

Contents

| | | |
|----------|-----------------------------|-----------|
| 1 | What is CGmapTools | 3 |
| 2 | File Formats | 5 |
| 2.1 | ATCGmap Format | 5 |
| 2.2 | CGmap Format | 5 |
| 2.3 | ATCGbz Format | 7 |
| 2.4 | CGbz Format | 8 |
| 3 | File Manipulation | 11 |
| 3.1 | convert | 11 |
| 3.2 | fetch | 12 |
| 3.3 | refill | 13 |
| 3.4 | intersect | 14 |
| 3.5 | merge2 | 15 |
| 3.6 | mergelist | 16 |
| 3.7 | sort | 18 |
| 3.8 | split | 19 |
| 3.9 | select | 19 |
| 4 | SNV calling | 21 |
| 4.1 | BaysWC strategy | 21 |
| 4.2 | BinomWC strategy | 23 |
| 4.3 | Performance | 23 |
| 5 | Methylation Analysis | 27 |
| 5.1 | dms | 27 |
| 5.2 | dmr | 28 |
| 5.3 | asm | 29 |
| 5.4 | mbed | 32 |
| 5.5 | mbin | 33 |
| 5.6 | mmbin | 35 |
| 5.7 | mfg | 35 |
| 5.8 | mstat | 37 |
| 5.9 | mtr | 38 |
| 6 | Coverage Analysis | 41 |
| 6.1 | oac | 41 |
| 6.2 | mec | 43 |
| 7 | Graphics | 47 |
| 7.1 | lollipop | 47 |
| 7.2 | heatmap | 50 |
| 7.3 | fragreg | 51 |

| | | |
|----------|------------------------|-----------|
| 7.4 | tanghulu | 54 |
| 8 | Other Utilities | 57 |
| 8.1 | findCCGG | 57 |
| 8.2 | bed2fragreg | 57 |

Chapter 1

What is CGmapTools

DNA methylation is crucial for a wide variety of biological processes. With the development of high throughput methylome profiling methods, huge volumes of data are generated and in egent need of computational tools for data analysis.

We proposed **CGmapTools**, a bisulfite sequencing analysis toolset with enhanced features on SNV calling and allele specific methylations and visualizations, in hope to set up a standard for bisulfite sequencing data related manipulation, including better data storage, extraction, visualization and improved performance in SNP calling. We also provide dozens of utilities and a seamless pipeline for bisulfite sequencing data analysis.

Command

```
cgmaptools -h
```

```
# Program : cgmaptools (Tools for analysis in CGmap/ATCGmap format)
# Version:  0.1.1
# Usage:   cgmaptools <command> [options]
# Commands:
#   -- File manipulation
#       convert      + data format conversion tools
#       fetch        + fetch a region by random accessing
#       refill       + refill the missing columns
#       intersect    + intersect two files
#       merge2       + merge two files into one
#       mergelist    + merge a list of files
#       sort         + sort lines by chromosome and position
#       split        + split file by chromosomes
#       select       + select lines by region/site
#   -- SNV analysis
#       snv          snv analysis
#   -- Methylation analysis
#       dms          differentially methylated site analysis
#       dmr          differentially methylated region analysis
#       asm          allele-specific methylation analysis
#       mbed         average methylation level in regions
#       mbin         * single sample, mC levels in bins
#       mmbin        multiple samples, mC levels in bins
#       mfg          methlation levels across fragmented region
#       mstat        * methyaltion statistic
#       mtr          methylation level to each region
```

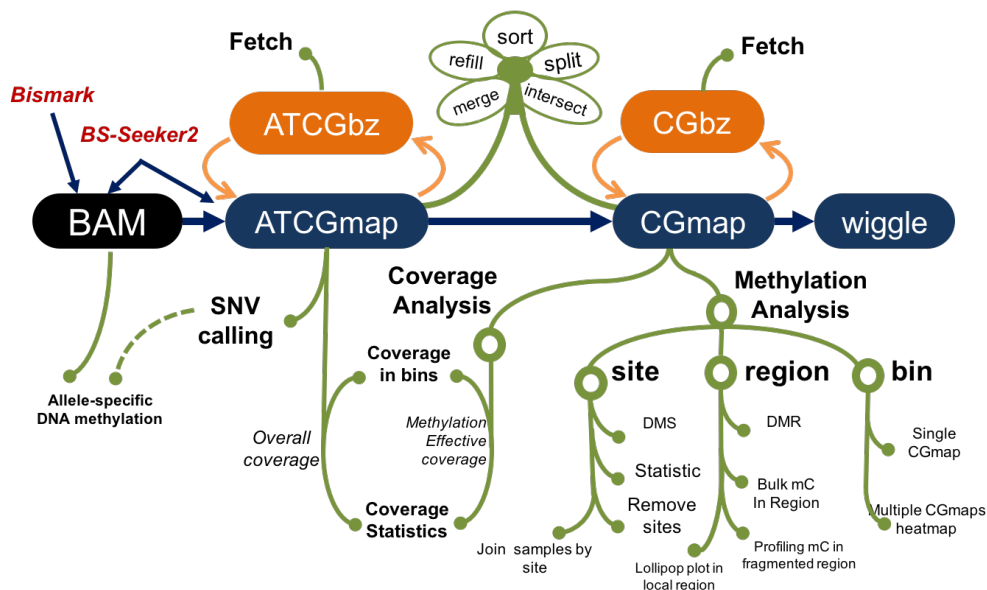


Figure 1.1: Schematic diagram of CGmapTools

```
# -- Coverage analysis
#   oac      ** overall coverage (for ATCGmap)
#   mec      ** methylation effective coverage (for CGmap)
# -- Graph related functions
#   lollipop  * show local mC levels as lollipop bars
#   heatmap   * global mC distribution for multiple samples
#   fragreg   * show mC profile across fragmented regions
#   tanghulu  * show local mapped reads in Tanghulu shape
# -- Other Utils
#   findCCGG  get MspI cutting sites for RRBS
#   bed2fragreg get fragmented region based on region
# Note:
#   Commands support figures generation are marked with "*"
#   Commands contain sub-commands are marked with "+"
# Authors:
#   GUO, Weilong; guoweilong@126.com; http://guoweilong.github.io
#   ZHU, Ping; pingzhu.work@gmail.com; http://perry-zhu.github.io
```

Chapter 2

File Formats

To facilitate high throughput data manipulation and reduce storage usage, several file format have been proposed and generally accepted as the standard. Due to these great efforts (e.g. SAM/BAM and VCF), data analysis and tool development become more easier and highly efficient. However, when it comes to bisulfite sequencing data, currently, available tools possess their own tool specific data format. In consequence, integrating results from several tools leads to extra efforts in unifying data format and developing customized tools, which is time consuming and error prone.

The widely-used BS-seq alignment software *BS-Seeker2* defines **CGmap** and **ATCGmap** file formats for the representation of DNA methylomes. In CGmapTools, we used **ATCGmap** and **CGmap** as the standard file format interface, so that to simplify the development of downstream DNA methylation analysis tools and to provide standard formats for storing and sharing the DNA methylomes.

In *CGmapTools*, we designed novel binary formats: **CGbz** and **ATCGbz** for less coverage and improvements in random-accessing data in large data in hard-disk.

2.1 ATCGmap Format

Similar with **pileup**, **ATCGmap** format summarizes the information of mapped reads covered on each nucleotide on both strands, specially designed for BS-seq data.

Here, we defined ATCGmap file format to integrate both mapping and coverage of non-cytosine and cytosine sites with estimated DNA methylation in a single file.

- **Example**

| | | | | | | | | | | | | | | | |
|------|---|---------|-----|----|---|----|----|---|---|---|----|---|---|---|------|
| chr1 | T | 3009410 | -- | -- | 0 | 10 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | na |
| chr1 | C | 3009411 | CHH | CC | 0 | 10 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0.0 |
| chr1 | C | 3009412 | CHG | CC | 0 | 10 | 0 | 0 | 0 | 0 | 9 | 1 | 0 | 0 | 0.0 |
| chr1 | C | 3009413 | CG | CG | 0 | 10 | 50 | 0 | 0 | 0 | 20 | 1 | 0 | 0 | 0.83 |

- **Column Description**

2.2 CGmap Format

In cases we only want to retain DNA methylation on cytosines to save storage usage, we defined another file format called **CGmap** which provides sequence context and estimated DNA methylation level of any covered cytosines on the reference genome.

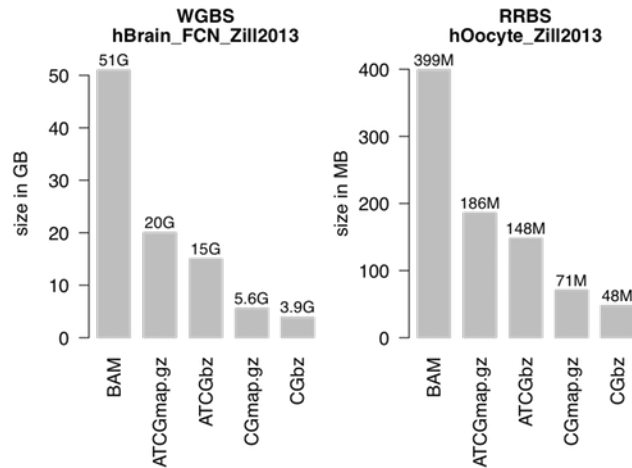


Figure 2.1: Size of multiple file formats

| Col | Field | Type | Regexp/Range | Brief description |
|-----|-------|--------|--------------------------------|---|
| 1 | CHR | String | [!-?A-~]{ 1,118 } | Query template NAME |
| 2 | NUC | Char | [ATCGN-] | The nucleotide on reference genome |
| 3 | POS | Int | [0,2 ³² -1] | 1-based leftmost mapping position |
| 4 | CONT | String | {"--", "CG", "CHG", "CHH"} | context |
| 5 | DINUC | String | {"--", "CA", "CT", "CC", "CG"} | Dinucleotide context |
| 6 | WA | Int | [0,2 ¹⁴ -1] | Counts of reads on Watson strand support Adenine |
| 7 | WT | Int | [0,2 ¹⁴ -1] | Counts of reads on Watson strand support Thymine |
| 8 | WC | Int | [0,2 ¹⁴ -1] | Counts of reads on Watson strand support Cytosine |
| 9 | WG | Int | [0,2 ¹⁴ -1] | Counts of reads on Watson strand support Guanine |
| 10 | WN | Int | [0,2 ⁶ -1] | Counts of reads on Watson strand support None |
| 11 | CA | Int | [0,2 ¹⁴ -1] | Counts of reads on Crick strand support Adenine |
| 12 | CT | Int | [0,2 ¹⁴ -1] | Counts of reads on Crick strand support Thymine |
| 13 | CC | Int | [0,2 ¹⁴ -1] | Counts of reads on Crick strand support Cytosine |
| 14 | CG | Int | [0,2 ¹⁴ -1] | Counts of reads on Crick strand support Guanine |
| 15 | CN | Int | [0,2 ⁶ -1] | Counts of reads on Crick strand support None |
| 16 | METH | Float | [0,1] or "na" | Methylation level or "Not Available" |

Figure 2.2: Description of ATCGmap

| Col | Field | Type | Regexp/Range | Brief description |
|-----|-------|--------|--------------------------------|--|
| 1 | CHR | String | [!-?A-~]{ 1,118 } | Query template NAME |
| 2 | NUC | Char | [ATCGN-] | The nucleotide on reference genome |
| 3 | POS | Int | [0,2 ³² -1] | 1-based leftmost mapping position |
| 4 | CONT | String | {"--", "CG", "CHG", "CHH"} | context |
| 5 | DINUC | String | {"--", "CA", "CT", "CC", "CG"} | Dinucleotide context |
| 6 | METH | Float | [0,1] or "na" | Methylation level or "Not Available" |
| 7 | MC | Int | [0,2 ¹² -1] | Counts of reads support methyalted Cytosine |
| 8 | NC | Int | [0,2 ¹² -1] | Counts of reads support all Cytosine |

Figure 2.3: Description of CGmap

| Field | Description | Type | Value |
|-----------------|------------------------------------|-------------|-------|
| N_chr | # chromosome | uint32_t | |
| List of ChrInfo | | | |
| CHR | Name of chromosome | char [118] | |
| count | # of ATCGbzT under this chromosome | uint32_t | |
| List of ATCGbzT | | | |
| pos | Position on this chromosome | uint32_t | |
| info | The mapping information | uint32_t[4] | |

Figure 2.4: Data structure of ATCGbz

- **Example**

```
chr1  G  3000851  CHH  CC  0.1  1  10
chr1  C  3001624  CHG  CA  0.0  0  9
chr1  C  3001631  CG   CG  1.0  5  5
chr1  G  3001632  CG   CG  0.9  9  10
```

- **Column Description**

2.3 ATCGbz Format

ATCGbz format is the binary compressed version for **ATCGmap** format. **ATCGmap** format is readable, while quite large for storing, and difficult for fetching information in a specific position. **ATCGbz** is defined as the sorted binary version, that storing all information of **ATCGmap** into standard binary form, largely reduced the storage requirement, and also supporting fast retrieval of methylation information for any position on genome.

