The MOABS Pipeline Document

A Technical Document
written by
DEQIANG SUN
Baylor College of Medicine

June 12, 2013
For MOABS v1.2.1

Contact: deqiangs@bcm.edu
Abstract

**MOABS: MOdel based Analysis of Bisulfite Sequencing data**

**Keywords:** Bioinformatics, Biostatistics, Methylation, Hydroxymethylation, WGBS, RRBS, DMR, Differential Methylation

In one sentence, MOABS is a complete, accurate and efficient solution for analysis of large scale base-resolution DNA methylation data, bisulfite sequencing or single molecule direct sequencing.

5-methylcytosine and 5-hydroxymethylcytosine can now be quantitatively measured at base level by whole genome bisulfite sequencing. However, lack of complete and accurate methods describing and utilizing digital methylation information from single base to region level, and lack of accurate and fast analysis pipeline are still two major challenges. They are now solved by MOABS, a complete, accurate and efficient solution for methylation data analysis. It seamlessly integrates alignment, methylation calling, identification of hypomethylation for one sample and differential methylation for multiple samples, and other downstream analysis. We show that it is aware of replicate reproducibility, measures biological significance, and is accurate even at low coverage. It uses advanced algorithms and efficiently utilizes threads and clusters so that 2 billion aligned reads from two conditions can be processed lightening fast in 1 hour (vs more than 1 day by other pipelines) analyzing methylation on around 30 million CpGs.
# TABLE OF CONTENTS

## CHAPTER | Page
--- | ---
1 | **OVERVIEW** | 2
   | 1. Introduction | 2
   | 2. Summary of available tools | 3
   | 3. Implementation and algorithmic approach | 3
   | 4. License and Availability | 4
   | 5. Cite MOABS | 4
   | 6. Contact | 4
2 | **INSTALLATION OF MOABS** | 5
   | 1. Downloads | 5
   | 2. Prerequisites for installation from sources | 5
   | 3. Install MOABS from sources | 8
3 | **MANUAL** | 10
   | 1. Usage of master script | 10
     | 1. The solution in brief | 10
     | 2. The configuration file | 11
     | 3. The command line options | 14
   | 2. Usage of mcall module | 14
     | 1. Summary of usage and option | 15
     | 2. Format of input and output files | 17
     | 3. Examples | 18
   | 3. Usage of mcomp module | 19
     | 1. Summary of usage and option | 19
     | 2. Format of input and output files | 21
     | 3. Examples | 21
4 | **MISC** | 23
   | 1. News | 23
REFERENCES | 24
CHAPTER 1

OVERVIEW

1. Introduction

The MOABS development was motivated by the challenges from the lack of accurate and efficient data analysis pipelines suited to large scale DNA methylation data.

5-methylcytosine and 5-hydroxymethylcytosine can now be quantitatively measured at base level by whole genome bisulfite sequencing. However, lack of complete and accurate methods describing and utilizing digital methylation information from single base to region level, and lack of accurate and fast analysis pipeline are still two major challenges.

They are now solved by MOABS, model based analysis of bisulfite sequencing data. It provides a complete, accurate and efficient and biologist friendly solution for analysis of large-scale DNA methylation data from single cytosine level to region level. It seamlessly integrates alignment, methylation calling, identification of hypomethylation for one sample and differential methylation for two samples, and other downstream analysis. It is aware of replicate reproducibility, measures biological significance, and is accurate even at low coverage. It uses advanced algorithms and efficiently utilizes threads and clusters so that 2 billion aligned reads from two conditions can be processed lightening fast in 1 hour (vs more than 1 day by other pipelines) analyzing methylation on around 30 million CpGs.

We also make the method used by MOABS to detect differentially methylated cytosines (DMCs) and differentially methylated regions (DMRs) publically available at the ExactNumCI project.
2. **Summary of available tools**

The main program moabs is actually wrapper of 4 modules, mmap, mcall, mone, and mcomp.

The bisulfite reads alignment module, named as 'mmap', is just a wrapper of popular mapping program so that alignment jobs can be efficiently processed in parallel on a cluster.

The methylation ratio calling module, named as 'mcall', accepts alignment from BSMAP, novoalign and Bismark and then performs methylation ratio calling on every covered cytosine.

For one condition analysis, the MOABS module, named as 'mone', detects hypomethylated regions in the highly methylated methylome (mammalian 5mc) or hypermethylated regions in the lowly methylated methylome (plant and fungi 5mc, and mammalian 5hmc).

For two conditions analysis, the MOABS module, named as 'mcomp', detects the differential methylation at DMC and DMR level by active search, or determines if the given regions are differentially methylated.

All the modules can be executed separately for different stages of research. They are also piped in the single master script 'moabs', such that the user can specify only the input sequences and all the modules will be executed one by one automatically.

3. **Implementation and algorithmic approach**

The master script 'moabs' was written in Perl. And other modules of MOABS were implemented in C++ and makes extensive use of data structures and fundamental algorithms from the BOOST and Numerical Recipes (NR) libraries.
4. **License and Availability**

MOABS is freely available under a GNU GPL v2 at google code site

https://code.google.com/p/moabs

5. **Cite MOABS**

To be updated.

