Methpipe Manual

Qiang Song Elena Harris Andrew Smith

February 3, 2012

The Methpipe software package is a comprehensive tool chain for analyzing whole genome bisulfite sequencing data (BS-Seq). This documentation will guide you step by step how to perform the data analysis in a BS-Seq project with Methpipe. To facilitate understanding of the work flow, we divide the analysis procedure into four steps: (1) Pre-mapping processing includes assessing read quality and trimming adapters (if users use rmapbs for mapping, then trimming adapters as a separate step is not necessary); (2) Mapping sequenced reads to a reference genome; (3) Analyzing methylation level at a single site (CpG sites or non-CpG cytosines). (4) Higher level analysis includes identifying scores profiles (differential methylation or allelic methylation) and methylation characteristic regions (hypomethylated regions or differentially methylated regions).

Next figure shows detailed work flow of analysis with Methpipe. Processes and corresponding tools are shown with rectangles: incoming and outgoing arrows represent the number of input and output files respectively. Files with intermediate input or output data are shown as parallelograms and files with the final results are shown as ovals. The order of processes is important.

We will use provided test data as an example how to run our tools. Suppose that in a project you would like to analyze methylomes of two cell types, ESC and NHFF. There are paired-end sequenced reads for ESC and single-end reads for NHFF:

```
test_ESC_1.fastq, test_ESC_2.fastq, test_NHFF.fastq
```

Two test references are in the directory *test_ref_dir*. We will discuss tools in the same order as they appear in Fig. 1. For each tool, we will describe what it does, the tool's options, and input/output formats. Most of the tools use files in BED format and in Mapped Reads format, therefore we will provide these formats first, and later only refer to it by the name.

BED format:

- reference name <string>
- start position within reference <integer>: starts with 0
- end position within reference <integer>: starts with 0
- name <string>: depends on the file (read ID, C– content, HMR ID, DMR ID, etc.)
- score <float>: dependes on the file (number of mismatches, methylation level, score of HMR or DMR, etc.)
- strand <string>

Mapped Reads format:

- reference name <string>
- start position within reference <integer>: starts with 0
- end position within reference <integer>: starts with 0
- name <string>: depends on the file (read ID, C– content, HMR ID, DMR ID, etc.)