- **Data structure**

- **Related command**

Command

```
cgmaptools fetch atcgbz -h
```

```
#
```

```
# Usage: cgmaptools fetch atcgbz -b <ATCGbz> -C <CHR> -L <LeftPos> -R <RightPos>
```

| info : uint32_t[4] 128 bit | | | | | | | | | | | | | | |
|----------------------------|----------------------------------|----------------|---|-------|----------|-------|-------|----------|------|-------|-------|----------|-------|-------|
| info [0] | | | | | info [1] | | | info [2] | | | | info [3] | | |
| 1 | 2,3 | 4 | 5-18 | 19-32 | 1-14 | 15-28 | 29-32 | 1-2 | 3-16 | 17-30 | 31-32 | 1-12 | 13-26 | 27-32 |
| strand | Dinuc | Context | WA | WT | WC | WG | WN | CA | CT | CC | CG | CN | | |
| 0 = + 1 = - | 00=CA 01=CC 10=CT 11=CG | 0=CNH 0=CNG | Count of reads mapped on Watson/Crick strands, supporting A, T, C, G or N | | | | | | | | | | | |
| | 111 = "--" not CGG | | | | | | | | | | | | | |

Figure 2.5: Data structure of info field of ATCGbz

| Field | Description | Type | Value |
|-----------------|----------------------------------|-----------|-------|
| N_chr | # chromosome | uint32_t | |
| List of ChrInfo | | | |
| CHR | Name of chromosome | char[118] | |
| count | # of CGbzT under this chromosome | uint32_t | |
| List of CGbzT | | | |
| pos | Position on this chromosome | uint32_t | |
| info | The mapping information | uint32_t | |

Figure 2.6: Data structure of ATCGbz

```
#      (aka ATCGbzFetchRegion)
#      Description: Convert ATCGbz format to ATCGmap format.
#      Contact:    Guo, Weilong; guoweilong@126.com
#      Last update: 2016-12-07
#
#      Options:
#
#      -h, --help          output help information
#      -b, --ATCGbz <arg> output ATCGbz file
#      -C, --CHR <arg>     specify the chromosome name
#      -L, --leftPos <arg> the left position
#      -R, --rightPos <arg> the right position
```

2.4 CGbz Format

CGbz format is the binary compressed version for **CGmap** format.

- Data structure
- Related command

Command

```
cgmaptools fetch cgbz -h
```

```
#
#      Usage: cgmaptools fetch cgbz -b <CGbz> -C <CHR> -L <LeftPos> -R <RightPos>
```


| info : uint32_t | | | | |
|-----------------|----------------------------------|----------------|---|---------------------------------|
| 1 | 2,3 | 4 | 5-18 | 19-32 |
| strand | Dinuc | Context | MC | NC |
| 0: + 1: - | 00=CA 01=CC 10=CT 11=CG | 0=CNH 0=CNG | # reads support methylated cytosine | # reads support all cytosine |
| | 111 = "--" not CGG | | | |

Figure 2.7: Data structure of info field of CGBz

```
#          (aka CGvzFetchRegion)
#  Description: Convert CGBz file to CGmap format.
#  Contact: Guo, Weilong; guoweilong@126.com
#  Last update: 2016-12-07
#
#  Options:
#
#    -h, --help          output help information
#    -b, --CGBz <arg>   output CGBz file
#    -C, --CHR <arg>     specify the chromosome name
#    -L, --leftPos <arg> the left position
#    -R, --rightPos <arg> the right position
```


Chapter 3

File Manipulation

CGmapTools provides multiple utilities to manipulate files in ATCGmap and CGmap format or compressed ATCGbz/CGbz format.

Usage: `cgmaptools <convert|fetch|refill|intersect|merge2|mergelist|sort|split|select|> [options]`

3.1 convert

- **Description** : File format conversion.
- **Table of command for converting formats:**

| Commands | From | To |
|-----------------------|---------|-----------------|
| bam2cgmap | BAM | CGmap & ATCGmap |
| atcgmap2atcgbz | ATCGmap | ATCGbz |
| atcgbz2atcgmap | ATCGbz | ATCGmap |
| atcgmap2cgmap | ATCGmap | CGmap |
| cgmap2cgbz | CGmap | CGbz |
| cgbz2cgmap | CGbz | CGmap |
| cgmap2wig | CGmap | WIG |
| bismark2cgmap | Bismark | CGmap |

- **Command**

```
cgmaptools convert -h
```

```
# Usage:    cgmaptools convert <command> [options]
# Version:  0.1.1
# Commands:
#   bam2cgmap      BAM      => CGmap & ATCGmap
#   atcgmap2atcgbz ATCGmap => ATCGbz
#   atcgbz2atcgmap ATCGbz  => ATCGmap
#   atcgmap2cgmap  ATCGmap => CGmap
#   cgmap2cgbz     CGmap   => CGbz
#   cgbz2cgmap     CGbz    => CGmap
#   cgmap2wig      CGmap   => WIG
#   bismark2cgmap  Bismark => CGmap
```

- **Example :**

- BAM to CGmap

```
cgmaptools convert bam2cgmap -b WG.bam -g genome.fa --rmOverlap -o WG
```

- BAM to CGmap

```
cgmaptools convert bam2cgmap -b RR.bam -g genome.fa --rmOverlap -o RR
```

- ATCGmap to ATCGbz

```
cgmaptools convert atcgmap2atcgbz -c WG.ATCGmap.gz -b WG.ATCGbz
```

- ATCGbz to ATCGmap

```
cgmaptools convert atcgbz2atcgmap -c WG2.ATCGmap.gz -b WG.ATCGbz
```

- CGmap to CGbz

```
cgmaptools convert cgmap2cgbz -c RR.CGmap.gz -b RR.CGbz
```

- CGbz to CGmap

```
cgmaptools convert cgbz2cgmap -c RR2.CGmap.gz -b RR.CGbz
```

- CGmap to WIG

```
cgmaptools convert cgmap2wig -i <CGmap> [-w <wig>] [-c <INT> -b <float>]
```

- bismark output to CGmap

```
cgmaptools convert bismark2cgmap -i bismark.dat -o output.CGmap
```

Note: please refer to the help message for usage details using -h option.

3.2 fetch

- **Description:** Fastly access methylation data in specified region.

- **Command**

```
cgmaptools fetch -h
```

```
# Usage:    cgmaptools fetch <command> [options]
# Version:  0.1.1
# Commands:
#   atcgbz    fetch lines from ATCGbz
#   cgbz      fetch lines from CGbz
```

3.2.1 fetch cgbz

- **Command**

```
cgmaptools fetch cgbz -h
```

```
#
# Usage: cgmaptools fetch cgbz -b <CGbz> -C <CHR> -L <LeftPos> -R <RightPos>
#       (aka CGvzFetchRegion)
# Description: Convert CGbz file to CGmap format.
# Contact: Guo, Weilong; guoweilong@126.com
# Last update: 2016-12-07
```

```
#
# Options:
#
# -h, --help          output help information
# -b, --CGbz <arg>    output CGbz file
# -C, --CHR <arg>     specify the chromosome name
# -L, --leftPos <arg> the left position
# -R, --rightPos <arg> the right position
```

- Example :

```
cgmmaptools fetch cgbz -b RR.CGbz -C chr3 -L 2200 -R 2400
```

3.2.2 fetch atcgbz

- Command

```
cgmmaptools fetch atcgbz -h
```

```
#
# Usage: cgmmaptools fetch atcgbz -b <ATCGbz> -C <CHR> -L <LeftPos> -R <RightPos>
#       (aka ATCGbzFetchRegion)
# Description: Convert ATCGbz format to ATCGmap format.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last update: 2016-12-07
#
# Options:
#
# -h, --help          output help information
# -b, --ATCGbz <arg> output ATCGbz file
# -C, --CHR <arg>     specify the chromosome name
# -L, --leftPos <arg> the left position
# -R, --rightPos <arg> the right position
```

- Example :

```
cgmmaptools fetch atcgbz -b WG.ATCGbz -C chr2 -L 90 -R 100
```

3.3 refill

- Command

```
cgmmaptools refill -h
```

```
# Usage: cgmmaptools refill [-i <CGmap>] -g <genome.fa> [-o output]
#       (aka CGmapFillContext)
# Description: Fill the CG/CHG/CHH and CA/CC/CT/CG context.
#              Other fields will not be affected.
#              Can be applied to ATCGmap file.
# Contact:    Guo, Weilong; guoweilong@126.com;
# Last Update: 2018-01-02
# Index Ex:
#   Chr1  C      3541  -      -      0.0    0      1
# Output Ex:
#   Chr1  C      3541  CG     CG     0.0    0      1
```

```
#
# Options:
#   -h, --help      show this help message and exit
#   -i STRING       Input CGmap file (CGmap or CGmap.gz)
#   -g STRING       genome file, FASTA format (gzipped if end with '.gz')
#   -o STRING       Output file name (gzipped if end with '.gz')
#   -0, --0-base    0-based genome if specified [Default: 1-based]
```

- **File formats:**

The input CGmap file, which is lacking C context on the 3rd and 4th columns:

```
Chr1    C      3541    -      -      0.0    0      1
```

After refill processing, the CGmap file would be as below, added C context information:

```
Chr1    C      3541    CG      CG      0.0    0      1
```

- **Example:**

```
zcat RR2.CGmap.gz | gawk -F"\t" -vOFS="\t" '{ $4="-"; $5="-"; print; }' | cgmaptools
refill -g genome.fa -o RR3.CGmap.gz
```

3.4 intersect

- **Command**

```
cgmaptools intersect -h
```

```
# Usage: cgmaptools intersect [-1 <CGmap_1>] -2 <CGmap_2> [-o <output>]
#       (aka CGmapIntersect)
# Description:
#   Get the intersection of two CGmap files. Contact: Guo, Weilong; guoweilong@126.com
# Last Update: 2018-04-10
# Output Format:
#   Chr1 C 3541 CG CG 0.8 4 5 0.4 4 10
# When 1st CGmap file is:
#   Chr1 C 3541 CG CG 0.8 4 5
# ,and 2nd CGmap file is:
#   Chr1 C 3541 CG CG 0.4 4 10
#
# Options:
#   -h, --help      show this help message and exit
#   -1 CGmap File   File name, end with .CGmap or .CGmap.gz.
#   -2 CGmap File   standard input if not specified
#   -o OUTFILE      To standard output if not specified. Compressed output
#                   if end with .gz
#   -C CONTEXT, --context=CONTEXT
#                   specific context: CG, CH, CHG, CHH, CA, CC, CT, CW
#                   use all sites if not specified
```

- **Example**

```
cgmaptools intersect -1 WG.CGmap.gz -2 RR.CGmap.gz -C CG -o intersect.CG.gz
```

- **Output format**

- **Example**

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|--------------------------------|--|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | NUC | Char | [ATCGN-] | The nucleotide on reference genome |
| 3 | POS | Int | [0,2 ³² -1] | 1-based leftmost mapping position |
| 4 | CONT | String | {"--", "CG", "CHG", "CHH"} | context |
| 5 | DINUC | String | {"--", "CA", "CT", "CC", "CG"} | Dinucleotide context |
| 6 | METH_1 | Float | [0,1] or "na" | Methylation level in Sample 1 |
| 7 | MC_1 | Int | [0,2 ¹² -1] | Counts of reads support methyalted Cytosine in Sample 1 |
| 8 | NC_1 | Int | [0,2 ¹² -1] | Counts of reads support all Cytosine in Sample 1 |
| 9 | METH_2 | Float | [0,1] or "na" | Methylation level in Sample 2 |
| 10 | MC_2 | Int | [0,2 ¹² -1] | Counts of reads support methyalted Cytosine in Sample 2 |
| 11 | NC_2 | Int | [0,2 ¹² -1] | Counts of reads support all Cytosine in Sample 2 |

Figure 3.1: Output format description for cgmaptools intersect

```

Chr1  C  3541  CG  CG  0.8  4  5  0.4  4  10
Chr1  C  3542  CG  CG  0.8  3  5  0.2  2  10
Chr1  C  3545  CHG CA  0.0  0  5  0.1  1  10

```

– Column Description

3.5 merge2

Command

```
cgmaptools merge2 -h
```

```

# Usage:    cgmaptools merge2 <command> [options]
# Version:  0.1.1
# Commands:
#   atcgmap    merge two ATCGmap files into one
#   cgmap      merge two CGmap files into one

```

3.5.1 merge2 atcgmap

Command

```
cgmaptools merge2 atcgmap -h
```

```

# Unknown option: -h
# Usage:  cgmaptools merge2 atcgmap -1 <ATCGmap> -2 <ATCGmap>
#         (aka ATCGmapMerge)
# Contact: Guo, Weilong; guoweilong@126.com;
# Last Update: 2016-12-07
# Options:
#   -1    Input, 1st ATCGmap file
#   -2    Input, 2nd ATCGmap file
# Output to STDOUT in ATCGmap format
# Tips: Two input files should have the same order of chromosomes

```

- Example

```
cgmaptools merge2 atcgmap -1 WG.ATCGmap.gz -2 RR.ATCGmap.gz | gzip > merge.ATCGmap.gz
```

3.5.2 merge2 cgmap

Command

```
cgmaptools merge2 cgmap -h
```

```
# Usage: cgmaptools merge2 cgmap -1 <CGmap_1> -2 <CGmap_2> [-o <output>]
#      (aka CGmapMerge)
# Description: Merge two CGmap files together.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Note: The two input CGmap files should be sorted in the same order first.
#
#
# Options:
#   -h, --help  show this help message and exit
#   -1 FILE      File name end with .CGmap or .CGmap.gz
#   -2 FILE      If not specified, STDIN will be used.
#   -o OUTFILE   CGmap, output file. Use STDOUT if omitted (gzipped if end with
#                '.gz').
```

- Example

- Example command :

```
cgmaptools merge2 cgmap -1 WG.CGmap.gz -2 RR.CGmap.gz | gzip > merge.CGmap.gz
```

3.6 mergelist

- Command

```
cgmaptools mergelist -h
```

```
# Usage:    cgmaptools mergelist <command> [options]
# Version:  0.1.1
# Commands:
#   tomatrix  mC levels matrix from multiple files
#   tosingle  merge list of input files into one
```

3.6.1 mergelist tomatrix

- Command

```
cgmaptools mergelist tomatrix -h
```

```
# Usage: cgmaptools mergelist tomatrix [-i <index>] -f <IN1,IN2,...> -t <tag1,tag2,...> [-o output]
#      (aka CGmapFillIndex)
# Description: Fill methylation levels according to the Index file for CGmap files in list.
# Contact:    Guo, Weilong; guoweilong@126.com;
# Last Updated: 2018-01-02
# Index format Ex:
```


| Col | Field | Type | Regexp/Range | Brief description |
|-----|-------|--------|------------------------|---------------------|
| 1 | CHR | String | [!-?A-~]* | Query template NAME |
| 2 | POS | Int | [0,2 ³² -1] | Position |