6. **Contact**

The package is developed by Deqiang Sun. Please post any questions, suggestions or problems to the MOABS Discussion google group or send email to Deqiang Sun at moabs_msuite@googlegroups.com. You are welcome to subscribe to the MOABS Discussion google group for updates.
CHAPTER 2

INSTALLATION OF MOABS

1. Downloads

There are three downloads available at
http://dldcc-web.brc.bcm.edu/lilab/deqiangs/moabs/.

“moabs-v1.2.2.tar.gz” includes only sources.

“moabs-v1.2.2.data.tar.gz” includes the sources, the binaries built on x86_64 Linux, and the test data.

2. Prerequisites for installation from sources

If you use x86_64 Linux, you may simply download the binaries. If you build MOABS from sources, you need install Boost C++ libraries and SAMTOOLS because the mcomp module depends on them. It’s optional to install several perl modules which are used by the wrapper script.

Install Samtools:

Download and unpack the Samtools tarball and cd to the Samtools source directory.

Build the Samtools by typing “make” at the command line.

Choose a directory into which you wish to copy the Samtools binary, the included library libbam.a, and the library headers. A common choice is /usr/local/. In this case, the system variable $SAMTOOLS is /usr/local/.

Copy libbam.a to the lib/ directory in the folder you’ve chosen (e.g. /usr/local/lib/).

Create a directory called “bam” in the include/ directory (e.g. /usr/local/include/bam/).
Copy the headers (files ending in .h) to the include/bam directory you’ve created above (e.g. /usr/local/include/bam).

Copy the samtools binary to some directory in your PATH.

If you install Samtools in your personal dir, then just replace the above string ‘/usr/local/’ by your desired installation directory.

**Install Boost:**

You must have the Boost C++ libraries (version 1.38 or higher) installed on your system. Build and install by typing the following two commands in the terminal.

```
./bootstrap.sh
./b2 install
```

The “./b2 install” command installs BOOST libraries into a system directory (e.g. /usr/local/ at default). You may install it to a personal directory by specifying the “prefix” option.

You may have to export `LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH` if you Boost lib is by default in /usr/local/lib/.

**Optionally install perl modules:**

Each module of MOABS is written by C++ but there is a perl script as the wrapper of all modules so that MOABS pipeline can be executed automatically on cluster of computer nodes or on many threads of a computer. This wrapper is written in perl so that you may simply modify cluster setting, or other parameters and so on.

To install the Config::Simple module, it will be easy to install by root user. Just simply type ‘cpan’ in the terminal, and ‘install Config::Simple’ in the interactive shell.

The following commands show how to install the Config::Simple module in a user
directory, say, /myperl/.
Type cpan to enter cpan interactive shell (marked by “cpan>”),
cpan> o conf makepl_arg 'PREFIX=/myperl'
cpan> install Config::Simple
Exit the cpan shell, and type the following command in terminal:
export PERL5LIB=$PERL5LIB:/myperl/lib/perl5/site_perl/5.8.8/
Test if the installation is successful by executing the command on terminal
perl -e 'use Config::Simple;' 
Seeing nothing means good. If you see “Can't locate Config/Simple.pm in @INC”,
it means either the installation is not successful or the variable PERL5LIB is not correct.

There are some other perl modules used by MOABS but most should be installed by perl at default. If you need install additional modules, just follow the same procedures above.

Install RInside:
The mcomp module used R to perform Fisher’s Exact Test and Anova statistics for multiple samples. By the way, the C code extracted from R for Fisher’s Exact Test is unfortunately not safe at Boost Threads. Please help me if you have such C/C++ sources. So I just used an quick solution to use R directly. For this purpose, you need install the R package RInside.

Commands for installation under Linux/Windows terminal:
wget http://cran.r-project.org/src/contrib/RInside_0.2.10.tar.gz
R CMD INSTALL RInside_0.2.10.tar.gz

Or you can install through the R terminal:
Set environment variables for installation:

After build and install of Boost and Samtools, you must have system variables SAMTOOLS and BOOST_ROOT defined for a successful MOABS installation. I do not have time to write and test a configure script, so let’s go with this easy choice. You may test them by executing command “echo $SAMTOOLS $BOOST_ROOT”.

On my system, I set them by inserting the following commands in ~/.bashrc file or by executing them in terminal:

```
export SAMTOOLS=/dsun/samtools/0.1.16
export BOOST_ROOT=/dsun/boost/1.46.1
```

These two commands instruct the build to find libbam.a at /dsun/samtools/0.1.16/lib/libbam.a,

- to find bam.h at /dsun/samtools/0.1.16/include/bam/bam.h,
- to find /dsun/boost/1.46.1/include/,
- to find /dsun/boost/1.46.1/lib/libboost_thread.so and so on.

3. Install MOABS from sources

Suppose you downloaded and unpacked MOABS to /home/dsun/ directory, commands for installation under Linux/Windows terminal are:

```
cd /home/dsun/moabs-v1.2.2
make install
```

The binaries are then installed in /home/dsun/moabs-v1.2.2/bin/. You may simply copy all files into your PATH directories. Or, you may add the complete path to moabs-v1.2.2/bin/ to your PATH variable by executing command

```
> install.packages("RInside",dependencies=TRUE)
```
export PATH=/home/dsun/moabs-v1.2.2/bin:$PATH

Congratulations, you have built MOABS!

