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 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/.j
"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:

```
> install.packages(“RInside”,dependencies=TRUE)
```

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

```
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.

CHAPTER 3

MANUAL

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 formated 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
```

```
[TASK]
Program=MMAP
Label=wt,ko
Parallel=THREAD
```

Here 2 sections of the configuration file are shown. In the [INPUT] section, the rightside 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.

-i	input files.
--cf	configuration file.
--def	overwrite definitions in configuration file. –def key=value

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 bsmap 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

Options	Description
help,h	Produce help message. Common options are provided with single letter format. Parameter defaults are in brackts. Example command: mCall -m Ko.bam; mCall -m wt_r1.bam -m wt_r2.bam -sampleName Wt; See doc for more details.)
mappedFiles,m	Specify the names of RRBS/WGBS alignment files for methylation calling. Multiple files can be provided to combine them(eg. lanes or replicates) into a single track;
sampleName	If two or more mappedFiles are specifed, this option generates a merged result; Ignored for one input file;
outputDir	The name of the output directory;
webOutputDir	The name of the web-accessible output directory for UCSC Genome Browser tracks;
genome,g	The UCSC Genome Browser identifier of source genome assembly; mm9 for example;
reference,r	Reference DNA fasta file; It's required if CHG methylation is wanted;
cytosineMinScore	Threshold for cytosine quality score (default: 20). Discard the base if threshold is not reached;
nextBaseMinScore	Threshold for the next base quality score(default: 3,ie, 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;
reportSkippedBase	Specify if bases that are not accepted for methylation analysis should be written to an extra output file;
qualityScoreBase	Specify quality score system: 0 means autodetection; Sanger=>33;Solexa=>59;Illumina=>64; See wiki FASTQ_format for details;
trimWGBSEndRepairPE2Seq	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>readLen is equivalent to use PE1 reads;
trimWGBSEndRepairPE1Seq	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 <= maxReadLen; -2: model determined n; Suggest 3;

processPEOverlapSeq	1/0 makes the program count once/twice the overlap seq of two pairs;
trimRRBSEndRepairSeq	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;
skipRandomChrom	Specify whether to skip random and hadrop chrom;
requiredFlag,f	Requiring samtools flag; 0x2(properly paried), 0x40(PE1), 0x80(PE2), 0x100(not unique), r=0x10(reverse); Examples: -f 0x10 <=> +-/-+ (Right) reads; -f 0x40 <=> ++/-+ (PE1) reads; -f 0x50 <=> -+ read; -f 0x90 <=> +- read;
excludedFlag,F	Excluding samtools flag; Examples: -f 0x2 -F 0x100 <=> uniquely mapped pairs; -F 0x10 <=> ++/-- (Left) reads; -F 0x40 <=> -f 0x80 +-/-- (PE2) reads; -f 0x40 -F 0x10 <=> ++ read; -f 0x80 -F 0x10 <=> -- read;
minFragSize	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;
minMMFragSize	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;
reportCpX	po::value<char>()->default_value('G'), "X=G generates a file for CpG methylation; A/C/T generates file for CpA/CpC/CpT meth;
reportCHX	po::value<char>()->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;
fullMode,a	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.
statsOnly	Off(0): no effect. On(1): only *_stat.txt is generated.
keepTemp	Specify whether to keep temp files;
threads,p	Number of threads on all mapped file. Suggest $1sim8$ on EACH input file depending RAM size and disk speed.

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

chrom	start	end	ratio	totalC	methC	strand	next	Plus	totalC	methC	Minus	totalC	methC
chr10	308	390	1	10	10	B	G	+	4	4	-	6	6
chr10	510	512	0.5	10	5	-	G	+	0	0	-	10	5

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:

```
Allreads = 46308748; Mapped reads = 46308748
```

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

next	strand	sites	mean	totalC	methC	global	depth
C	+	36752215	0.76%	76555557	565290	0.74%	2.08302
C	-	36416416	0.76%	74628171	563779	0.76%	2.0493
C	B	73168631	0.76%	151183728	1129069	0.75%	2.06624
G	+	7227348	85.71%	25700731	22522713	87.63%	3.55604
G	-	7188816	85.65%	23335829	20324113	87.09%	3.24613
G	B	14416164	85.68%	49036560	42846826	87.38%	3.4015
A	+	59108520	0.85%	134242394	1061459	0.79%	2.27112
A	-	58644042	0.85%	132252113	1050043	0.79%	2.25517
A	B	117752562	0.85%	266494507	2111502	0.79%	2.26317
T	+	53335539	0.81%	119944928	885216	0.74%	2.24887
T	-	52891643	0.81%	116964923	879988	0.75%	2.21141
T	B	106227182	0.81%	236909851	1765204	0.75%	2.23022

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 different 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.