Figure 3.2: Format description for INDEX file

```
#      chr10    100005504
#  Output format Ex:
#      chr      pos      tag1      tag2      tag3
#      Chr1     111403  0.30      nan      0.80
#      Chr1     111406  0.66      0.40     0.60
#
#  Options:
#      -h, --help  show this help message and exit
#      -i FILE     TXT file, index file, use STDIN if omitted
#      -f STRING   List of (input) CGmap files (CGmap or CGmap.gz)
#      -t STRING   List of tags, same order with '-f'
#      -c INT      minimum coverage [default: 1]
#      -C INT      maximum coverage [default: 200]
#      -o STRING   Output file name (gzipped if end with '.gz')
```

- **Example**

```
zcat RR*.CGmap.gz WG.CGmap.gz | gawk '$8>=5' | cut -f1,3 | sort -u | cgmapttools sort
-c 1 -p 2 > index
```

```
cgmapttools mergelist tomatrix -i index -f RR.CGmap.gz,RR2.CGmap.gz,WG.CGmap.gz -t
RR,RR2,WG -c 5 -C 100 -o matrix.CG.gz
```

- **Format for Index file**

- **Example**

```
Chr1    940
Chr1    1840
Chr2    9060
```

- **Column Description**

- **Format for output file**

- **Example**

```
chr      pos      tag1      tag2      tag3
Chr1     111403  0.05      nan      0.02
Chr1     111500  1.00      0.80     0.60
Chr2     20000   0.96      0.33     0.66
```

- **Column Description**

3.6.2 mergelist tosingle

- **Command**

```
cgmapttools mergelist tosingle -h
```

```
#  Usage: cgmapttools mergelist tosingle -i f1,f2,...,fn [-o <output>]
```

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|------------------------|-------------------------------|
| 1 | CHR | String | [!-?A-~]* | Query template NAME |
| 2 | POS | Int | [0,2 ³² -1] | Position |
| 3 | METH_1 | Float | [0.00, 1.00] | Methylation level in sample 1 |
| ... | ... | ... | ... | ... |
| n | METH_n | Float | [0.00, 1.00] | Methylation level in sample n |

Figure 3.3: Output format description for cgmaptools fill tomatrix

```
#      (aka MergeListOfCGmap)
# Description: Merge multiple CGmap/ATCGmap files into one.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-04-10
# Note: Large memory is needed.
#      Split input by chromosome for merge will save some memory.
#
#
# Options:
#   -h, --help  show this help message and exit
#   -i FILE     List of input files; gzipped file ends with '.gz'; seperated by
#               comma without gap
#   -f FILE     cgmap or atcgmap [Default: cgmap]
#   -o OUTFILE  To standard output if not specified; gzipped file if end with
#               '.gz'
```

- Example

3.7 sort

- Command

```
cgmaptools sort -h
```

```
# Usage: Sort_chr_pos [-i <input>] [-c 1] [-p 3] [-o output]
# Author : Guo, Weilong; guoweilong@gmail.com; 2014-05-11
# Last Update: 2018-01-02
# Description: Sort the input files by chromosome and position.
#      The order of chromosomes would be :
#      "chr1 chr2 ... chr11 chr11_random ... chr21 ... chrM chrX chrY"
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE             File name end with .CGmap or .CGmap.gz. If not specified,
#                       STDIN will be used.
#   -c INT, --chr=INT   The column of chromosome [default: 1]
#   -p INT, --pos=INT   The column of position [default: 2]
#   -o OUTFILE          To standard output if not specified
```

- Example

```
zcat RR*.CGmap.gz WG.CGmap.gz | gawk '$8>=5' | cut -f1,3 | sort -u | cgmaptools sort
-c 1 -p 2 > index
```

3.8 split

- Command

```
cgmaptools split -h
```

```
# Usage: cgmaptools split -i <input> -p <prefix[.chr.]> -s <[.chr.]suffix>
#       (aka CGmapSplitByChr)
# Description: Split the files by each chromosomes.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
#
# Options:
#   -h, --help  show this help message and exit
#   -i FILE     Input file, CGmap or ATCGmap format, use STDIN when not
#               specified.(gzipped if end with 'gz').
#   -p STRING   The prefix for output file
#   -s STRING   The suffix for output file (gzipped if end with 'gz').
```

- Example

```
cgmaptools split -i WG.CGmap.gz -p WG -s CGmap.gz
```

3.9 select

- Command

```
cgmaptools select -h
```

```
# Usage:    cgmaptools select <command> [options]
# Version:  0.1.1
# Commands:
#   region   select or exclude lines by region lists
#   site     select or exclude lines by site list
```

3.9.1 select region

- Command

```
cgmaptools select region -h
```

```
# Usage: cgmaptools select region [-i <CGmap/ATCGmap>] -r <BED> [-R]
#       (aka CGmapSelectByRegion)
# Description: Lines in input CGmap/ATCGmap be selected/excluded by BED file.
#               Strand is NOT considered.
#               Output to STDOUT in same format with input.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2016-12-07
# Options:
#   -i Input, CGmap/ATCGmap file; use STDIN if not specified
#       Please use "gunzip -c <input>.gz " and pipe as input for gzipped file.
#       Ex: chr12 G   19898796   ...
#   -r Input, Region file, BED file to store regions
#       At least 3 columns are required
#       Ex: chr12 19898766 19898966 XX XXX XXX
```

```
# -R [optional] Reverse selection. Sites in region file will be excluded when specified
# -h help
# Tips: program will do binary search for each site in regions
```

- **Example**

```
for CHR in 1 2 3 4 5; do (for P in 1 2 3 4 5; do echo | gawk -vC=$CHR -vP=$P -vOFS="\t"
'{print "chr"C, P*1000, P*1000+200, "+"};}' ; done) ; done > region.bed

zcat WG.CGmap.gz | cgmmaptools select region -r region.bed | head
```

3.9.2 select site

- **Command**

```
cgmmaptools select site -h
```

```
# Usage: cgmmaptools select site -i <index> [-f <CGmap/ATCGmap>] [-r] [-o output]
# (aka CGmapSelectBySite)
# Description: Select lines from input CGmap/ATCGmap in index or reverse.
# Contact: Guo, Weilong; guoweilong@126.com
# Last Update: 2016-12-07
# Index format example:
# chr10 100504
# chr10 103664
#
# Options:
# -h, --help show this help message and exit
# -i FILE Name of Index file required (gzipped if end with '.gz').
# -r reverse selected, remove site in index if specified
# -f STRING Input CGmap/ATCGmap files. Use STDIN if not specified
# -o STRING CGmap, Output file name (gzipped if end with '.gz').
```

- **Example**

```
gawk 'NR%100==50' index > site
cgmmaptools select site -f RR.CGmap.gz -i site -o RR_select.CGmap.gz
```

Chapter 4

SNV calling

Bisulfite sequencing data contains information of both methylation and genome sequences. In addition to DNA methylation analysis, we can also call variants using bisulfite data. Due to bisulfite conversion and PCR amplification during library preparation, the unmethylated cytosines on the DNA fragments would be converted to thymines. Thus, it's difficult to distinguish thymine produced by bisulfite conversion with the real thymine allele.

In recent years, few tools are adapted to bisulfite data for SNP calling. The main idea is removing vague reads that may contain unmethylated cytosines for a given position. Consequently, the rest reads can be regarded as reads generated from a normal genome DNA without bisulfite treatment and can be used to call variants using regular methods without consideration of bisulfite conversion.

However, removing the vague reads leads to information lost in most cases making variant calling less confident, especially when the sequencing depth is low.

To solve this problem, we tried to introduce wild-card in genotype calling. Even for these ambiguous genotypes, we can still learn something.

We proposed two independent methods called BinomWC (based on binomial) and BayesWC (based on bayesian), taking vague reads into consideration.

4.1 BaysWC strategy

BinomWC strategy combines bayesian method and wildcard strategy for predicting the genotype. The likelihood matrix is designed as following.

| ATCGmap table | | | | |
|--|----------|----------|----------|----------|
| Support the nucleotide on reference genome | A | T | C | G |
| Read count on Watson strand | $A_w^\#$ | $T_w^\#$ | $C_w^\#$ | $G_w^\#$ |
| Read count on Crick strand | $A_c^\#$ | $T_c^\#$ | $C_c^\#$ | $G_c^\#$ |

Figure 4.1: ATCGmap table used for SNV calling

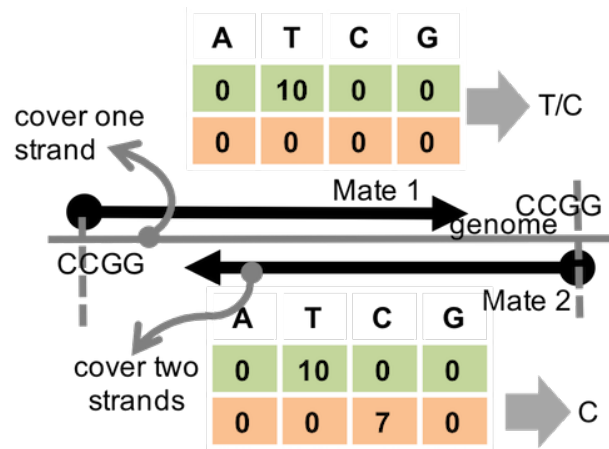


Figure 4.2: Example that alignments refer to vague genotypes

| Ambiguous GN symbol | Possible genotypes | Hete- or Homo- zygous | sure to be SNV if reference is |
|------------------------|--------------------|--------------------------|-----------------------------------|
| Y | TT / TC / CC | not sure | A, G |
| R | AA / AG / GG | not sure | T, C |
| A,Y | AT / AC | heterozygous | A, T, C, G |
| C,Y | CT / CC | not sure | A, T, G |
| G,Y | GT / GC | heterozygous | A, T, C, G |
| T,Y | TT / TC | not sure | A, C, G |
| A,R | AA / AG | not sure | T, C, G |
| C,R | CA / CG | heterozygous | A, T, C, G |
| G,R | GA / GG | not sure | A, T, C |
| T,R | TA / TG | heterozygous | A, T, C, G |

The wildcard characters are defined as: Y=T/C and R=A/G

Figure 4.3: Table for definition of amibiguous genotype

| $\Pr(I^g = 1 g)$ | $g = A$ | $g = T$ | $g = C$ | $g = G$ |
|--------------------|---------|---------|---------|---------|
| $A_w^\# = 1$ | p | e | e | e |
| $T_w^\# = 1$ | $2e$ | $p + e$ | $p + e$ | $2e$ |
| $C_w^\# = 1$ | e | e | p | e |
| $G_w^\# = 1$ | e | e | e | p |
| $A_c^\# = 1$ | $p + e$ | $2e$ | $2e$ | $p + e$ |
| $T_c^\# = 1$ | e | p | e | e |
| $C_c^\# = 1$ | e | e | p | e |
| $G_c^\# = 1$ | e | e | e | p |

Figure 4.4: The likelihood matrix for BayesWC strategy

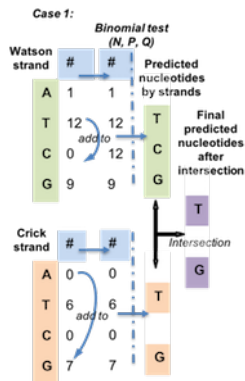


Figure 4.5: Case 1 for BinomWC strategy

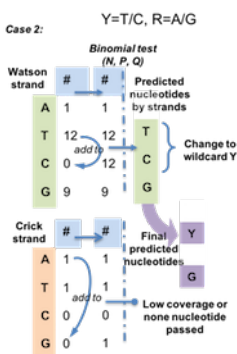


Figure 4.6: Case 2 for BinomWC strategy

4.2 BinomWC strategy

BinomWC (Binomial-WildCard) strategy works as following for 3 cases.

4.3 Performance

Performances on simulation data

Command

```
cgmaptools snv -h
```

```
# Usage: cgmaptools snv [-i <ATCGmap>] [-o <output> -v <VCF>]
# (aka SNVFromATCGmap)
# Description: Predict the SNV from ATCGmap file.
# Contact: Guo, Weilong; guoweilong@126.com
# Last update: 2018-01-02
# Output format example:
# #chr nuc pos ATCG_watson ATCG_crick predicted_nuc p_value
# chr1 G 4752 17,0,0,69 0,0,0,0 A,G 9.3e-07
# chr1 A 4770 40,0,0,29 0,0,0,0 A,G 0.0e+00
# chr1 T 8454 0,39,0,0 0,0,0,0 T/C 1.00e-01
#
#
```

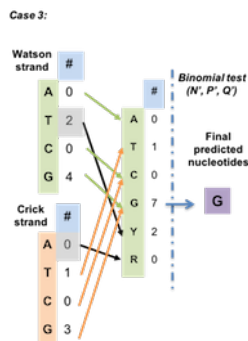


Figure 4.7: Case 3 for BinomWC strategy

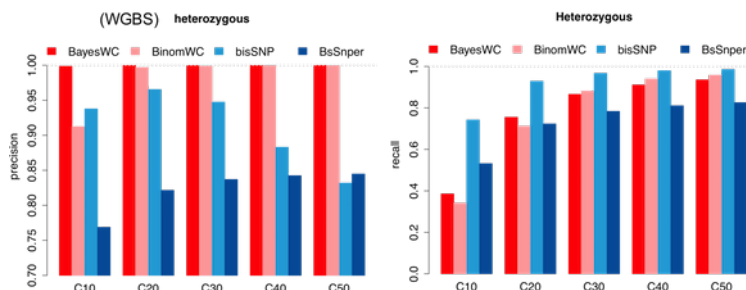


Figure 4.8: Precision-Recall analysis on simulated WGBS data

```
# Options:
# -h, --help          show this help message and exit
# -i FILE             ATCGmap format, STDIN if not specified
# -v FILE, --vcf=FILE VCF format file for output
# -a, --all_nt        Show all sites with an predictable genotype. Only show
#                     SNP sites if not specified.
# -o OUTFILE          STDOUT if not specified
# -m MODE, --mode=MODE Mode for calling SNP [Default: binom]
#                     binom: binomial, separate strands
#                     bayes: bayesian mode
# --bayes-e=BAYES_ER  (BayesWC mode) Error rate for calling a nucleotide
#                     [Default: 0.05]
# --bayes-p=BAYES_PV  (BayesWC mode) P value as cut-off [Default: 0.001]
# --bayes-dynamicP    (BayesWC mode) Use dynamic p-value for different
#                     coverages install of specific p-value. (Recomended)
#                     "--bayes-p" will be ignored if "--bayes-dynamicP" is
#                     specified.
# --binom-e=BINOM_ER  (BinomWC mode) Error rate for calling a nucleotide
#                     [Default: 0.05]
# --binom-p=BINOM_PV  (BinomWC mode) P value as cut-off [Default: 0.01]
# --binom-cov=BINOM_COV
#                     (BinomWC mode) The coverage checkpoint [Default: 10]
```