You may type moabs to see if you need install any additional perl modules. You may also type mcomp to see if the Boost lib files can be found. If not, use "echo $LD_LIBRARY_PATH" to check it. Note that for the mcomp program to run, the current working dir (where you type mcomp) can not contain the hidden file '.RData', which is auto-saved by R. Hopefully, this problem will be solved in later versions.
1. Usage of master script

The main program moabs is actually wrapper of 4 modules, mmap, mcall, mone, and mcom in order to automate the process in a cluster or thread environment. You may invoke all modules through moabs in addition to directly use them.

1. The solution in brief

One may simply finish the whole processing of bisulfite data for two conditions by typing

```moabs -i wt_r1.fq -i wt_r2.fq -i ko_r1.fq -i ko_r2.fq```

or

```moabs --cf my_research_config_file```

Done! Here for purpose of easy illustration, I have only used 4 files with 2 conditions and 2 replicates each condition. The four file names are formatted such that the underscore ‘_’ separates sample name, replicate name, paired-end mate(if possible). Without specifying the configuration file, the moabs pipeline at default starts with the mmap module in thread mode and parses the condition labels to be wt and ko.

The configuration file “my_research_config_file” looks like the following

```
[INPUT]
s1_r1=wt_r1.fq
s1_r2=wt_r2.fq
s2_r1=ko_r1.fq
s2_r2=ko_r2.fq
```
Here 2 sections of the configuration file are shown. In the [INPUT] section, the
eightside is the real name of the file and the leftside denotes the sample name,
replicate name, paired-end mate(if possible). You may include as many files as
you want. In the [TASK] section, Program denotes the first module to run, Label
denotes the comma-separated sample names corresponding to s1 and s2, Parallel
denotes whether to use thread or qsub to parallelize the tasks.

2. The configuration file

Since there are multiple modules and three ways to execute the pipeline over the
computing resources, I think a configuration file is much more flexible than command
line options. With the use of configuration file, you can simply run the MOABS
pipeline by entering command

```
moabs --cf my_research_config_file
```

By setting “Program=MCOMP” in the config invokes the mcomp module through
moabs. The config looks like the following for example:

```
[INPUT]
#Input is parsed to be wt.G.bed and ko.G.bed from 'Label'

[TASK]
Program=MCOMP
Label=wt,ko
```

where the task is defined such that mcomp module is invoked for file wt.G.bed and
file ko.G.bed.

By setting “Program=MCALL” in the config invokes the mcall module through moabs. The config looks like the following for example:

```
[INPUT]
s1_r1=wt_r1.bam
s1_r2=wt_r2.bam
s2_r1=ko_r1.bam
s2_r2=ko_r2.bam

[TASK]
Program=MCALL
Label=wt,ko
```

where the task is defined such that mcall module is invoked for 4 bam files and the mcomp module follows to process the two condition labels.

By setting “Program=MMAP” in the config invokes the mmap module through moabs. The last subsection [?] showed a elegant and simple example configuration. The following is a example a little bit complicated:

```
[INPUT]
s1_r1=s1_r1.fq
s1_r2_1=s1_r2_1.fq
s1_r2_2=s1_r2_2.fq
s2_r1=ko_r1.fq
s2_r2_1=ko_r2_1.fq
s2_r2_2=ko_r2_1.fq

[TASK]
Program=MCALL
```
Label=wt,ko

where the task is defined such that mmap and mcall module is invoked for the fastq file(s) for each condition and replicate, and then the mcomp module follows to process the two condition labels.

There are three allowed values for key Parallel: NONE, THREAD, or QSUB. For the QSUB mode, the master script need run on the head node. It assigns jobs across the cluster based on a template file. The template file bin/template_for_qsub for is written for Sun Grid Engine (SGE). However, you can easily modify that file for similar cluster manager like PBS. You may also modify the template file bin/template_for_qsub for it to be suitable for your cluster.

The parameters for each module can also be defined in the config file by block instruction [MMAP], [MCALL] and [MCOMP]. In addition to the original options from the module, each module comes with an additional key “Path” which defines the full path to the module. For example, you may use Novoalign as the aligner for the mmap module through the configuration like the following:

[MMAP]
Path=/dir/to/novoalign
b=2

At default with the fastq input files, the master script will automatically call mmap, mcall, and mcomp modules one after one. However the process may be interrupted by computer crash or other problems. You may want to restart the program but do not want to start from the very beginning. The master script moabs checks if there exists the result file from each step even if you start with the mmap module. So if you want to rerun the mmap module, you can move away the all or some bam files. So if you want to rerun the mcall module, you can move away the
all or some *.G.bed files. So if you want to rerun the mcomp module, you can move away the comp.* file.

3. The command line options

There’s also traditional usage of program moabs through command line options. By simply entering moabs in the command line, you notice there are only three options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>input files.</td>
</tr>
<tr>
<td>--cf</td>
<td>configuration file.</td>
</tr>
<tr>
<td>--def</td>
<td>overwrite definitions in configuration file. --def key=value</td>
</tr>
</tbody>
</table>

Here -i specifies the input files, the option --cf specifies a configuration file, and --def provides a way to overwrite the definitions in the configuration file.

