HiC-Bench Manual

Stephen M. Kelly\textsuperscript{1,2}, Charalampos Lazaris\textsuperscript{3,4,5}, Aristotelis Tsirigos\textsuperscript{1,2,3,4,5}

\textsuperscript{1} Applied Bioinformatics Center, Office of Collaborative Science, NYU School of Medicine, NY 10016, USA
\textsuperscript{2} Genome Technology Center, Office of Collaborative Science, NYU School of Medicine, NY 10016, USA
\textsuperscript{3} Department of Pathology, NYU School of Medicine, New York, New York 10016, USA
\textsuperscript{4} NYU Cancer Institute and Helen L. and Martin S. Kimmel Center for Stem Cell Biology, NYU School of Medicine, New York, New York 10016, USA
\textsuperscript{5} Center for Health Informatics & Bioinformatics, NYU School of Medicine, NY 10016, USA

March 22, 2016

Contents

1 Introduction ................................................. 2
  1.1 Installation ........................................... 2
  1.2 Compile Binaries .................................... 2
  1.3 Setting up a new analysis ......................... 2
  1.4 Setting input files .................................. 3
  1.5 Create project sample sheet ..................... 3
  1.6 Running the Pipeline ............................... 3
  1.7 Dependencies ....................................... 5

2 Default Pipeline Components ......................... 6
  2.1 Parent Directory Overview ...................... 6
  2.2 Code Directories .................................. 7
  2.3 Data Directory .................................... 7
  2.4 Inputs Directory .................................. 7
  2.5 Pipeline Directory ................................. 8

3 Custom Pipeline Steps ................................ 12
  3.1 Overview ............................................ 12
  3.2 How To Add Steps ................................... 12

4 HiC-Seq Pipeline ........................................... 19
  4.1 Pipeline Steps ..................................... 19
  4.2 Alignment .......................................... 22
  4.3 Filter ............................................... 23
  4.4 Filter Stats ........................................ 24
  4.5 Tracks .............................................. 27
  4.6 Matrix Filtered ..................................... 29
  4.7 Matrix Prep ........................................ 31
  4.8 Matrix IC ........................................... 32
  4.9 Matrix HiCNorm .................................... 33
  4.10 Matrix Stats ....................................... 34
  4.11 Compare Matrices .................................. 36
  4.12 Compare Matrices Stats ......................... 38
  4.13 Boundary Scores ................................... 41
  4.14 Boundary Scores PCA ............................. 42
  4.15 Domains ............................................ 44
  4.16 Compare Boundaries ............................... 45
  4.17 Compare Boundaries Stats ..................... 46
  4.18 HiC Plotter ......................................... 49
  4.19 Interactions ....................................... 51
  4.20 Annotations ........................................ 53
  4.21 Annotations Stats .................................. 54
1 Introduction

1.1 Installation

The analysis pipeline can be installed by cloning its git repository from GitHub, located here: https://github.com/NYU-BFX/hic-bench. In the Terminal (OS X, Linux), run a command such as the following:

```bash
git clone https://github.com/NYU-BFX/hic-bench.git
```

Once a clone of the pipeline repository has been made, it will be used as a blank template to start future analysis; analysis is not performed directly in the pipeline repository.

1.2 Compile Binaries

Source code for needed binaries has been included in the repository, and must be compiled. Navigate to the code/src directory from within the Terminal, and run the command make to automatically run the compilation scripts needed. The program bedGraphToBigWig is also required, and available as a binary file from UCSC at their page here: http://hgdownload.cse.ucsc.edu/admin/exe/. To directly install a version compatible with the Linux operating system, navigate to the code/bin directory and run the command wget http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/bedGraphToBigWig.

1.3 Setting up a new analysis

Assuming your pipeline repository clone exists at ~/hic-bench, use the following terminal command to create a new analysis:

```bash
~/hic-bench/code/code.main/pipeline-new-analysis hicseq-standard <project_name>
```

This will create a new directory at the given location, and copy into it all the basic files and sub-directories needed for analysis from the pipeline repository.
### 1.4 Setting input files

Manual setup for the pipeline input files requires the creation of the directories `<project_name>/pipeline/input/fastq` or `<project_name>/pipeline/input/bam`, corresponding to the type of input files to be used. Sub-directories within these should be created with the name of each sample to be included in the analysis. A naming scheme similar to the following is suggested:

```
<Cell_line>--<treatment>--<SampleID>
```

Importantly, the '-' should be used as a delimiter, since this is recognized by the sample sheet creation script. Within each sub-directory, place all fastq / fastq.gz or bam files for the sample. Symlinks can be used if the files are not contained in the same location as the project analysis directory, and are preferable in order to save storage space. Since this part of the pipeline setup is custom for each analysis, it must be completed manually. A script used to automatically create the correct directories and symlinks might look like this:

```bash
#!/bin/bash
Fastq_dir="/data/sequence/results/smithlab/2016-01-28/fastq"
Inputs_dir="/home/$(whoami)/projects/SmithLab_HiC_2016-02-09/inputs/fastq"

# make inputs dir
mkdir -p "$Inputs_dir"

for i in $Fastq_dir/*.{fastq,fastq.gz}; do
    echo "$i"
    TMP_NAME=$(echo "$(basename "$i")" | sed -nr '/^[[:alnum:]][[:alpha:]]+$/p')
    echo "$TMP_NAME"
    mkdir -p "$Inputs_dir/$TMP_NAME"
    ln -s "$i" "$Inputs_dir/$TMP_NAME"
done
```

### 1.5 Create project sample sheet

A sample sheet must be created for the analysis project. After the inputs directory has been set up, the follow command can be used to automatically create a sample sheet template:

```
inputs$ ./code/create-sample-sheet.tclsh <genome> <fragment-size>
```

Where genome is hg19, hg38, etc. The fragment-size entry is optional and should be a numeric argument such as 300, representing the library size of the sequencing sample. After creation of the sample sheet (sample-sheet.tsv), a manual review process is required to match the correct control or input samples with experimental samples, verify proper grouping names, files, and other entries. If not entered prior, fragment-size should be filled in for each sample. This process can be completed within Microsoft Excel, but saving the file in Excel should be avoided due to the introduction of invisible formatting errors by Microsoft Office products. It is advisable to instead copy the finalized sheet from Excel and paste directly into a terminal text editor such as vi or nano for saving under the file name sample-sheet.tsv.

### 1.6 Running the Pipeline

After navigating to the parent directory of the analysis project, run the pipeline with:

```
./code/main/pipeline--execute PROJECT-NAME E-MAIL
```
Figure 1: Example sample sheet
1.7 Dependencies

This pipeline was developed for use in a High Performance Computing environment, running CentOS 6. Additionally, tcsh and bash shells are required, along with R version 3.2.0. The following includes software used in the HiC-Seq pipeline:

OGS/Grid Engine 2011.11
Linux 2.6.32−573.3.1.el6.x86_64 #1 SMP Thu Aug 13 22:55:16 UTC 2015 x86_64 GNU/Linux

tcsh 6.17.00 (Astron) 2009−07−10 (x86_64−unknown−linux)
GNU bash, version 4.1.2(1)−release (x86_64−redhat−linux−gnu)
armatus/2014−05−19
bedtools/2.22.0
bowtie2/2.2.6
caltdaS/0.1.0
gmm/0.9
java/1.7
matlab/R2013a
picard−tools
python/2.7.3
r/3.2.0
r/3.2.3
samtools/1.2.1

The following R packages are used in the pipeline:

plyr 1.8.1
VennDiagram 1.6.16
flsa 1.05
genlasso 1.3
ggplot2 1.0.1
optparse 1.3.0
pasteS 1.3−18
plotrix 3.5−11
reshape2 1.4.1
zoo 1.7−12
preprocessCore 1.24.0
MASS 7.3−35
gplots 2.17.0
reshape 0.8.5
corrplot 0.73
RColorBrewer 1.1−2
lattice 0.20−33
grid
stringr 1.0.0

For software information specific to the creation of this document, see Section 5.4
2 Default Pipeline Components

2.1 Parent Directory Overview

A default pipeline will have the following basic structure within its parent directory:

<table>
<thead>
<tr>
<th>Command</th>
<th>User</th>
<th>Date</th>
<th>Time</th>
<th>Command Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>hicseq_analysis</td>
<td>at570</td>
<td>29 Feb 14</td>
<td>19:28</td>
<td>__01a-align -&gt; pipeline/align</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>14 Feb 14</td>
<td>19:28</td>
<td>__02a-filter -&gt; pipeline/filter</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>15 Feb 14</td>
<td>19:28</td>
<td>__02b-filter --stats -&gt; pipeline/filter --stats</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>15 Feb 14</td>
<td>19:28</td>
<td>__03a-tracks -&gt; pipeline/tracks</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>20 Feb 14</td>
<td>19:28</td>
<td>__04a-matrix--filtered -&gt; pipeline/matrix--filtered</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>20 Feb 14</td>
<td>19:28</td>
<td>__05a-matrix--prep -&gt; pipeline/matrix--prep</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>18 Feb 14</td>
<td>19:28</td>
<td>__06a-matrix--ic -&gt; pipeline/matrix--ic</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>23 Feb 14</td>
<td>19:28</td>
<td>__07a-matrix--hicnorm -&gt; pipeline/matrix--hicnorm</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>21 Feb 14</td>
<td>19:28</td>
<td>__08a-matrix--stats -&gt; pipeline/matrix--stats</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>25 Feb 14</td>
<td>19:28</td>
<td>__09a-compare--matrices -&gt; pipeline/compare--matrices</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>31 Feb 14</td>
<td>19:28</td>
<td>__09b-compare--matrices--stats -&gt; pipeline/compare--matrices--stats</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>24 Feb 14</td>
<td>19:28</td>
<td>__10a-boundary--scores -&gt; pipeline/boundary--scores</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>28 Feb 14</td>
<td>19:28</td>
<td>__10b-boundary--scores--pca -&gt; pipeline/boundary--scores--pca</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>16 Feb 14</td>
<td>19:28</td>
<td>__11a-domains -&gt; pipeline/domains</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>27 Feb 14</td>
<td>19:28</td>
<td>__12a-compare--boundaries -&gt; pipeline/compare--boundaries</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>33 Feb 14</td>
<td>19:28</td>
<td>__12b-compare--boundaries--stats -&gt; pipeline/compare--boundaries--stats</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>19 Feb 14</td>
<td>19:28</td>
<td>__13a-hicplotter -&gt; pipeline/hicplotter</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>21 Feb 14</td>
<td>19:28</td>
<td>__14a--interactions -&gt; pipeline/interactions</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>20 Feb 14</td>
<td>19:28</td>
<td>__15a-annotations -&gt; pipeline/annotations</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>28 Feb 14</td>
<td>19:28</td>
<td>__15b-annotations--stats -&gt; pipeline/annotations--stats</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>30 Feb 18</td>
<td>11:16</td>
<td>code -&gt; code.repo/code.hicseq-standard</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>14 Nov 12</td>
<td>11:55</td>
<td>code.main -&gt; code/code.main</td>
</tr>
<tr>
<td>drwxr-xr-x</td>
<td>10 at570</td>
<td>238 Feb 15</td>
<td>19:53</td>
<td>code.repo</td>
</tr>
<tr>
<td>lrwxrwxrwx</td>
<td>1 at570</td>
<td>36 Mar 10</td>
<td>16:30</td>
<td>data -&gt; /ifs/home/at570/pipeline--master/data</td>
</tr>
<tr>
<td>drwx-r-xr-x</td>
<td>5 at570</td>
<td>230 Jan 5</td>
<td>09:24</td>
<td>inputs</td>
</tr>
<tr>
<td>drwx-r-xr-x</td>
<td>25 at570</td>
<td>834 Feb 14</td>
<td>19:28</td>
<td>pipeline</td>
</tr>
<tr>
<td>-rw-r-xr-x</td>
<td>1 at570</td>
<td>981 Jan 5</td>
<td>19:40</td>
<td>run</td>
</tr>
<tr>
<td>-rw-r-xr-x</td>
<td>1 at570</td>
<td>554 Dec 18</td>
<td>17:02</td>
<td>run.dry</td>
</tr>
</tbody>
</table>