- Example commands :

```
cgmaptools snv -i WG.ATCGmap.gz -m bayes -v bayes.vcf -o bayes.snv --bayes-dynamicP
cgmaptools snv -i WG.ATCGmap.gz -m binom -o binom.snv
```


| Col | Field | Type | Regexp/Range | Brief description |
|-----|---------|--------|------------------------|---|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | NUC | Char | [ATCGN-] | The nucleotide on reference genome |
| 3 | POS | Int | [0,2 ³² -1] | 1-based leftmost mapping position |
| 4 | W_Count | String | [0-9,]+ | Count of reads support A, T, C, G on Watson strand, seprated by “,” |
| 5 | C_Count | String | [0-9,]+ | Count of reads support A, T, C, G on Crick strand, seprated by “,” |
| 6 | PRDNUC | String | [ATCG./]+ | Predicted genotype (“,” indicate two allele; “/” means “or”) |
| 7 | PVALUE | Float | [0, 1] | P_value for confidence of this prediction |

Figure 4.9: Output format description for cgmtools snv

- **Output format**

- **Example**

| #chr | nuc | pos | ATCG_watson | ATCG_crick | predicted_nuc | p_value |
|------|-----|------|--------------|------------|---------------|----------|
| chr1 | G | 4752 | 17, 0, 0, 69 | 0, 0, 0, 0 | A,G | 9.3e-07 |
| chr1 | A | 4770 | 40, 0, 0, 29 | 0, 0, 0, 0 | A,G | 0.0e+00 |
| chr1 | T | 8454 | 0, 39, 0, 0 | 0, 0, 0, 0 | T/C | 1.00e-01 |

- **Column Description**

Chapter 5

Methylation Analysis

The *CGmapTools* supports both differentially methylated site (**DMS**) analyses and differentially methylated region (**DMR**) analyses are supported.

As the current available DNA methylome are either low coverage (such as WGBS) or fragmented in covered region (such as RRBS). In *CGmapTools*, we proposed a novel method **dynamic fragmentation strategy** for identifying DMRs between a pair of CGmap files.

5.1 dms

Differentially methylated site analysis, supporting *Chi-square* and *Fisher* tests.

- Command

```
cgmaptools dms -h
```

```
# Usage: cgmaptools dms [-i <CGmapInter>] [-m 5 -M 100] [-o output]
# (aka CGmapInterDiffSite)
# Description:
#   Get the differentially methylated sites for two samples.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2017-01-20
# Input Format, same as the output of CGmapIntersect.py:
#   Chr1 C 3541 CG CG 0.8 4 5 0.4 4 10
# Output Format:
#   chr1 C 4654 CG CG 0.92 1.00 8.40e-01
#   chr1 C 4658 CHH CC 0.50 0.00 3.68e-04
#   chr1 G 8376 CG CG 0.62 0.64 9.35e-01
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE             File name for CGmapInter, STDIN if omitted
#   -m INT, --min=INT   min coverage [default : 0]
#   -M INT, --max=INT   max coverage [default : 100]
#   -o OUTFILE          To standard output if omitted. Compressed output if
#                       end with .gz
#   -t STRING, --test-method=STRING
#                       chisq, fisher [default : chisq]
```

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|--------------------------------|------------------------------------|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | NUC | Char | [ATCGN-] | The nucleotide on reference genome |
| 3 | POS | Int | [0,2 ³² -1] | 1-based position |
| 4 | CONT | String | {"--", "CG", "CHG", "CHH"} | context |
| 5 | DINUC | String | {"--", "CA", "CT", "CC", "CG"} | Dinucleotide context |
| 6 | METH_1 | Float | [0,1] or "na" | Methylation level in Sample 1 |
| 9 | METH_2 | Float | [0,1] or "na" | Methylation level in Sample 2 |
| 11 | PVALUE | Float | [0, 1] | P-value |

Figure 5.1: Output format description for cgmaptools dms

- **Example**

```
cgmaptools dms -i intersect_CG.gz -m 4 -M 100 -o DMS.gz -t fisher
```

- **Output format**

- **Example**

```
chr1    C    4654    CG    CG    0.92    1.00    8.40e-01
chr1    C    4658    CHH   CC    0.50    0.00    3.68e-04
chr1    G    8376    CG    CG    0.62    0.64    9.35e-01
```

- **Column Description**

5.2 dmr

Differentially methylated region analysis, using **dynamic fragmentation strategy** .

- **Command**

```
cgmaptools dmr -h
```

```
# Usage: cgmaptools dmr [-i <CGmapInter>] [-m 5 -M 100] [-o output]
# (aka CGmapInterDiffReg)
# Description:
# Get the differentially methylated regions using dynamic fragment strategy.
# Author: Guo, Weilong; guoweilong@126.com;
# Last Updated: 2018-01-02
# Input Format, same as the output of CGmapIntersect.py:
# chr1 C 3541 CG CG 0.8 4 5 0.4 4 10
# Output Format, Ex:
# #chr start end t pv mC_A mC_B N_site
# chr1 1004572 1004574 inf 0.00e+00 0.1100 0.0000 20
# chr1 1009552 1009566 -0.2774 8.08e-01 0.0200 0.0300 15
# chr1 1063405 1063498 0.1435 8.93e-01 0.6333 0.5733 5
#
#
# Options:
# -h, --help show this help message and exit
# -i FILE File name for CGmapInter, STDIN if omitted
# -c INT, --minCov=INT min coverage [default : 4]
```

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|---------------------------------|---------------------------------------|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | POS_L | Int | [0,2 ³² -1] | leftmost position for region |
| 3 | POS_R | Int | [0,2 ³² -1] | rightmost mapping position for region |
| 4 | T | Float | [0,2 ³² -1) or "inf" | Statistics of T test |
| 5 | PV | Float | [0, 1] | P-value of t test |
| 6 | METH_1 | Float | [0,1] or "na" | Methylation level in Sample 1 |
| 7 | METH_2 | Float | [0,1] or "na" | Methylation level in Sample 2 |

Figure 5.2: Output format description for cgmaptools dmr

```
# -C INT, --maxCov=INT max coverage [default : 500]
# -s INT, --minStep=INT
# min step in bp [default : 100]
# -S INT, --maxStep=INT
# max step in bp [default : 1000]
# -n INT, --minNSite=INT
# min N sites [default : 5]
# -o OUTFILE
# To standard output if omitted. Compressed output if
# end with .gz
```

- **Example**

```
cgmaptools dmr -i intersect_CG.gz -o DMR.gz
```

- **Output format**

- **Example**

```
chr1 1004572 1004574 inf 0.00e+00 0.1100 0.0000
chr1 1009552 1009566 -0.2774 8.08e-01 0.0200 0.0300
chr1 1063405 1063498 0.1435 8.93e-01 0.6333 0.5733
chr1 1082130 1082133 -0.0822 9.42e-01 0.5000 0.5550
chr1 1123931 1123933 inf 0.00e+00 0.0600 0.0000
```

- **Column Description**

- **Dynamic Fragment Strategy**

5.3 asm

Feeding with the precisely predicted heterozygous SNVs (by `cgmaptools snv`), *CGmapTools* can identify Allele-Specific Methylated (**ASM**) regions from BAM files.

Following showed an interesting ASM region by analysing a previous cohort (Weilong Guo, et al., Scientific Report, 2016).

- **Command**

```
cgmaptools asm -h
```

```
# DESCRIPTION
# Allele specific methylated region/site calling
# * Fisher exact test for site calling.
```

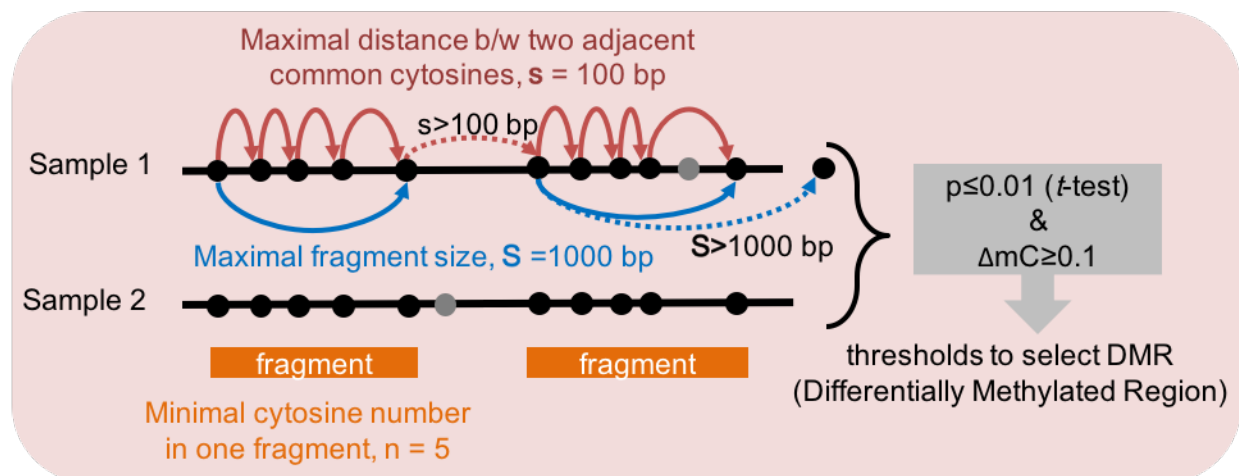


Figure 5.3: Dynamic Fragmentation Strategy

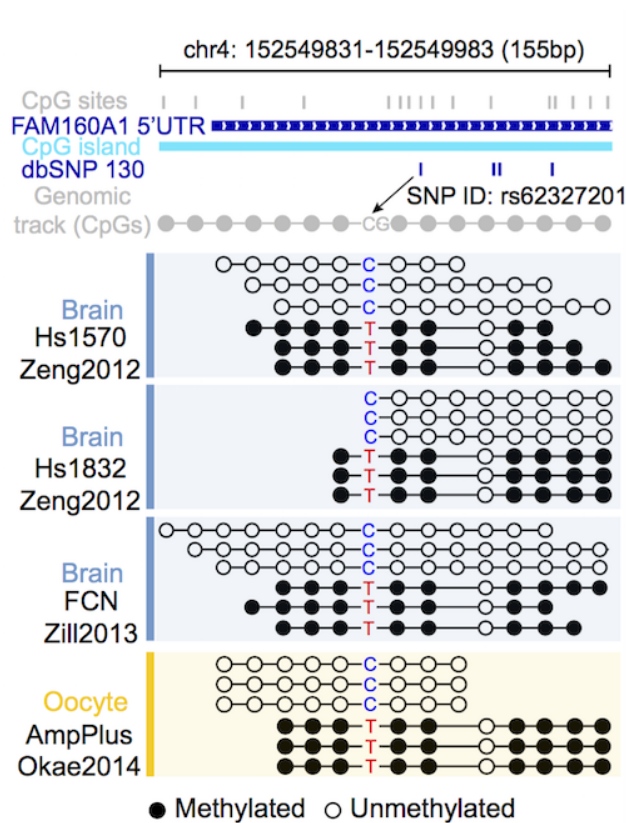


Figure 5.4: Examples showed allele-specific methylated region reported by CGmapTools

```

#           * Students' t-test for region calling.
#
#  USAGE
#
#  cgmaptools asm [options] -r <ref.fa> -b <input.bam> -l <snp.vcf>
#  (aka ASM)
#
#  Options:
#  -r      Samtools indexed reference genome sequeunce, fasta format. eg. hg19.fa
#          - use samtools to index reference first: samtools faidx hg19.fa
#  -b      Samtools indexed Bam format file.
#          - use samtools to index bam file first: samtools index <input.bam>
#  -l      SNPs in vcf file format.
#  -s      Path to samtools eg. /home/user/bin/samtools
#          - by default, we try to search samtools in your system PATH,
#  -o      Output results to file. [default: STDOUT]
#  -t      C context. [default: CG]
#          - available context: C, CG, CH, CW, CC, CA, CT, CHG, CHH
#  -m      Specify calling mode. [default: asr]
#          - alternative: ass
#          - asr: allele specific methylated region
#          - ass: allele specific methylated site
#  -d      Minimum number of read for each allele linked site to call ass. [default: 3]
#          - ass specific.
#  -n      Minimum number of C site each allele linked to call asr. [default: 2]
#          - asr specific.
#  -D      Minimum read depth for C site to call methylation level when calling asr. [default: 1]
#          - asr specific.
#  -L      Low methylation level threshold. [default: 0.2]
#          - allele linked region [or site] with low methylation level should be no greater than
#  -H      High methylation level threshold. [default: 0.8]
#          - allele linked region[or site] with high methylation level should be no less than th
#  -q      Adjusted p value using Benjamini & Hochberg (1995) ("BH" or its alias "fdr"). [default
#  -h      Help message.
#
#  AUTHOR
#
#  Contact:      Zhu, Ping; pingzhu.work@gmail.com
#  Last update: 2016-12-07

```

- **Example**

```

gawk '{if(/^#/){print}else{print "chr"$0;}}' bayes.vcf > bayes2.vcf
cgmaptools asm -r genome.fa -b WG.bam -l bayes2.vcf > WG.asm

