stored in one fasta file and can build the index. We tried it successfully with *hisat2*, *bowtie2* and *bwa*. Run the mapping with one of them and do not mix them!

### hisat2

```
hisat2-build -p 8 genome_mm10/mm10.fa hisat2/mm10_index
```

You can find more information about [hisat](#)

### bowtie2

```
bowtie2-build genome_mm10/mm10.fa bowtie2/mm10_index --threads 8
```

You can find more information about [bowtie](#)

## bwa

```
bwa index -p bwa/mm10_index genome_mm10/mm10.fa
```

You can find more information about [bwa](#)

## Mapping the RAW files

Mates have to be mapped individually to avoid mapper specific heuristics designed for standard paired-end libraries.

It is important to have in mind for the different mappers:

- for either *bowtie2* or *hisat2* use the *-reorder* parameter which tells bowtie2 or hisat2 to output the *sam* files in the **exact** same order as in the *.fastq* files.
- use local mapping, in contrast to end-to-end. A fraction of Hi-C reads are chimeric and will not map end-to-end thus, local mapping is important to increase the number of mapped reads.
- Tune the aligner parameters to penalize deletions and insertions. This is important to avoid aligned reads with gaps if they happen to be chimeric.

## hisat2

```
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956527_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956527_1.bam
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956527_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956527_2.bam
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956528_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956528_1.bam
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956528_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956528_2.bam
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956529_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956529_1.bam
hisat2 -x hisat2/mm10_index --threads 8 -U ../original_data/SRR1956529_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956529_2.bam
```

## bowtie2

```
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956527_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956527_1.bam
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956527_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956527_2.bam
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956528_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956528_1.bam
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956528_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956528_2.bam
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956529_1.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956529_1.bam
bowtie2 -x bowtie2/mm10_index --threads 8 -U ../original_data/SRR1956529_2.fastq.gz --
↪reorder | samtools view -Shb - > SRR1956529_2.bam
```

```

bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956527_1.fastq.gz
↪| samtools view -Shb -> SRR1956527_1.bam
bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956527_2.fastq.gz
↪| samtools view -Shb -> SRR1956527_2.bam
bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956528_1.fastq.gz
↪| samtools view -Shb -> SRR1956528_1.bam
bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956528_2.fastq.gz
↪| samtools view -Shb -> SRR1956528_2.bam
bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956529_1.fastq.gz
↪| samtools view -Shb -> SRR1956529_1.bam
bwa mem -A 1 -B 4 -E 50 -L 0 -t 8 bwa/mm10_index original_data/SRR1956529_2.fastq.gz
↪| samtools view -Shb -> SRR1956529_2.bam

```

## Build, visualize and correct Hi-C matrix

### Create a Hi-C matrix using the aligned files

In the following we will create three Hi-C matrices and merge them to one.

### Build Hi-C matrix

hicBuildMatrix builds the matrix of read counts over the bins in the genome, considering the sites around the given restriction site. We need to provide:

- the input BAM/SAM files: *-samFiles SRR1956527\_1.sam SRR1956527\_2.sam*
- binsize: *-binSize 1000*
- restriction sequence: *-restrictionSequence GATC*
- the name of output bam file which contains the accepted alignments: *-outBam SRR1956527\_ref.bam*
- name of output matrix file: *-outFileName hicMatrix/SRR1956527\_10kb.h5*
- the folder for the quality report: *-QCfolder hicMatrix/SRR1956527\_QC*
- the number of to be used threads. Minimum value is 3: *-threads 8*
- the buffer size for each thread buffering *inputBufferSize* lines of each input BAM/SAM file: *-inputBufferSize 400000*

To build the Hi-C matrices:

```

mkdir hicMatrix
hicBuildMatrix --samFiles SRR1956527_1.bam SRR1956527_2.bam --binSize 10000 --
↪restrictionSequence GATC --outBam SRR1956527_ref.bam --outFileName hicMatrix/
↪SRR1956527_10kb.h5 --QCfolder hicMatrix/SRR1956527_10kb_QC --threads 8 --
↪inputBufferSize 400000
hicBuildMatrix --samFiles SRR1956528_1.bam SRR1956528_2.bam --binSize 10000 --
↪restrictionSequence GATC --outBam SRR1956528_ref.bam --outFileName hicMatrix/
↪SRR1956528_10kb.h5 --QCfolder hicMatrix/SRR1956528_10kb_QC --threads 8 --
↪inputBufferSize 400000
hicBuildMatrix --samFiles SRR1956529_1.bam SRR1956529_2.bam --binSize 10000 --
↪restrictionSequence GATC --outBam SRR1956529_ref.bam --outFileName hicMatrix/
↪SRR1956529_10kb.h5 --QCfolder hicMatrix/SRR1956529_10kb_QC --threads 8 --
↪inputBufferSize 400000

```

The output bam files show that we have around 34M, 54M and 58M selected reads for SRR1956527, SRR1956528 & SRR1956529, respectively. Normally 25% of the total reads are selected. The output matrices have counts for the genomic regions. The extension of output matrix files is *.h5*.

A quality report is created in e.g. *hicMatrix/SRR1956527\_10kb\_QC*, have a look at the report *hicQC.html*.

