Hi-C User Pipeline (HiCUP) Manual

HiCUP is a bioinformatics pipeline written in Perl for processing Hi-C data

HICUP V0.5.12
MANUAL EDIT 2
22 NOVEMBER 2014

HICUP QUICK START GUIDE

HiCUP is a bioinformatics pipeline for processing Hi-C data. The pipeline receives FASTQ data which is mapped against a reference genome and filtered to remove frequently encountered experimental artefacts. The pipeline produces paired read files in SAM/BAM format, each read pair corresponding to a putative Hi-C di-tag. HiCUP also produces summary statistics at each stage of the pipeline to assist with quality control and identify potential experimental problems and help refine the protocol for the future.

The HiCUP pipeline is not intended for determining contact probabilities between loci, which requires normalising the data for a range of biases.

Another Babraham Institute project, SeqMonk, enables the visualisation of Hi-C data (www.bioinformatics.babraham.ac.uk/projects/seqmonk).

REQUIREMENTS

HiCUP requires a working version of Perl and Bowtie installed on your machine. For further details go to:

http://www.perl.org/

http://bowtiebio.sourceforge.net/index.shtml

For full functionality HiCUP requires the R to be installed. (For more details of R go to http://www.r-project.org).

INSTALLATION

HiCUP is written in Perl and executed from the command line. To install HiCUP copy the hicup_v0.X.Y.tar.gz file into a HiCUP installation folder and extract all files by typing:

```
tar -xvzf hicup_v0.X.Y.tar.gz
```

Check after extracting that the Perl scripts are executable by all.

RUNNING HICUP

1) CREATE A DIGESTED REFERENCE GENOME

To filter out common experimental artefacts, HiCUP requires the positions at which the restriction enzyme(s) used in the protocol cut the genome. The script hicup_digester creates this reference genome digest file. The example below will digest all '.fa' files in the current working directory with HindIII. The digest output file will be labelled as the genome 'Human_GRCh37'. Provide the full path to hicup_digester or the sequence files to be digested if they are not in the current working directory.

Execute the script:

```
hicup_digester -g Human_GRCh37 -1 A^AGCTT,HindIII *.fa
```

2) CREATE BOWTIE INDICES

HiCUP uses the aligner Bowtie to map sequences to the reference genome, requiring the construction of Bowtie indices. **These indices need to be constructed from the same reference genome files as used by the hicup_digester script.** In the command line enter 'bowtie-build' to construct the indices, followed by a comma-separated list of the sequence files and then a space followed by the name of the output indices. For example:

```
bowtie-build 1.fa,2.fa,…,MT.fa Human_GRCh37
```

Further instructions on building a Bowtie index can be found at:

http://bowtie-bio.sourceforge.net/manual.shtml#the-bowtie-build-indexer

3) RUN THE HICUP PIPELINE

To start the pipeline, copy the configuration file hicup.conf and then modify the copy as required, adhering to the format below:

```
#Directory to which output files should be written (optional parameter)
#Set to current working directory by default
Outdir:

#Number of threads to use
Threads: 1

#Suppress progress updates (0: off, 1: on)
Quiet:0

#Retain intermediate pipeline files (0: off, 1: on)
Keep:0

#Compress outputfiles (0: off, 1: on)
Zip:1

#Path to the alignment program Bowtie (include the executable Bowtie filename)
Bowtie: /usr/local/bowtie/bowtie

#Path to R (required for full functionality)
R: /bi/apps/R/3.0.3/bin/R

#Path to the reference genome indices
#Remember to include the basename of the genome indices

Index: /data/public/Genomes/Mouse/NCBIM37/Mus_musculus.NCBIM37
```

```
#Path to the genome digest file produced by hicup_digester
Digest: Digest_Mouse_genome_HindIII_None_12-32-06_17-02-2012.txt

#FASTQ file format (phred33-quals, phred64-quals, solexa-quals or solexa1.3-quals)
#If not specified, HiCUP will try to determine the format automatically by analysing
#one of the FASTQ files. All input FASTQ will assumed to be in this format
Format: phred33-quals

#Maximum di-tag length (optional parameter)
Longest: 800

#Minimum di-tag length (optional parameter)
Shortest: 150

#FASTQ files to be analysed, paired files on adjacent lines
s_1_1_sequence.fastq
s_1_2_sequence.fastq

s_2_1_sequence.fastq
s_2_2_sequence.fastq

s_3_1_sequence.txt.fastq.gz
s_3_2_sequence.txt.fastq.gz
```

Enter the following text in the command line to run the whole HiCUP pipeline using the parameters specified in the configuration file:

```
hicup -c hicup.conf
```

The '-c' flag is used to specify the configuration filename. Also remember to provide the full path to the hicup script or the configuration file if they are not in the current working directory.