```

- **Output format for ASS (Allele-Specific methylated Site)**

- **Example**

| Chr | SNP_Pos | Ref | Allele1 | Allele2 | C_Pos | Allele1_linked_C | Allele2_linked_C | Allele1_linked_C | Allele2_linked_C | Allele1_linked_C | Allele2_linked_C |
|------|---------|-----|---------|---------|---------|------------------|------------------|------------------|------------------|------------------|------------------|
| Chr1 | 8949221 | T | T | A | 8949252 | 30,2 | 6,0 | 0.94 | 1.00 | 1.00 | 1.00 |
| Chr1 | 8965481 | A | A | T | 8965494 | 12,3 | 12,4 | 0.80 | 0.75 | 0.75 | 0.75 |

- **Column Description**

- **Output format for ASR (Allele-Specific methylated Region)**

- **Example**

| Chr | Pos | Ref | Allele1 | Allele2 | Allele1_linked_C | Allele2_linked_C | Allele1_linked_C | Allele2_linked_C | Allele1_linked_C | Allele2_linked_C |
|-----|-----|-----|---------|---------|------------------|------------------|------------------|------------------|------------------|------------------|
|-----|-----|-----|---------|---------|------------------|------------------|------------------|------------------|------------------|------------------|

| Col | Field | Type | Regexp/Range | Brief description |
|-----|----------------------|---------|-------------------------|--|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | SNP_Pos | Int | [0,2 ³² -1] | 1-based leftmost mapping position of SNP site |
| 3 | Ref | Char | [ATCGN-] | The nucleotide on reference genome |
| 4 | Allele1 | Char | [ATCG] | The nucleotide of allele1 |
| 5 | Allele2 | Char | [ATCG] | The nucleotide of allele2 |
| 6 | C_Pos | Int | [0,2 ³² -1] | 1-based leftmost mapping position of C site |
| 7 | Allele1_linked_C | Int | [0, 2 ³² -1] | Comma separated number of reads support methylated and unmethylated C linked by allele1 respectively |
| 8 | Allele2_linked_C | Int | [0, 2 ³² -1] | Comma separated number of reads support methylated and unmethylated C linked by allele2 respectively |
| 9 | Allele1_linked_C_met | Float | [0,1] | Methylation level of allele1 linked C site |
| 10 | Allele2_linked_C_met | Float | [0,1] | Methylation level of allele2 linked C site |
| 11 | pvalue | Float | [0,1] | P value of t test |
| 12 | fdr | Float | [0,1] | Adjusted p value using Benjamini & Hochberg method |
| 13 | ASM | Logical | TRUE/FALSE | TRUE indicates this C site is allele specific methylated. FALSE otherwise. |

Figure 5.5: Output format description for cgmaptools asm -m ass

| | | | | | | | |
|------|---------|---|---|---|----------|----------|-------|
| chr1 | 8943402 | A | A | T | 1-1 | | 0.8-1 |
| chr1 | 8966879 | C | C | G | 0.93-0-0 | 0.81-0-0 | 0.31 |

– Column Description

5.4 mbed

The `cgmaptools mbed` command will calculate **one** DNA methylation level for **all the investigated regions**, which is different from `cgmaptools mtr`.

For example, this function can be applied when calculating the average DNA methylation levels in regions, such as promoter, gene body, specific Transposon Elements (TEs).

- Command

```
cgmaptools mbed -h
```

```
# Usage: cgmaptools mbed [-i <CGmap>] -b <regin.bed> [-c 5 -C 500 -s]
# (aka CGmapMethylInBed)
# Description: Calculated bulk average methylation levels in given regions.
# Contact: Guo, Weilong; guoweilong@126.com
# Last Update: 2017-01-20
# Options:
# -i String, CGmap file; use STDIN if not specified
# Please use "gunzip -c <input>.gz " and pipe as input for gzipped file.
# Ex: chr1 G 3000851 CHH CC 0.1 1 10
# -b String, BED file, should have at least 4 columns
# Ex: chr1 3000000 3005000 -
# -c Int, minimum Coverage [Default: 5]
# -C Int, maximum Coverage [Default: 500]
```


| Col | Field | Type | Regexp/Range | Brief description |
|-----|----------------------|---------|------------------------|--|
| 1 | CHR | String | [!-?A-~]{1,118} | Query template NAME |
| 2 | Pos | Int | [0,2 ³² -1] | 1-based leftmost mapping position of SNP site |
| 3 | Ref | Char | [ATCGN-] | The nucleotide on reference genome |
| 4 | Allele1 | Char | [ATCG] | The nucleotide of allele1 |
| 5 | Allele2 | Char | [ATCG] | The nucleotide of allele2 |
| 6 | Allele1_linked_C | Float | [0,1] | '-' separated methylation level of C sites linked by allele1 |
| 7 | Allele2_linked_C | Float | [0,1] | '-' separated methylation level of C sites linked by allele2 |
| 9 | Allele1_linked_C_met | Float | [0,1] | Average methylation level of allele1 linked C sites |
| 10 | Allele2_linked_C_met | Float | [0,1] | Average methylation level of allele2 linked C sites |
| 11 | pvalue | Float | [0,1] | P value of t test |
| 12 | fdr | Float | [0,1] | Adjusted p value using Benjamini & Hochberg method |
| 13 | ASM | Logical | TRUE/FALSE | TRUE indicates this region is allele specific methylated. FALSE otherwise. |

Figure 5.6: Output format description for cgmaptools asm -m asr

```
# -s Strands would be distinguished when specified
# -h help
#
# Output to STDOUT:
# Title      Count    mean_mC
# sense      34      0.2353
# antisense  54      0.2778
# total      88      0.2614
# Notice:
# The overlapping of regions would not be checked.
# A site might be considered multiple times.
```

- **Example**

```
zcat WG.CGmap.gz | cgmaptools mbed -b region.bed
```

- **Output format**

- **Example**

| chr | sense_Count | sense_mC | anti_Count | anti_mC | all_Count | all_mC |
|------|-------------|----------|------------|---------|-----------|---------|
| chr1 | 203 | 0.08127 | 178 | 0.1148 | 381 | 0.09692 |
| chr2 | 185 | 0.07045 | 257 | 0.05586 | 442 | 0.06197 |
| chr3 | 313 | 0.1042 | 250 | 0.1358 | 563 | 0.1182 |
| chr4 | 300 | 0.1218 | 271 | 0.13 | 571 | 0.1257 |
| chr5 | 282 | 0.1272 | 222 | 0.1589 | 504 | 0.1412 |

5.5 mbin

This function will calculate the average methylation levels in equal-length bins, across genome, generating both summary table and distribution graph.

- **Command**

```
cgmaptools mbin -h
```

```
# Usage: cgmaptools mbin [-i <CGmap>] [-c 10 --CXY 5 -B 5000000]
# (aka CGmapMethInBins)
```

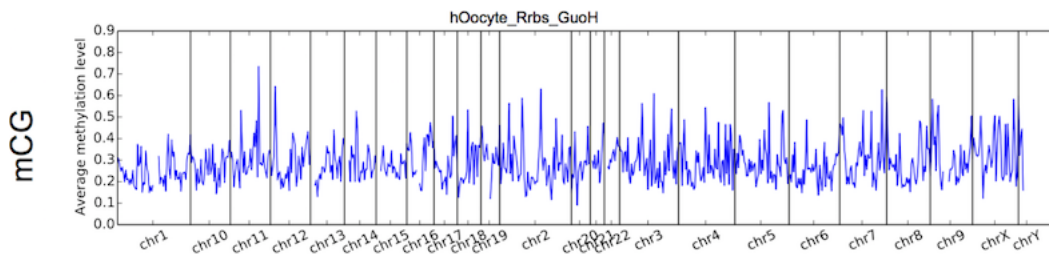


Figure 5.7: Output figure example for cgmaptools mbin

```
# Description: Generate the methylation in Bins.
# Contact:      Guo, Weilong; guoweilong@126.com
# Last Update: 2016-10-26
# Output Ex:
#   chr1      1      5000      0.0000
#   chr1     5001     10000     0.0396
#   chr2       1      5000     0.0755
#   chr2     5001     10000     0.0027
#   chr3       1      5000      na
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE              File name end with .CGmap or .CGmap.gz. If not
#                       specified, STDIN will be used.
#   -B BIN_SIZE          Define the size of bins [Default: 5000000]
#   -c COVERAGE          The minimum coverage for site selection [Default: 10]
#   -C CONTEXT, --context=CONTEXT
#                       specific context: CG, CH, CHG, CHH, CA, CC, CT, CW
#                       use all sites if not specified
#   --cXY=COVERAGEXY    Coverage for chrX/Y should be half that of autosome
#                       for male [Default: same with -c]
#   -f FIGTYPE, --figure-type=FIGTYPE
#                       png, pdf, eps. Will not generate figure if not
#                       specified
#   -H FLOAT             Height of figure in inch [Default: 4]
#   -W FLOAT             Width of figure in inch [Default: 8]
#   -p STRING            Prefix for output figures
#   -t STRING, --title=STRING
#                       title in the output figures
```

- **Example**

```
cgmaptools mbin -i WG.CGmap.gz -B 500 -c 4 -f png -t WG -p WG > mbin.WG.data
```

- **File format**

The output format:

```
chr1      1      5000      0.0000
chr1     5001     10000     0.0396
chr2       1      5000     0.0755
chr2     5001     10000     0.0027
chr3       1      5000      na
```

5.6 mmbin

This function will calculate the average methylation levels in equal-length bins for **multiple** samples, generating a summary table.

- **Command**

```
cgmaptools mmbin -h
```

```
# Usage: cgmaptools mmbin [-l <1.CGmap[,2.CGmap,...]>] [-c 10 --CXY 5 -B 5000000]
#      (aka CGmapsMethInBins)
# Description: Generate the methylation in Bins.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Output Ex:
#   chr1    1      5000    0.0000
#   chr1   5001    10000    0.0396
#   chr2     1      5000    0.0755
#   chr2   5001    10000    0.0027
#   chr3     1      5000     na
#
# Options:
#   -h, --help          show this help message and exit
#   -l FILE             File name list, end with .CGmap or .CGmap.gz. If not
#                       specified, STDIN will be used.
#   -t FILE             List of samples
#   -B BIN_SIZE         Define the size of bins [Default: 5000000]
#   -C CONTEXT, --context=CONTEXT
#                       specific context: CG, CH, CHG, CHH, CA, CC, CT, CW
#                       use all sites if not specified
#   -c COVERAGE         The minimum coverage for site selection [Default: 10]
#   --cXY=COVERAGEXY   Coverage for chrX/Y should be half that of autosome
#                       for male [Default: same with -c]
```

- **Example**

```
cgmaptools mmbin -l WG.CGmap.gz,RR.CGmap.gz,RR2.CGmap.gz,merge.CGmap.gz -c 4 -B 2000
| gawk '{printf("%s:%s-%s", $1, $2, $3); for(i=4;i<=NF;i++){printf("\t%s", $i);}
printf("\n");}' > mmbin
""
```

- **Output format**

- **Example**

| chr | pos1 | pos2 | tag1 | tag2 | tag3 |
|------|--------|--------|------|------|------|
| Chr1 | 111403 | 113403 | 0.05 | nan | 0.02 |
| Chr1 | 111500 | 112500 | 1.00 | 0.80 | 0.60 |
| Chr2 | 20000 | 20500 | 0.96 | 0.33 | 0.66 |

- **Column Description**

5.7 mfg

The `cgmaptools mfg` supports studys of Methylation in FraGmented regions. The function can be applied to draw mC distribution across gene body, transposon elements, and other user-provided regions. The

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|------------------------|-------------------------------|
| 1 | CHR | String | [!-?A-~]* | Query template NAME |
| 2 | POS1 | Int | [0,2 ³² -1] | Leftmost Position |
| 3 | POS2 | Int | [0,2 ³² -1] | Rightmost Position |
| 4 | METH_1 | Float | [0.00, 1.00] | Methylation level in sample 1 |
| ... | ... | ... | ... | ... |
| n | METH_n | Float | [0.00, 1.00] | Methylation level in sample n |

Figure 5.8: Output format description for cgmaptools mmbin

fragmented regions can be generated using `cgmaptools bed2fragreg`.