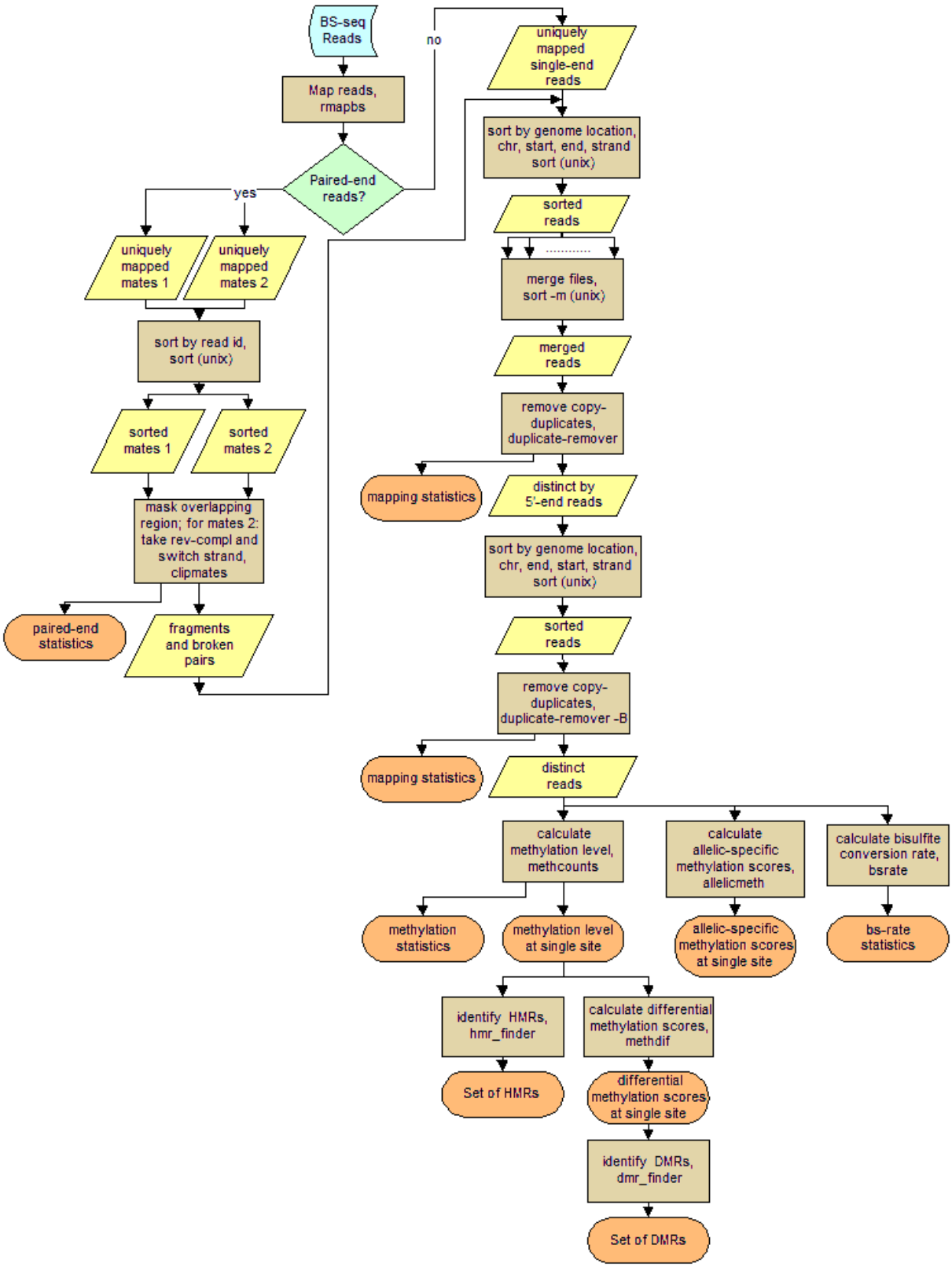


Figure 1: Methpipe's Workflow

- score <float>: dependes on the file (number of mismatches, methylation level, score of HMR or DMR, etc.)
- strand <string>
- read <string>
- score <string>

We will start with the tools for pre-mapping analysis (not shown in the workflow).

1 Pre-mapping processing

Before mapping sequenced reads to a reference genome, we need to pre-process the raw read sequences. In particular, we need to trim adapter sequences retained in the 3' end of raw reads. Further, we may be interested in examining the quality of reads in our library and visualizing raw reads in UCSC Genome Browser.

1.1 Trim adapters

As the length of sequenced reads has been increasing all the time, it is possible there are adapter sequences in the 3' end of some reads. These retained adapter sequences affect the mappability of the reads. Even if the reads are somehow mapped to the reference genome, the adapter sequences may introduce bias to our estimate of methylation frequency. Therefore it is necessary to trim these retained adapter sequences.

The program **trim-adapter** is used to trim adapter sequences, if any, from the 3' end of raw reads. Suppose the adapter sequence is *GATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGATCTCGTATGCCG*, we mask the adapter sequences from raw reads in the file *test_NHFF.fastq* with the following command:

```
$ ./trim-adapter -o test_NHFF_adapter_masked.fastq
-a GATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGATCTCGTATGCCG test_NHFF.fastq
```

If you would like to know the effective read length distribution after trimming adapter sequences and Ns from 3' end, you can add **-S** option to specify the output file for the distribution of effective read lengths. For example,

```
./trim-adapter -a GATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGATCTCGTATGCCG
       -S test_NHFF_read_length_distr.txt
               -o test_NHFF_adapter_masked.fastq test_NHFF.fastq
```

The file *test_NHFF_read_length_distr.txt* contains distribution of effective reads lengths after masking adapter sequences. Effective read length is the length of a read without Ns at the end of the read.

While we provide this **trim-adapter** as a standalone program for users who choose to map reads with another tool than rmapbs, its functionality is included in the **rmapbs** tool.