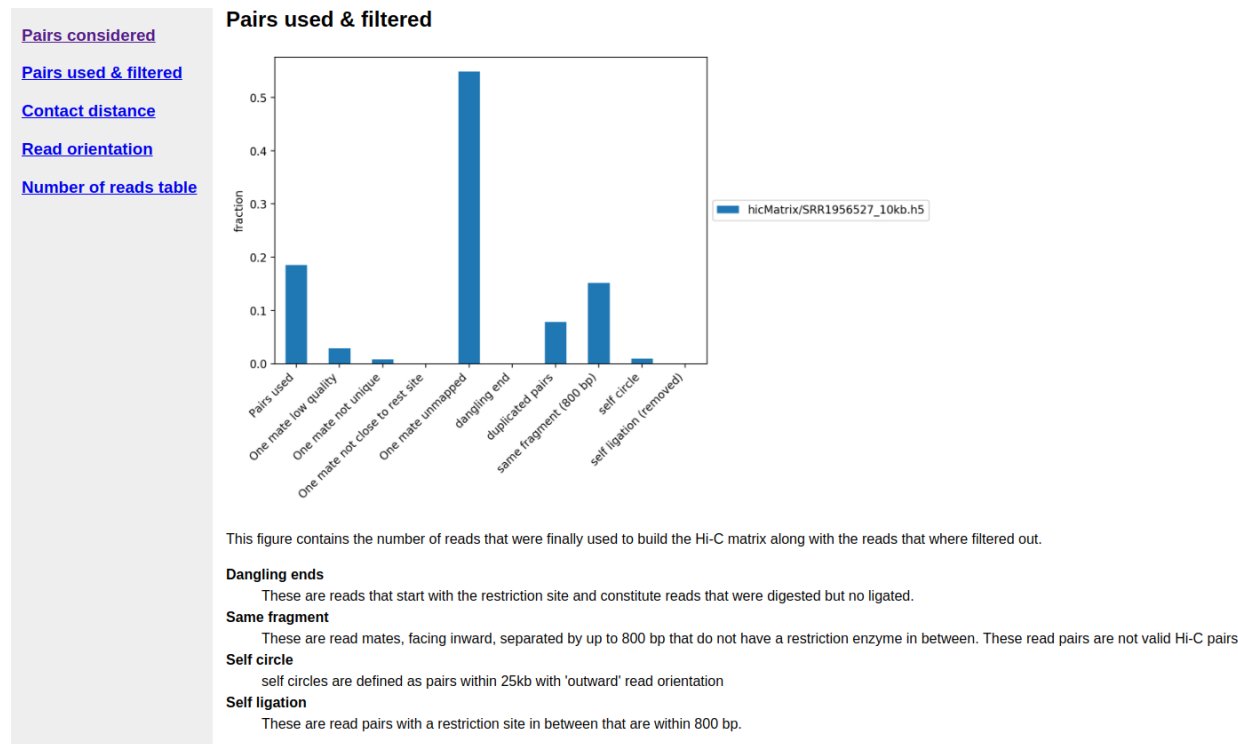


Fig. 4.1: A segment of Hi-C quality report.

## Merge (sum) matrices from replicates

To increase the depth of reads we merge the counts from these three replicates.

```
hicSumMatrices --matrices hicMatrix/SRR1956527_10kb.h5 hicMatrix/SRR1956528_10kb.h5 \
hicMatrix/SRR1956529_10kb.h5 --outFileName hicMatrix/replicateMerged_10kb.h5
```

## Plot Hi-C matrix

A 10kb bin matrix is quite large to plot and is better to reduce the resolution (to know the size of a Hi-C matrix use the tool *hicInfo*), i.e. we usually run out of memory for a 1 kb or a 10 kb matrix and second, the time to plot is very long (minutes instead of seconds). For this we use the tool *hicMergeMatrixBins*.

## Merge matrix bins for plotting

*hicMergeMatrixBins* merges the bins into larger bins of given number (specified by *-numBins*). We will merge 1000 bins in the original (uncorrected) matrix and then correct it. The new bin size is going to be  $10.000 \text{ bp} * 100 = 1.000.000 \text{ bp} = 1 \text{ Mb}$

```
hicMergeMatrixBins \  
--matrix hicMatrix/replicateMerged_10kb.h5 --numBins 100 \  
--outFileName hicMatrix/replicateMerged.100bins.h5
```

## Plot the corrected Hi-C matrix

**hicPlotMatrix** can plot the merged matrix. We use the following options:

- the matrix to plot: `--matrix hicMatrix/replicateMerged.100bins.h5`
- logarithmic values for plotting: `--log1p`
- the resolution of the plot: `--dpi 300`
- masked bins should not be plotted: `--clearMaskedBins`
- the order of the chromosomes in the plot: `--chromosomeOrder chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13 chr14 chr15 chr16 chr17 chr18 chr19 chrX chrY`
- the color map: `--colorMap jet`
- the title of the plot: `--title "Hi-C matrix for mESC"`
- the plot image itself: `--outFileName plots/plot_1Mb_matrix.png`

```
mkdir plots  
hicPlotMatrix \  
--matrix hicMatrix/replicateMerged.100bins.h5 \  
--log1p \  
--dpi 300 \  
--clearMaskedBins \  
--chromosomeOrder chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12_  
↪chr13 chr14 chr15 chr16 chr17 chr18 chr19 chrX chrY \  
--colorMap jet \  
--title "Hi-C matrix for mESC" \  
--outFileName plots/plot_1Mb_matrix.png
```

## Correct Hi-C Matrix

**hicCorrectMatrix** corrects the matrix counts in an iterative manner. For correcting the matrix, it's important to remove the unassembled scaffolds (e.g. NT\_) and keep only chromosomes, as scaffolds create problems with matrix correction. Therefore we use the chromosome names (1-19, X, Y) here. **Important:** Use 'chr1 chr2 chr3 etc.' if your genome index uses chromosome names with the 'chr' prefix.

Matrix correction works in two steps: first a histogram containing the sum of contact per bin (row sum) is produced. This plot needs to be inspected to decide the best threshold for removing bins with lower number of reads. The second steps removes the low scoring bins and does the correction.

In the following we will use a matrix with a bin size of 20 kb:  $10\text{kb} * 2 = 20\text{ kb}$

```
hicMergeMatrixBins \  
--matrix hicMatrix/replicateMerged_10kb.h5 --numBins 2 \  
--outFileName hicMatrix/replicateMerged.matrix_20kb.h5
```

(1-19, X, Y) variant:

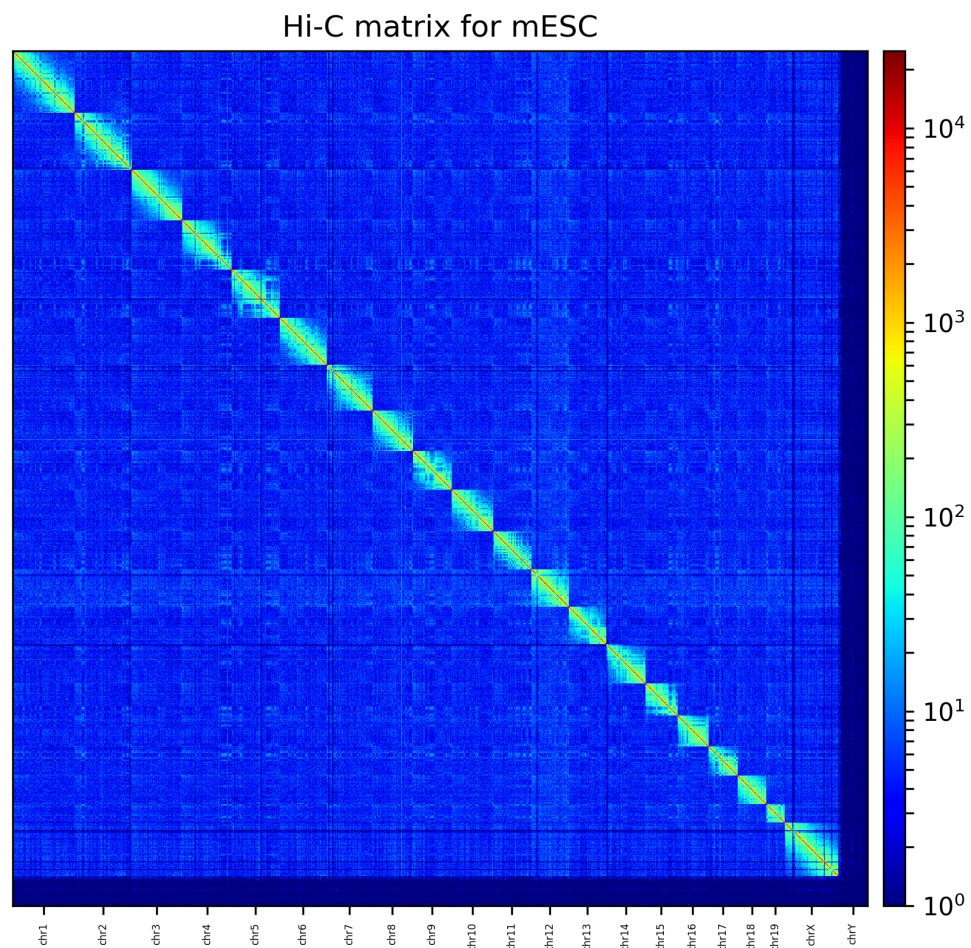


Fig. 4.2: The Hi-C interaction matrix with a resolution of 1 MB.

```
hicCorrectMatrix diagnostic_plot \
--chromosomes 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 X Y \
--matrix hicMatrix/replicateMerged.matrix_20kb.h5 --plotName hicMatrix/diagnostic_
→plot.png
```

(chr1-ch19, chrX, chrY) variant:

```
hicCorrectMatrix diagnostic_plot \
--chromosomes chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13_
→chr14 chr15 chr16 chr17 chr18 chr19 chrX chrY \
--matrix hicMatrix/replicateMerged.matrix_20kb.h5 --plotName hicMatrix/diagnostic_
→plot.png
```

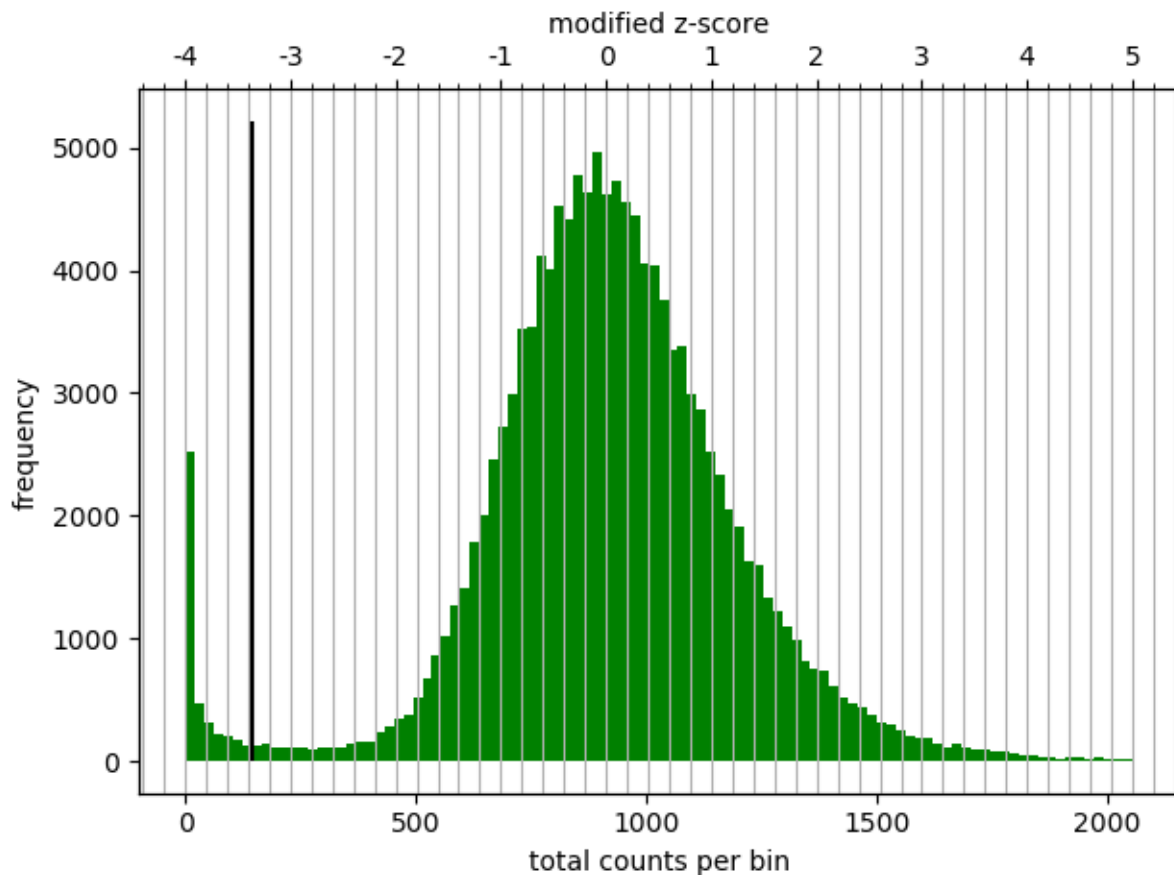


Fig. 4.3: Diagnostic plot for the Hi-C matrix at a resolution of 20 kb

The output of the program prints a threshold suggestion that is usually accurate but is better to revise the histogram plot. The threshold is visualized in the plot as a black vertical line. See [Example usage](#) for an example and for more info.

**The threshold parameter needs two values:**

- low z-score
- high z-score

“The absolute value of  $z$  represents the distance between the raw score and the population mean in units of the standard



deviation. z is negative when the raw score is below the mean, positive when above.” (Source). For more information see [wikipedia](#).

Fig. 4.4: The z-score definition.

In our case the distribution describes the counts per bin of a genomic distance. To remove all bins with a z-score threshold less / more than X means to remove all bins which have less / more counts than X of mean of their specific distribution in units of the standard deviation.

Looking at the above distribution, we can select the value of -2 (lower end) and 3 (upper end) to remove. This is given by the **-filterThreshold** option in hicCorrectMatrix.

(1-19, X, Y) variant:

```
hicCorrectMatrix correct \
--chromosomes 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 X Y \
--matrix hicMatrix/replicateMerged.matrix_20kb.h5 \
--filterThreshold -2 3 --perchr --outFileName hicMatrix/replicateMerged.Corrected_
↪20kb.h5
```

(chr1-ch19, chrX, chrY) variant:

```
hicCorrectMatrix correct \
--chromosomes chr1 chr2 chr3 chr4 chr5 chr6 chr7 chr8 chr9 chr10 chr11 chr12 chr13_
↪chr14 chr15 chr16 chr17 chr18 chr19 chrX chrY \
--matrix hicMatrix/replicateMerged.matrix_20kb.h5 \
--filterThreshold -2 3 --perchr --outFileName hicMatrix/replicateMerged.Corrected_
↪20kb.h5
```

It can happen that the correction stops with:

```
`ERROR:iterative correction:*Error* matrix correction produced extremely large values.
This is often caused by bins of low counts. Use a more stringent filtering of bins.`
```

This can be solved by a more stringent z-score values for the filter threshold or by a look at the plotted matrix. In our case we see that chromosome Y is having more or less 0 counts in its bins. This chromosome can be excluded from the correction by not defining it for the set of chromosomes that should be corrected, parameter **-chromosomes**.