- **Command**

```
cgmaptools mfg -h
```

```
# Usage: cgmaptools mfg [-i <CGmap>] -r <region> [-c 5 -C 500]
# Description: Calculated methylation profile across fragmented regions.
# Contact: Guo, Weilong; guoweilong@126.com
# Last Update: 2017-01-20
# Options:
#   -i String, CGmap file; use STDIN if not specified
#       Please use "gunzip -c <input>.gz " and pipe as input for gzipped file.
#       chr1 G 851 CHH CC 0.1 1 10
#   -r String, Region file, at least 4 columns
#       Format: chr strand pos_1 pos_2 pos_3 ...
#       Regions would be considered as [pos_1, pos_2), [pos_2, pos_3)
#       Strand information will be used for distinguish sense/antisense strand
#       Ex:
#       #chr strand U1 R1 R2 D1 End
#       chr1 + 600 700 800 900 950
#       chr1 - 1600 1500 1400 1300 1250
#   -c Int, minimum Coverage [Default: 5]
#   -C Int, maximum Coverage [Default: 500]
#       Sites exceed the coverage range will be discarded
#   -x String, context [use all sites by default]
#       string can be CG, CH, CHG, CHH, CA, CC, CT, CW
#   -h help
# Output to STDOUT:
#   Region_ID      U1      R1      R2      D1
#   sense_ave_mC   0.50    0.40    0.30    0.20
#   sense_sum_mC   5.0     4.0     3.0     2.0
#   sense_sum_NO   10      10      10      10
#   anti_ave_mC    0.40    0.20    0.10    NaN
#   anti_sum_mC    8.0     4.0     2.0     0.0
#   anti_sum_NO    20      20      20      0
#   total_ave_mC   0.43    0.27    0.17    0.2
#   total_sum_mC   13.0    8.0     5.0     2.0
#   total_sum_NO   30      30      30      10
```

- **Example**

```
for CHR in 1 2 3 4 5; do    for P in 1 2 3 4 5; do    echo | gawk -vC=$CHR -vP=$P
```

```
-vOFS="\t" '{print "chr"C, P*1000, P*1000+1000, "+";}' done done | cgmmaptools
bed2fragreg \ -n 30 -F 50,50,50,50,50,50,50,50,50,50,50 -T 50,50,50,50,50,50,50,50,50,50
> fragreg.bed

gunzip -c WG.CGmap.gz | cgmmaptools mfg -r fragreg.bed -c 2 -x CG > WG.mfg
```

- **Output format**

- **Example**

| Region_ID | R_1 | R_2 | R_3 | R_4 |
|--------------|------|------|------|------|
| sense_ave_mC | 0.50 | 0.40 | 0.30 | 0.20 |
| sense_sum_mC | 5.0 | 4.0 | 3.0 | 2.0 |
| sense_sum_NO | 10 | 10 | 10 | 10 |
| anti_ave_mC | 0.40 | 0.20 | 0.10 | NaN |
| anti_sum_mC | 8.0 | 4.0 | 2.0 | 0.0 |
| anti_sum_NO | 20 | 20 | 20 | 0 |
| total_ave_mC | 0.43 | 0.27 | 0.17 | 0.2 |
| total_sum_mC | 13.0 | 8.0 | 5.0 | 2.0 |
| total_sum_NO | 30 | 30 | 30 | 10 |

- **Column Description**

- * Tab-delimited file with header line
- * Content in 1st col is fixed
- * Column number is dynamic
- * Invalid number is annotated as “NaN”

5.8 mstat

CGmapTools provide resourceful statistic analysis on DNA methylation globally. The *cgmmaptools mstat* command will generated a table summary, together with several graphs:

- mC contributions of different contexts in Pie chart
- Bulk mC levels of different contexts
- Fragmented distribution of mC in different contexts

The methylation contexts are different for plants and animals. - For plants, the contexts for DNA methylations are known as **CG**, **CHG** and **CHH**, where $H = \{A, C, T\}$. - For animals, the situation is different. In 2014, Weilong Guo, et al. showed that it is unnecessary to separate CHG methylations and CHH methylations in human. In 2016, Weilong Guo, et al. designed *MiDD* method and de novo predicted the main separated contexts for non-CG (CH) methylation should be CW ($W = \{A, T\}$) and CC, and mCW is cell-type specific and conserved between human and mice. In *CGmapTools*, we support both human view (**CG**, **CW** and **CC**) and plant view (**CG**, **CHG**, and **CHH**) for DNA methylation contexts.

- **Command**

```
cgmmaptools mstat -h
```

```
# Usage: cgmmaptools mstat [-i <CGmap>]
# (aka CGmapStatMeth)
# Description: Generate the bulk methylation.
# Contact: Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Output Ex:
# MethStat context C CG CHG CHH CA CC CT CH CW
# mean_mC global 0.0798 0.3719 0.0465 0.0403 0.0891 0.0071 0.0241 0.0419 0.0559
# sd_mCbyChr global 0.0078 0.0341 0.0163 0.0110 0.0252 0.0049 0.0076 0.0096 0.0148
```

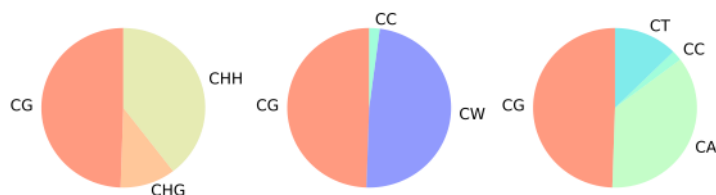


Figure 5.9: mC contribution example

```
# count_C      global 10000 1147 2332 6521 3090 2539 3224 8853 6314
# contrib_mC   global 1.0000 0.5348 0.1360 0.3292 0.3452 0.0228 0.0973 0.4652 0.4424
# quant_mC     [0]    8266 471 2012 5783 2422 2421 2952 7795 5374
# quant_mC     (0.00 ,0.20] 705 182 155 368 272 97 154 523 426
# mean_mC_byChr chr1   0.0840 0.4181 0.0340 0.0412 0.0794 0.0065 0.0251 0.0393 0.0513
# mean_mC_byChr chr10 0.0917 0.4106 0.0758 0.0421 0.0968 0.0097 0.0349 0.0502 0.0655
```

```
# Options:
# -h, --help          show this help message and exit
# -i FILE             File name end with .CGmap or .CGmap.gz. If not
#                   specified, STDIN will be used.
# -c COVERAGE         The minimum coverage for site selection [Default: 10]
# -f FILE, --figure-type=FILE
#                   png, pdf, eps. Will not generate figure if not
#                   specified
# -H FLOAT            Height of figure in inch [Default: 3]
# -W FLOAT            Width of figure in inch [Default: 8]
# -p STRING           Prefix for output figures
# -t STRING, --title=STRING
#                   title in the output figures
```

- **Example**

```
cgmaptools mstat -i WG.CGmap.gz -c 4 -f png -p WG -t WG > WG.mstat.data
```

- **File format**

The output format:

| MethStat | context | C | CG | CHG | CHH | CA | CC | CT | CH | CW |
|---------------|--------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| mean_mC | global | 0.0798 | 0.3719 | 0.0465 | 0.0403 | 0.0891 | 0.0071 | 0.0241 | 0.0419 | 0.0559 |
| sd_mCbyChr | global | 0.0078 | 0.0341 | 0.0163 | 0.0110 | 0.0252 | 0.0049 | 0.0076 | 0.0096 | 0.0148 |
| count_C | global | 10000 | 1147 | 2332 | 6521 | 3090 | 2539 | 3224 | 8853 | 6314 |
| contrib_mC | global | 1.0000 | 0.5348 | 0.1360 | 0.3292 | 0.3452 | 0.0228 | 0.0973 | 0.4652 | 0.4424 |
| quant_mC | [0] | 8266 | 471 | 2012 | 5783 | 2422 | 2421 | 2952 | 7795 | 5374 |
| quant_mC | (0.00 ,0.20] | 705 | 182 | 155 | 368 | 272 | 97 | 154 | 523 | 426 |
| mean_mC_byChr | chr1 | 0.0840 | 0.4181 | 0.0340 | 0.0412 | 0.0794 | 0.0065 | 0.0251 | 0.0393 | 0.0513 |
| mean_mC_byChr | chr10 | 0.0917 | 0.4106 | 0.0758 | 0.0421 | 0.0968 | 0.0097 | 0.0349 | 0.0502 | 0.0655 |

- **Output figures**

5.9 mtr

The `cgmaptools mtr` command will calculate the DNA methylation levels for each investigated region.

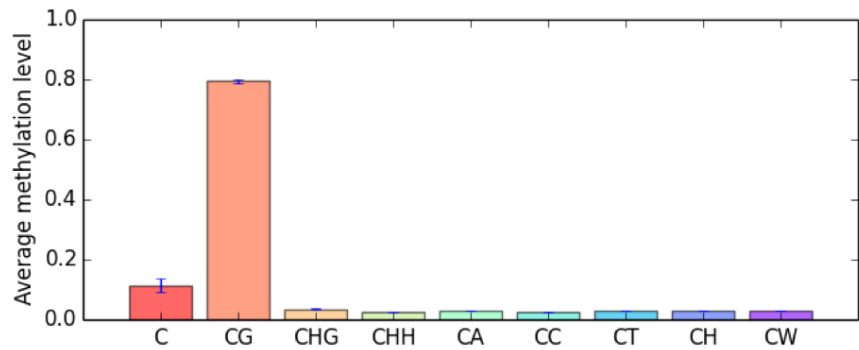


Figure 5.10: Bulk mC example

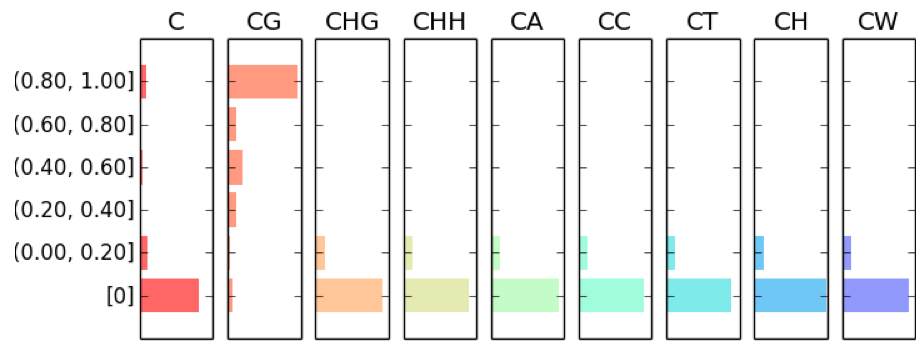


Figure 5.11: mC fragmented distribution example

| Col | Field | Type | Regexp/Range | Brief description |
|-----|-------|--------|-------------------------|--|
| 1 | CHR | String | [!-?A-~]* | Query template NAME |
| 2 | POS_L | Int | [0,2 ³² -1] | Leftmost Position |
| 3 | POS_R | Int | [0,2 ³² -1] | Rightmost Position |
| 4 | MC_s | Float | [0, 2 ³² -1) | Average methylation levels by each site |
| 5 | NC_s | Int | [0,2 ³² -1] | Count of Cytosines in this region |
| 6 | MC_r | Float | [0, 2 ³² -1) | Average methylation levels recalculated by region |
| 7 | NC_r | Int | [0,2 ³² -1] | Sum of effective coverage for all cytosines in this region |

Figure 5.12: Output format description for cgmaptools mtr

- **Command**

```
cgmaptools mtr -h
```

```
# Usage: cgmaptools mtr [-i <CGmap>] -r <region> [-o <output>]
#       (aka CGmapToRegion)
# Description: Calculated the methylation levels in regions in two ways.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Format of Region file:
#   #chr    start_pos  end_pos
#   chr1    8275       8429
# Output file format:
#   #chr    start_pos  end_pos  mean(mC)  #_C  #read(C)/#read(T+C)  #read(T+C)
#   chr1    8275       8429      0.34     72    0.40                 164
# Note: The two input CGmap files should be sorted by Sort_chr_pos.py first.
#       This script would not distinguish CG/CHG/CHH contexts.
#
# Options:
#   -h, --help  show this help message and exit
#   -i FILE     File name end with .CGmap or .CGmap.gz. If not specified, STDIN
#               will be used.
#   -r FILE     Filename for region file, support *.gz
#   -o OUTFILE  To standard output if not specified.
```

- **Example**

```
cgmaptools mtr -i WG.CGmap.gz -r region.bed -o WG.mtr.gz
```

- **Input region format**

```
#chr    start_pos  end_pos
chr1    8275       8429
```

- **Output format**

- **Example**

```
chr1    8275    8429    0.34    72    0.40    164
chr1    8899    8999    0.20    40    0.33    198
chr2    8275    8429    0.50    12    0.45    40
```

- **Column Description**

Chapter 6

Coverage Analysis

Read coverage is an important factor for interpreting DNA methylomes.

It requires different coverage levels for different purpose. For example, SNV calling requires higher coverage than it is required for DMR study. The SNV calling process is depending on all nucleotides (A, T, C and G), whereas DNA methylation levels only depend on T and C read counts aligned to cytosines.

In *CGmapTools*, we proposed two ways for evaluating the coverages of DNA methylations: **OverAll Coverage (OAC)** and **Methylation-Effective Coverage (MEC)**.

- OAC is calculated as the average read coverage on all nucleotides on both strands, which are calculated from the ATCGmap file.
- MEC is calculated as the average read coverage only for cytosines, which is calculated from the CGmap file. Generally, the MEC is slightly higher than half of the OAC.

In *CGmapTools*, we provides function for basic statistics of coverages (`cgmaptools oac stat` and `cgmaptools mec stat`) and visualization of coverages in bins across genome (`cgmaptools oac bin` and `cgmaptools mec stat`).

6.1 oac

- Command

```
cgmaptools oac -h
```

```
# Usage:    cgmaptools oac <command> [options]
# Version:  0.1.1
# Commands:
#      bin      * overall coverage in bins
#      stat     * overall coverage statistics globally
```

6.1.1 oac bin

- Command

```
cgmaptools oac bin -h
```

```
# Usage: cgmaptools oac bin [-i <ATCGmap>] [-B 5000000]
#      (aka ATCGmapCovInBins)
# Description: Generate the overall coverage in Bins.
```

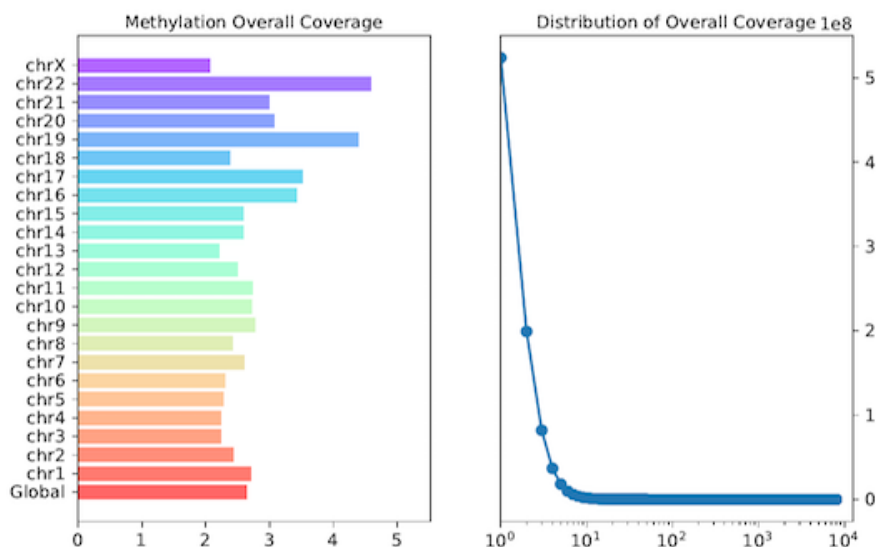


Figure 6.1: MEC example

```
# Contact:      Guo, Weilong; guoweilong@126.com;
# Last Update: 2016-12-07
# Output Ex:
#   chr1      1      5000      29.0000
#   chr1     5001     10000     30.0396
#   chr2      1      5000     35.0755
#   chr2     5001     10000     40.0027
#   chr3      1      5000      na
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE              File name end with .ATCGmap or .ATCGmap.gz. If not
#                       specified, STDIN will be used.
#   -B BIN_SIZE          Define the size of bins [Default: 5000000]
#   -f FILE, --figure-type=FILE
#                       png, pdf, eps. Will not generate figure if not
#                       specified
#   -H FLOAT             Height of figure in inch [Default: 4]
#   -W FLOAT             Width of figure in inch [Default: 8]
#   -p STRING            Prefix for output figures
#   -t STRING, --title=STRING
#                       title in the output figures
```