The following components can be seen here:

- __01a-align ... __15b-annotations-stats: Symlinks to each step in the pipeline, in alphanumeric order of execution.
- code: Symlink to the directory containing scripts and code specific to the current analysis type e.g. ChIP-Seq.
- code.main: Symlink to the directory containing scripts and code used for all pipelines.
- code.repo: Directory containing all code for the project, copied from the main pipeline repository.
- data: Symlink to a directory containing reference genome data; set this in your original repository clone.
- inputs: Directory containing information on the files used as inputs.
- pipeline: Directory containing the files needed for each step in the pipeline.
- project_notes: A bare directory in which you can place miscellaneous notes and documents concerning the analysis.
- run: File containing code for running the pipeline.
- run.dry: File containing code for testing the pipeline without execution of pipeline steps.
2.2 Code Directories

The code needed for the execution of the analysis pipeline is divided among several sub-directories, based on usage. Within an analysis pipeline, the directory `code.repo` contains all of these sub-directories.

- `bin`: A directory containing symlinks to binary files for programs used by the pipeline.
- `code.chipseq-standard`, `code.hicseq-standard`: Directories containing scripts specific to the execution of each step in the given type of pipeline analysis.
- `code.main`: A directory containing code and scripts used for all analysis pipelines.

2.3 Data Directory

The reference genome information needed for analysis is contained in the `data` directory. This can be contained in an external location and symlinked to the project directory if it has not already been set in the cloned HiC-bench repository template. Our example `data` contains only the subdirectory `genomes`, which is configured as such:

Also included are indexes for bowtie2, which can be obtained from bowtie-bio.sourceforge.net/bowtie2/manual.shtml or http://support.illumina.com/sequencing/sequencing_software/igenome.html.

2.4 Inputs Directory

The `inputs` directory contains files needed to run the pipeline.
• README: File containing usage notes for the inputs directory.
• code: Symlink to code one level up in the parent directory.
• data: Symlink to data one level up in the parent directory.
• fastq: Directory containing sub-directories for each sample to be used in the analysis. This directory is not created automatically, it must be created and populated manually. Alternatively, the directory bam can be used in its place if .bam files are to be used.
• genomes: Symlink to the directory containing reference genome information, within the data directory.
• params: Directory containing the parameters files associated with the input files.
• sample-sheet.tsv: Sample sheet for pipeline execution.

2.4.1 FASTQ Directory
The contents of an example fastq directory can be seen here:

```
1  hicseq . analysis−for−hicbench/inputs/fastq$
  lrwxrwxrwx 1 at570 97 Feb 12 15:43 CD34−HindIII−rep1
2  lrwxrwxrwx 1 at570 95 Feb 12 15:43 GM−HindIII−rep1
3  lrwxrwxrwx 1 at570 92 Feb 12 15:43 GM−NcoI−rep1
4  lrwxrwxrwx 1 at570 103 Feb 12 15:43 H1−HindIII−Ren2015_rep1
5  lrwxrwxrwx 1 at570 103 Feb 12 15:43 H1−HindIII−Ren2015_rep2
6  lrwxrwxrwx 1 at570 95 Feb 12 15:43 H1−HindIII−rep1
7  lrwxrwxrwx 1 at570 95 Feb 12 15:43 H1−HindIII−rep2
8  lrwxrwxrwx 1 at570 98 Feb 12 15:43 IMR90−HindIII−rep1
9  lrwxrwxrwx 1 at570 98 Feb 12 15:43 IMR90−HindIII−rep2
10 lrwxrwxrwx 1 at570 103 Feb 12 15:43 T47D_T0−HindIII−rep1
11 lrwxrwxrwx 1 at570 97 Feb 12 15:43 T47D_T0−NcoI−rep1
12 lrwxrwxrwx 1 at570 101 Feb 12 15:43 T47D_T60−HindIII−rep1
13 lrwxrwxrwx 1 at570 98 Feb 12 15:43 T47D_T60−NcoI−rep1
14 lrwxrwxrwx 1 at570 100 Feb 12 15:43 mESC_J1−HindIII−rep1
15 lrwxrwxrwx 1 at570 100 Feb 12 15:43 mESC_J1−HindIII−rep2
16 lrwxrwxrwx 1 at570 97 Feb 12 15:43 mESC_J1−NcoI−rep1
17 lrwxrwxrwx 1 at570 97 Feb 12 15:43 mESC_J1−NcoI−rep2
```

Each directory name contains information about the sample, in the format <CellLine>-<treatment>-<SampleID>. This format can be modified to suit your purposes, though it is recommended to retain the "-" character as a delimiter since it is used downstream in the sample sheet generation steps. Each directory should contain all of the .fastq / .fastq.gz files associated with the sample; symlinks pointing to each file can be used as well, and are encouraged in order to save disk space. The same protocol should be followed if .bam files are to be used. As per standard Linux Terminal guidelines, spaces and special characters should be avoided in file names and directory names.

2.5 Pipeline Directory
The pipeline directory contains information for each step in the pipeline. An example pipeline directory will have the following structure:

```
1  hicseq . analysis−for−hicbench/pipeline$
  drwxr−xr−x 5 at570 226 Feb 15 16:47 align
2  drwxr−xr−x 5 at570 207 Jan 19 22:12 annotations
3  drwxr−xr−x 5 at570 213 Feb 16 17:24 annotations−stats
4  drwxr−xr−x 5 at570 211 Feb 6 17:46 boundary−scores
5  drwxr−xr−x 5 at570 389 Mar 10 18:16 boundary−scores−pca
6  lrwxrwxrwx 1 at570 7 Dec 2 12:39 code → ../code
7  lrwxrwxrwx 1 at570 12 Dec 2 12:39 code . main → ../code . main
8  drwxr−xr−x 5 at570 385 Jan 19 22:08 compare−boundaries
9  drwxr−xr−x 5 at570 437 Mar 10 18:16 compare−boundaries−stats
```
• align ... qc: Directories containing the information for each pipeline step.
  • code: Symlink to the directory containing code specific to current analysis type.
  • code.main: Symlink to the directory containing code used for all analyses.
  • inputs: Symlink to the inputs directory containing the .fastq or .bam files for the pipeline.
  • index.txt: A text file containing a list of pipeline steps to be executed. Entries in this document match the names of the pipeline directories.

2.5.1 Pipeline Index

The file index.txt contains a list of the pipeline steps to be completed during the analysis, listed in order of completion. An example index.txt would have the following structure:

```
hicseq.analysis—for—hicbench$ cat index.txt
hicseq.analysis—for—hicbench/pipeline$ cat index.txt
align
  filter
    filter—stats
  tracks
    matrix—filtered
  matrix—prep
  matrix—ic
  matrix—hicnorm
  #matrix—estimated
  #
  matrix—stats
  compare—matrices
  compare—matrices—stats
  boundary—scores
  boundary—scores—pca
  domains
  compare—boundaries
  compare—boundaries—stats
  #diff—domains
  #
  hicplotter
  interactions
```
Each entry in the index.txt file matches the name of the pipeline step to be completed, represented by the corresponding name of the step’s sub-directory in the pipeline directory. One entry is allowed per line in the index.txt file. Entries that begin with a ‘#’ character will be ignored, and pipeline steps that are not included in the index.txt file will not be included in the analysis pipeline.

### 2.5.2 Example Pipeline Step Directory Structure

Each step in the pipeline is represented by a sub-directory in the pipeline directory. An example sub-directory for a pipeline step would have the following structure:

```
$ tree

hicseq.analysis-for-hicbench/pipeline/align$
  lrwxrwxrwx 1 at570 15 Oct 28 12:10 clean.tcsh -> code/clean.tcsh
  lrwxrwxrwx 1 at570 7 Sep 29 13:31 code -> ../code
  drwxr-xr-x 2 at570 24 Feb 15 16:47 inpdirs
  drwxr-xr-x 2 at570 9 Sep 29 13:31 inputs -> ../inputs
  drwxr-xr-x 3 at570 62 Feb 16 12:27 results
  lrwxrwxrwx 1 at570 14 Jan 19 15:50 run -> run-align.tcsh
  lrwxrwxrwx 1 at570 971 Jan 19 15:49 run-align.tcsh
```

- **clean.tcsh**: Script for cleaning the directory; remove results and error logs.
- **code**: Symlink to the directory containing code specific to the analysis type e.g. code.chipseq-standard in this case.
- **error.log**: File containing errors encountered during execution of the pipeline step, generated at runtime.
- **inpdirs**: Directory containing symlinks to directories containing input files for use during execution of the pipeline step.
- **inputs**: Symlink to the directory containing input files.
- **params**: Directory containing the parameters files associated with the pipeline step files.
- **run**: Symlink to the ‘run’ file for the pipeline step.
- **run-align.tcsh**: ‘Run’ file for the pipeline step, containing a script that passes pipeline execution information to the wrapper script located in ./code/code.main/pipeline-master-explorer.r.

### 2.5.3 Example Pipeline Step Results Directory

The base level of a results directory for a pipeline step will have the following structure:

```
$ tree

hicseq.analysis-for-hicbench/pipeline/align/results/align.by_sample.bowtie2/CD34-HindIII-rep1$
  lrwxr-xr-x 1 at570 49G Jan 13 01:02 alignments.bam
  lrwxr-xr-x 1 at570 473 Jan 13 01:02 job.id
  lrwxr-xr-x 1 at570 47 Jan 12 18:42 job.err
  lrwxr-xr-x 1 at570 0 Jan 12 18:42 job.out
  lrwxr-xr-x 1 at570 136 Jan 12 18:42 job.sh
  lrwxr-xr-x 1 at570 2.3K Jan 13 01:02 job.vars.tsv
```

- **alignments.bam**: Example alignment output file.
- **job.err**: File containing the standard error output of the pipeline step.
• job.id: File containing the ID number of the job after submission for execution on the HPC cluster.
• job.out: File containing the standard output of the pipeline step.
• job.sh: File containing the command submitted for execution on the HPC cluster.
• job.vars.tsv: File containing the variables used in the completion of the pipeline step.
3 Adding Custom Pipeline Steps

3.1 Custom Pipeline Step Overview

The following basic steps should be taken to create a custom pipeline step:

- Copy an existing step as a template
- Update the new pipeline step name and add it to the entries in the `index.txt` and as a symlink in the parent level of the analysis directory
- Set the input directories (`inpdirs`)
- Edit the 'run' file and add needed parameter files
- Add a script in the `code` directory containing the commands needed to run the programs used in the pipeline step

3.2 How To Add Custom Pipeline Steps

The steps needed to create a custom pipeline step are explained in detail here:

1. Within the `pipeline` directory, use a command such as `cp -r` to make a copy of an existing pipeline step as a template for the new one.