## Plot corrected matrix

We can now plot the one of the chromosomes (e.g. chromosome X) , with the corrected matrix.

**New parameter:**

- The region to plot: **-region chrX:10000000-2000000** or **-region chrX**

(1-19, X, Y) variant:

```
hicPlotMatrix \
--loglp --dpi 300 \
--matrix hicMatrix/replicateMerged.Corrected_20kb.npz \
--region X --title "Corrected Hi-C matrix for mESC : chrX" \
--outFileName plots/replicateMerged_Corrected-20kb_plot-chrX.png
```

(chr1-ch19, chrX, chrY) variant:

```
hicPlotMatrix \  
--loglp --dpi 300 \  
--matrix hicMatrix/replicateMerged.Corrected_20kb.npz \  
--region chrX --title "Corrected Hi-C matrix for mESC : chrX" \  
--outFileName plots/replicateMerged_Corrected-20kb_plot-chrX.png
```

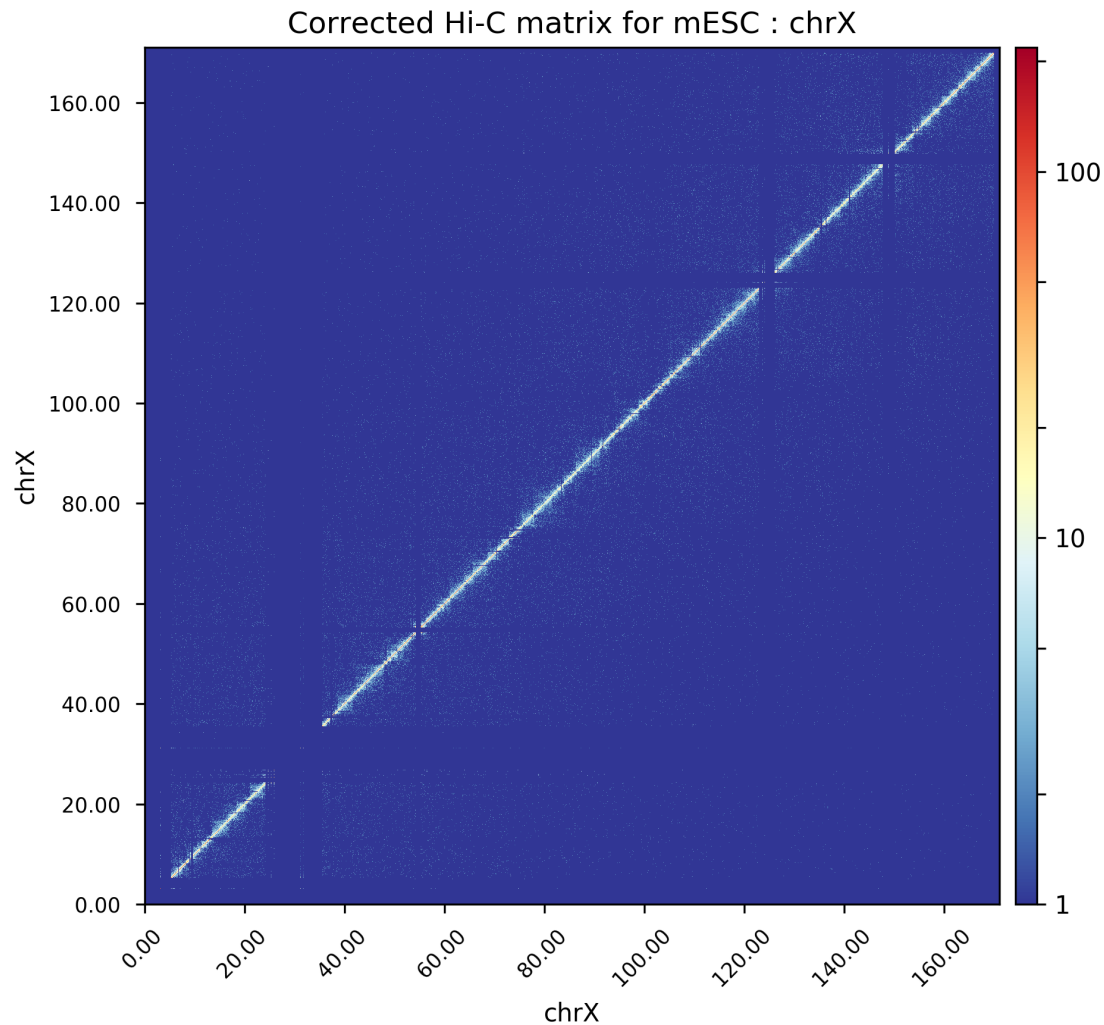


Fig. 4.5: The Hi-C interaction matrix for chromosome X.

### Plot TADs

“The partitioning of chromosomes into topologically associating domains (TADs) is an emerging concept that is reshaping our understanding of gene regulation in the context of physical organization of the genome” [Ramirez et al. 2017].

## Find TADs

TAD calling works in two steps: First HiCEXplorer computes a TAD-separation score based on a z-score matrix for all bins. Then those bins having a local minimum of the TAD-separation score are evaluated with respect to the surrounding bins to decide assign a p-value. Then a cutoff is applied to select the bins more likely to be TAD boundaries.

hicFindTADs tries to identify sensible parameters but those can be change to identify more stringent set of boundaries.

```
mkdir TADs
hicFindTADs --matrix hicMatrix/replicateMerged.Corrected_20kb.h5 \
--minDepth 60000 --maxDepth 120000 --numberOfProcessors 8 --step 20000 \
--outPrefix TADs/marks_et-al_TADs_20kb-Bins --minBoundaryDistance 80000 \
--correctForMultipleTesting fdr --threshold 0.05
```

As an output we get the boundaries, domains and scores separated files. We will use in the plot below only the TAD-score file.

## Build Tracks File

We can plot the TADs for a given chromosomal region. For this we need to create a track file containing the instructions to build the plot. The tools/hicPlotTADs documentation contains the instructions to build the track file.

In following plot we will use the listed track file. Please store it as track.ini.

```
[hic]
file = hicMatrix/replicateMerged.Corrected_20kb.h5
title = HiC mESC chrX:99974316-101359967
colormap = RdYlBu_r
depth = 2000000
width = 7
transform = loglp
x labels = yes
type = interaction
file_type = hic_matrix
boundaries_file = TADs/marks_et-al_TADs_20kb-Bins_domains.bed

[x-axis]
fontsize=16
where=top

[tad score]
file = TADs/marks_et-al_TADs_20kb-Bins_score.bedgraph
title = "TAD separation score"
width = 2
type = lines
color = blue
file_type = bedgraph

[spacer]

[gene track]
file = mm10_genes_sorted.bed
width = 10
title = "mm10 genes"
width = 5
labels = off
```

We used as a gene track [mm10 genes](#) and sorted with *sortBed* from *bedtools*.