The HiCUP pipeline may take several hours to complete, so it may be preferable to run as a background job that will not terminate when ending the session. To do this, run the pipeline with the command:

```
nohup hicup -c hicup.conf &
```

TEST DATASET

To confirm HiCUP functions correctly on your system please download the test Hi-C dataset from the HiCUP homepage. The test files 'test_dataset1.fastq' and 'test_dataset2.fastq' both contain human Hi-C reads in Phred33 FASTQ format.

1) Extract the tar archive before processing:

```
tar -xvzf hicup_v0.X.Y.tar.gz
```

2) If necessary, create Bowtie indices of the Homo sapiens GRCh37 genome (chromosomes 1…22, X, Y and MT).

Example command:

```
bowtie-build 1.fa,2.fa,…,MT.fa human_GRCh37
```

3) Using hicup_digester create a reference genome of Homo sapiens GRCh37 all chromosomes (1…22, X, Y and MT) digested with HindIII (A^AGCTT).

Example command:

```
hicup_digester –g Human_GRCh37 -1 A^AGCTT,HindIII *.fa
```

4) Edit a copy of the hicup.conf configuration file so it has the following parameters:

```
Zip: 1
Keep: 0
Threads: 1

Bowtie: [Path to Bowtie on your system]
Digest: [Path to digest file on your system]
Index: [Path to Bowtie indices on your system]
R: [Path to R on your system]

Format: phred33-quals
Shortest: 150
Longest: 800

test_dataset1.fastq | test_dataset2.fastq
```

5) Run the pipeline using the command:

```
hicup –c hicup.conf
```

HICUP - GENERAL INFORMATION

HI-C OVERVIEW

Chromosome folding can bring distant elements – such as promoters and enhancers – close together, which may affect genome activity. Consequently, investigating genomic conformation may improve understanding of processes such as transcription and replication.

Hi-C, developed from 3C, identifies long-range genomic interactions. The Hi-C protocol involves formaldehyde-fixing cells to create DNA-protein bonds that cross-link interacting DNA loci. The DNA is then digested with a restriction enzyme, fragmenting the molecule but maintaining the cross-links. The resulting overhanging 5' ends are then filled in with the concomitant incorporation of a biotinylated residue. Next, blunt-end ligation is performed under dilute conditions, favouring ligation between cross-linked DNA fragments. This produces a library of ligation products that were close to each other in the nucleus. The library is the further fragmented, either by using a second restriction enzyme or by sonication. The Hi-C fragments are then pulled-down with streptavidin beads, which adhere with great affinity to the biotin tag at the ligation junction. The purified Hi-C fragments are then sequenced (Lieberman-Aiden *et al.*)

WHAT IS HICUP?

HiCUP (Hi-C User Pipeline) comprises six Perl scripts for analysing Hi-C sequence data:

i) hicup_digester
ii) hicup_truncater
iii) hicup_mapper
iv) hicup_filter
v) hicup_deduplicator
vi) hicup (controls other pipeline scripts)

The pipeline enables multiplexed sequences to be sorted and mapped to the genome, the pairing of Hi-C fragment ends and filtering for valid Hi-C interaction products.

DEPENDENCIES

HiCUP requires a working version of Perl and Bowtie installed on your machine. For further details go to:
http://www.perl.org/
http://bowtiebio.sourceforge.net/index.shtml

Bugs should be reported at:
http://www.bioinformatics.bbsrc.ac.uk/bugzilla/

We would like to hear your comments or suggestions regarding HiCUP. Please email them to:
steven.wingett@babraham.ac.uk.

The rest of this manual discusses each script in more detail and provides instructions on how to execute pipeline scripts separately i.e. without running the 'hicup' control script.

HICUP_DIGESTER

The 'hicup_digester' Perl script cuts throughout a selected genome at one or two specified restriction sites

SYNOPSIS

hicup_digester [OPTIONS] --config [Configuration FILE]…

hicup_digester [OPTIONS] [FASTA format sequence files]

FUNCTION

The Perl script 'hicup_mapper' generates a file of paired mapped reads. While the majority of those reads are expected to be valid Hi-C ligation products, a substantial minority probably will not and should be removed.