Example pipeline directory:

```
$ ls -l
total 818K
-rwxr-xr-x 2 at570 at570 78  Mar  8 17:14 .
-rwxr-xr-x 1 at570 at570 834 Mar  8 17:14 ..
drwxr-xr-x 5 at570 at570  98 Mar 21 19:09 annotations
-rw-r--r-- 1 at570 at570 204 Mar 21 19:40 annotations-stats
-rw-r--r-- 1 at570 at570 204 Mar 21 20:44 boundaries
-rw-r--r-- 1 at570 at570 242 Mar 21 16:37 boundary-scores
-rw-rwxrwx 1 at570 at570    0 Mar  9 13:34 code
lrwxrwxrwx 1 at570 at570    0 Mar  9 13:34 code > . . / code
-rw-rwxrwx 1 at570 at570  12 Mar 10 16:20 code.main --> ../code.main
drwxr-xr-x 5 at570 at570 241 Mar 21 16:37 compare-boundaries
-rw-r--r-- 1 at570 at570 247 Mar 21 16:37 compare-boundaries-stats
drwxr-xr-x 5 at570 at570 239 Mar 18 20:20 compare-matrices
-rw-r--r-- 1 at570 at570 245 Mar 21 16:37 compare-matrices-stats
drwxr-xr-x 5 at570 at570 245 Mar 21 16:37 diff-domains
-rw-rwxrwx 1 at570 at570  20 Mar 18 21:22 domains
drwxr-xr-x 5 at570 at570 229 Mar 10 16:20 filter
-rw-r--r-- 1 at570 at570 229 Mar 10 16:20 filter-stats
drwxr-xr-x 7 at570 at570 323 Mar 21 16:41 hicplotter
-rw-r--r-- 1 at570 at570 331 Mar 10 16:20 index.txt
lrwxrwxrwx 1 at570 at570  20 Mar 10 16:20 inputs --> ../inputs
-rw-rwxrwx 1 at570 at570 335 Mar 10 16:20 interactions
-rw-rwxrwx 1 at570 at570  187 Mar 21 16:37 matrix-estimated
-rw-rwxrwx 1 at570 at570 238 Mar 21 16:20 matrix-filtered
-rw-rwxrwx 1 at570 at570 260 Mar 10 16:20 matrix-hicnorm
-rw-rwxrwx 1 at570 at570  98 Mar 18 17:31 matrix-ic
-rw-rwxrwx 1 at570 at570 234 Mar 21 16:37 matrix-prep
-rw-rwxrwx 1 at570 at570 235 Mar 21 16:37 matrix-stats
lrwxrwxrwx 1 at570 at570  40 Dec  8 14:56 psync --> ../psync
-rw-rwxrwx 1 at570 at570  48 Mar 20 16:20 template
-rw-rwxrwx 1 at570 at570 229 Mar 10 16:20 tracks
```

2. Adjust the name of the new directory to match the desired name of the new pipeline step. Add this name as an entry in the `index.txt` file, and make a symlink to this directory from the parent directory in the same style of the existing symlinks to other pipeline steps. The alpha-numeric prefix on the symlink will determine the order in which it will be executed. The command to do this might look like this:
Example index.txt file contents:

```bash
> hicseq.analysis --for hicbench$ ls -l
  hiwixrwxwx 1 at570 at570 14 Mar 10 16:20 __01a-align -> pipeline/align
  hiwixrwx wx 1 at570 at570 15 Mar 10 16:20 __02a-filter -> pipeline/filter
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __02b-filter-stats -> pipeline/filter-stats
  hiwixrwx wx 1 at570 at570 15 Mar 10 16:20 __03a-tracks -> pipeline/tracks
  hiwixrwx wx 1 at570 at570 24 Mar 10 16:20 __04a-matrix-filtered -> pipeline/matrix-filtered
  hiwixrwx wx 1 at570 at570 20 Mar 10 16:20 __05a-matrix-prep -> pipeline/matrix-prep
  hiwixrwx wx 1 at570 at570 18 Mar 10 16:20 __06a-matrix-ic -> pipeline/matrix-ic
  hiwixrwx wx 1 at570 at570 23 Mar 10 16:20 __07a-matrix-hicnorm -> pipeline/matrix-hicnorm
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __08a-matrix-stats -> pipeline/matrix-stats
  hiwixrwx wx 1 at570 at570 25 Mar 10 16:20 __09a-compare-matrices -> pipeline/compare-matrices
  hiwixrwx wx 1 at570 at570 31 Mar 10 16:20 __09b-compare-matrices-stats -> pipeline/compare-matrices-stats
  hiwixrwx wx 1 at570 at570 24 Mar 10 16:20 __10a-boundary-scores -> pipeline/boundary-scores
  hiwixrwx wx 1 at570 at570 28 Mar 10 16:20 __10b-boundary-scores-pca -> pipeline/boundary-scores-pca
  hiwixrwx wx 1 at570 at570 16 Mar 10 16:20 __11a-domains -> pipeline/domains
  hiwixrwx wx 1 at570 at570 27 Mar 10 16:20 __12a-compare-boundaries -> pipeline/compare-boundaries
  hiwixrwx wx 1 at570 at570 33 Mar 10 16:20 __12b-compare-boundaries-stats -> pipeline/compare-boundaries-stats
  hiwixrwx wx 1 at570 at570 19 Mar 10 16:20 __13a-hicplotter -> pipeline/hicplotter
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __14a-interactions -> pipeline/interactions
  hiwixrwx wx 1 at570 at570 20 Mar 10 16:20 __15a-annotations -> pipeline/annotations
  hiwixrwx wx 1 at570 at570 26 Mar 10 16:20 __15b-annotations-stats -> pipeline/annotations-stats
  hiwixrwx wx 1 at570 at570 30 Mar 14 18:25 code -> code.repo/code.hicseq-standard
  hiwixrwx wx 1 at570 at570 14 Mar 10 16:20 code.main -> code/code.main
  hiwixrwx wx 10 at570 at570 238 Mar 13 21:45 code.repo
  hiwixrwx wx 1 at570 at570 104 Mar 14 18:25 data -> /ifs/home/at570/disk1/Resources/Code/pipeline-master/code/code.main/.../pipelines/hicseq-standard/data
  drwxr-xr-x 5 at570 at570 274 Mar 10 16:20 inputs
```

Example parent directory structure:

```bash
> hicseq.analysis --for hicbench$ ls -l
  hiwixrwxwx 1 at570 at570 14 Mar 10 16:20 __01a-align -> pipeline/align
  hiwixrwx wx 1 at570 at570 15 Mar 10 16:20 __02a-filter -> pipeline/filter
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __02b-filter-stats -> pipeline/filter-stats
  hiwixrwx wx 1 at570 at570 15 Mar 10 16:20 __03a-tracks -> pipeline/tracks
  hiwixrwx wx 1 at570 at570 24 Mar 10 16:20 __04a-matrix-filtered -> pipeline/matrix-filtered
  hiwixrwx wx 1 at570 at570 20 Mar 10 16:20 __05a-matrix-prep -> pipeline/matrix-prep
  hiwixrwx wx 1 at570 at570 18 Mar 10 16:20 __06a-matrix-ic -> pipeline/matrix-ic
  hiwixrwx wx 1 at570 at570 23 Mar 10 16:20 __07a-matrix-hicnorm -> pipeline/matrix-hicnorm
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __08a-matrix-stats -> pipeline/matrix-stats
  hiwixrwx wx 1 at570 at570 25 Mar 10 16:20 __09a-compare-matrices -> pipeline/compare-matrices
  hiwixrwx wx 1 at570 at570 31 Mar 10 16:20 __09b-compare-matrices-stats -> pipeline/compare-matrices-stats
  hiwixrwx wx 1 at570 at570 24 Mar 10 16:20 __10a-boundary-scores -> pipeline/boundary-scores
  hiwixrwx wx 1 at570 at570 28 Mar 10 16:20 __10b-boundary-scores-pca -> pipeline/boundary-scores-pca
  hiwixrwx wx 1 at570 at570 16 Mar 10 16:20 __11a-domains -> pipeline/domains
  hiwixrwx wx 1 at570 at570 27 Mar 10 16:20 __12a-compare-boundaries -> pipeline/compare-boundaries
  hiwixrwx wx 1 at570 at570 33 Mar 10 16:20 __12b-compare-boundaries-stats -> pipeline/compare-boundaries-stats
  hiwixrwx wx 1 at570 at570 19 Mar 10 16:20 __13a-hicplotter -> pipeline/hicplotter
  hiwixrwx wx 1 at570 at570 21 Mar 10 16:20 __14a-interactions -> pipeline/interactions
  hiwixrwx wx 1 at570 at570 20 Mar 10 16:20 __15a-annotations -> pipeline/annotations
  hiwixrwx wx 1 at570 at570 26 Mar 10 16:20 __15b-annotations-stats -> pipeline/annotations-stats
  hiwixrwx wx 1 at570 at570 30 Mar 14 18:25 code -> code.repo/code.hicseq-standard
  hiwixrwx wx 1 at570 at570 14 Mar 10 16:20 code.main -> code/code.main
  hiwixrwx wx 10 at570 at570 238 Mar 13 21:45 code.repo
  hiwixrwx wx 1 at570 at570 104 Mar 14 18:25 data -> /ifs/home/at570/disk1/Resources/Code/pipeline-master/code/code.main/.../pipelines/hicseq-standard/data
  drwxr-xr-x 5 at570 at570 274 Mar 10 16:20 inputs
```
3. Edit the contents of the directory you have created to hold the information for your new pipeline step.