## Plot

We plot the result with:

(1-19, X, Y) variant:

```
hicPlotTADs --tracks track.ini --region X:98000000-105000000 \
--dpi 300 --outFileName plots/marks_et-al_TADs.png \
--title "Marks et. al. TADs on X"
```

(chr1-ch19, chrX, chrY) variant:

```
hicPlotTADs --tracks track.ini --region chrX:98000000-105000000 \
--dpi 300 --outFileName plots/marks_et-al_TADs.png \
--title "Marks et. al. TADs on X"
```

The result is:

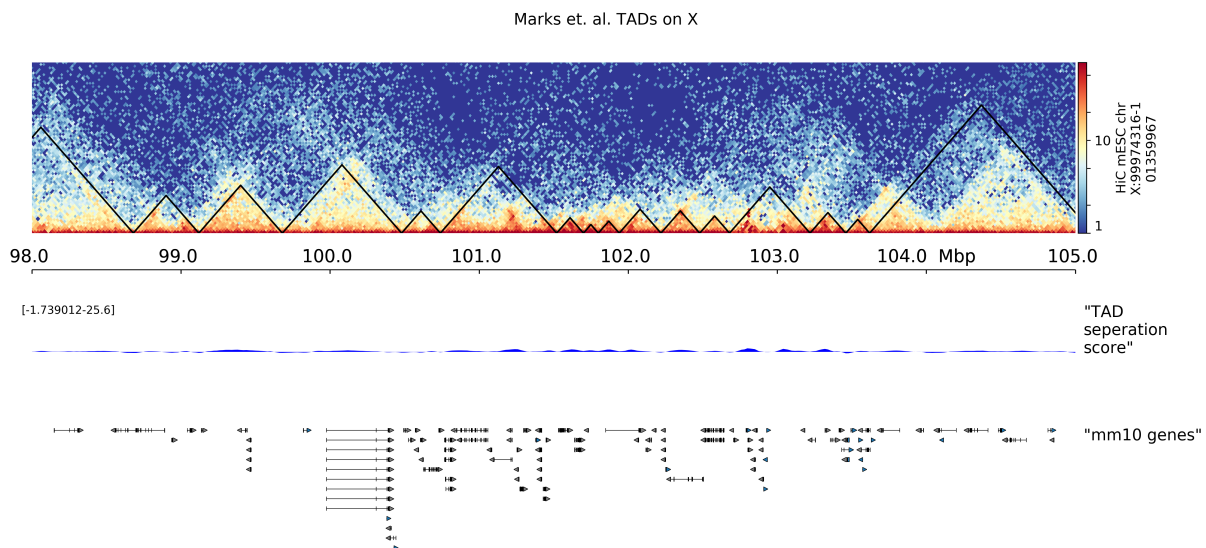


Fig. 4.6: TADplot

## Importing and Exporting HiCEXplorer data

### Exporting HiCEXplorer output to Bioconductor

It's possible to export Hi-C Matrices produced by HiCEXplorer to [bioconductor](#) in R, which allows us to use existing bioconductor infrastructure for differential Hi-C analysis. The tool **hicExport** allows us to write Hi-C matrices in a format that can easily be imported in bioconductor as **GInteractions** object. Below is an example.

```
## Assuming HiCEXplorer is installed in ~/programs
hicExport --inFile ~/programs/HiCEXplorer/test/test_data/Li_et_al_2015.h5 \
-o GInteraction_example --outputFormat GInteractions
```

The output file is in tsv format. It looks like this :

V1	V2	V3	V4	V5	V6	V7
X	19537	20701	X	19537	20701	1054.47483
X	19537	20701	X	20701	22321	375.86990
X	19537	20701	X	22321	24083	222.53900
X	19537	20701	X	24083	25983	114.26340
X	19537	20701	X	25983	27619	95.87463

This file can now be loaded into R as a **GInteractions** object, as shown below :

```
## INSIDE R
library(GenomicRanges)
library(InteractionSet)

hic <- read.delim("GInteraction_example.tsv", header = FALSE)

# Converting data.frame to GInteraction
convertToGI <- function(df) {
  row.regions <- GRanges(df$V1, IRanges(df$V2, df$V3)) # interaction start
  col.regions <- GRanges(df$V4, IRanges(df$V5, df$V6)) # interaction end
  gi <- GInteractions(row.regions, col.regions)
  gi$norm.freq <- df$V7 # Interaction frequencies
  return(gi)
}

hic.gi <- convertToGI(hic)
```

Multiple files can be loaded, and converted to an **InteractionSet** object. If you have prepared matrices using binning, the intervals in the matrices must be the same. Therefore it's easy to merge these matrices together in an **InteractionSet** object. In case some bins don't match, we can merge the **GInteraction** objects based on matching bins, as follows.

```
# assuming hic.gi is a list of two GInteraction objects hic.gi1 and hic.gi2
# hic.gi <- list(hic.gi1, hic.gi2)

# Get common regions between the two objects
combined <- unique(c(hic.gi$hic.gi1, hic.gi$hic.gi2))

# replace original regions with the common regions
replaceRegions(hic.gi$hic.gi1) <- regions(combined)
replaceRegions(hic.gi$hic.gi2) <- regions(combined)

# Get the matching indexes between the two objects
matched <- lapply(hic.gi, function(x) {
  match(x, combined)
})

# Create a count matrix (for interaction frequencies)
counts <- matrix(0, ncol = 2, nrow=length(combined)) # counts for unmatched bins set
↳to zero

# fill in the counts for matched bins
counts[matched$hic.gi1,1] <- hic.gi$hic.gi1$norm.freq
counts[matched$hic.gi2,2] <- hic.gi$hic.gi2$norm.freq
```

```
# Finally, create the InteractionSet object
iset <- InteractionSet(counts, combined)
```

InteractionSet objects can be used for packages like [diffHic](#), for differential Hi-C analysis.

- For more information on working with GInteraction and InteractionSet objects in bioconductor check out [this vignette](#).

## How we use HiCEXplorer

To generate a Hi-C contact matrix is necessary to perform the following basic steps

1. Map the Hi-C reads to the reference genome
2. Filter the aligned reads to create a contact matrix
3. Filter matrix bins with low or zero read coverage
4. Remove biases from the Hi-C contact matrices

After a corrected Hi-C matrix is created other tools can be used to visualize it, call TADS or compare it with other matrices.

## Reads mapping

Mates have to be mapped individually to avoid mapper specific heuristics designed for standard paired-end libraries.