OPTIONS:

- **-a** <string>: adapter sequence
- **-o** <string>: output file (default: stdout)
- **-S** <string>: output file with effective read length distribution
- **-N** : for counting effective read length (without this option effective length includes the length of read without masked adapter sequence if any, and with this option it also includes the length of read without Ns at the end of the read)

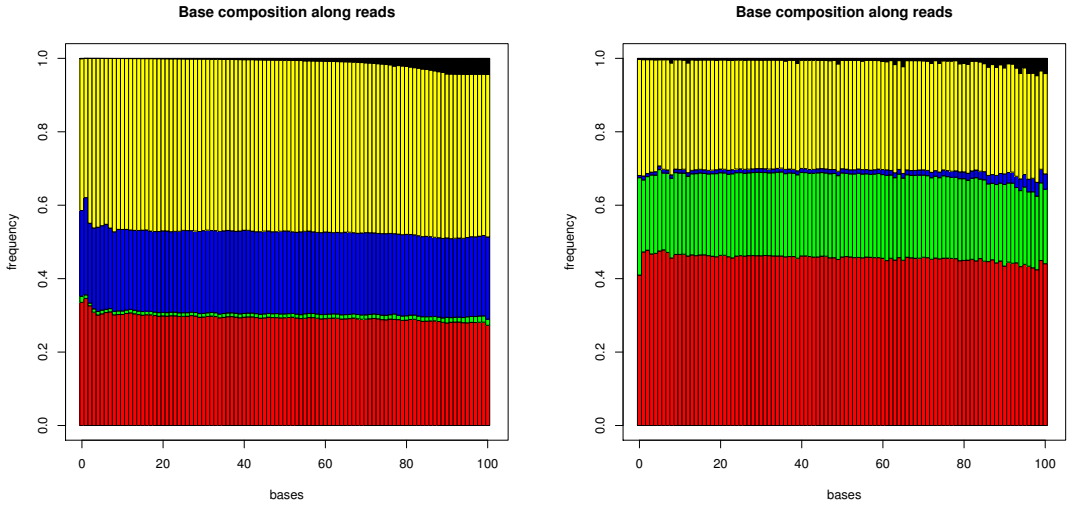
INPUT: reads in FASTQ format.
OUTPUT: reads in FASTQ format.

1.2 Assessing read quality

It is optional to assess read quality before mapping, however such assessment may help us to spot potential problems during library preparation and/or bisulfite sequencing. The program **read-quality-prof** is used to generate an average summary of base composition and quality scores from 5' to 3' along all reads. We run **read-quality-prof** as following

```
$ ./read-quality-prof -o test_NHFF_qual.txt test_NHFF.fastq
```

The output file *test_NHFF_qual.txt* can be visualized with the R software. Fig. 2 shows the base composition profile in the pair-end sequencing sample, where the left panel shows T-rich reads and the right panel A-rich reads. Note the small proportion of C and G reflects the effect of bisulfite conversion.



(a) T-rich reads

(b) A-rich reads

Figure 2: Base composition

OPTIONS:

- **-o** <string>: output file (default: stdout)

INPUT: reads in FASTQ format.
OUTPUT: self-explanatory column names are provided in the output file

2 Mapping

In the mapping step, you will map sequence reads to a reference genome. During bisulfite treatment, unmethylated cytosines in the original DNA sequences are converted to uracils, which are in turn incorporated as thymines during PCR amplification. We call such strand T-rich and its complementary strand A-rich (adenine-rich). When mapping T-rich reads to the reference genome, either a cytosine (C) or a thymine (T) in reads is considered valid match to a cytosine in the reference genome. But for A-rich reads, a adenine or a guanine is consider valid match to a guanine in the reference genome.

If using single-end sequencing, you will get T-rich reads only. If using pair-end sequencing, you will get both T-rich reads and A-rich reads (also referred to as $5'$ end and $3'$ mates respectively, or as simply mates 1 and mates 2). Next we will learn how to map bisulfite treated reads with **rmapbs** from either single-end sequencing or pair-end sequencing.