The script 'hicup_filter' removes many of those invalid pairs, but before it can do this it requires a digested reference genome as input, along with the paired sequence files. The hicup_digester program cuts a specified genome with one or two Type II restriction enzymes that recognise single undivided palindromic sequences, in accordance with the experimental protocol followed. The script prints the results to file for subsequent processing by hicup_filter.

The names of the files to be processed and the digestion parameters may be passed to the script by a configuration file or command line arguments. The configuration file contains: i) restriction site 1; ii) restriction site 2 (optional); iii) the name of the genome to be processed (optional) and iv) list of FASTA files to be processed.

For example:
re1: A^GATCT, BglII
re2: AG^CT, AluI
genome: Mouse

/Genomes/Mouse/NCBIM37/Mus_musculus.NCBIM37.52.dna.chromosome.1.fa
/Genomes/Mouse/NCBIM37/Mus_musculus.NCBIM37.52.dna.chromosome.2.fa
/Genomes/Mouse/NCBIM37/Mus_musculus.NCBIM37.52.dna.chromosome.3.fa
.
.
.
/Genomes/Mouse/NCBIM37/ Mus_musculus.NCBIM37.52.dna.chromosome.Y.fa

The re1 flag refers to the sequence at which the first restriction enzyme used in the Hi-C protocol cuts the genome. The carat symbol ('^') marks the position where the enzyme cuts the DNA. As an option the name of the restriction enzyme may be included after the sequence, using a comma to separate the two. Some Hi-C protocols may use two enzymes at this stage to create Hi-C ligation junctions. Specify two enzymes as follows: -1 A^GATCT,BglII:A^AGCTT,HindIII.

Specify restriction enzyme 2 e.g. AG^CT,AluI. Restriction site 2 refers to the second, optional (other DNA shearing techniques such as sonication may be used) enzymatic digestion. This restriction site does NOT form a Hi-C ligation junction. This is the restriction enzyme that is used when the Hi-C sonication protocol is not followed. Typically the sonication protocol is followed.

The program creates a tab-delimited file listing:

column 1: chromosome name (as named in the header row of the FASTA file)
column 2: restriction fragment start position (chromosomes considered to start at base 1)

column 3: restriction fragment end position (chromosomes considered to start at base 1)
column 4: restriction fragment number (the fragment at the five-prime end of each chromosome is named fragment 1)
column 5: the restriction fragment number if the genome underwent a single digest with restriction enzyme 1 (the fragment at the five-prime end of each chromosome is named fragment 1)
column 6: the restriction site at the five-prime end of the restriction fragment
column 7: the restriction site at the three-prime end of the restriction fragment

Important note: chromosome names in the sequence file to be digested need to be identical to those in the Bowtie indices.

COMMAND LINE OPTIONS

--re1	Restriction enzyme used to digest the genome (the enzyme that forms the ligation junction) e.g. A^GATCT,BglII. Some Hi-C protocols may use two enzymes at this stage. To specify two enzymes: -1 A^GATCT,BglII:A^AGCTT,HindIII.
--re2	To specify a restriction enzyme instead of sonication to shorten di-tags. This restriction site does NOT form a Hi-C ligation junction. 2 e.g. AG^CT,AluI. Typically the sonication protocol is followed.
--config	Specify the name of the optional configuration file
--genome	Name of the genome to be digested (not the path to the genome file or files, but the genome name to include in the output file)
--help	Print program help and exit
--outdir	Specify the directory to which the output files should be written
--quiet	Suppress all progress reports
--version	Print the program version and exit
--zip	Print the results to a gzip file

HICUP

The hicup Perl script controls the other programs in the HiCUP pipeline

SYNOPSIS

hicup [OPTIONS]... [Configuration FILE]...

FUNCTION

The HiCUP pipeline comprises the scripts 'hicup_truncater', 'hicup_mapper', 'hicup_filter' and 'hicup_deduplicator' ('hicup_digester generates the genome_digest file used by hicup_filter). The pipeline receives FASTQ files and generates Hi-C di-tag paired reads aligning to a specified reference genome. The hicup script regulates the pipeline, executing each script in turn and passing output from one program to the next.

The designated configuration file sets the parameters for the whole pipeline. The configuration file lists the names of the FASTQ file pairs to be processed.