We have used the HiCEXplorer successfully with *bwa*, *bowtie2* and *hisat2*. However, it is important to:

- for either *bowtie2* or *hisat2* use the *-reorder* parameter which tells bowtie2 or hisat2 to output the *sam* files in the **exact** same order as in the *.fastq* files.
- use local mapping, in contrast to end-to-end. A fraction of Hi-C reads are chimeric and will not map end-to-end thus, local mapping is important to increase the number of mapped reads.
- Tune the aligner parameters to penalize deletions and insertions. This is important to avoid aligned reads with gaps if they happen to be chimeric.

```
# map the reads, each mate individually using
# for example bwa
#
# bwa mem mapping options:
#   -A INT          score for a sequence match, which scales options -TdBOELU_
↪unless overridden [1]
#   -B INT          penalty for a mismatch [4]
#   -O INT[,INT]    gap open penalties for deletions and insertions [6,6]
#   -E INT[,INT]    gap extension penalty; a gap of size k cost '{-O} + {-E}*k' [1,
↪1] # this is set very high to avoid gaps
#                                   at restriction sites. Setting the gap extension_
↪penalty high, produces better results as
#                                   the sequences left and right of a restriction site_
↪are mapped independently.
#   -L INT[,INT]    penalty for 5'- and 3'-end clipping [5,5] # this is set to no_
↪penalty.

$ bwa mem -A1 -B4 -E50 -L0 index_path \
  -U mate_R1.fastq.gz 2>>mate_R1.log | samtools view -Shb - > mate_R1.bam
```

```
$ bwa mem -A1 -B4 -E50 -L0 index_path \
-U mate_R2.fastq.gz 2>>mate_R2.log | samtools view -Shb - > mate_R2.bam
```

## Creation of a Hi-C matrix

Once the reads have been mapped the Hi-C matrix can be built. For this, the minimal extra information required is the *binSize* used for the matrix. Is it best to enter a low number like 10.000 because lower resolution matrices (larger bins) can be easily constructed using `hicMergeMatrixBins`. Matrices at restriction fragment resolution can be created by providing a file containing the restriction sites, this file can be created with the tool `findRestSite`

`findRestSite` that is part of HiCEXplorer.

```
# build matrix from independently mated read pairs
# the restriction sequence GATC is recognized by the DpnII restriction enzyme

$ hicBuildMatrix --samFiles mate_R1.bam mate_R2.bam \
--binSize 10000 \
--restrictionSequence GATC \
--threads 4
--inputBufferSize 100000
--outBam hic.bam \
-o hic_matrix.h5
--QCfolder ./hicQC
```

`hicBuildMatrix` creates two files, a bam file containing only the valid Hi-C read pairs and a matrix containing the Hi-C contacts at the given resolution. The bam file is useful to check the quality of the Hi-C library on the genome browser. A good Hi-C library should contain piles of reads near the restriction fragment sites. In the *QCfolder* a html file is saved with plots containing useful information for the quality control of the Hi-C sample like the number of valid pairs, duplicated pairs, self-ligations etc. Usually, only 25%-40% of the reads are valid and used to build the Hi-C matrix mostly because of the reads that are on repetitive regions that need to be discarded.

An important quality control measurement to check is the *inter chromosomal* fraction of reads as this is an indirect measure of random Hi-C contacts. Good Hi-C libraries have lower than 10% inter chromosomal contacts. The *hicQC* module can be used to compare the QC measures from different samples.

## Correction of Hi-C matrix

The Hi-C matrix has to be corrected to remove GC, open chromatin biases and, most importantly, to normalize the number of restriction sites per bin. Because a fraction of bins from repetitive regions contain few contacts it is necessary to filter those regions first. Also, in mammalian genomes some regions enriched by reads should be discarded. To aid in the filtering of regions `hicCorrectMatrix` generates a diagnostic plot as follows:

```
$ hicCorrectMatrix diagnostic_plot -m hic_matrix.h5 -o hic_corrected.h5
```

The plot should look like this:

For the upper threshold is only important to remove very high outliers and thus a value of 5 could be used. For the lower threshold it is recommended to use a value between -2 and -1. What it not desired is to try to correct low count bins which could result simply in an amplification of noise. For the upper threshold is not so concerning because those bins will be scaled down.

Once the thresholds have been decided, the matrix can be corrected

```
# correct Hi-C matrix
$ hicCorrectMatrix -m hic_matrix.h5 --filterThreshold -1.5 5 -o hic_corrected.h5
```

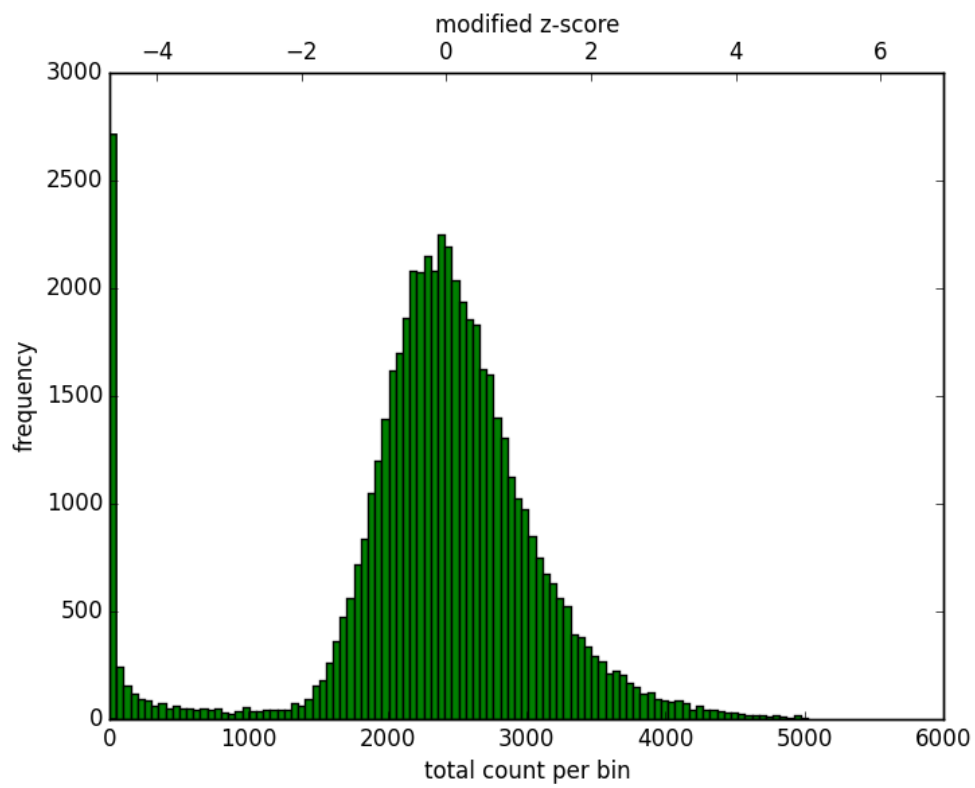


Fig. 4.7: Histogram of the number of counts per bin.