2.1 Mapping single-end sequencing reads

Mapping read sequences to the reference genome is done by **rmapbs**. To map paired-end reads, use **rmapbs** separately for $5'$ end mates and for $3'$ end mates. **IMPORTANT**: use **-A** option when mapping mates 2. Failure to use option **-A** with A-rich mates will result in mapping significantly less reads.

```
$ ./rmapbs -o test_ESC_1_mapped.mr -c test_ref_dir -m 3 -y test_ESC_1.fastq
$ ./rmapbs -o test_ESC_2_mapped.mr -c test_ref_dir -m 3 -y -A test_ESC_2.fastq
```

To map single end reads, run the command:

```
$ ./rmapbs -o test_NHFF_mapped.mr -c test_ref_dir -m 3 -y test_NHFF.fastq
```

OPTIONS:

- **-y** : allow N in read to match anything
- **-o** <string>: output file (default: stdout)
- **-c** <string>: FASTA file or directory containing files with chromosomes in FASTA format (if option **-s** is not used, files with chromosomes must have **.fa** extension)
- **-s** <string>: suffix of FASTA files (assumes -c indicates dir)
- **-F** <string>: file listing names of chromosome files (absolute paths to chromosome files)
- **-S** <integer>: number of seeds
- **-h** <integer>: weight of hit
- **-w** <integer>: width of the shortest reads in the input
- **-m** <integer>: maximum allowed mismatches
- **-a** <string>: file to write names of ambiguously mapped reads
- **-M** <integer>: maximum allowed mappings for a read
- **-W** : run in wildcard matching mode
- **-P** <float>: wildcard cutoff probability
- **-Q** : use quality scores (input must be FASTQ)
- **-A** : map using A/G bisulfite wildcards (ALWAYS use with mates 2)
- **-B** : allow CpG non-conversion to assist
- **-f** : use faster seeds (sensitive to 2 mismatches)
- **-C** <string>: clip the specified adaptor
- **-v** : print more run info
- **-G** : use original output format (BED)

INPUT: Reads are in FASTQ or FASTA format, and references are in FASTA format.
OUTPUT: Mapped Reads format.

3 Post-mapping processing

3.1 Paired-end reads

Next we will discuss some necessary processing steps that are applied to paired-end reads only. As has been noted before, if using pair-end sequencing, you will get both T-rich reads and A-rich reads. When estimating methylation frequency of a cytosine residue, we count the number of cytosines, which originate from methylated cytosines, and the number of thymines, which originate from unmethylated cytosines, and then we use the ratio $\frac{\#C}{\#C+\#T}$ as the estimate of methylation level of that base. Since A-rich reads are complementary to the original DNA fragments, from which they were sequenced, A-rich reads reflect methylation status of the cytosines on the strand, from which DNA fragments were originated. To avoid a bias in calculating methylation level, A-rich reads need to be converted to T-rich reads before we count Cs and Ts from these reads. Another bias in counting may arise from overlapping regions of two mates; to avoid this we mask overlapping region in one mate with Ns preserving all valid information. Conversion of A-rich reads to T-rich reads (taking reverse-complement of A-rich reads and switching the strand to which they mapped) and masking overlapping regions is done by **clipmates**. This tool works on sorted by read id (read name) mapped reads.

To sort mapped paired-end reads by read id, use **sort (unix)**:

```
$ sort -k 4,4 test_ESC_1_mapped.mr > test_ESC_1_sortname.mr
$ sort -k 4,4 test_ESC_2_mapped.mr > test_ESC_2_sortname.mr
```

Then use **clipmates** that takes two input files: one with T-rich mates and the other with A-rich mates.

```
$ ./clipmates -T test_ESC_1_sortname.mr -A test_ESC_2_sortname.mr
              -S test_ESC_clipmates_stat.txt -o test_ESC_clipmates.mr -L 500
```

This tool finds corresponding mates (whose names except for the last character match), and if such mates exist and are mapped correctly (to the same chromosome, to correct strands, with correct orientation) and within specified distance from each other (option **-L**, maximum fragment size), then the mates are combined into a single fragment with Ns filled between the mates if there is a gap between them. If mates could not be matched, then each mate is present in the output. Thus, the output file has Mapped Reads format and contains both entire DNA fragments or single mates. Note that this tool takes reverse-complement and switches the strands for mates 2.