Example pipeline step directory:

```
$ ls

rwxr-xr-x 1 at570 at570 25 2011 00:10 pipeline
-rwxr-xr-x 1 at570 at570 834 2011 17:14 pipeline
```

First, edit the `run` file. A sample `run` file looks like this:

```
#!/bin/tcsh
source ./code/code.main/custom-tcsrhrc # customize shell environment
#
## USAGE: run-domains.tcsh [---dry-run]
#
# This section holds information that will be used in future updates of the software for reporting
## This step identifies topologically-associated domains (TADs) using different methods.
## TABLES:
## FIGURES:
# process command-line inputs
# check to make sure that the proper number of arguments have been passed to the script,
# if not then print the script lines starting with '##' and exit
if ( $#argv > 1) then
  grep "##" $0 | scripts-send2err
  exit
endif

set opt = "$1"

# setup
# set the 'operation' to be performed, aka name of the pipeline step
set op = domains
# the directories to be used for inputs
set inpdirs = "inpdirs/"
# an expression which specified which input branches to include
set filter = "*.res_40kb" # work only with 40kb resolution
# the name of the results directory
set results = results
# create the results directory
scripts-create-path $results/
# sends a message to the error logging script
scripts-send2err "### Operation = $op "
# 'resources' argument to be passed to qsub, referring to CPU cores and GB of RAM to be reserved for the job
set resources = 1,20G
# command to be passed to the 'pipeline-master-explorer.r' script
set cmd = "/code/code.main/scripts-qsub-wrapper $resources ./hicseq-$op.tcsh"

# generate run script
# the 'pipeline-master-explorer.r' script parses the items set above to create a line of text containing
# the commands to be submitted to qsub
Rscript ./code/code.main/pipeline-master-explorer.r -v -F "$filter" "$cmd" $results/$op "params/params...tcsh" "$inpdirs" "sample" 1

# run and wait until done!
if ("$opt" != "---dry-run") scripts-submit-jobs $results/.db/run
```

As listed in the above run file, the following 'setup' items need to be set for the custom pipeline step:

- set op = <name_of_pipeline_step>

The 'operation' to be performed is set as 'op' and should be the name of the pipeline step, as listed in the directory name and in the index.txt file.

- set inpdirs = "inpdirs/*"

The 'inpdirs', or input directories, should be set as the file path to the directory containing symlinks to the input directories. In this case, the contents of inpdirs is as follows:

```
$ ls -l inpdirs/
```

The setting "inpdirs/*" will cause all input directories to be used. The entries in the inpdirs directory should be set as needed for the execution of the custom pipeline step.

- set filter = "*.res_40kb"

The 'filter' setting to be used when parsing the 'branches' of the input directory results, for inclusion in the execution of the pipeline step. In this example, only input branches that match the pattern "*.res_40kb" will be included. In this example, the following input branches are available:

```
$ ls -l inpdirs/ matrix-filtered/results/
```

Based on the given 'filter' setting, only the following branch will be included:

```
$ ls -l inpdirs/ matrix-filtered/results/
```

This allows for the exclusion of unnecessary analysis branches.

- set resources = 1,20G

This sets the number of computer resources to be reserved by qsub, listed as CPU cores and GB of RAM. If RAM is not a concern, only CPU cores need to be listed. A range of values can be used for CPU cores, such as 8-64, though the utility of this depends on many factors related to your high-performance computing infrastructure and the specifics of the program being run; more cores may not necessarily speed up execution of the task at hand.
This line does not need to be modified by the user, but should be noted since it refers to the file in the code directory that will be created later and used to execute the program used in the pipeline step. Importantly, the entry `./code/hicseq-$op.tcsh` in this case refers to the file `./code/hicseq-domains.tcsh`.

Since this calls the settings that have already been made, this line of the 'run' file does not need to be edited unless the grouping and splitting variables need to be changed. In this case, the command uses the following arguments (as per Section 5.3):

Importantly, the 'split-variable' and 'output-object-variable' come from the headings of columns used as grouping factors in the inputs/sample-sheet.tsv file for the analysis; custom grouping factors can be included in the sample sheet and used here. The output of the `pipeline-master-explorer.r` script is stored in the file `results/.db/run` which is created when the run file is executed (the command `./run--dry-run` can be used to generate this without running the commands). An example entry will look like this:

During pipeline step execution, these lines will be submitted to `qsub` by the script `./code/code.main/scripts-qsub-wrapper`.

4. Next, the parameter files must be set for the new pipeline step. These files are contained in the params directory:

Importantly, files must use the following naming scheme: `params.<name>.tcsh`. All files included in the params directory following this naming scheme will be evaluated as a separate 'branch' for analysis. An example parameters file looks like this:
5. A script containing the commands needed to run the desired program must be created and placed in the `results` directory for each entry in the `params` directory, as can be seen here:

```bash
import sys
import os

if (sys.argv[1] == ' hicseq- domains.tabs')
    # Run domains
    source ./code/hicseq-domains-$tool.$branch $outdir $params $branch "$objects"
```
The preamble of the script should require little user intervention, while the bulk of the user's custom pipeline code should be inserted between the 'MAIN CODE' blocks specified within the document. For reference on how to structure your custom code, compare the 'USAGE' entry with the evaluated command to be passed to the script in the results/.db/run file. For convenience, the sample entry is repeated below:

```
hicseq.analysis --for=--hicbench/pipeline/domains$ head -n 1 results/.db/run
./code/code_main/scripts--qub--wrapper 1,20G ./code/hicseq--domains.tcsh results/domains_by_sample.armatus.gamma_0.5/matrix-prep_by_sample_scale/matrix--filtered_by_sample.res_40kb/filter_by_sample.standard/align_by_sample_bowtie2/CD34-HindIII-rep1 params/params.armatus.gamma_0.5.tcsh indirs/matrix--prep/results/matrix-prep_by_sample_scale/matrix--filtered_by_sample.res_40kb/filter_by_sample.standard/align_by_sample_bowtie2 'CD34-HindIII-rep1'
```

While this primary script should be in the .tcsh format, subsequent scripts in the user's preferred language can be called. They should follow the same naming conventions as shown in the code directory example above.
4 HiC-Seq Pipeline

4.1 Pipeline Steps

Within the parent directory of an analysis, the default pipeline steps are listed as symlinks, in alpha-numeric order starting with "__", as seen here:

```
lrwxrwxrwx 1 at570 165 Dec 26 08:09 run . usage
lrwxrwxrwx 1 at570 210 Dec 2 14:23 psync

This functions in informing the user of the order of pipeline steps. Each symlink points back to a directory in the pipeline directory for the corresponding pipeline step, as shown here:
```

```
pipeline$
```

```
pipeline$ 4 at570 203 Jan 19 15:50 align
```

The pipeline directory contains files and symlinks needed for each step in the pipeline. The steps to be executed are defined in two ways:
1. A file called 'index.txt' lists the names of each step in the pipeline, in the order in which they will be completed. This file is located in the 'pipeline' directory.

2. A subdirectory within the 'pipeline' directory with the same name as its corresponding entry in the 'index.txt' file must be included to hold the parameters and commands to be run, and the results produced.

Index file:

```
pipeline/index.txt
align
filter
filter−stats
tracks
matrix−filtered
matrix−prep
matrix−ic
#matrix−estimated
# matrix−stats
compare−matrices
compare−matrices−stats
boundary−scores
boundary−scores−pca
domains
compare−boundaries
compare−boundaries−stats
#diff−domains
# hicplotlib
interactions
annotations
```

Steps listed in 'index.txt' which have been commented out (i.e. start with a # character) will not be included in the analysis. Custom pipeline steps can be easily included by adding the corresponding entry to the 'index.txt' and creating a subdirectory within the 'pipeline' directory. For details on adding custom pipeline steps, see Section ??

4.1.1 Default Parameters

These parameters are used by default across pipeline steps.

```
#!/inputs/params/params.tcsh

# load basic tools
module unload samtools
module unload java
module unload gcc
module unload python
module load samtools/1.2.1
module load bedtools/2.22.0
module load java/1.7
module load picard−tools

# load tools required for each step of the pipeline (this can be overridden in local param scripts)
module load bowtie2/2.2.6
module load armatus/2014−06−19
module load caltads/0.1.0
```
4.1.2 Pipeline Step Execution Flowchart

![Pipeline Step Execution Flowchart](image)

**Figure 2:** Overview of pipeline step execution for default analysis steps. See Section 4.1.2.
4.2 Alignment

4.2.1 Input

Raw data in fastq or fastq.gz files (Section 2.4).

4.2.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
set aligner = bowtie2
set genome = `./code/read–sample-sheet.tcsh $sheet $object genome`
set genome_index = inputs/genomes/$genome/bowtie2.index/genome
set align_params = "--very-sensitive--local --local"
```

4.2.3 Output

Default output:

```
-rw-r--r--  1 at570  49G Jan 13 01:02 alignments.bam
-rw-r--r--  1 at570  473 Jan 13 01:02 job.err
-rw-r--r--  1 at570  47 Jan 12 18:42 job.id
-rw-r--r--  1 at570  0 Jan 12 18:42 job.out
-rw-r--r--  1 at570  136 Jan 12 18:42 job.sh
-rw-r--r--  1 at570  2.3K Jan 13 01:02 job.vars.tsv
```
4.3 Filter

4.3.1 Input

Data from the pipeline align step is used as input (Section 4.2).

4.3.2 Analysis

Default parameters:

```bash
params.standard.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set filter_params = "--mapq 30 --min-dist 25000 --max-offset 500 --filter-dups"
```

4.3.3 Output

Default output:

```
-rw-r--r--  1 at570  1.7G Jan 13 14:26 filtered.reg.gz
-rw-r--r--  1 at570  65K Jan 13 14:25 job.err
-rw-r--r--  1 at570   47 Jan 13 13:15 job.id
-rw-r--r--  1 at570    0 Jan 13 13:15 job.out
-rw-r--r--  1 at570  195 Jan 13 13:15 job.sh
-rw-r--r--  1 at570  2.1K Jan 13 14:26 job.vars.tsv
-rw-r--r--  1 at570   378 Jan 13 14:24 stats.tsv
```
4.4 Filter Stats

4.4.1 Input

Data from the pipeline filter step is used as input (Section 4.3).

4.4.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
```

4.4.3 Output

See Figure 3 and Figure 4. Default output:

```
-rw-r--r-- 1 at570 6.5K Feb 11 15:27 counts.pdf
-rw-r--r-- 1 at570  34 Feb 11 15:27 job.err
-rw-r--r-- 1 at570  47 Feb 11 15:27 job.id
-rw-r--r-- 1 at570  52 Feb 11 15:27 job.out
-rw-r--r-- 1 at570 226 Feb 11 15:27 job.sh
-rw-r--r-- 1 at570  6.7K Feb 11 15:27 percent.pdf
```
Figure 3: Filter Stats counts sample output
Figure 4: Filter Stats percentage sample output
4.5 Tracks

4.5.1 Input

Data from the pipeline filter step is used as input (Section 4.3).