For example if you want to use a different number of threads for mcall and different version of bsmmap than the setting in config, you may change the config file or use the --def option: moabs --cf myrun.cfg --def MMAP.Path=/mydir/bsmap --def MCALL.p=1

2. Usage of mcall module

The methylation ratio calling module, named as mcall, accepts alignment from BSMAP (at default), novoalign and Bismark and then performs methylation ratio calling on every covered cytosine. The procedure may perform differently depending on whether it is RRBS or WGBS data. In short, methylation ratio calling module adjusts amplification bias, end-repair bias, short fragment bias, accurately reports CpG or CpH methylation and corresponding methylation confidence interval, in addition to general statistics and estimated bisulfite conversion ratio.
1. Summary of usage and option

**Usage:** $ mcall [options] -m bam/sam -m bam/sam

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>help,h</td>
<td>Produce help message. Common options are provided with single letter format. Parameter defaults are in brackets. Example command: mCall -m Ko.bam; mCall -m wt_r1.bam -m wt_r2.bam -sampleName Wt; See doc for more details.)</td>
</tr>
<tr>
<td>mappedFiles,m</td>
<td>Specify the names of RRBS/WGBS alignment files for methylation calling. Multiple files can be provided to combine them (e.g., lanes or replicates) into a single track;</td>
</tr>
<tr>
<td>sampleName</td>
<td>If two or more mappedFiles are specified, this option generates a merged result; Ignored for one input file;</td>
</tr>
<tr>
<td>outputDir</td>
<td>The name of the output directory;</td>
</tr>
<tr>
<td>webOutputDir</td>
<td>The name of the web-accessible output directory for UCSC Genome Browser tracks;</td>
</tr>
<tr>
<td>genome,g</td>
<td>The UCSC Genome Browser identifier of source genome assembly; mm9 for example;</td>
</tr>
<tr>
<td>reference,r</td>
<td>Reference DNA fasta file; It’s required if CHG methylation is wanted;</td>
</tr>
<tr>
<td>cytosineMinScore</td>
<td>Threshold for cytosine quality score (default: 20). Discard the base if threshold is not reached;</td>
</tr>
<tr>
<td>nextBaseMinScore</td>
<td>Threshold for the next base quality score (default: 3, i.e., better than B or #); Possible values: -1 makes the program not to check if next base matches reference; any positive integer or zero makes the program to check if next base matches reference and reaches this score threshold;</td>
</tr>
<tr>
<td>reportSkippedBase</td>
<td>Specify if bases that are not accepted for methylation analysis should be written to an extra output file;</td>
</tr>
<tr>
<td>qualityScoreBase</td>
<td>Specify quality score system: 0 means autodetection; Sanger=&gt;33; Solexa=&gt;59; Illumina=&gt;64; See wiki FASTQ_format for details;</td>
</tr>
<tr>
<td>trimWGBSEndRepairPE2Seq</td>
<td>How to trim end-repair sequence from begin of +/- -- reads from Pair End WGBS Sequencing; 0: no trim; n (positive integer): trim n bases from begin of +/- -- reads; -2: model determined n; -1: trim from beginning to before 1st methylated C; Suggest 3; n&gt;readLen is equivalent to use PE1 reads;</td>
</tr>
<tr>
<td>trimWGBSEndRepairPE1Seq</td>
<td>How to trim end-repair sequence from end of ++/+- reads from Pair End WGBS Sequencing; 0: no trim; n (positive integer): trim n + NM bases from end of ++/+ reads if fragSize &lt;= maxReadLen; -2: model determined n; Suggest 3;</td>
</tr>
<tr>
<td><strong>processPEOverlapSeq</strong></td>
<td>1/0 makes the program count once/twice the overlap seq of two pairs;</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>trimRRBSEndRepairSeq</strong></td>
<td>How to trim end-repair sequence for RRBS reads; RRBS or WGBS protocol can be automatically detected; 0: no trim; 2: trim the last CG at exactly end of ++/+- reads and trim the first CG at exactly begin of +/-/-- reads like the WGBS situation;</td>
</tr>
<tr>
<td><strong>skipRandomChrom</strong></td>
<td>Specify whether to skip random and hadrop chrom;</td>
</tr>
<tr>
<td><strong>requiredFlag,f</strong></td>
<td>Requiring samtools flag; 0x2(properly paired), 0x40(PE1), 0x80(PE2), 0x100(not unique), r=0x10(reverse); Examples: -f 0x10 &lt;=&gt; ++/-+ (Right) reads; -f 0x40 &lt;=&gt; ++/-+ (PE1) reads; -f 0x50 &lt;=&gt; --/+ read; -f 0x90 &lt;=&gt; ++ read;</td>
</tr>
<tr>
<td><strong>excludedFlag,F</strong></td>
<td>Excluding samtools flag; Examples: -f 0x2 -F 0x100 &lt;=&gt; uniquely mapped pairs; -F 0x10 <code>&lt;=&gt; ++/-- (Left) reads; -F 0x40 </code>&lt;=&gt; -f 0x80 ++/-- (PE2) reads; -f 0x40 -F 0x10 <code>&lt;=&gt; ++ read; -f 0x80 -F 0x10 </code>&lt;=&gt; -- read;</td>
</tr>
<tr>
<td><strong>minFragSize</strong></td>
<td>Requiring min fragment size, the 9th field in sam file; Since non-properly-paired read has 0 at 9th field, setting this option is requiring properly paired and large enough fragment size;</td>
</tr>
<tr>
<td><strong>minMMFragSize</strong></td>
<td>Requiring min fragment size for multiply matched read; Same as option above but only this option is only applicable to reads with flag 0x100 set as 1;</td>
</tr>
<tr>
<td><strong>reportCpX</strong></td>
<td>po::value&lt;char&gt;()-&gt;default_value(‘G’), ‘X=G generates a file for CpG methylation; A/C/T generates file for CpA/CpC/CpT meth;</td>
</tr>
<tr>
<td><strong>reportCHX</strong></td>
<td>po::value&lt;char&gt;()-&gt;default_value(‘X’), ‘X=G generates a file for CHG methylation; A/C/T generates file for CHA/CHC/CHT meth; This file is large;</td>
</tr>
<tr>
<td><strong>fullMode,a</strong></td>
<td>Specify whether to turn on full mode. Off(0): only *.G.bed, *.HG.bed and *_stat.txt are allowed to be generated. On(1): file *.HG.bed, *.bed, *.skip.bed, and *.strand.bed are forced to be generated. Extremely large files will be generated at fullMode.</td>
</tr>
<tr>
<td><strong>statsOnly</strong></td>
<td>Off(0): no effect. On(1): only *_stat.txt is generated.</td>
</tr>
<tr>
<td><strong>keepTemp</strong></td>
<td>Specify whether to keep temp files;</td>
</tr>
<tr>
<td><strong>threads,p</strong></td>
<td>Number of threads on all mapped file. Suggest 1sim8 on EACH input file depending RAM size and disk speed.</td>
</tr>
</tbody>
</table>
2. Format of input and output files