OPTIONS:

- **-T** <string>: file with mates 1 (T-rich mates)
- **-A** <string>: file with mates 2 (A-rich mates)
- **-S** <string>: file with mapping statistics for paired-end reads
- **-o** <string>: output file
- **-L** <integer>: maximum length of DNA fragment (default: 1000)

INPUT: Two files with mates 1 and mates 2 in Mapped Reads format.

OUTPUT: Mapped Reads format with merged fragments and single mates. File with paired-end mapping statistics has self-explanatory content.

3.2 Paired-end and Single-end reads

During the post-mapping procession of mapped reads, we may need to sort reads according to different criterions. This is done with the **sort** utility provided in UNIX environment. Before calculating methylation level, users might consider removing copy-duplicates, the reads that were mapped to the same genomic location and that most likely are by-product of PCR rather than representatives of two distinct DNA molecules. To remove copy-duplicates, users must

aplly the following steps: (1) sort reads by genomic location (chrom, start, end, strand), (2) merge files with reads that came from the same DNA library and the same PCR and therefore likely to have shared copy-duplicates (3) run duplicate-remover that removes copy-duplicates mapped to the same start, (4) sort reads by genomic location (chrom, end, start, strand), and (5) run duplicate-remover with option **-B** that removes copy-duplicates mapped to the same end positions.

1. Sort reads by genomic location: chrom, start, end, strand.

```
$ sort -k 1,1 -k 2,2g -k 3,3g -k 6,6
       test_ESC_clipmates.mr > test_ESC_sorted_start.mr
$ sort -k 1,1 -k 2,2g -k 3,3g -k 6,6
       test_NHFF_mapped.mr > test_NHFF_sorted_start.mr
```

2. Merge sorted files using **merge -m (unix)**. (We do not provide test data for this function)

3. Run **duplicate-remover** to remove copy-duplicates sharing the same start position.

```
$ ./duplicate-remover -S test_ESC_duplremove_start_stat.txt
                      -o test_ESC_duplremove_start.mr  test_ESC_sorted_start.mr
```

4. Sort reads by genomic location: chrom, end, start, strand.

```
$ sort -k 1,1 -k 3,3g -k 2,2g -k 6,6
       test_ESC_duplremove_start.mr > test_ESC_sorted_end.mr
$ sort -k 1,1 -k 3,3g -k 2,2g -k 6,6
       test_NHFF_duplremove_start.mr > test_NHFF_sorted_end.mr
```

5. Run **duplicate-remover** to remove copy-duplicates sharing the same end position.

```
$ ./duplicate-remover -S test_ESC_duplremove_end_stat.txt
                       -o test_ESC_distinct.mr -B test_ESC_sorted_end.mr

$ ./duplicate-remover -S test_NHFF_duplremove_end_stat.txt
                       -o test_NHFF_distinct.mr -B test_NHFF_sorted_end.mr
```

The tool **duplicate-remover** collects reads and fragments mapped to the same genomic location (start or end), and chooses a random one to be the representative of the original DNA fragment. If there are any fragments (both mates are mapped and matched, and merged into a fragment), then the representative is chosen from fragments. When considering fragments mapped to the same genomic location, say fragments sharing same start, we treat fragments of different lengths as distinct DNA fragments. Below we provide options, and input/output format information for **duplicate-remover**.

OPTIONS:

- **-o** <string>: output file
- **-S** <string>: file with mapping statistics
- **-B** : With paired-end reads duplicate-remover must be used twice. This option is used when we run duplicate-remover for the second time on reads sorted by chrom, end, start, strand. This option assumes this sorted order and removes copy-duplicates that share the same end position.

INPUT: A single file in Mapped Reads format.
OUTPUT: Output file is in Mapped Reads format and statistics file has self-explanatory output.