For example:

s_1_1_sequence.txt
s_1_2_sequence.txt

s_2_1_sequence.txt
s_2_2_sequence.txt

s_3_1_sequence.txt
s_3_2_sequence.txt

This configuration instructs the pipeline to process and pair the files s_1_1_sequence.txt / s_1_2_sequence.txt; s_2_1_sequence.txt / s_2_2_sequence.txt; and s_3_1_sequence.txt / s_3_2_sequence.txt (remember, a file pair generates one output file).

HiCUP also requires the paths to Bowtie and the genome digest file.

Print the example configuration file for more details (hicup -e) or read the help documentation for the other scripts of the pipeline.

COMMAND LINE OPTIONS

--bowtie	Specify the path to Bowtie
--bowtie2	Specify the path to Bowtie 2
--config	Specify the configuration file
--example	Produce an example configuration file
--format	Specify FASTQ format Options: Sanger, Solexa_Illumina_1.0, Illumina_1.3, Illumina_1.5
--help	Print help message and exit
--keep	Keep intermediate pipeline files
--longest	Maximum allowable insert size (bps)
--nofill	Hi-C protocol did NOT include a fill-in of sticky ends prior to ligation step and therefore FASTQ reads shall be truncated at the Hi-C restriction enzyme cut site (if present) sequence is encountered
--outdir	Directory to write output files
--quiet	Suppress progress reports (except warnings)
--shortest	Minimum allowable insert size (bps)
--temp	Write intermediate files (i.e. all except summary files and files generated by HiCUP Deduplicator) to a specified directory
--threads	Specify the number of threads, allowing simultaneous processing of multiple files
--version	Print the program version and exit
--zip	Compress output

HICUP_TRUNCATER

The hicup_truncater Perl script terminates sequence reads at specified Hi-C ligation junctions

SYNOPSIS

truncate [OPTIONS]... --config [Configuration FILE]...

truncate [OPTIONS]... [FASTQ FILES]...

FUNCTION

Valid Hi-C pairs comprise two DNA fragments from different regions of the genome ligated together. Typically, a forward read maps to one ligation fragment; the reverse read maps to the other. However, this is not always true since the Hi-C ligation junction may be found within the sequenced region. Such reads will most likely be removed from the Hi-C pipeline during the mapping process, thereby losing potentially valid data. The hicup_truncater script helps remedy this by identifying ligation junctions within reads and deleting sequence downstream of the restriction enzyme recognition site.

The names of the files to be processed and the restriction site may be passed to the scrip using a configuration file or command line arguments. The configuration file contains: i) the recognition sequence of the first (or only) restriction enzyme used in the Hi-C protocol and ii) the sequence files to be processed by the hicup_truncater.

Example:

re1: A^GATCT
s_6_1_sequence.txt_CCTT.txt
s_6_2_sequence.txt_CCTT.gz

This configuration instructs the script to process the files 's_6_1_sequence.txt_CCTT.txt' and 's_6_2_sequence.txt_CCTT.gz'. The script identifies any reads containing the Hi-C ligation sequence 'AGATCGATCT' and discards sequence downstream of the restriction cut site. (The caret symbol ('^') denotes the cut position in the restriction enzyme recognition sequence.)

(Alternatively, instead of specifying the restriction enzyme, specify the Hi-C ligation junctions as a comma-delimited list. The underscore character '_' is used to denote the position at which the ligation junction no longer matches the reference genome e.g. --sequences AAGCT_AGCTT. This enables the user to specify multiple truncation sites.)

The program creates sequence files named the same as the input files, only suffixed with '_trunc'. The script also produces a date-stamped summary file (e.g. 'truncate_sequence_summary_11-02-45_21-02-2011.txt') listing the number of reads truncated or not truncated for each input sequence file.

COMMAND LINE OPTIONS

--config	Name of the optional configuration file
--help	Print program help and exit
--nofill	Hi-C protocol did NOT include a fill-in of sticky ends prior to re-ligation and therefore reads shall be truncated at the restriction site sequence
--outdir	Directory to write output files
--quiet	Suppress all progress reports
--re1	Restriction enzyme recognition sequence
--sequences	Instead of specifying a restriction enzyme recognition sequence, specify the ligation sequences directly
--threads	Number of threads to use, allowing simultaneous processing of different files
--version	Print the program version and exit
--zip	Compress output

HICUP_MAPPER

The hicup_mapper script aligns sequence reads to a genome specified by the user

SYNOPSIS

hicup_mapper [OPTIONS]... --config [Configuration FILE]...

hicup_mapper [OPTIONS]... [FASTQ FILES]...