## Visualization of results

There are two ways to see the resulting matrix, one using `hicPlotMatrix` and the other is using `hicPlotTADs`. The first one allows the visualization over large regions while the second one is preferred to see specific parts together with other information, for example genes or bigwig tracks.

Because of the large differences in counts found in the matrix, it is better to plot the counts using the `-logIp` option.

```
$ hicPlotMatrix -m hic_corrected.h5 -o hic_plot.png --region 1:200000000-800000000 --  
↪ logIp
```

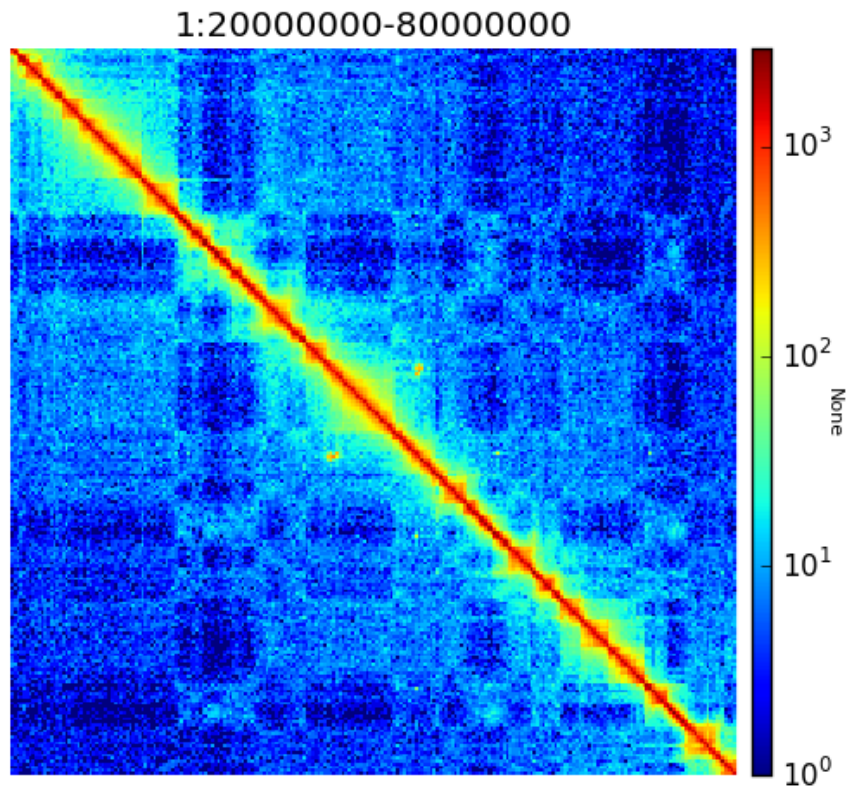


Fig. 4.8: Corrected Hi-C counts in log scale.

## Quality control of Hi-C data and biological replicates comparison

HiCEXplorer integrates multiple tools that allow the evaluation of the quality of Hi-C libraries and matrices.

- `hicQC` on the log files produced by `hicBuildMatrix` and control of the pdf file produced.

Proportion of useful reads is important to assess the efficiency of the HiC protocol, which is dependant of proportion of dangling ends detected... Proportion of inter chromosomal, short range and long range contacts are important for....

- `hicPlotDistVsCounts` to compare the distribution of corrected Hi-C counts in relation with the genomic distance between multiple samples. If some differences are observed between biological replicates, these can be investigated more precisely by computing `log2ratio` matrices.
- `hicCompareMatrices` `log2ratio` of matrices of biological replicates to identify where the potential changes are located.
- `hicPlotPCA` bins correlation of two biological replicates.

### TAD calling

To call TADs a corrected matrix is needed. Restriction fragment resolution matrices provide the best results. TAD calling works in two steps: First HiCEXplorer computes a TAD-separation score based on a z-score matrix for all bins. Then those bins having a local minimum of the TAD-separation score are evaluated with respect to the surrounding bins to decide assign a p-value. Then a cutoff is applied to select the bins more likely to be TAD boundaries.

```
$ hicFindTADs -m hic_corrected.h5 --outPrefix hic_corrected --numberOfProcessors 16
```

This code will produce several files: 1. The TAD-separation score file, 2. the z-score matrix, 3. a bed file with the boundary location, 4. a bed file with the domains, 5. a bedgraph file with the TAD-score that can be visualized in a genome browser.

The TAD-separation score and the matrix can be visualized using `hicPlotTADs`.

### A / B compartment analysis

To compute the A / B compartments the matrix needs to be transformed to an observed/expected matrix in the way [Lieberman-Aiden](#) describes it. In a next step a pearson correlation matrix and based on it a covariance matrix is computed. Finally the eigenvectors based on the covariance matrix are computed. All these steps are computed with the command:

```
$ hicPCA -m hic_corrected.h5 --outFileName pca1.bedgraph pca2.bedgraph
```

If the intermediate matrices of this process should be used for plotting run:

```
$ hicTransform -m hic_corrected.h5 --outFileName all.h5 --method all
```

This creates all intermediate matrices: `obs_exp_all.h5`, `pearson_all.h5` and `covariance_all.h5`.

The A / B compartments can be plotted with `hicPlotMatrix`.

```
$ hicPlotMatrix -m pearson_all.h5 --outFileName pca1.png --perChr --pca pca1.bedgraph
```

```
//.. figure:: ../images/eigenvector1_lieberman.png // :scale: 90 % // :align: center
```

## News and Developments

### Release 2.0

**December 21, 2017**

This release makes HiCEXplorer ready for the future:

- Python 3 support

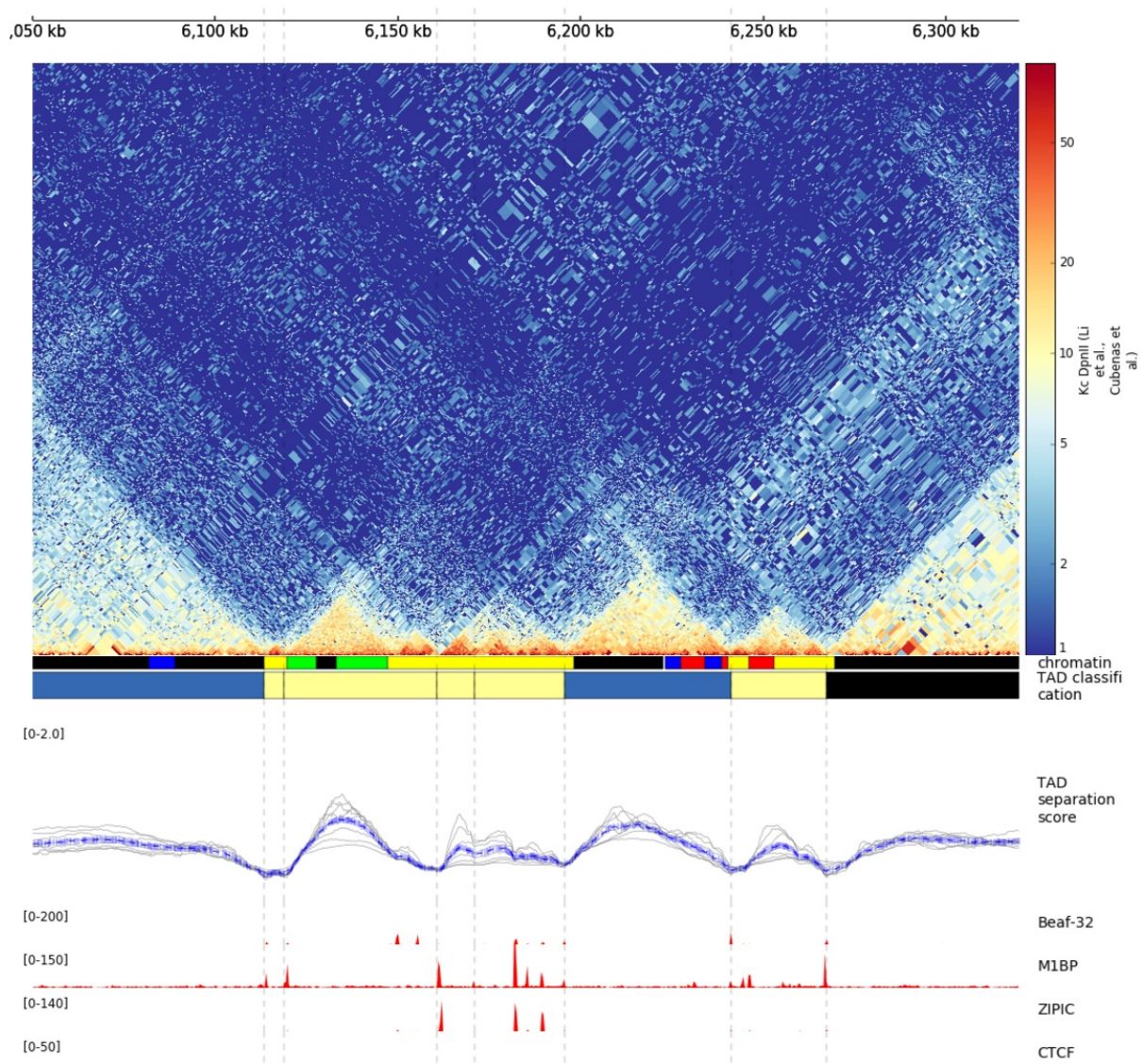


Fig. 4.9: Example output from `hicPlotTADs` from <http://chorogenome.ie-freiburg.mpg.de/>

- Cooler file format support
- A/B compartment analysis
- Improved visualizations
- bug fixes for `-perChr` option in `hicPlotMatrix`
- eigenvector track with `-pca` for `hicPlotMatrix`
- visualization of interactions around a reference point or region with `hicPlotViewpoint`
- Higher test coverage
- re-licensing from GPLv2 to GPLv3

### Release 1.8.1

**November 27, 2017**

Bug fix release:

- a fix concerning the handling chimeric alignments in `hicBuildMatrix`. Thanks to Aleksander Jankowski @ajank
- handling of dangling ends was too strict
- improved help message in `hicBuildMatrix`

### Release 1.8

**October 25, 2017**

This release is adding new features and fixes many bugs:

- `hicBuildMatrix`: Added multicore support, new parameters `-threads` and `-inputBufferSize`
- `hicFindTADs`:
  - One call instead of two: `hicFindTADs TAD_score` and `hicFindTADs find_TADs` merged to `hicFindTADs`.
  - New multiple correction method supported: False discovery rate. Call it with `-correctForMultipleTesting fdr` and `-threshold 0.05`.
  - Update of the tutorial: mES-HiC analysis.
  - Additional test cases and docstrings to improve the software quality
  - Fixed a bug occurring with bigwig files with frequent NaN values which resulted in only NaN averages
- `hicPlotTADs`: Support for plotting points
- Moved galaxy wrappers to <https://github.com/galaxyproject/tools-iuc>
- Fixed multiple bugs with saving matrices
- `hicCorrelate`: Changes direction of dendograms to left

## Release 1.7.2

**April 3, 2017**

- Added option to plot bigwig files as a line hicPlotTADs
- Updated documentation
- Improved hicPlotMatrix –region output
- Added compressed matrices. In our tests the compressed matrices are significantly smaller.

**March 28, 2017**

## Release 1.7

**March 28, 2017**

This release adds a quality control module to check the results from hicBuildMatrix. By default, now hicBuildMatrix generates a HTML page containing the plots from the QC measures. The results from several runs of hicBuildMatrix can be combined in one page using the new tool hicQC.

Also, this release added a module called hicCompareMatrices that takes two Hi-C matrices and computes the difference, the ratio or the log2 ratio. The resulting matrix can be plotted with hicPlotMatrix to visualize the changes.

## Preprint introducing HiCEXplorer is now online

**March 8, 2017**

Our #biorXiv preprint on DNA sequences behind Fly genome architecture is online!

Read the article here : <http://biorxiv.org/content/early/2017/03/08/115063>

In this article, we introduce HiCEXplorer : Our easy to use tool for Hi-C data analysis, also available in [Galaxy](#).

We also introduce [HiCBrowser](#) : A standalone software to visualize Hi-C along with other genomic datasets.

Based on HiCEXplorer and HiCBrowser, we built a useful resource for anyone to browse and download the chromosome conformation datasets in Human, Mouse and Flies. It's called [the chorogenome navigator](#)

Along with these resources, we present an analysis of DNA sequences behind 3D genome of Flies. Using high-resolution Hi-C analysis, we find a set of DNA motifs that characterize TAD boundaries in Flies and show the importance of these motifs in genome organization.

We hope that these resources and analysis would be useful for the community and welcome any feedback.

## HiCEXplorer wins best poster prize at VizBi2016

**March 20, 2016**

We are excited to announce that HiCEXplorer has won the [NVIDIA Award for Best Scientific Poster](#) in VizBi2016, the international conference on visualization of biological data.

[Read more here](#)

This was our poster :



---

### Citation

---

Please cite HiCEplorer as follows:

Fidel Ramirez, Vivek Bhardwaj, Jose Villaveces, Laura Arrigoni, Bjoern A Gruening, Kin Chung Lam, Bianca Habermann, Asifa Akhtar, Thomas Manke. **“High-resolution TADs reveal DNA sequences underlying genome organization in flies”**. *Nature Communications*, Volume 9, Article number: 189 (2018), doi: <https://doi.org/10.1038/s41467-017-02525-w>



This tool suite is developed by the Bioinformatics Unit at the Max Planck Institute for Immunobiology and Epigenetics, Freiburg.