4 Methylation level

The tool for calculating methylation level, **methcounts**, takes a single input file. Therefore, for a single methylome, users might need to merge multiple files from different libraries, and for this task please use **merge -m (unix)**. In this section we will learn how to estimate the methylation probability at a single cytosine loci. In mammals, DNA methylation exists mostly at cytosines in the context of CpG dinucleotides. This kind of methylation is symmetric because the cytosine on the complementary strand, which is base-paired with the guanine, is also methylated due to the interesting property of maintenance DNA methyltransferases. In mammalian stem cells and in plant cells, cytosines in sequence contexts other than CpG dinucleotides, such as CHG or CHH (H denotes adenines, thymines or cytosines), may also be methylated. This type of methylation is called asymmetric as the cytosines on the complementary strand are unnecessarily methylated. To calculate methylation level, use **methcounts**. This tool calculates methylation level for CpG sites by default; to calculate methylation level for all cytosines, please use option **-N**.

```
$ ./methcounts -M 40 -c test_ref_dir
               -o test_ESC_methcounts.bed test_ESC_distinct.mr
$ ./methcounts -N -M 40 -c test_ref_dir
               -o test_NHFF_methcounts.bed test_NHFF_distinct.mr
```

OPTIONS:

- **-o** <string>: Output file (default: stdout)
- **-c** <string>: FASTA file or directory containing chromosome(s)
- **-s** <string>: suffix of FASTA files (assumes -c indicates dir)
- **-N** : process non-CpG cytosines
- **-B** <integer>: buffer size (in records, not bytes; default: 100000)
- **-M** <integer or float>: max mismatches (can be fractional)
- **-C** <float>: cutoff for high-quality bases (assumes fastq reads)
- **-S** <string>: Output file with methylation statistics
- **-v** : print more run info

INPUT: A single file in Mapped Reads format.

OUTPUT: Output file is in BED format. File with statistics has self-explanatory content. Field **name** has format **CpG:X**, where **CpG** is cytosine content and has three possible values, namely **CpG, CHH, CHG**, and **X** is the total number of Cs and Ts mapped to the site. Field **score** is methylation level.

5 Estimating bisulfite conversion rate

By bisulfite sodiom treatment, unmethylated cytosines are converted to uracils, which are later read out as thymines. However this conversion may be incomplete due to various reasons, such as insufficient concentration, insufficient time of treatment or sequencing error. Therefore, bisulfite conversion rate, defined as ratio of converted unmethylated cytosines to all unmethylated cytosines, is an important parameter to assess the data quality of a BS-Seq experiment. To stimate bisulfite conversion rate, we need first to have some cytosines in the genome that are unmethylated. This includes three scenarios: (i) non-CpG cytosines in most mammalian tissues, (ii) spike-in Lamda genome and (iii) the chloroplast genome in plants. In these three scenarios, unmethylated cytosines should all be converted if the bisulfite treatment is perfect. Otherwise, some cytosines are unchanged. Therefore we count the number of cytosines covering these unmethylated cytosines and the number of thymines, the ratio of thymines to cytosines and thymines combined is the estimator of the bisulfite conversion rate. To estimate bisulfite conversion rate in mammals (over non-CpG cytosines), run:

```
$ ./bsrate -c test_ref_dir -M 40 -o test_ESC_bsrate.txt test_ESC_distinct.mr
$ ./bsrate -c test_ref_dir -M 40 -o test_NHFF_bsrate.txt test_NHFF_distinct.mr
```

To estimate bisulfite conversion rate over all cytosines, use option **-N**.

OPTIONS:

- **-o** <string>: Output file with bs-rate statistics
- **-c** <string>: FASTA file or directory containing chromosome(s)
- **-s** <string>: suffix of FASTA files (assumes -c indicates dir)
- **-N** : process all Cs
- **-B** <integer>: buffer size (in records; default: 100000)
- **-C** <integer>: cutoff for high-quality bases
- **-M** <integer>: max mismatches (can be fractional)
- **-v** : print more run info

INPUT: Mapped Reads format.
OUTPUT: Self-explanatory content.