The input files defined by -m (i.e. --mappedFiles) are sam or bam files. If it’s a bam file, it need be sorted. The sam/bam must include a field to specify which bisulfite strand the read is sequenced from. In the BSMAP result this field is for example “ZS:Z:+”.

If the --fullMode option is set to 0, only *.G.bed, *.HG.bed and *_stat.txt are allowed to be generated.

The *.G.bed and *.HG.bed files report the CG and CHG methylation. Here is the format

<table>
<thead>
<tr>
<th>chrom</th>
<th>start</th>
<th>end</th>
<th>ratio</th>
<th>totalC</th>
<th>methC</th>
<th>strand</th>
<th>next</th>
<th>Plus</th>
<th>totalC</th>
<th>methC</th>
<th>Minus</th>
<th>totalC</th>
<th>methC</th>
</tr>
</thead>
<tbody>
<tr>
<td>chr10</td>
<td>308</td>
<td>390</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>B</td>
<td>G</td>
<td>+</td>
<td>4</td>
<td>4</td>
<td>-</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>chr10</td>
<td>510</td>
<td>512</td>
<td>0.5</td>
<td>10</td>
<td>5</td>
<td>-</td>
<td>G</td>
<td>+</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

where B means info is from Both strands, next is the nucleotide after C, Plus and following two columns show the info from the + strand, and Minus and following two columns show the info from the - strand.

The *_stat.txt file reports various statistics.

It first reports the number of all reads in sam/bam file and number of mapped reads:

```
All reads = 46308748; Mapped reads = 46308748
```

It then reports “Strand specific” statistics where the CG dimer on two strands are regarded as two Cytosines.

<table>
<thead>
<tr>
<th>next</th>
<th>strand</th>
<th>sites</th>
<th>mean</th>
<th>totalC</th>
<th>methC</th>
<th>global</th>
<th>depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>+</td>
<td>36752215</td>
<td>0.76%</td>
<td>76555557</td>
<td>565290</td>
<td>0.74%</td>
<td>2.08302</td>
</tr>
<tr>
<td>C</td>
<td>-</td>
<td>36411616</td>
<td>0.76%</td>
<td>74628171</td>
<td>563779</td>
<td>0.76%</td>
<td>2.0493</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>73168631</td>
<td>0.76%</td>
<td>151183728</td>
<td>1129069</td>
<td>0.75%</td>
<td>2.06624</td>
</tr>
<tr>
<td>G</td>
<td>+</td>
<td>7227348</td>
<td>85.71%</td>
<td>25700731</td>
<td>22522713</td>
<td>87.63%</td>
<td>3.55604</td>
</tr>
<tr>
<td>G</td>
<td>-</td>
<td>7188816</td>
<td>85.65%</td>
<td>23335829</td>
<td>20324113</td>
<td>87.09%</td>
<td>3.24613</td>
</tr>
<tr>
<td>G</td>
<td>B</td>
<td>14416164</td>
<td>85.68%</td>
<td>49036560</td>
<td>42846826</td>
<td>87.38%</td>
<td>3.4015</td>
</tr>
<tr>
<td>A</td>
<td>+</td>
<td>59108520</td>
<td>0.85%</td>
<td>134242394</td>
<td>1061459</td>
<td>0.79%</td>
<td>2.27112</td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>58644042</td>
<td>0.85%</td>
<td>132252113</td>
<td>1059043</td>
<td>0.79%</td>
<td>2.25517</td>
</tr>
<tr>
<td>T</td>
<td>+</td>
<td>53335539</td>
<td>0.81%</td>
<td>119944928</td>
<td>885216</td>
<td>0.74%</td>
<td>2.24887</td>
</tr>
<tr>
<td>T</td>
<td>-</td>
<td>52891643</td>
<td>0.81%</td>
<td>116964923</td>
<td>879988</td>
<td>0.79%</td>
<td>2.21141</td>
</tr>
<tr>
<td>T</td>
<td>B</td>
<td>106227182</td>
<td>0.81%</td>
<td>236909851</td>
<td>1765204</td>
<td>0.75%</td>
<td>2.23022</td>
</tr>
</tbody>
</table>