4.5.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
set bin_size = 40000  # this is a commonly used bin size
```

4.5.3 Output

Default output:

```
-rw-r--r-- 1 at570  4.1K Jan 13 15:55 job.err
-rw-r--r-- 1 at570  47 Jan 13 15:10 job.id
-rw-r--r-- 1 at570  0 Jan 13 15:11 job.out
-rw-r--r-- 1 at570 242 Jan 13 15:10 job.sh
-rw-r--r-- 1 at570  2.6K Jan 13 15:55 job.vars.tsv
-rw-r--r-- 1 at570  1.1G Jan 13 15:54 track.washu.tsv.gz
-rw-r--r-- 1 at570  789K Jan 13 15:55 track.washu.tsv.gz.tbi
```
Figure 5: WashU tracks loaded in browser.
4.6 Matrix Filtered

4.6.1 Input

Data from the pipeline filter step is used as input (Section 4.3).

4.6.2 Analysis

Default parameters files:

```
- rwxr-x 1 at570 195 Nov 24 11:20 params.res_1000kb.tcsh
- rwxr-x 1 at570 194 Nov 25 15:11 params.res_1000kb.tcsh
- rwxr-x 1 at570 210 Nov 25 15:11 params.res_100kb.maxd_5Mb.rotate45.tcsh
- rwxr-x 1 at570 193 Nov 30 16:22 params.res_10kb.tcsh
- rwxr-x 1 at570 193 Nov 24 11:20 params.res_40kb.tcsh
```

Default parameters:

```
params.res_1000kb.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set bin_size = 1000000
set max_dist = 0
set ref = $genome_dir/genome.bed
set matrix_params = "--bin-size $bin_size --max-dist $max_dist -R $ref"
```

```
params.res_1000kb.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set bin_size = 1000000
set max_dist = 0
set ref = $genome_dir/genome.bed
set matrix_params = "--bin-size $bin_size --max-dist $max_dist -R $ref"
```

```
params.res_10kb.maxd_5Mb.rotate45.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set bin_size = 10000
set max_dist = 5000000
set ref = $genome_dir/genome.bed
set matrix_params = "--bin-size $bin_size --max-dist $max_dist --rotate45 -R $ref"
```

```
params.res_10kb.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set bin_size = 10000
set max_dist = 0
set ref = $genome_dir/genome.bed
set matrix_params = "--bin-size $bin_size --max-dist $max_dist -R $ref"
```

```
params.res_40kb.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
```
4.6.3 Output

Default output:

- rw-r--r-- 1 at570 56K Jan 13 15:57 ignored_loci.txt
- rw-r--r-- 1 at570 9.6K Jan 13 16:02 job.err
- rw-r--r-- 1 at570 47 Jan 13 15:57 job.id
- rw-r--r-- 1 at570 0 Jan 13 15:57 job.out
- rw-r--r-- 1 at570 266 Jan 13 15:57 job.sh
- rw-r--r-- 1 at570 2.7K Jan 13 16:02 job.vars.tsv
- rw-r--r-- 1 at570 73M Jan 13 16:01 matrix.chr1.tsv
- rw-r--r-- 1 at570 23M Jan 13 16:01 matrix.chr10.tsv
- rw-r--r-- 1 at570 22M Jan 13 16:01 matrix.chr11.tsv
- rw-r--r-- 1 at570 22M Jan 13 16:01 matrix.chr12.tsv
- rw-r--r-- 1 at570 16M Jan 13 16:01 matrix.chr13.tsv
- rw-r--r-- 1 at570 14M Jan 13 16:01 matrix.chr14.tsv
- rw-r--r-- 1 at570 13M Jan 13 16:01 matrix.chr15.tsv
- rw-r--r-- 1 at570 9.9M Jan 13 16:01 matrix.chr16.tsv
- rw-r--r-- 1 at570 8.0M Jan 13 16:01 matrix.chr17.tsv
- rw-r--r-- 1 at570 7.4M Jan 13 16:01 matrix.chr18.tsv
- rw-r--r-- 1 at570 4.3M Jan 13 16:01 matrix.chr19.tsv
- rw-r--r-- 1 at570 7.1M Jan 13 16:01 matrix.chr2.tsv
- rw-r--r-- 1 at570 4.9M Jan 13 16:01 matrix.chr20.tsv
- rw-r--r-- 1 at570 2.9M Jan 13 16:01 matrix.chr21.tsv
- rw-r--r-- 1 at570 3.3M Jan 13 16:01 matrix.chr22.tsv
- rw-r--r-- 1 at570 40M Jan 13 16:01 matrix.chr3.tsv
- rw-r--r-- 1 at570 44M Jan 13 16:01 matrix.chr4.tsv
- rw-r--r-- 1 at570 40M Jan 13 16:01 matrix.chr5.tsv
- rw-r--r-- 1 at570 30M Jan 13 16:02 matrix.chr6.tsv
- rw-r--r-- 1 at570 31M Jan 13 16:02 matrix.chr7.tsv
- rw-r--r-- 1 at570 26M Jan 13 16:02 matrix.chr8.tsv
- rw-r--r-- 1 at570 24M Jan 13 16:02 matrix.chr9.tsv
- rw-r--r-- 1 at570 29 Jan 13 16:02 matrix.chrM.tsv
- rw-r--r-- 1 at570 29M Jan 13 16:02 matrix.chrX.tsv
- rw-r--r-- 1 at570 4.3M Jan 13 16:02 matrix.chrY.tsv
4.7 Matrix Prep

4.7.1 Input

Data from the pipeline matrix-filtered step is used as input (Section 4.6).

4.7.2 Analysis

Default parameters:

```bash
# $/bin/tcsh
source ./inputs/params/params.tcsh

set chrom_excluded = 'chr[MY]' # excluded chromosomes
set prep_params = "−−scale−−impute"
```

4.7.3 Output

Default output:

```bash
drwxr-xr-x 2 at570 3.4K Jan 13 16:16 __jdata
−rw−r−r−− 1 at570 4.0K Jan 13 16:18 job.err
−rw−r−r−− 1 at570 47 Jan 13 16:14 job.id
−rw−r−r−− 1 at570 0 Jan 13 16:15 job.out
−rw−r−r−− 1 at570 345 Jan 13 16:14 job.sh
−rw−r−r−− 1 at570 3.3K Jan 13 16:18 job.vars.tsv
−rw−r−r−− 1 at570 371M Jan 13 16:17 matrix.chr1.tsv
−rw−r−r−− 1 at570 110M Jan 13 16:16 matrix.chr10.tsv
−rw−r−r−− 1 at570 109M Jan 13 16:15 matrix.chr11.tsv
−rw−r−r−− 1 at570 107M Jan 13 16:16 matrix.chr12.tsv
−rw−r−r−− 1 at570 80M Jan 13 16:16 matrix.chr13.tsv
−rw−r−r−− 1 at570 69M Jan 13 16:16 matrix.chr14.tsv
−rw−r−r−− 1 at570 63M Jan 13 16:16 matrix.chr15.tsv
−rw−r−r−− 1 at570 49M Jan 13 16:16 matrix.chr16.tsv
−rw−r−r−− 1 at570 40M Jan 13 16:16 matrix.chr17.tsv
−rw−r−r−− 1 at570 37M Jan 13 16:16 matrix.chr18.tsv
−rw−r−r−− 1 at570 21M Jan 13 16:16 matrix.chr19.tsv
−rw−r−r−− 1 at570 353M Jan 13 16:17 matrix.chr2.tsv
−rw−r−r−− 1 at570 24M Jan 13 16:16 matrix.chr20.tsv
−rw−r−r−− 1 at570 14M Jan 13 16:16 matrix.chr21.tsv
−rw−r−r−− 1 at570 234M Jan 13 16:17 matrix.chr22.tsv
−rw−r−r−− 1 at570 219M Jan 13 16:17 matrix.chr3.tsv
−rw−r−r−− 1 at570 196M Jan 13 16:17 matrix.chr4.tsv
−rw−r−r−− 1 at570 175M Jan 13 16:17 matrix.chr5.tsv
−rw−r−r−− 1 at570 152M Jan 13 16:17 matrix.chr6.tsv
−rw−r−r−− 1 at570 128M Jan 13 16:17 matrix.chr7.tsv
−rw−r−r−− 1 at570 120M Jan 13 16:17 matrix.chr8.tsv
−rw−r−r−− 1 at570 144M Jan 13 16:17 matrix.chr9.tsv
−rw−r−r−− 1 at570 234M Jan 13 16:17 matrix.chrX.tsv
```
4.8 Matrix IC

4.8.1 Input

Data from the pipeline matrix-filtered step is used as input (Section 4.6).

4.8.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source /inputs/params/params.tcsh
module unload gcc  # this is necessary in order to take care of module conflicts in our system
module unload python
module load python/2.7.3
set chrom_excluded = 'chr[MY]'  # excluded chromosomes
set cutoff = 0.05
```

4.8.3 Output

Default output:

```
```

Page 32 of 62
4.9 Matrix HiCNorm

4.9.1 Input

Data from the pipeline `matrix-filtered` step is used as input (Section 4.6).

4.9.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source /inputs/params/params.tcs
set chrom_excluded = 'chr [MY]'  # excluded chromosomes
```

4.9.3 Output

Default output:

```bash
drwxr-xr-x 2 at570 3.4K Feb 8 17:10 __jdata
-rw-r--r-- 1 at570 13K Feb 8 17:25 job.err
-rw-r--r-- 1 at570  47 Feb 8 17:06 job.id
-rw-r--r-- 1 at570  34 Feb 8 17:06 job.out
-rw-r--r-- 1 at570  86M Feb 8 17:19 matrix.chr1.tsv
-rw-r--r-- 1 at570  28M Feb 8 17:10 matrix.chr10.tsv
-rw-r--r-- 1 at570  28M Feb 8 17:12 matrix.chr11.tsv
-rw-r--r-- 1 at570  21M Feb 8 17:09 matrix.chr12.tsv
-rw-r--r-- 1 at570  18M Feb 8 17:09 matrix.chr13.tsv
-rw-r--r-- 1 at570  16M Feb 8 17:09 matrix.chr14.tsv
-rw-r--r-- 1 at570  10M Feb 8 17:09 matrix.chr15.tsv
-rw-r--r-- 1 at570   9M Feb 8 17:09 matrix.chr16.tsv
-rw-r--r-- 1 at570   7M Feb 8 17:09 matrix.chr17.tsv
-rw-r--r-- 1 at570   5M Feb 8 17:09 matrix.chr18.tsv
-rw-r--r-- 1 at570   5M Feb 8 17:09 matrix.chr19.tsv
-rw-r--r-- 1 at570  32M Feb 8 17:09 matrix.chr20.tsv
-rw-r--r-- 1 at570  3.6M Feb 8 17:09 matrix.chr21.tsv
-rw-r--r-- 1 at570  3.8M Feb 8 17:09 matrix.chr22.tsv
-rw-r--r-- 1 at570  55M Feb 8 17:17 matrix.chr3.tsv
-rw-r--r-- 1 at570  49M Feb 8 17:15 matrix.chr4.tsv
-rw-r--r-- 1 at570  45M Feb 8 17:15 matrix.chr5.tsv
-rw-r--r-- 1 at570  43M Feb 8 17:15 matrix.chr6.tsv
-rw-r--r-- 1 at570  39M Feb 8 17:15 matrix.chr7.tsv
-rw-r--r-- 1 at570  30M Feb 8 17:14 matrix.chr8.tsv
-rw-r--r-- 1 at570  29M Feb 8 17:13 matrix.chr9.tsv
-rw-r--r-- 1 at570   3M Feb 8 17:15 matrix.chrX.tsv
```
4.10 Matrix Stats

4.10.1 Input

Data from the pipeline steps matrix-filtered (Section 4.6), matrix-hicnorm (Section 4.9), matrix-prep (Section 4.7), and matrix-ic (Section 4.8) are used as input.

4.10.2 Analysis

Default parameters:

```
params.standard.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set chrom_excluded = 'chr[MY]' # excluded chromosomes
```

4.10.3 Output

See Figure 6. Default output:

```
-rw-r--r--  1 at570  39K Feb 11 16:11 job.err
-rw-r--r--  1 at570   47 Feb 11 15:48 job.id
-rw-r--r--  1 at570  86 Feb 11 15:48 job.out
-rw-r--r--  1 at570 480 Feb 11 15:48 job.sh
-rw-r--r--  1 at570  5.3K Feb 11 16:11 job.vars.tsv
-rw-r--r--  1 at570  59K Feb 11 16:11 stats.pdf
```
Figure 6: Matrix Stats sample output
4.11 Compare Matrices

4.11.1 Input

Data from the pipeline steps matrix-filtered (Section 4.6), matrix-hicnorm (Section 4.9), matrix-prep (Section 4.7), and matrix-ic (Section 4.8) are used as input.

4.11.2 Analysis

Default parameters:

```bash
# only used if estimation was done with max-lambda=inf
param standard.tcsh

source ./inputs/params/params.tcsh

set chrom_excluded = 'chr [MYX]'  # excluded chromosomes

set max_dist = `echo 10000000/$bin_size | bc`  # number of bins (max distance = 10Mb)

set compare_params = "--max-dist=$max_dist --dist=1 --min-lambda=0.0 --max-lambda=1.0 --n-lambda=6 --gamma=0"
```

4.11.3 Output

Default output:

```
-rw------- 1 at570 17 Feb 9 01:26 chr1.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:26 chr1.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:27 chr10.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:27 chr10.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:28 chr11.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:28 chr11.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:28 chr12.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:28 chr12.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:29 chr13.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:29 chr13.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:29 chr14.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:29 chr14.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:30 chr15.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:30 chr15.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:30 chr16.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:30 chr16.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr17.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr17.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr18.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr18.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr19.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:31 chr19.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr2.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr2.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr20.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr20.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr21.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr21.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr22.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:33 chr22.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:35 chr3.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:35 chr3.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:36 chr4.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:36 chr4.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:37 chr5.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:37 chr5.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:38 chr6.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:38 chr6.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:39 chr7.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:39 chr7.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:40 chr8.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:40 chr8.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:41 chr9.cor.pearson.tsv
-rw------- 1 at570 17 Feb 9 01:41 chr9.cor.spearman.tsv
-rw------- 1 at570 17 Feb 9 01:41 cor.pearson.tsv
```

4.12 Compare Matrices Stats

4.12.1 Input

Data from the pipeline compare-matrices step is used as input (Section 4.11).

4.12.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
```

4.12.3 Output

See Figure 7, and See Figure 8. Default output:

```
-rw-r---r-- 1 at570  97 Feb 12 11:25 task.err
-rw-r---r-- 1 at570  47 Feb 12 11:24 task.id
-rw-r---r-- 1 at570  52 Feb 12 11:25 task.out
-rw-r---r-- 1 at570  3.4K Feb 12 11:24 task.sh
-rw-r---r-- 1 at570  44K Feb 12 11:25 task.vars.tsv
drwxr-xr-x 2 at570  96 Feb 12 11:25 spearman
drwxr-xr-x 2 at570  97 Feb 12 11:25 spearman
```

```
spearman/
-rw-r---r-- 1 at570  161K Feb 12 11:25 cor.spearman.tsv
-rw-r---r-- 1 at570  8.9K Feb 12 11:25 correlograms.pdf
-rw-r---r-- 1 at570  12K Feb 12 11:25 summary.tsv
```

```
pearson/
-rw-r---r-- 1 at570  159K Feb 12 11:25 cor.pearson.tsv
-rw-r---r-- 1 at570  9.0K Feb 12 11:25 correlograms.pdf
-rw-r---r-- 1 at570  12K Feb 12 11:25 summary.tsv
```
**Figure 7**: Compare Matrices Stats Spearman sample correlograms. See Section 4.12.
Figure 8: Compare Matrices Stats Pearson sample correlograms. See Section 4.12.
4.13 Boundary Scores

4.13.1 Input

Data from the pipeline steps matrix-filtered (Section 4.6), matrix-hicnorm (Section 4.9), matrix-prep (Section 4.7), and matrix-ic (Section 4.8) are used as input.

4.13.2 Analysis

Default parameters:

```bash
#!/bin/tcsh

source ./inputs/params/params.tcsh

set chrom_excluded = 'chr[MYX]' # excluded chromosomes

set boundary_scores_params = ( 
    --min-lambda=0.0 --max-lambda=1.0 --n-lambda=6 --gamma=0 
    --distance=`echo 500000/$bin_size | bc` 
    --distance2=`echo 500000/$bin_size | bc` 
    --skip-distance=0 
    --flank-dist=`echo 500000/$bin_size | bc` 
    --tolerance=0.01 
    --alpha=0.50 
    --track-dist=`echo 2000000/$bin_size | bc` 
    --presentation=none )
```

4.13.3 Output

Default output:

```bash
-rw-r--r--  1 at570  9.0M Feb 15 14:25 all_scores.k=001.tsv
-rw-r--r--  1 at570 17K Feb 15 14:25 job.err
-rw-r--r--  1 at570  47 Feb 15 14:07 job.id
-rw-r--r--  1 at570   0 Feb 15 14:11 job.out
-rw-r--r--  1 at570  345 Feb 15 14:07 job.sh
-rw-r--r--  1 at570  3.2K Feb 15 14:25 job.vars.tsv
```
4.14 Boundary Scores PCA

4.14.1 Input

Data from the pipeline `boundary-scores` step is used as input (Section 4.13).

4.14.2 Analysis

Default parameters:

```
params_standard.tcsh
#!/bin/tcsh
source /inputs/params/params.tcsh
set chrom_excluded = 'chr[MYX]'  # excluded chromosomes
set group_var = 'cell_type'       # grouping variable (from sample sheet) to be used for color assignment
```

4.14.3 Output

See Figure 9. Default output:

```
-rw-r-r--  1 at570  4.1K Feb 15 15:20 job.err
-rw-r-r--  1 at570    47 Feb 15 15:18 job.sh
-rw-r-r--  1 at570  936 Feb 15 15:20 job.out
-rw-r-r--  1 at570  564 Feb 15 15:18 job.out
-rw-r-r--  1 at570  4.8K Feb 15 15:20 job.vars.tsv
-rw-r-r--  1 at570  211 Feb 15 15:18 labels.tsv
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.DL.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.DL2.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:20 pca.diffratio.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.intra-left.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.intra-max.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.intra-min.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:20 pca.novel-max.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:20 pca.novel-min.k=001.pdf
-rw-r-r--  1 at570  4.4K Feb 15 15:19 pca.ratio.k=001.pdf
```
Figure 9: Boundary Scores PCA sample output. See Section 4.14.
4.15 Domains

4.15.1 Input

Data from the pipeline steps matrix-filtered (Section 4.6), matrix-hicnorm (Section 4.9), matrix-prep (Section 4.7), and matrix-ic (Section 4.8) are used as input.

4.15.2 Analysis

Default parameters:

```bash
params.armatus.gamma_0.5.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set tool = armatus
set chrom_excluded = 'chr[MY]'
set armatus_params = "-g 0.5"
```

```bash
params.hicmatrix.tcsh$
#!/bin/tcsh
source ./inputs/params/params.tcsh
set tool = hicmatrix
set chrom_excluded = 'chr[MY]'
set hicmatrix_params = 
  "−−min−lambda=0.0 −−max−lambda=1.0 −−n−lambda=6 −−gamma=0 "
  "−−preprocess=none "
  "−−method=ratio "
  "−−distance=echo 500000/$bin_size | bc "
  "−−distance2=echo 500000/$bin_size | bc "
  "−−skip−distance=0 "
  "−−flank−dist=echo 500000/$bin_size | bc "
  "−−tolerance=0.01 "
  "−−alpha=0.25 "
  "−−track−dist=echo 2000000/$bin_size | bc "
  "−−presentation=none ")
```

```bash
params.topdom.tcsh$
#/usr/bin/tcsh
source ./inputs/params/params.tcsh
set tool = topdom
set topdompath = "./code/TopDom.R"
set chrom_excluded = 'chr[MY]'
set winsize = 5
```

4.15.3 Output

Default output:

```
-rw-r--r--  1 at570  288K Feb 15 16:31 domains.k=001.bed
-rw-r--r--  1 at570  288K Feb 15 16:31 job.err
-rw-r--r--  1 at570  47K Feb 15 16:13 job.id
-rw-r--r--  1 at570  5.6K Feb 15 16:31 job.out
-rw-r--r--  1 at570  347 Feb 15 16:13 job.sh
-rw-r--r--  1 at570  2.7K Feb 15 16:31 job.vars.tsv
```
4.16  Compare Boundaries

4.16.1  Input

Data from the pipeline domains step is used as input (Section 4.15).

4.16.2  Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
set flank_dist = $bin_size
set black_lists = ($genome_dir/centrotelo.bed)
```

4.16.3  Output

Default output:

```bash
-rw-r--r--  1 at570  573K Feb 16 00:18 boundaries1.k=001.bed
-rw-r--r--  1 at570  573K Feb 16 00:18 boundaries2.k=001.bed
-rw-r--r--  1 at570  268K Feb 16 00:18 common_boundaries.k=001.bed
-rw-r--r--  1 at570  154  Feb 16 00:18 comparison.tsv
-rw-r--r--  1 at570  268K Feb 16 00:18 intersection.k=001.bed
-rw-r--r--  1 at570   70  Feb 16 00:18 job.err
-rw-r--r--  1 at570   47  Feb 16 00:18 job.id
-rw-r--r--  1 at570    0  Feb 16 00:18 job.out
-rw-r--r--  1 at570   456 Feb 16 00:18 job.sh
-rw-r--r--  1 at570  4.5K Feb 16 00:18 job.vars.tsv
-rw-r--r--  1 at570  268K Feb 16 00:18 union.k=001.bed
```
4.17 Compare Boundaries Stats

4.17.1 Input

Data from the pipeline compare-boundaries step is used as input (Section 4.16).

4.17.2 Analysis

Default parameters:

```bash
#!/bin/tcsh

source ./inputs/params/params.tcsh
```

4.17.3 Output

See Figure 11 and Figure 10. Default output:

```bash
-rw-r---r-- 1 at570  6.9K Feb 12 12:52 comparisons.tsv
-rw-r---r-- 1 at570  27K Feb 12 12:52 correlograms.pdf
-rw-r---r-- 1 at570  238 Feb 12 12:52 job.err
-rw-r---r-- 1 at570  47 Feb 12 12:51 job.id
-rw-r---r-- 1 at570  52 Feb 12 12:52 job.out
-rw-r---r-- 1 at570  3.5K Feb 12 12:51 job.sh
-rw-r---r-- 1 at570  51K Feb 12 12:52 job.vars.tsv
-rw-r---r-- 1 at570  5.6K Feb 12 12:52 raw_comparisons.pdf
```
### Number of boundaries

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CD34–HindIII–rep1</td>
<td>9158</td>
<td>5231</td>
<td>5277</td>
<td>4045</td>
<td>4157</td>
<td>5006</td>
<td>3532</td>
<td>4869</td>
<td>3546</td>
</tr>
<tr>
<td>H1–HindIII–rep1</td>
<td>5231</td>
<td>7258</td>
<td>4724</td>
<td>3860</td>
<td>4059</td>
<td>4481</td>
<td>3215</td>
<td>4481</td>
<td>3232</td>
</tr>
<tr>
<td>H1–HindIII–rep2</td>
<td>5277</td>
<td>4724</td>
<td>6564</td>
<td>3615</td>
<td>3719</td>
<td>4186</td>
<td>2860</td>
<td>4149</td>
<td>2890</td>
</tr>
<tr>
<td>IMR90–HindIII–rep1</td>
<td>4045</td>
<td>3860</td>
<td>3615</td>
<td>5464</td>
<td>4291</td>
<td>3539</td>
<td>2666</td>
<td>3576</td>
<td>2678</td>
</tr>
<tr>
<td>IMR90–HindIII–rep2</td>
<td>4157</td>
<td>4059</td>
<td>3719</td>
<td>4291</td>
<td>5800</td>
<td>3767</td>
<td>2813</td>
<td>3749</td>
<td>2823</td>
</tr>
<tr>
<td>T47D_T0–HindIII–rep1</td>
<td>5006</td>
<td>4481</td>
<td>4186</td>
<td>3539</td>
<td>3767</td>
<td>7364</td>
<td>3252</td>
<td>5027</td>
<td>3264</td>
</tr>
<tr>
<td>T47D_T0–Ncol–rep1</td>
<td>3532</td>
<td>3215</td>
<td>2860</td>
<td>2666</td>
<td>2813</td>
<td>3252</td>
<td>5417</td>
<td>3205</td>
<td>4467</td>
</tr>
<tr>
<td>T47D_T60–HindIII–rep1</td>
<td>4869</td>
<td>4481</td>
<td>4149</td>
<td>3576</td>
<td>3749</td>
<td>5027</td>
<td>3205</td>
<td>7215</td>
<td>3234</td>
</tr>
<tr>
<td>T47D_T60–Ncol–rep1</td>
<td>3546</td>
<td>3232</td>
<td>2890</td>
<td>2678</td>
<td>2823</td>
<td>3264</td>
<td>4467</td>
<td>3234</td>
<td>5401</td>
</tr>
</tbody>
</table>

**Figure 10:** Example raw comparisons. See Section 4.17.
Figure 11: Example correlograms. See Section 4.17.
4.18 HiC Plotter

4.18.1 Input

Data from the pipeline steps matrix-filtered (Section 4.6), matrix-hicnorm (Section 4.9), matrix-prep (Section 4.7), and matrix-ic (Section 4.8) are used as input.

4.18.2 Analysis

Default parameters:

```
#!/bin/tcsh
source ./inputs/params/params.tcsh
set hicplotter_path = ./code/HiCPlotter2.py

# create bedgraphs for boundary scores
set bscores_branch = `echo $objects[1] | cut -d'-' -f1`
set f = bscores_branch/$objects[1]/all_scores.k=001.tsv
set methods = (intra-max DI ratio)
set bedgraphs = ()
set bedgraph_labels = ($methods)
foreach m ($methods)
    set k = `head -1 $f | tr 't' 'n' | grep ^$m | cut -d':-' -f1`
    cat $f | sed '1d' | cut -f1,5k | sed 's:/\t/\t/' | sed 's/-/\t/' >! $outdir/bscores.$m.bedGraph
    set bedgraphs = ( $bedgraphs $outdir/bscores.$m.bedGraph )
end

# add CTCF ChIP-seq
if (-e inputs/data.external/$cell_type/CTCF.bedGraph) then
    set bedgraphs = ($bedgraphs inputs/data.external/$cell_type/CTCF.bedGraph)
    set bedgraph_labels = ($bedgraph_labels CTCF)
endif

# regions to plot
set regions = "chr8:125000000-133000000"
set tiles = "params-regions.bed"
set tiles_labels = "regions"
set highlight = 1
set highlight_bed = "params/highlight.bed"
set insulation_score = 0 # Either 1 or 0 (header / no header)
```

4.18.3 Output

See Figure 12. Default output:

```
-rw-r--r-- 1 at570 2.3M Feb 15 14:49 bscores.DI.bedGraph
-rw-r--r-- 1 at570 2.3M Feb 15 14:49 bscores.intra-max.bedGraph
-rw-r--r-- 1 at570 2.3M Feb 15 14:49 bscores.ratio.bedGraph
-rw-r--r-- 1 at570 146K Feb 15 14:50 chr8:125000000-133000000.pdf
-rw-r--r-- 1 at570 107 Feb 15 14:50 job.err
-rw-r--r-- 1 at570 47 Feb 15 14:49 job.id
-rw-r--r-- 1 at570 40 Feb 15 14:50 job.out
-rw-r--r-- 1 at570 335 Feb 15 14:49 job.sh
-rw-r--r-- 1 at570 8.4K Feb 15 14:50 job.vars.tsv
```
Figure 12: HiCPlotter sample output
4.19 Interactions

4.19.1 Input

Data from the pipeline matrix-filtered step is used as input (Section 4.6).

4.19.2 Analysis

Default parameters:

```bash
#!/bin/tcsh
source ./inputs/params/params.tcsh
set chrom_excluded = 'chr[MYX]'           # excluded chromosomes
set loop_params = "−−bin−size=$bin_size
−−lambda−id=6
−−rpk2b−cutoff=1.0
−−loop−cutoff=4.0
−−min−distance=40000"
# parameters for identifying significant interactions
```

4.19.3 Output

See Figure 13. Default output:

```bash
drwxr-xr-x  2 at570  3.3K Feb  5 10:12 __jdata
−rw−r−r−r—  1 at570  4.8K Feb  5 10:17 job.err
−rw−r−r−r—  1 at570   47 Feb  5 10:11 job.id
−rw−r−r−r—  1 0 Feb  5 10:11 job.out
−rw−r−r−r—  1 at570  375 Feb  5 10:11 job.sh
−rw−r−r−r—  1 at570  3.6K Feb  5 10:17 job.vars.tsv
drwxr-xr-x  2 at570  54 Feb  5 10:15 matrix.chr1
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr10
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr11
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr12
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr13
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr14
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr15
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr16
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr17
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr18
drwxr-xr-x  2 at570  54 Feb  5 10:16 matrix.chr2
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr20
drwxr-xr-x  2 at570  54 Feb  5 10:12 matrix.chr21
drwxr-xr-x  2 at570  54 Feb  5 10:13 matrix.chr22
drwxr-xr-x  2 at570  54 Feb  5 10:15 matrix.chr3
drwxr-xr-x  2 at570  54 Feb  5 10:15 matrix.chr4
drwxr-xr-x  2 at570  54 Feb  5 10:15 matrix.chr5
drwxr-xr-x  2 at570  54 Feb  5 10:15 matrix.chr6
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr7
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr8
drwxr-xr-x  2 at570  54 Feb  5 10:14 matrix.chr9
```

```
matrix.chr1$
−rw−r−r−r—  1 at570  4.7M Feb  5 10:15 loops.tsv
−rw−r−r−r—  1 at570   27K Feb  5 10:16 plots.pdf
```
Figure 13: Interactions sample output
4.20 Annotations

4.20.1 Input

Data from the pipeline interactions step is used as input (Section 4.19).

4.20.2 Analysis

Default parameters:

```
params.standard.tcsh
#!/bin/tcsh

source /inputs/params/params.tcsh

set genes_bed = $genome_dir/gene.bed # gene BED6 file for annotation of interactions
set cell_type = `echo $objects[1] | cut -d'-' -f1`
if (! -e inputs/data.external/$cell_type) then
  set loci_bed = ()
else
  set loci_bed = `find inputs/data.external/$cell_type -maxdepth 1 -name '*.bed'`
endif
```

4.20.3 Output

Default output:

```
-rw-r-r-- 1 at570 5.9M Feb 5 17:33 bin.annotated.tsv
-rw-r-r-- 1 at570 3.8M Feb 5 17:33 bin.gene.tsv
-rw-r-r-- 1 at570 7.5M Feb 5 17:33 bin.loci.tsv
-rw-r-r-- 1 at570 8.7M Feb 5 17:33 bin.reg
-rw-r-r-- 1 at570 5.4K Feb 5 17:33 job.err
-rw-r-r-- 1 at570 47 Feb 5 17:32 job.id
-rw-r-r-- 1 at570 0 Feb 5 17:33 job.out
-rw-r-r-- 1 at570 434 Feb 5 17:32 job.sh
-rw-r-r-- 1 at570 3.1K Feb 5 17:33 job.vars.tsv
-rw-r-r-- 1 at570 42M Feb 5 17:33 loci.reg
-rw-r-r-- 1 at570 49M Feb 5 17:33 table.annotated.tsv
```
4.21 Annotations Stats

4.21.1 Input

Data from the pipeline annotations step is used as input (Section 4.20).

4.21.2 Analysis

Default parameters:

```
paramsstandard.tcsh$
# /bin/tcsh
source./inputs/params/params.tcsh
set nbest = 10000        # choose top-scoring interactions to calculate enrichments
```

4.21.3 Output

See Figure 14. Default output:

```
-rw-r--r--  1 at570  77 Feb 16 17:26 counts.tsv
-rw-r--r--  1 at570  350 Feb 16 17:26 enrich.tsv
-rw-r--r--  1 at570  7.0K Feb 16 17:26 enrichment.pdf
-rw-r--r--  1 at570  121 Feb 16 17:26 job.err
-rw-r--r--  1 at570   47 Feb 16 17:25 job.id
-rw-r--r--  1 at570   62 Feb 16 17:26 job.out
-rw-r--r--  1 at570  507 Feb 16 17:25 job.sh
-rw-r--r--  1 at570  3.1K Feb 16 17:26 job.vars.tsv
-rw-r--r--  1 at570  184 Feb 16 17:25 top_counts.tsv
```
Figure 14: Annotation Stats enrichment sample output. See Section 4.21.
5  Appendix

5.1  Error Logs

Errors encountered during pipeline execution can be viewed with:

```
<project_directory>$ code.main/pipeline-errors
```

Analysis results can be removed with:

```
<project_directory>$ code/clean-all
```

5.2  Other Pipeline Software: gtools-hic