FUNCTION

Valid Hi-C ligation products comprise two restriction fragments from different regions of the genome ligated together. This program maps Hi-C di-tags against a reference genome to determine from where each restriction fragment is derived. Following mapping the forward and reverse reads are paired i.e. two input files result in one output file.

hicup_mapper calls the sequence alignment program 'Bowtie' to perform the mapping (http://bowtie-bio.sourceforge.net/index.shtml) using the following parameters:

-p 1 -n 1 -m 1 --best

-p 1: launches Bowtie with only one search thread. Although limiting the search to one processor is slower on multi-core machines, it ensures the order of the returned mapped reads is the same as found in the input file (ignoring omitted unmapped sequences). Maintaining the order is essential for the correct functioning of the hicup_pairer script further along the pipeline.(Bowtie actually defaults to -p 1, but this has been made explicit in the Perl script due to the importance of preserving the read order.)

-n 1: alignments may have no more than 1 mismatch in the first 28 bases (seed) and the sum of the Phred quality values at all mismatched positions (not just in the seed) may not exceed 70.Bowtie rounds quality values to the nearest 10, saturating at 30.

-m 1: report only unique alignments

--best: reports alignments in best-to-worst order

The configuration file 'mapper.conf' sets the parameters for the hicup_mapper script (the name of the configuration file may be changed as required). The configuration file contains: i) names of files to be mapped; ii) local path to Bowtie; iii) path to the relevant reference genome Bowtie indices; iv) the sequence format.

For example:
Bowtie: /usr/local/bowtie/bowtie
Index: /data/public/Genomes/Mouse/NCBIM37/Mus_musculus.NCBIM37
Format: solexa1.3-quals
Threads: 4

s_1_1_sequence.txt_CCTT_trunc
s_1_2_sequence.txt_CCTT_trunc

s_1_1_sequence.txt_AAGT_trunc
s_1_2_sequence.txt_AAGT_trunc

The above example sends four files to Bowtie for mapping against the mouse NCBIM37 genome. The flags 'Bowtie:', 'Index:' and 'Format' need to be retained in the configuration file. Any text not preceded with a flag is assumed to be a sequence filename.

The sequence format options are:

phred33-quals	Phred scores are ASCII character values plus 33
phred64-quals	Phred scores are ASCII character values plus 64
solexa-quals	Convert input qualities from Solexa (which can be negative) to Phred (which cannot)
solexa1.3-quals	Same as phred64-quals

The output filenames will be based on the input filenames but suffixed with '.pair'. The script also produces a date-stamped summary (e.g. 'hicup_mapper_summary_17-04-42_03-08-2011.txt') listing, among other things, the number of reads that successfully mapped to the genome.

COMMAND LINE OPTIONS

--bowtie	Specify the path to Bowtie
--bowtie2	Specify the path to Bowtie 2
--config	Specify the configuration file
--format	Specify FASTQ format Options: Sanger, Solexa_Illumina_1.0, Illumina_1.3, Illumina_1.5
--help	Print help message and exit
--index	Path to the relevant reference genome Bowtie indices
--outdir	Directory to write output files
--quiet	Suppress progress reports (except warnings)
--threads	Specify the number of threads, allowing simultaneous processing of different files (default: 1)
--version	Print the program version and exit
--zip	Compress output

HICUP_FILTER

The 'hicup_filter' Perl script classifies read pairs, identifying valid Hi-C di-tags

SYNOPSIS

hicup_filter [OPTIONS] --config [Configuration FILE]…

hicup_filter [OPTIONS] [hicup_mapper output file]

FUNCTION

The majority of reads generated by the hicup_mapper script are most likely valid Hi-C products, but a substantial minority are probably not and should be removed. The hicup_filter script processes paired reads together with the file created by hicup_digester to identify valid Hi-C pairs.

The names of the files to be processed and other parameters may be passed to the script using a configuration file or by command line arguments. As a minimum requirement the script requires: i) a list of hicup_mapper output file(s) and ii) a digested genome produced by hicup_digester. The presence of one or two restriction sites in the header line of the genome digest file instructs hicup_filter as to whether a sonication or double-digest protocol was followed.

Example input files:
Digest: Digest_Mouse_Genome_BglII_AluI_11-12-29_08-02-2012.txt
s_1_sequence.txt_truncated.pair
s_2_sequence.txt_truncated.pair
s_3_sequence.txt_truncated.pair

The program writes valid Hi-C read pairs to a file named the same as the original, but suffixed with '.sam'.