Here the “mean” is mean ratio of all cytosine specified by “next” and “strand” (e.g. “T” and “-”). The “global” is “totalC” divided by “methC”, and hence slightly dif-
ferent than “mean”. The strand “B” denotes all Cytosines from both strands but
still considers each CG dimer as two Cytosines.

It then reports the “Strand combined” statistics where the CG dimer on two strands
are regarded as one Cytosine.

<table>
<thead>
<tr>
<th>next</th>
<th>strand</th>
<th>sites</th>
<th>mean</th>
<th>totalC</th>
<th>methC</th>
<th>global</th>
<th>depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>B</td>
<td>7791723</td>
<td>0.76%</td>
<td>16057280</td>
<td>1198962</td>
<td>0.75%</td>
<td>2.06081</td>
</tr>
<tr>
<td>G</td>
<td>B</td>
<td>12946070</td>
<td>85.30%</td>
<td>59940906</td>
<td>44443423</td>
<td>87.25%</td>
<td>3.93479</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>125662492</td>
<td>0.85%</td>
<td>281423311</td>
<td>22324490</td>
<td>0.79%</td>
<td>2.25026</td>
</tr>
<tr>
<td>T</td>
<td>B</td>
<td>112814040</td>
<td>0.81%</td>
<td>250154072</td>
<td>1868150</td>
<td>0.75%</td>
<td>2.2174</td>
</tr>
</tbody>
</table>

Here the “Strand combined” numbers are different than the “Strand specific” num-
bers as expected.

It then reports the “bisulfite Conversion ratio” as

```
bisulfiteConversionFail: 0.00765666, or bisulfite conversion ratio = 0.992343
```

It then reports the “Strand combined” statistics for mean methylation ratio of C,
CG, CH, CHG, and CHH at different depth cutoff values.

<table>
<thead>
<tr>
<th>depth</th>
<th>NumC</th>
<th>NumCG</th>
<th>NumCH</th>
<th>NumCHG</th>
<th>NumCHH</th>
<th>MeanC</th>
<th>MeanCG</th>
<th>MeanCH</th>
<th>MeanCHG</th>
<th>MeanCHH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>534146840</td>
<td>21342779</td>
<td>512803261</td>
<td>112610026</td>
<td>400193235</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>1</td>
<td>328739835</td>
<td>12946070</td>
<td>315793765</td>
<td>71982073</td>
<td>243811692</td>
<td>4.14%</td>
<td>85.30%</td>
<td>0.81%</td>
<td>0.80%</td>
<td>0.82%</td>
</tr>
<tr>
<td>2</td>
<td>156636760</td>
<td>8038102</td>
<td>148598658</td>
<td>34782601</td>
<td>113816057</td>
<td>5.25%</td>
<td>85.85%</td>
<td>0.89%</td>
<td>0.85%</td>
<td>0.90%</td>
</tr>
<tr>
<td>3</td>
<td>79872726</td>
<td>4350784</td>
<td>75031943</td>
<td>17860683</td>
<td>57181296</td>
<td>6.03%</td>
<td>86.27%</td>
<td>0.77%</td>
<td>0.74%</td>
<td>0.78%</td>
</tr>
<tr>
<td>4</td>
<td>42049689</td>
<td>2839717</td>
<td>39209972</td>
<td>9429506</td>
<td>29780466</td>
<td>6.56%</td>
<td>86.54%</td>
<td>0.73%</td>
<td>0.72%</td>
<td>0.78%</td>
</tr>
<tr>
<td>5</td>
<td>22619338</td>
<td>1645117</td>
<td>20973311</td>
<td>5083334</td>
<td>15850977</td>
<td>6.93%</td>
<td>86.68%</td>
<td>0.67%</td>
<td>0.64%</td>
<td>0.68%</td>
</tr>
</tbody>
</table>

If the --fullMode option is set to 1, the files *.HG.bed, *.bed, *_skip.bed, and
*_strand.bed are forced to be generated. Extremely large files will be generated at
fullMode. It’s not recommended to turn this on, unless you want to diagnose the
procedures or want to see what bases or what reads are “skip” at quality control
step.