```
code.repo/bin/gtools-hic$
```

**USAGE:**

```
gtools-hic OPERATION [OPTIONS] <REGION-SET>
```

**VERSION:**

```
genomic-tools 3.0.0
```

**DESCRIPTION:**

Pipeline for HiC-seq data analysis. For detailed description and list of options choose an operation and use the "--help" option.

**OPERATION:**

- **align**  Iteratively aligns HiC-seq read pairs to reference genome using bowtie2.
- **classify**  Classifies and computes various metrics for HiC-seq aligned read pairs.
- **filter**  Filters HiC-seq aligned read pairs for common experimental artifacts.
- **bin**  Bins filtered read pairs to genomic bins of desired resolution.
- **matrix**  Create Hi-C count matrix.
- **convert**  Convert contact matrix into WashU Epigenome Browser format.

5.2.1  gtools-hic align

```
code.repo/bin/gtools-hic align --help
```

**USAGE:**

```
gtools-hic align [OPTIONS] READ1-FASTQ READ2-FASTQ
```

**DESCRIPTION:**

Iteratively aligns HiC-seq read pairs to reference genome using bowtie2.

**DETAILS:**

* Input: FASTQ files
* Output: aligned reads in SAM format (same order as in fastq files)

**OPTIONS:**

```
--help  help
--help  help [true]
-v     verbose mode
--work-dir  working directory (required)
--min-len  minimum truncated read length
--len-diff  read truncation step
-p     number of threads for bowtie2 run
--bowtie-path  full bowtie2 path (version >=2.1.0)
--bowtie-index  full bowtie2 index prefix path
```

Page 56 of 62
5.2.2  gtools-hic classify

```bash
USAGE:
gtools-hic classify [OPTIONS] <ALIGNED-READS>

DESCRIPTION:
Classifies and computes various metrics for HiC-seq aligned read pairs.

DETAILS:
- Input: aligned reads in SAM format (sorted by read-id, at most one alignment per read)
- Output: tab-separated table

OPTIONS:
--help   help [true]
--h      help [true]
--v      verbose mode [false]
--E      enzyme fragments (BED/GFF/SAM/REG) []
--min-dist minimum allowed distance between 5's of reads in read pair [500]
--max-offset maximum allowed offset of 5's of reads from fragment ends [500]
```

5.2.3  gtools-hic filter

```bash
USAGE:
gtools-hic filter [OPTIONS] <ALIGNED-READS>

DESCRIPTION:
Filters HiC-seq aligned read pairs for common experimental artifacts.

DETAILS:
- Input: aligned reads in SAM format (sorted by read-id, at most one alignment per read)
- Output: filtered read pairs in REG format

OPTIONS:
--help   help [true]
--h      help [true]
--v      verbose mode [false]
--E      enzyme fragments (BED/GFF/SAM/REG) []
--min-dist minimum allowed distance between 5's of reads in read pair [500]
--max-offset maximum allowed offset of 5's of reads from fragment ends [500]
--filter-dups filter duplicate read pairs as PCR artifacts [false]
--stats output statistics file (default=stderr) []
```

5.2.4  gtools-hic bin

```bash
USAGE:
gtools-hic bin [OPTIONS] <FILTERED-READ-PAIRS>

DESCRIPTION:
Bins filtered read pairs to genomic bins of desired resolution.

DETAILS:
- Input: filtered read pairs in REG format
- Output: binned read pairs

OPTIONS:
--help   help [true]
--h      help [true]
--v      verbose mode [false]
--bin-size genomic bin size [1000000]
--g      genome region file (BED/REG) []
```
### 5.2.5 gtools-hic matrix

```bash
code.repo/bin/gtools-hic matrix --help$
```

**USAGE:**
```bash
gtools-hic matrix [OPTIONS] <FILTERED-READ-PAIRS>
```

**DESCRIPTION:**
Create Hi-C count matrix.

**DETAILS:**
- Input: filtered read pairs in REG format
- Output: contact matrix

**OPTIONS:**
- `--help` help [true]
- `-h` help [true]
- `-v` verbose mode [false]
- `--bin-size` genomic bin size (in nucleotides) [5000]
- `--max-dist` maximum distance between bins (in nucleotides; default = no restriction) [0]
- `--rotate45` rotate matrix by 45 degrees (applicable if `--max-dist > 0`) [false]
- `--R` reference region file (BED/REG) []
- `--p` output file prefix []

### 5.2.6 gtools-hic convert

```bash
code.repo/bin/gtools-hic convert --help
```

**USAGE:**
```bash
gtools-hic convert [OPTIONS] <CONTACT-MATRIX>
```

**DESCRIPTION:**
Convert contact matrix into WashU Epigenome Browser format.

**DETAILS:**
- Input: locus-labelled contact matrix
- Output: WashU Epigenome Browser format

**OPTIONS:**
- `--help` help [true]
- `-h` help [true]
- `-v` verbose mode [false]
- `--col-labels` input matrix has column labels [false]
- `-t` matrix element separator [ ]
- `--C` normalization constant [1.000000e+00]
- `-min` score cutoff (values below this are set to zero) [0.0000000e+00]
- `--d` maximum distance between interacting loci (default = no limit) [0]
5.3 Other Pipeline Software: pipeline-master-explorer.r

The `pipeline-master-explorer.r` script, located in the `code.main` directory, is the driver of combinatorial parameter exploration during the execution of each pipeline step.

```r
code.main$ ./pipeline-master-explorer.r --help
Usage: pipeline-master-explorer.r [OPTIONS] SCRIPT OUTDIR PREFIX PARAM SCRIPTS INPUT BRANCHES SPLIT VARIABLE TUPLES

Options:
- v, --verbose
  Print more messages.
- S SAMPLE-SHEET, --sample-sheet=SAMPLE-SHEET
  Sample sheet file name (required) [default "inputs/sample-sheet.tsv"].
- F FILTER-BRANCH, --filter-branch=FILTER-BRANCH
  Regular expression for filtering input branches [default ""].
- --exclude-branch=EXCLUDE-BRANCH
  Regular expression for excluding input branches [default ""].
- --exclude-obj=EXCLUDE-OBJ
  Regular expression for excluding input objects [default ""].
- --exclude-outdir=EXCLUDE-OUTDIR
  Regular expression for excluding output directories [default ""].
- h, --help
  Show this help message and exit
```
5.4 System and Session Information

\LaTeX version: \LaTeX 2e 2005/12/01

\input{system}

\begin{verbatim}
\input{sessionInfo}
\end{verbatim}

\LaTeX File List

<table>
<thead>
<tr>
<th>File</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>article.cls</td>
<td>2005/09/16</td>
<td>v1.4f Standard \LaTeX document class</td>
</tr>
<tr>
<td>size10.clo</td>
<td>2005/09/16</td>
<td>v1.4f Standard \LaTeX file (size option)</td>
</tr>
<tr>
<td>graphicx.sty</td>
<td>1999/02/16</td>
<td>v1.0f Enhanced \LaTeX Graphics (DPC,SPQR)</td>
</tr>
<tr>
<td>keyval.sty</td>
<td>1999/03/16</td>
<td>v1.13 key-value parser (DPC)</td>
</tr>
<tr>
<td>graphics.sty</td>
<td>2006/02/20</td>
<td>v1.00 Standard \LaTeX Graphics (DPC,SPQR)</td>
</tr>
<tr>
<td>trig.sty</td>
<td>1999/03/16</td>
<td>v1.09 sin cos tan (DPC)</td>
</tr>
<tr>
<td>graphics.cfg</td>
<td>2007/01/18</td>
<td>v1.5 graphics configuration of \TeX/\TeXLive</td>
</tr>
<tr>
<td>pdftex.def</td>
<td>2007/08/08</td>
<td>v0.04d Graphics/color for pd\TeX</td>
</tr>
<tr>
<td>color.sty</td>
<td>1999/02/16</td>
<td></td>
</tr>
<tr>
<td>color.cfg</td>
<td>2007/01/18</td>
<td>v1.5 color configuration of \TeX/\TeXLive</td>
</tr>
<tr>
<td>framed.sty</td>
<td>2011/10/22</td>
<td>v 0.96: framed or shaded text with page breaks</td>
</tr>
<tr>
<td>alltt.sty</td>
<td>1997/08/16</td>
<td>v2.0g defines alltt environment</td>
</tr>
<tr>
<td>mathpazo.sty</td>
<td>2005/04/12</td>
<td>PSNFSS-v9.2a Palatino w/ Pazo Math (D.Puga, WaS)</td>
</tr>
<tr>
<td>helvet.sty</td>
<td>2005/04/12</td>
<td>PSNFSS-v9.2a (WaS)</td>
</tr>
<tr>
<td>fontenc.sty</td>
<td>2005/09/27</td>
<td>v1.99g Standard \LaTeX file</td>
</tr>
<tr>
<td>t1enc.def</td>
<td>2005/02/07</td>
<td>v3.2 Page Geometry</td>
</tr>
<tr>
<td>geometry.sty</td>
<td>2002/07/08</td>
<td></td>
</tr>
<tr>
<td>geometry.cfg</td>
<td>2002/07/08</td>
<td></td>
</tr>
<tr>
<td>cite.sty</td>
<td>2003/11/04</td>
<td>v 4.01 Customising captions (AR)</td>
</tr>
<tr>
<td>caption.sty</td>
<td>2007/01/07</td>
<td>v3.0k Customising captions (AR)</td>
</tr>
<tr>
<td>caption3.sty</td>
<td>2007/01/07</td>
<td>v3.0k caption3 kernel (AR)</td>
</tr>
<tr>
<td>hyperref.sty</td>
<td>2002/07/07</td>
<td>v6.75r Hyperref links for \LaTeX</td>
</tr>
<tr>
<td>pdftex.def</td>
<td>2002/07/07</td>
<td>v6.75r Hyperref: PDFDocEncoding definition (HO)</td>
</tr>
<tr>
<td>hyperref.cfg</td>
<td>2002/06/06</td>
<td>v1.2 hyperref configuration of Te\TeX</td>
</tr>
<tr>
<td>ksvoptions.sty</td>
<td>2006/08/22</td>
<td>v2.4 Connects package keyval with \LaTeX options (HO)</td>
</tr>
<tr>
<td>puenc.def</td>
<td>2002/07/07</td>
<td>v6.75r Hyperref: PDF Unicode definition (HO)</td>
</tr>
<tr>
<td>url.sty</td>
<td>2005/06/27</td>
<td>v3.2 Verb mode for urls, etc.</td>
</tr>
<tr>
<td>pdftex.def</td>
<td>2002/07/07</td>
<td>v6.75r Hyperref driver for pd\TeX</td>
</tr>
<tr>
<td>breakurl.sty</td>
<td>2006/08/26</td>
<td>v1.20 Breakable hyperref URLs</td>
</tr>
<tr>
<td>xkeyval.sty</td>
<td>2006/11/18</td>
<td>v2.5f package option processing (HA)</td>
</tr>
<tr>
<td>xkeyval.tex</td>
<td>2006/11/18</td>
<td>v2.5f key-value parser (HA)</td>
</tr>
</tbody>
</table>