next	strand	sites	mean	totalC	methC	global	depth
C	B	77917233	0.76%	160572980	1198962	0.75%	2.06081
G	B	12946070	85.30%	50940096	44443423	87.25%	3.93479
A	B	125062492	0.85%	281423311	2232449	0.79%	2.25026
T	B	112814040	0.81%	250154072	1868150	0.75%	2.2174

Here the “Strand combined” numbers are different than the “Strand specific” numbers 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.

depth	NumC	NumCG	NumCH	NumCHG	NumCHH	MeanC	MeanCG	MeanCH	MeanCHG	MeanCHH
0	534146040	21342779	512803261	112610026	400193235	NA	NA	NA	NA	NA
1	328739835	12946070	315793765	71982073	243811692	4.14%	85.30%	0.81%	0.80%	0.82%
2	156636760	8038102	148598658	34782601	113816057	5.25%	85.85%	0.89%	0.85%	0.90%
3	79872726	4830783	75041943	17860683	57181260	6.03%	86.27%	0.86%	0.82%	0.87%
4	42049689	2839717	39209972	9429506	29780466	6.56%	86.54%	0.77%	0.74%	0.78%
5	22619828	1646517	20973311	5083334	15889977	6.93%	86.68%	0.67%	0.64%	0.68%

--

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 command performs the methylation ratio calling for ko_r1.bam and ko_r2.bam parallely 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

Usage: $ mcomp [options] -r wt.G.bed -m ko.G.bed -c comp.wt.vs.ko.txt
Usage: $ mcomp [options] -r wt_r1.bam.G.bed,wt_r2.bam.G.bed
Usage: $ mcomp [options] -r wt_r1.bed,wt_r2.bed -r ko.bed -c comp.wt.vs.ko.txt
Usage: $ mcomp [options] -c comp.wt.vs.ko.txt -f CGI.bed

Options	Description
help,h	produce help message;
email	Specify email;
ratiosFiles,r	Specify the names of ratio files from methCall. Multiple lane files can be separated by , to be combined into a single track; example: -r sample1 -r sample2 -r s_r1,s_r2,s_r3;
mergedRatiosFiles,m	If --ratiosFiles is ’,’ separated, then this option must be set;
labels,l	Name labels for samples, defaut 0, 1, ...;
outputDir	Specify the name of the output directory;
webOutputDir	Specify the name of the web-accessible output directory for UCSC Genome Browser tracks;
compFile,c	Name of the comparison file resulted from statistical tests;
inGenome	Specify the UCSC Genome Browser identifier of source genome assembly;
outGenome	Specify the UCSC Genome Browser identifier of destination genome assembly;
xVector	Specify the x vector for R.lm() function;x is comma(,) separated float numbers; default, 1.0,2.0,...,;
precision	Specify the precision of float numbers in output files (default: 3);
threads,p	Specify number of threads; suggest number 6-12; default 6;
lmFit	Specify if lenear model fitting is performed; default true; Note that ’na’ is generated if slope is 0;

mergeNotIntersect	Specify if genomic locations are merged or intersected among samples; 1 for merge(default) and 0 for intersect;
doMergeRatioFiles	Internal parameter. Is true when -m parameter is ',' separated and program will merge ratio Files that are separated by ',' and the output files are named according to option -x;
doStrandSpecifiMeth	whether strand specific methylation analysis will be performed;
doComp	doComp;
minDepthForComp,d	If a site has depth $<$ 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 rescan to filter sites with depth $<$ d;
	Below are options for Dmc and Dmr scan;)
doDmcScan	doDmcScan;
doDmrScan	doDmrScan;
filterCredibleDif	if absolute value of cDif for a site $<$ 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;
dmrMethods	dmrMethods: add $2^x methodx; examples : 7forthreemethods, 4formethod3only;$
pFetDmc	Cutoff of P value from Fisher Exact Test for Dmc scan;
pFetDmr	Cutoff of P value from Fisher Exact Test for Dmr scan;
minNominalDif	min nominal meth diff for Dmc Dmr;
pSimDmc	Cutoff P value from Similarity Test for Dmc scan; Since p is alwasys less than 1, default 1 means not a criteria;
pSimDmr	Cutoff P value from Simlarity Test for Dmr scan;
minCredibleDif	min credible meth diff for Dmc calling, used in M2 or predefined regions;
topRankByCDif	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 prefered; The cutoff cDif will be used as Dmr criteria;
topRankByPSim	filter Dmc by asking it to be in top (default 100%) percent by ranking P value from Similarity Test;
minDmcsInDmr	minimum number of Dmcs in a Dmr;
maxDistConsDmcs	max distance between two consective Dmcs for them to be considered in a Dmr;
predefinedFeature,f	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

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.

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

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.

CHAPTER 4

MISC

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

http://schneider.ncifcrf.gov/latex.html.

The R command output is generated by Sweave.

REFERENCES