3. Examples

```
$ mcall -m ko_r1.bam -m ko_r2.bam --sampleName ko -p 4 -r hg19.fa
```

The easiest use is just use mcall process for all files from one condition. This com-
mand performs the methylation ratio calling for ko_r1.bam and ko_r2.bam paral-
lely and then generate the new ko.G.bed and ko.HG.bed and ko_stat.txt files. This
command is more or less same as “mcall -m ko_r1.bam” and “mcall -m ko_r2.bam”
followed by a merging command “mcomp -r ko_r1.bam.G.bed,ko_r2.bam.G.bed -m ko.G.bed”.

3. **Usage of mcomp module**

For two conditions analysis, the MOABS module, named as mcomp, detects the differential methylation at DMC and DMR level by active search, or determines if the given regions are differentially methylated.

1. **Summary of usage and option**

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>help,h</td>
<td>produce help message;</td>
</tr>
<tr>
<td>email</td>
<td>Specify email;</td>
</tr>
<tr>
<td>ratiosFiles,r</td>
<td>Specify the names of ratio files from methCall. Multiple lane files can be</td>
</tr>
<tr>
<td></td>
<td>separated by , to be combined into a single track; example: -r sample1 -r</td>
</tr>
<tr>
<td></td>
<td>sample2 -r s_r1,s_r2,s_r3;</td>
</tr>
<tr>
<td>mergedRatiosFiles,m</td>
<td>If --ratiosFiles is ',', separated, then this option must be set;</td>
</tr>
<tr>
<td>labels,l</td>
<td>Name labels for samples, default 0, 1, ...;</td>
</tr>
<tr>
<td>outputDir</td>
<td>Specify the name of the output directory;</td>
</tr>
<tr>
<td>webOutputDir</td>
<td>Specify the name of the web-accessible output directory for UCSC Genome</td>
</tr>
<tr>
<td></td>
<td>Browser tracks;</td>
</tr>
<tr>
<td>compFile,c</td>
<td>Name of the comparison file resulted from statistical tests;</td>
</tr>
<tr>
<td>inGenome</td>
<td>Specify the UCSC Genome Browser identifier of source genome assembly;</td>
</tr>
<tr>
<td>outGenome</td>
<td>Specify the UCSC Genome Browser identifier of destination genome assembly;</td>
</tr>
<tr>
<td>xVector</td>
<td>Specify the x vector for R.lm() function;x is comma(,) separated float</td>
</tr>
<tr>
<td></td>
<td>numbers; default, 1.0,2.0,...,;</td>
</tr>
<tr>
<td>precision</td>
<td>Specify the precision of float numbers in output files (default: 3);</td>
</tr>
<tr>
<td>threads,p</td>
<td>Specify number of threads; suggest number 6-12; default 6;</td>
</tr>
<tr>
<td>lmFit</td>
<td>Specify if linear model fitting is performed; default true; Note that 'na'</td>
</tr>
<tr>
<td></td>
<td>is generated if slope is 0;</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>mergeNotIntersect</td>
<td>Specify if genomic locations are merged or intersected among samples; 1 for merge (default) and 0 for intersect;</td>
</tr>
<tr>
<td>doMergeRatioFiles</td>
<td>Internal parameter. Is true when <code>-m</code> parameter is <code>,</code> separated and program will merge ratio files that are separated by ``,<code>and the output files are named according to option</code>-x`;</td>
</tr>
<tr>
<td>doStrandSpecifiMeth</td>
<td>whether strand specific methylation analysis will be performed;</td>
</tr>
<tr>
<td>doComp</td>
<td>doComp;</td>
</tr>
<tr>
<td>minDepthForComp,d</td>
<td>If a site has depth &lt; d then this site is ignored for statistical tests; This option affects much of nominal ratios but none of credible ratios; Suggest 10 for method 2 and 3 for method 2; You may also reset this option during later DMC/DMR rescans to filter sites with depth &lt; d;</td>
</tr>
<tr>
<td>doDmcScan</td>
<td>doDmcScan;</td>
</tr>
<tr>
<td>doDmrScan</td>
<td>doDmrScan;</td>
</tr>
<tr>
<td>filterCredibleDif</td>
<td>if absolute value of cDif for a site &lt; filterCredibleDif, then this site is ignored for regional calculation. Use 0.01 (for example) to filter all sites with no difference; use 0.20 (for example) to select DMCs; Any negative number = no filter;</td>
</tr>
<tr>
<td>dmrMethods</td>
<td>dmrMethods: add $2^x$ methods; examples: 7 for three methods, 4 for method 3 only;</td>
</tr>
<tr>
<td>pFetDmc</td>
<td>Cutoff of P value from Fisher Exact Test for Dmc scan;</td>
</tr>
<tr>
<td>pFetDmr</td>
<td>Cutoff of P value from Fisher Exact Test for Dmr scan;</td>
</tr>
<tr>
<td>minNominalDif</td>
<td>min nominal meth diff for Dmc Dmr;</td>
</tr>
<tr>
<td>pSimDmc</td>
<td>Cutoff P value from Similarity Test for Dmc scan; Since p is always less than 1, default 1 means not a criteria;</td>
</tr>
<tr>
<td>pSimDmr</td>
<td>Cutoff P value from Similarity Test for Dmr scan;</td>
</tr>
<tr>
<td>minCredibleDif</td>
<td>min credible meth diff for Dmc calling, used in M2 or predefined regions;</td>
</tr>
<tr>
<td>topRankByCDif</td>
<td>filter Dmc by asking it to be in top (default 100%) percent by ranking absolute value of credibleDif; suggest 0.05 as the only condition to call Dmc if cDif condition is not preferred; The cutoff cDif will be used as Dmr criteria;</td>
</tr>
<tr>
<td>topRankByPSim</td>
<td>filter Dmc by asking it to be in top (default 100%) percent by ranking P value from Similarity Test;</td>
</tr>
<tr>
<td>minDmcsInDmr</td>
<td>minimum number of Dmcs in a Dmr;</td>
</tr>
<tr>
<td>maxDistConsDmcs</td>
<td>max distance between two consecutive Dmcs for them to be considered in a Dmr;</td>
</tr>
<tr>
<td>predefinedFeature,f</td>
<td>supply bed files as predefined feature; -f promoter.bed -f CpgIsland.bed -f LINE.bed is same as -f promoter.bed,CpgIsland.bed,Line.bed</td>
</tr>
</tbody>
</table>
2. Format of input and output files