The script also creates a date-stamped file providing an overview of the types of ligation products created e.g. hicup_filter_summary_results_12-03-24_14-02-2012.txt. Rejected paired sequences are written to different files in a separate date-stamped folder e.g. HiC_rejects_12-03-24_14-02-2012.

Rejected paired reads:

A) Double-digest protocol

Unmapped: di-tags did not map to expected genomic restriction sites.

No ligation: read pair members both map to a single restriction enzyme 1 / restriction enzyme 2 double-digest fragment.

Wrong size: calculated di-tag length is not within the limits set by the size-selection step in the experimental protocol

Re-ligation: the original restriction enzyme 1 cut re-anneals. The resulting di-tags map to both the adjacent fragments; one read mapping to one fragment, the other member of the pair mapping to the adjacent fragment.

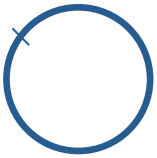
Self-ligation: a DNA fragment cut with restriction enzyme 1 circularises, ligating to itself, and is then linearized by the action of restriction enzyme 2.

Internal restriction enzyme 2 fragments: DNA fragments cut only by restriction enzyme 2 i.e. contain no Hi-C junction or restriction 1 cut sites.

Unclassified: while the ends of the fragments map to expected locations within the genome, the orientation of the cut sites do not correspond to any of the aforementioned categories.

B) Sonication protocol:

Same circularized: DNA fragment cut with the restriction enzyme circularizes, ligating to itself, and is then linearized by sonication



Same dangling ends: di-tag pairs map to the same restriction fragment and at least one end overlaps the restriction fragment cut site

Same internal: di-tag maps to a single restriction fragment but neither end of the di-tag overlaps the restriction fragment cut site

Re-ligation fragments: di-tag pairs map to adjacent restriction fragments which have re-ligated in the same orientation as found in the genome

Wrong size: calculated di-tag length is not within the limits set by the size-selection step in the experimental protocol

Contiguous: the di-tag could theoretically represent a contiguous DNA strand spanning several restriction fragments

COMMAND LINE OPTIONS

--config	Specify the optional configuration file
--digest	Specify the genome digest file (created by hicup_digester)
--help	Print program help and exit
--longest	Maximum allowable insert size (bps)
--outdir	Directory to write output files
--quiet	Suppress all progress reports
--shortest	Minimum allowable insert size (bps)
--threads	Specify the number of threads, allowing simultaneous processing of multiple files
--version	Print the program version and exit
--zip	Compress final output files using gzip, or if SAMtools is installed, to BAM format

HICUP_DEDUPLICATOR

The 'hicup_deduplicator' Perl script removes duplicated di-tags (retaining one copy of each) from the data set

SYNOPSIS

hicup_deduplicator [OPTIONS]... --config [Configuration FILE]...

hicup_deduplicator [OPTIONS]... [SAM/BAM FILES]...

FUNCTION

The Hi-C experimental protocol involves a PCR amplification step to generate enough material for sequencing. Consequently, the dataset generated by hicup_filter may contain PCR copies of the same di-tag. These PCR duplicates could result in incorrect inferences being drawn regarding the genomic conformation and so should be removed from the data set.

The names of the files to process can be passed to the script either by using a configuration file or command line arguments.

Example:
sample_544_PC_FL_500_lane2.sam
sample_545_PC_TAM_4_lane3.bam

The program creates SAM/BAM files named the same as the input files, only prefixed with 'uniques_':
sample_544_PC_FL_500_lane2.sam -> uniques_sample_544_PC_FL_500_lane2.sam
sample_545_PC_TAM_4_lane3.bam -> uniques_sample_545_PC_TAM_4_lane3.bam

The script also produces a date-stamped summary file (e.g. hicup_deduplicater_summary_results_12-15-44_11-06-2012.txt) reporting the number of unique di-tags present found in the data set and then classifies those unique di-tags as either cis di-tags (in which both reads are derived from the same chromosome) or trans di-tags (different chromosomes).

COMMAND LINE OPTIONS

--config	Specify the configuration file
--help	Print help message and exit
--outdir	Directory to write output files
--quiet	Suppress progress reports (except warnings)
--threads	Specify the number of threads, allowing simultaneous processing of multiple files
--version	Print the program version and exit
--zip	Compress output

REFERENCES

1. Lieberman-Aiden, E. et al. Comprehensive mapping of long-range interactions reveals folding principles of the human genome. *Science* **326**, 289–293 (2009).