- Example

```
cgmaptools oac bin -i WG.ATCGmap.gz -B 1000 -f png -p WG -t WG > WG.oac_bin.data
```

- Output figure

6.1.2 oac stat

- Command

```
cgmaptools oac stat -h
```

```
# Usage: cgmaptools oac stat [-i <ATCGmap>]
#       (aka ATCGmapStatCov)
# Description: Get the distribution of overall coverages.
# Contact:    Guo, Weilong; guoweilong@126.com;
# Last Update: 2018-01-02
# Output Ex:
#   OverAllCov      global  47.0395
#   OverAllCov      chr1    45.3157
#   OverAllCov      chr10   47.7380
#   CovAndCount     1       1567
#   CovAndCount     2       655
#   CovAndCount     3       380
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE              File name end with .ATCGmap or .ATCGmap.gz. If not
#                       specified, STDIN will be used.
#   -f FILE, --figure-type=FILE
#                       png, pdf, eps. Will not generate figure if not
#                       specified
#   -H FLOAT             Scale ratio for the Height of figure [Default: 4]
#   -W FLOAT             Width of figure in inch [Default: 8]
#   -p STRING            Prefix for output figures
```

- Example

```
cgmaptools oac stat -i WG.ATCGmap.gz -p WG -f png > WG.oac_stat.data
```

- output format:

The output format of bin:

```
chr1    1      5000    29.0000
chr1    5001   10000    30.0396
chr2    1      5000    35.0755
chr2    5001   10000    40.0027
chr3    1      5000    na
```

The output format of stat:

```
OverAllCov      global  47.0395
OverAllCov      chr1    45.3157
OverAllCov      chr10   47.7380
CovAndCount     1       1567
CovAndCount     2       655
CovAndCount     3       380
```

6.2 mec

- Command

```
cgmaptools mec -h
```

```
# Usage:    cgmaptools mec <command> [options]
# Version:  0.1.1
# Commands:
#     bin      * methylation effective coverage in bins
#     stat     * methylation effective coverage statistics globally
```

6.2.1 mec bin

- Command

```
cgmaptools mec bin -h
```

```
# Usage: cgmaptools mec bin [-i <CGmap>] [-B 5000000]
#       (aka CGmapCovInBins)
# Description: Generate the methylation-effective coverage in Bins.
# Contact:    Guo, Weilong; guoweilong@126.com;
# Last Update: 2018-01-02
# Output Ex:
#   chr1    1      5000    29.0000
#   chr1    5001   10000   30.0396
#   chr2    1      5000   35.0755
#   chr2    5001   10000   40.0027
#   chr3    1      5000    na
#
# Options:
#   -h, --help                show this help message and exit
#   -i FILE                   File name end with .CGmap or .CGmap.gz. If not
#                             specified, STDIN will be used.
#   -B BIN_SIZE               Define the size of bins [Default: 5000000]
#   -f FILE, --figure-type=FILE
#                             png, pdf, eps. Will not generate figure if not
#                             specified
#   -H FLOAT                  Height of figure in inch [Default: 4]
#   -W FLOAT                  Width of figure in inch [Default: 8]
#   -p STRING                  Prefix for output figures
#   -t STRING, --title=STRING
#                             title in the output figures
#   -C CONTEXT, --context=CONTEXT
#                             specific context: CG, CH, CHG, CHH, CA, CC, CT, CW
#                             use all sites if not specified
```

- Example

```
cgmaptools mec bin -i WG.CGmap.gz -B 1000 -f png -p WG -t WG > WG.mec_bin.data
```

6.2.2 mec stat

- Command

```
cgmaptools mec stat -h
```

```
# Usage: cgmaptools mec stat [-i <CGmap>]
#       (aka CGmapStatCov)
```

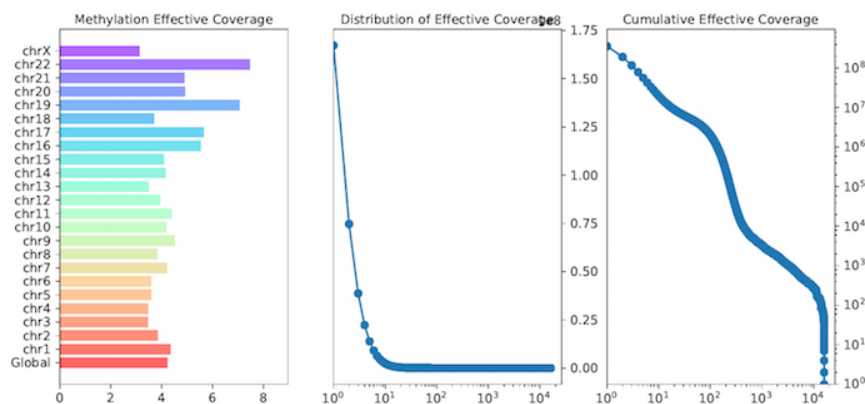


Figure 6.2: MEC example

```
# Description: Get the distribution of methylation-effective coverages.
# Contact:      Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Output Ex:
#   MethEffectCove  global  47.0395
#   MethEffectCove  chr1    45.3157
#   MethEffectCove  chr10   47.7380
#   CovAndCount     1       1567
#   CovAndCount     2       655
#   CovAndCount     3       380
#
# Options:
#   -h, --help          show this help message and exit
#   -i FILE              File name end with .CGmap or .CGmap.gz. If not
#                       specified, STDIN will be used.
#   -f FILE, --figure-type=FILE
#                       png, pdf, eps. Will not generate figure if not
#                       specified
#   -H FLOAT            Scale factor for the Height of figure [Default: 4]
#   -W FLOAT            Width of figure in inch [Default: 11]
#   -p STRING           Prefix for output figures
#   -C CONTEXT, --context=CONTEXT
#                       specific context: CG, CH, CHG, CHH, CA, CC, CT, CW
#                       use all sites if not specified
```

- Example

```
cgmmaptools mec stat -i WG.CGmap.gz -p WG -f png > WG.mec_stat.data
```

- Output figure

Chapter 7

Graphics

7.1 lollipop

The consideration that we design novel **Lollipop** plot, is to be able to distinguish **un-methylated sites** and **un-detected sites**. Each covered cytosine would have a large round head; color and height of the bar represent DNA methylation level.

- Command

```
cgmaptools lollipop -h
```

```
# Usage: cgmaptools lollipop [options] file
#       (aka mCLollipop)
# Description: Plot local mC level for multiple samples
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-04-10
# Example:
#       mCLollipop [-i input] -o gene.png
# -Input Format (-i)
#       Can be output by "cgmaptools mergelist tomatrix". Use STDIN if omitted.
#       The 1st line (header line) is required.
#       Example:
#           chr    pos    tag1    tag2    tag3
#           Chr1   111403  0.30    nan     0.80
#           Chr1   111406  0.66    0.40    0.60
# -Site File (-s)
#       >= 3 columns, the 1st line (header line) is required, using R color name or "NaN".
#       To show specific sites (such as DMS, SNV) at the bottom as triangles.
#       Example:
#           chr    pos    A_vs_B  B_vs_C  A_vs_C
#           chr1   13116801  NaN     NaN     darkgreen
#           chr1   13116899  NaN     red     NaN
# -Region File (-b)
#       the first 4 columns are required.
#       To show specific region (such as DMR, Repeats) at the bottom as blocks.
#       Example:
#           chr1   213941196  213942363  hyper-DMR
#           chr1   213942363  213943530  hypo-DMR
#           # chr    left    right    region-description
```

```

# -annotation file (-a), refFlat Format:
#   To show the structure of genes/transcripts. One-line in annotation, one-track in figure.
#   Example:
#       GeneA  TransA  chr2  +      1000      2000      1100      1950      3      1100,1500,1700, 1
#       #  GeneID  TrandID ChrID Strand TransLeft TransRight CDSLeft CDSRight nExon ExonLefts      E
#
# Options:
#   -i INFILE, --infile=INFILE
#       input file, use STDIN if omitted, multiple-chr is not suggested
#
#   -a ANNOTATION, --annotation=ANNOTATION
#       [opt] annotation file name, refFlat format
#
#   -o OUTFILE, --outfile=OUTFILE
#       [opt] output file
#
#   -f FORMAT, --format=FORMAT
#       [opt] the format for output figure: pdf (default), png, eps
#
#   -l LEFT, --left=LEFT
#       [opt] Left-most position, use the 1st position if omitted
#
#   -r RIGHT, --right=RIGHT
#       [opt] Right-most position, use the last position of input if omitted
#
#   -c CHR, --chr=CHR
#       [opt] chromosome name, use the chr in 1st line of input file if omitted
#
#   -s SITE, --site=SITE
#       [opt] file of site to be marked
#
#   -b BED, --bed=BED
#       [opt] BED file for region to be marked
#
#   -t TITLE, --title=TITLE
#       [opt] text shown on title
#
#   -w WIDTH, --width=WIDTH
#       [opt] width (in inch). Default: 8.
#
#   --height=HEIGHT
#       [opt] height (in inch). Default: 8.
#
#   -h, --help
#       Show this help message and exit

```

- **Example**

```
cgmmaptools lollipop -i matrix.CG.gz -a anno.refFlat -f pdf
```

- **Figure examples**

- **refFlat** format

- **Example**

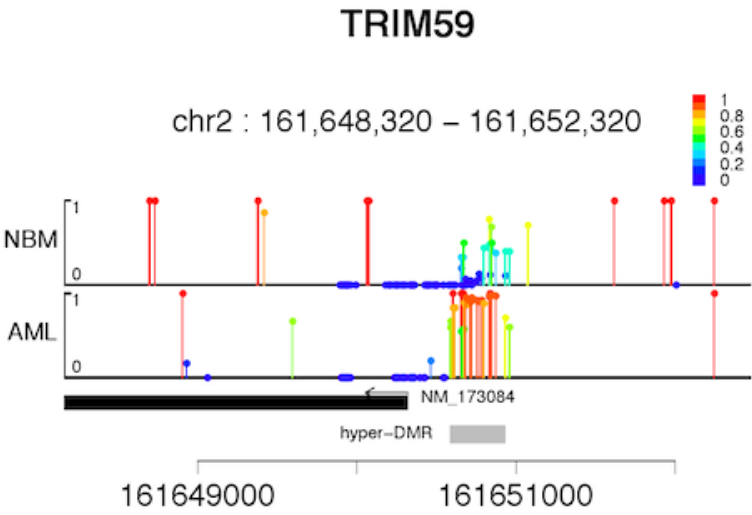


Figure 7.1: Lollipop example-1

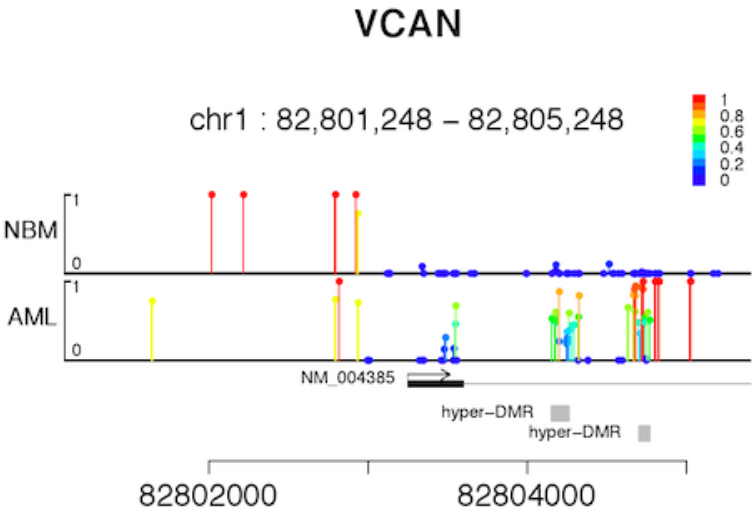


Figure 7.2: Lollipop example-2

```
GeneA TransA chr2 + 1000 2000 1100 1950 3 1100,1500,1700, 1200,1580,1950,
```

- Description
 - Col 1: Gene ID
 - Col 2: Transcript ID
 - Col 3: chromatine ID
 - Col 4: strand, “+” or “-”
 - Col 5: The left-most position of transcript
 - Col 6: The right-most position of transcript
 - Col 7: The left-most position of CDS
 - Col 8: The right-most position of CDS
 - Col 9: Number of exons
 - Col 10: List of left-most position of exons, seperated by “,”
 - Col 11: List of right-most position of exons, seperated by “,”
- Convert GTF format to refFlat format

The following is an example for *Z. mays*.