6 Higher level analysis

6.1 Allelic methylation scores

We provide two tools to calculate methylation scores profiles. The tool **allelicmeth** calculates allelic-specific methylation score for each CpG site. The higher score, the more likely that the site has allelic-specific methylation. The allelic scores profile can be visualized using a genome browser. Use the following command to generate allelic scores profile:

```
$ ./allelicmeth -M 40 -c test_ref_dir
                -o test_ESC_allelicmeth.bed test_ESC_distinct.mr
$ ./allelicmeth -N -M 40 -c test_ref_dir
                -o test_NHFF_allelicmeth.bed test_NHFF_distinct.mr
```

OPTIONS:

- **-o** <string>: Output file
- **-c** <string>: FASTA file or directory containing chromosome(s)
- **-s** <string>: suffix of FASTA files (assumes -c indicates dir)
- **-N** : process all Cs
- **-B** <integer>: buffer size (in records; default: 100000)
- **-C** <integer>: cutoff for high-quality bases
- **-M** <integer>: max mismatches (can be fractional)
- **-v** : print more run info

INPUT: Mapped Reads format.
OUTPUT FORMAT:

- reference name <string>
- start position within reference <integer>: starts with 0
- end position within reference <integer>: starts with 0
- name <string>: in format **CpG:X**, where **X** is total number of reads mapped to the site
- score <float>: allelic methylation score
- mm <integer>: total number of reads with methylation status of two consecutive CpGs methylated and methylated
- mu <integer>: total number of reads with methylation status of two consecutive CpGs methylated and unmethylated
- um <integer>: total number of reads with methylation status of two consecutive CpGs unmethylated and methylated
- uu <integer>: total number of reads with methylation status of two consecutive CpGs unmethylated and unmethylated

6.2 Differential methylation scores

Our tool **methyldiff** generates differential methylation scores profile between two methylomes. Given two methylomes (the result of **methcounts**), it calculates the probability that a CpG is significantly less methylated in one methylome than in the other. To generate differential methylation scores profile, run **methyldiff**:

```
$ ./methdiff -o test_NHFF_ESC_methdiff.bed
             test_NHFF_methcounts.bed test_ESC_methcounts.bed
```

OPTIONS:

- **-o** <string>: Output file.
- **-p** <integer>: pseudocount for the contingency table
- **-v** : print more run info

INPUT: Two files in BED format, the results of **methcounts**.
OUTPUT: BED format file, with field **name** having format **CpG:X:Y**, where **X** is the total number of reads mapped to the site in one methylome, and **Y** is the total number of reads mapped to the site in the other methylome. Field **score** contains differential methylation score.

6.3 Hypomethylated regions, HMRs

Hypomethylated regions, HMRs, are contiguous regions with very low methylation. To identify HMRs, use the results of **methcounts** as input and run **hmr_finder**:

```
$ ./hmr_finder -i 20 -o test_ESC_HMR.bed test_ESC_methcounts.bed
$ ./hmr_finder -i 20 -o test_NHFF_HMR.bed test_NHFF_methcounts.bed
```

OPTIONS:

- **-o** <string>: Output file

- **-i** <integer>: maximum number of iterations
- **-P** <string>: input file with parameters for HMM
- **-p** <string>: output file with parameters for HMM
- **-v** : print more run info

INPUT: BED format file, results of **methcounts**.
OUTPUT: BED format file with HMRs.

6.4 Differentially methylated regions, DMRs

Once differential methylation scores are calculated, the tool **dmr_finder** can be used to identify differentially methylated regions, DMRs. Option **-l** regulates the output: without this option, DMRs are calculated, in which methylation is less in one methylome, and with this option DMRs are calculated with methylation less in the other methylome. Run **dmr_finder** to identify DMRs.

```
$ ./dmr_finder -o test_NHFF_ESC_DMR.bed -d 500 -b 10 -C 10 -m 200
               -c 0.7 test_NHFF_ESC_methdif.bed
```

OPTIONS:

- **-o** <string>: Output file
- **-b** <integer>: bandwidth in bases used for smoothing scores profile
- **-c** <float>: cutoff for peak confidence score
- **-C** <integer>: minimum number of CpGs per DMR
- **-m** <integer>: minimum size of DMR in bases
- **-l** : use lower cutoff (¡ 1 - cutoff)
- **-v** : print more run info

INPUT: BED format file, the result of **methdiff**.
OUTPUT: BED format file with DMRs.