The input files defined by -r (i.e. --ratiosFiles) are the *.G.bed files generated by mcall module. If you have methylation ratio files from other programs, you can easily convert them to the *.G.bed format.

The file defined by -c (i.e. --compFile) is the major result of comparison between two conditions. Each row represents a Cytosine. The selected columns are explained below.

<table>
<thead>
<tr>
<th>Header</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@chrom</td>
<td>chr1</td>
<td>chrom</td>
</tr>
<tr>
<td>start</td>
<td>3002598</td>
<td>start</td>
</tr>
<tr>
<td>end</td>
<td>3002600</td>
<td>end</td>
</tr>
<tr>
<td>totalC_0</td>
<td>3</td>
<td>total number of reads for s0</td>
</tr>
<tr>
<td>nominalRatio_0</td>
<td>0.333</td>
<td>nominal ratio for s0</td>
</tr>
<tr>
<td>ratioCI_0</td>
<td>0,0.751</td>
<td>confidence interval of ratio for s0</td>
</tr>
<tr>
<td>totalC_1</td>
<td>4</td>
<td>total number of reads for s1</td>
</tr>
<tr>
<td>nominalRatio_1</td>
<td>0</td>
<td>nominal ratio for s1</td>
</tr>
<tr>
<td>ratioCI_1</td>
<td>0,0.451</td>
<td>confidence interval of ratio for s1</td>
</tr>
<tr>
<td>nominalDif_1-0</td>
<td>-0.333</td>
<td>nominal difference of s1 - s0</td>
</tr>
<tr>
<td>credibleDif_1-0</td>
<td>-0</td>
<td>credible difference for s1 - s0</td>
</tr>
<tr>
<td>difCI_1-0</td>
<td>-0.639,0.17</td>
<td>confidence interval for s1 - s0</td>
</tr>
<tr>
<td>p_sim_1_v_0</td>
<td>0.0214</td>
<td>pvalue from similarity test</td>
</tr>
<tr>
<td>p_fet_1_v_0</td>
<td>0.429</td>
<td>pvalue from fisher’s exact test</td>
</tr>
</tbody>
</table>

Here the credible difference balances the sequencing depth and nominal differences.

For more details, please refer the Chapter [xx] for methods.

3. Examples

This module contains multiple functions depending on how the input files and options are set. Here I list several common usages of this module.

```
$ mcomp [options] -r wt.G.bed -m ko.G.bed -c comp.wt.vs.ko.txt
```

This is in general the first step in your differential analysis for two conditions. It will generate the result for differential test on each Cytosine (file -c comp.wt.ko.txt). It will also generate reports of DMCs and DMRs.

```
$ mcomp [options] -r wt_r1.G.bed,wt_r2.G.bed -r ko_r1.G.bed,ko_r2.G.bed -c comp.wt.vs.ko.txt
```

This command does the same thing but includes the replicate information which may
be used by particular options.

$ mcomp [options] -r wt_r1.bam.G.bed,wt_r2.bam.G.bed
This command will just merge two methylation ratio files.

$ mcomp -c comp.wt.vs.ko.txt --doDmcScan=0 --doDmcScan=1 --dmrMethods=2
--minCredibleDif 0.1
This will rerun the DMR scan using parameters different than default values. The
input here is just the test file so that the lengthy differential testing step neednot
be repeated. You need probably create a different directory because it overwrite
previous results.

$ mcomp -c ../comp.wt.vs.ko.txt -f CGI.bed
This command generates the statistics file and methylation report on all predefined
regions in file CGI.bed.
1. **News**

2013.06.15

The documentation is updated for MOABS v1.2.2. I have used two good references for writing latex and bibtex codes:

http://groups.mrl.uiuc.edu/chiang/czoschke/latex.html

and


The R command output is generated by Sweave.
REFERENCES