“gtfToGenePred” is a command tool downloaded from UCSC utility.

```
...
```

```
gtfToGenePred -genePredExt -geneNameAsName2 -allErrors AGPv4.gtf AGPv4.GenePred
```

```
paste <(cut -f13 AGPv4.GenePred) <(cut -f1-10 AGPv4.GenePred) > AGPv4.refFlat
```

```
paste <(cut -f13 AGPv4.GenePred) <(cut -f1-10 AGPv4.GenePred) | sed -i s/transcript://g | cut -f9 | gawk
```

```
cut -f1-10 AGPv4.GenePred > AGPv4.refFlat.tmp
```

```
gawk -F"\t" -vOFS="\t" 'ARGIND==1{GeneID[$1]=$2;} ARGIND==2{printf GeneID[$1]"\\t"$0}' trans_gene_ID AGPv4.refFlat.tmp
```

```
rm ${GN}.refFlat.txt AGPv4.GenePred
```

```
...
```

7.2 heatmap

- Command

```
cgmmaptools heatmap -h
```

```
# Usage: cgmmaptools heatmap [options]
#       (aka mCBinHeatmap)
# Description: Plot methylation dynamics of target region for multiple samples [heatmap]
# Contact:    Zhu, Ping; pingzhu.work@gmail.com
# Last update: 2017-09-16
# Example:
#   mCBinHeatmap.R -i input -m white -o chr1.xxx-xxx.pdf
#   -Input File Format:
#   1st line is the header.
#   Each column contains methylation measurements of a sample.
#   Example:
#   Region Sample1 Sample2 ...
#   Region1 0.1      0.1      ...
#   Region2 0.1      0.1      ...
```

```

#
#
# Options:
#   -i INFILE, --infile=INFILE
#       input file
#
#   -o OUTFILE, --outfile=OUTFILE
#       [opt] output file name. [default: mCBinHeatmap.SysDate.pdf]
#
#   -c, --cluster
#       [opt] cluster samples by methylation in regions. [default: FALSE]
#
#   -l COLORLOW, --colorLow=COLORLOW
#       [opt] color used for the lowest methylation value. [default: cyan3]
#
#   -m COLORMID, --colorMid=COLORMID
#       [opt] color used for the middle methylation value. [default: null]
#
#   -b COLORHIGH, --colorHigh=COLORHIGH
#       [opt] color used for the highest methylation value. [default: coral2]
#
#   -n COLORNUMBER, --colorNumber=COLORNUMBER
#       [opt] desired number of color elements in the panel. [default: 10]
#
#   -W WIDTH, --width=WIDTH
#       [opt] width of figure (inch). [default: 7]
#
#   -H HEIGHT, --height=HEIGHT
#       [opt] height of figure (inch). [default: 7]
#
#   -f FORMAT, --format=FORMAT
#       [opt] format of output figure. Alternative: png. [default: pdf]
#
#   -R RESOLUTION, --resolution=RESOLUTION
#       [opt] Resolution in ppi. Only available for png format. [default: 300]
#
#   -h, --help
#       Show this help message and exit

```

- Example:

```
cgmaptools mmbin -l 1.CGmap,2.CGmap,3.CGmap > mmbin.tab
cgmaptools heatmap -i mmbin.tab -c -o cluster.pdf -f pdf
```

- Figure examples

7.3 fragreg

- Command

```
cgmaptools fragreg -h
```

```

# Usage: cgmaptools fragreg [options]
#       (aka mCFragRegView)
# Description: Plot methylation dynamics of target and flanking region for multiple samples

```

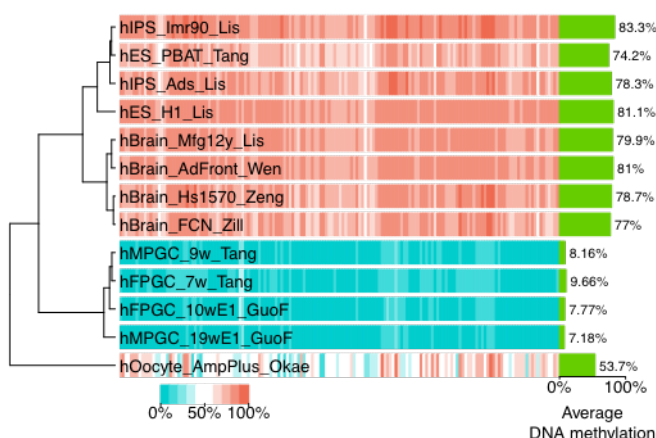


Figure 7.3: heatmap example-1

```
# Contact:      Zhu, Ping; pingzhu.work@gmail.com
# Last update: 2018-02-12
# Example:
#   FragRegView.R -i input -r 5 -o genebody.pdf
# -Input File Format:
#   1st line is the header.
#   Each row contains methylation measurements of a sample.
#   The user may need to use shell script to generate following format
#   based on the results of "cgmaptools mfg".
# Example:
# Sample Up1 Up2 ... Region1 Region2 ... Down1 Down2 ...
# Sample1 0.1 0.1 ... 0.2 0.2 ... 0.3 0.3 ...
# Sample2 0.1 0.1 ... 0.2 0.2 ... 0.3 0.3 ...
#
# Options:
#   -i INFILE, --infile=INFILE
#       input file
#
#   -r RATIO, --ratio=RATIO
#       [opt] range ratio between target region and flanking region in plot. [default: 5]
#
#   -o OUTFILE, --outfile=OUTFILE
#       [opt] output file name. [default: FragRegView.SysDate.pdf]
#
#   -W WIDTH, --width=WIDTH
#       [opt] width of figure (inch). [default: 7]
#
#   -H HEIGHT, --height=HEIGHT
#       [opt] height of figure (inch). [default: 7]
#
```

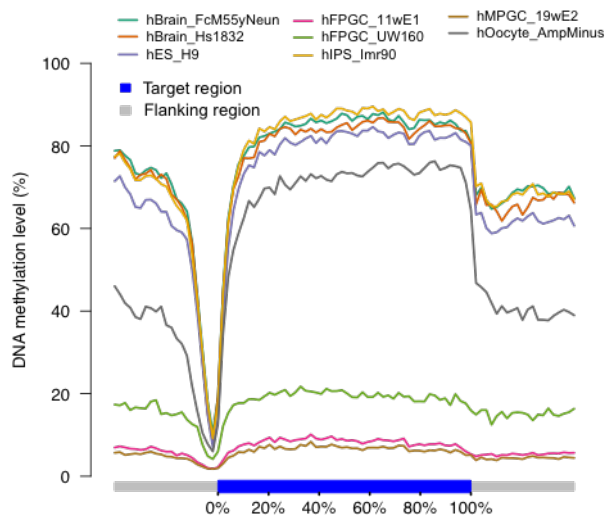


Figure 7.4: DNA methylation distribution across gene body

```
# -f FORMAT, --format=FORMAT
#     [opt] format of output figure. Alternative: png. [default: pdf]
#
# -R RESOLUTION, --resolution=RESOLUTION
#     [opt] Resolution in ppi. Only available for png format. [default: 300]
#
# -h, --help
#     Show this help message and exit
```

- Example

The input file can be generated from the output of **cgmaptools mfg**.

```
...
cgmaptools mfg -i S1.CGmap.gz -r fragreg.bed -c 2 -x CG > S1.mfg
cgmaptools mfg -i S2.CGmap.gz -r fragreg.bed -c 2 -x CG > S2.mfg
(head -1 S1.mfg | gawk '{ $1="Sample"; print $0; }');
for F in *.mfg; do
    gawk -vSampleName=`echo $F | sed s/.mfg//g` '/total_ave_mC/{ $1=SampleName; print $0; }'
done
) > mfg_merge.xls
cgmaptools fragreg -i mfg_merge.xls -o merge.fragreg.pdf -f pdf
...
```

- Output figure

7.4 tanghulu

The **Tanghulu** plot is designed as show the methylation state on each cytosine by reads. (See what does “*Tanghulu*” strand for? Wikipedia)

- Command

```
cgmaptools tanghulu -h
```

```
# DESCRIPTION
#       Circle plot representing DNA methylation of each C [default CpG] site
#       on each mapped reads.
#
# USAGE
#       cgmaptools tanghulu [options] -r <ref> -b <bam> -l chr1:133-144
#       or: cgmaptools tanghulu [options] -r <ref> -b <bam> -l chr1:133
#       (aka mCTanghulu)
#
#       Options:
#       -r      Samtools indexed reference genome sequeunce, fasta format. eg. hg19.fa
#               - use samtools to index reference: samtools faidx <hg19.fa>
#       -b      Samtools indexed Bam file to view.
#               - use samtools to index bam file: samtools index <input.bam>
#       -l      Region in which to display DNA methylation.
#               - or specify a single position (eg. heterozygous SNP site), we will show allele specif
#       -s      Path to samtools eg. /home/user/bin/samtools
#               - by default, we try to search samtools in your system PATH.
#       -o      Output results to file [default: CirclePlot.Ctype.region.Date.pdf].
#       -t      C context. [default: CG]
#               - available context: C, CG, CH, CW, CC, CA, CT, CHG, CHH
#       -d      Ouput device. [default: pdf]
#               - alternative: png
#       -c      Seperate reads by chain. [default: OFF]
#               - specify this option to turn ON.
#       -v      Show vague allele linked reads. [ default: OFF]
#       -g      Genotype of heterozygous SNP site.
#               - This option provides two alleles of htSNP site. eg. AT
#               - The genotype information can be used to reduce vague alleles.
#               - This option is specific to display methylation in allele specific mode.
#       -D      Minimum number of reads (depth) covered in this region or allele linked. [default: 0|0]
#       -C      Minimum number of C (specified type) covered in this region or allele linked. [default
#       -W      Width of graphics reigon in inches. [default: 4]
#       -H      Height of graphics reigon in inches. [default: 4]
#       -R      Resolution in ppi. [default: 300]
#               - only available for png device.
#       -h      Help message.
#
# AUTHOR
#       Contact:      Zhu, Ping; pingzhu.work@gmail.com
#       Last update: 2016-12-07
```

- Example

```
cgmaptools tanghulu -r genome.fa -b WG.bam -l chr1:2000-2400 -t CG
```

- Output figure

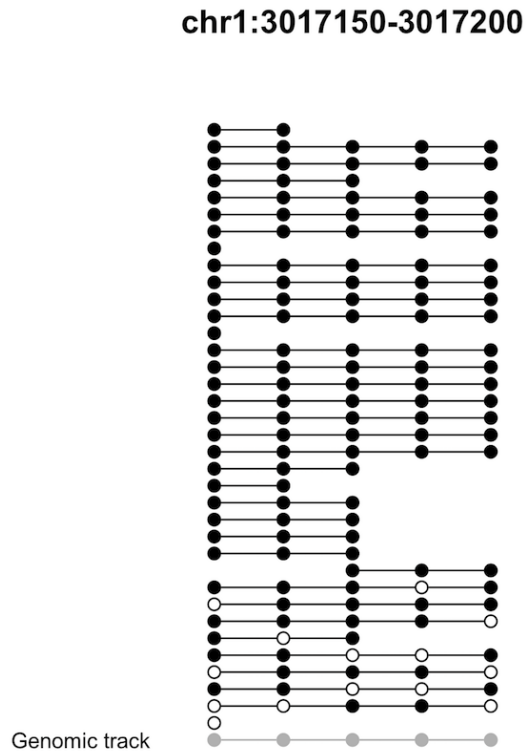


Figure 7.5: Tanghulu plot example

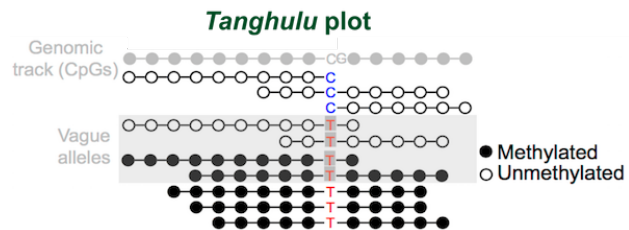


Figure 7.6: Tanghulu plot show vague-reads

We also designed **Tanghulu** plot for visualizing reads that are support methylated, un-methylated, and vague reads for Allele-Specific Methylation (ASM) region.

Chapter 8

Other Utilities

8.1 findCCGG

- Command

```
cgmaptools findCCGG -h
```

```
# Usage: cgmaptools findCCGG -i <genome.fa> [-o <output>]
#       (aka FiindCCGG)
# Description: Get the positions of all the C'CGG---CCG'G fragments.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
# Output Ex:
#       chr1    4025    5652
#       chr1    8274    8431
#
# Options:
# -h, --help  show this help message and exit
# -i FILE     Genome sequence file in Fasta format
# -o FILE     Name of the output file (standard output if not specified).
#             Format: chr cCgg_pos ccGg_pos (0-base)
```

- Example

```
cgmaptools findCCGG -i genome.fa -o genome.ccgg
```

8.2 bed2fragreg

- Command

```
cgmaptools bed2fragreg -h
```

```
# Usage: cgmaptools bed2fragreg [-i <BED>] [-n <N>] [-F <50,50,...> -T <50,...>] [-o output]
#       (aka FragRegFromBED)
# Description: Generate fragmented regions from BED file.
# Contact:    Guo, Weilong; guoweilong@126.com
# Last Update: 2018-01-02
#       Split input region into N bins, get fragments from 5' end and 3' end.
# Input Ex:
```

| Col | Field | Type | Regexp/Range | Brief description |
|-----|--------|--------|------------------------|--|
| 1 | CHR | String | [!-?A-~]* | Query template NAME |
| 2 | STRAND | Char | [+-] | Strand |
| 3 | POS_1 | Int | [0,2 ³² -1] | The 1 st position from 5' end |
| 4 | POS_2 | Int | [0,2 ³² -1] | The 2 nd position from 5' end |
| ... | ... | ... | ... | ... |
| n+2 | POS_n | Int | [0,2 ³² -1] | The nth position from 5' end |

Figure 8.1: Output format description for `cgmmaptools bed2fragreg`

```
# chr1 1000 2000 +
# chr2 9000 8000 -
# Output Ex:
# chr1 + 940 950 1000 1200 1400 1600 1800 1850
# chr2 - 9060 9050 9000 8800 8600 8400 8200 8150
#
#
# Options:
# -h, --help show this help message and exit
# -i FILE BED format, STDIN if omitted
# -F INT_list List of region lengths in upstream of 5' end, Ex: 10,50. List
# is from 5'end->3'end
# -T INT_list List of region lengths in downstream of 3' end, Ex: 40,20. List
# is from 5'end->3'end
# -n INT Number of bins to be equally split [Default:1]
# -o OUTFILE To standard output if omitted. Compressed output if end with
# .gz
```

- Example

- Output format

- Example

```
chr1 + 940 950 1000 1200 1400 1600 1800 2000 2060 2080
chr2 - 9060 9050 9000 8800 8600 8400 8200 8000 7960 7940
```

- Column Description

[POS_1, POS_2), [POS_2, POS_3), ... [POS_(n-1), POS_n) will be used as input for `cgmmaptools mfg`