The MiClip package

Tao Wang
The University of Texas Southwestern Medical Center,
5323 Harry Hines Boulevard Dallas, Texas, 75390
tao.wang@utsouthwestern.edu

November 16, 2013

Abstract

There has been increasing interest in the role of RNA-binding proteins in biological processes. Crosslinking and immunoprecipitation (CLIP) experiments have made it possible to identify binding sites of RNA-binding proteins in various cell culture and tissue types. The two most commonly used types of CLIP-Seq experiments are HITS-CLIP and PAR-CLIP. Here we present MiClip, an R package for identification of binding sites in CLIP-Seq experiments. The MiClip package employs two rounds of Hidden Markov Model (HMM) to identify enriched regions and further high-confidence binding sites from raw sequencing data.

Contents

1 Installation

R (`http://www.r-project.org/`) needs to be installed first for *MiClip* and the installation of the *MiClip* package follows the regular method for R package installation.

However, *MiClip* also requires Perl to be installed. Perl should ship along with any standard UNIX and MacOS distribution. But Windows users probably need to install Perl themselves (`http://www.perl.com/`). The users can type the following line in the command console to check if Perl has been installed properly.

```
perl -v
```

2 Preparation of input files

2.1 Trimming adaptor

During CLIP-Seq experiments, RNAs are usually digested to short fragments. It is quite often for sequenced reads to have adaptor contamination at the 3' end, while it is relatively rare for the 5' end of short reads to have adaptor contamination. Thus, it is necessary to trim contaminating adaptor sequences from 3' end before running alignment. Users can use published softwares like Trimmomatic [3] to trim adaptors.

We encourage users to use these more professional softwares, but we also provide a very simple helper function to remove adaptor sequence. Here we use a small portion of the data from [1] for demonstration. A new file with ".removed" suffix will be generated in the same folder as the original file. On my computer, it takes 40 minutes to process a fastq file of 20 million reads (80 million lines).

```
> library("MiClip")
> MiClip.adaptor(file=system.file("extdata/test.fastq",package="MiClip"),
+                  adaptor="TGGAATTCTCGGGTGCCAAGGAACTCCAGTCAC")
```

2.2 Alignment

The raw sequencing file can be single-end or paired-end in basespace or colorspace. Any mainstream alignment software can be used to align the short reads. The output format must be SAM/BAM format and in basespace. *MiClip* can work on both single-end and paired-end alignment files. In the case where the user wishes to pool the alignment files from several experiments, the user can just concatenate the SAM files simply by typing the following in the command console.

```
cat example1.sam example2.sam > example.sam
```

The MiClip algorithm collects mutation information from the CIGAR and MD fields of each short read. The MD field is a string for mismatching positions characterized by "MD:Z:" towards the end of each entry in the alignment file (`http://samtools.sourceforge.net/SAM1.pdf`). Please make sure the MD fields are present in the aligned reads. If not, the user should install and use samtools to populate the MD fields, please see the instructions by typing the following command in command console.

```
samtools fillmd
```

However, this command runs very slowly. So it will be much better if the user can choose an alignment software which will give MD fields to all mapped tags in the very beginning. I myself only know that bowtie and novoalign produce correct MD fields and tophat cannot. It won't be a bad idea to try your aligner first on a small test dataset and see if MD field is attached before aligning all your samples.

2.3 Multiple-mapping reads

"Multiple mapping" reads are reads that can be mapped to more than one place in the genome. In the alignment process, the user can specify whether/how many hits per read to report in the alignment file while MiClip will take in all reported hits.

2.4 Mapping across splice junctions

Reads that are mapped across splice junctions are discarded (these are different from reads that are mapped with short deletions). These reads typically only occupy less than 3% of total mapped reads. If you insist on analyzing these reads too, please map your reads to trancriptome and then analyze them using *MiClip*.

2.5 Paired-end reads

For paired-end reads, the users must look at the sequencing files and provide the suffix for the forward strand and the backward strand. For example, the mate in the sequencing dataset may be named like "694_122_1972-F3" and "694_122_1972-F5-RNA", where "694_122_1972" is the id number of the mate, "F3" means forward strand and "F5-RNA" means backward strand. Then the suffix should be "F3" and "F5-RNA" or "F3" and "F5-RNA" or "3" and "5-RNA".

Sometimes, the aligner will trim the suffix. For example, "HWI-ST188:8:2217:5190:132924#0/1" and "HWI-ST188:8:2217:5190:132924#0/2" are one mate and certain aligners will only write "HWI-ST188:8:2207:5196:132923#0" for both segments in the alignment file. In such cases, please set the suffix to "" and "" or "#0" and "#0". The point is to make the remaining part of the read names the same for a mate.

MiClip treats paired-end sequencing data as fr-secondstrand by default (consult Tophat manul for definition). It can also handle fr-firststrand if you use opposite bases when specifying substitution mutations. For example, instead of specify T2C, you can use A2G. *MiClip* cannot handle any other type of fr or any type of ff libraries.

3 Running *MiClip*

3.1 Construct a *MiClip* class object for following analysis

The analysis of *MiClip* starts by constructing a `MiClip` object. Here we used a small portion of the single-end HITS-CLIP data provided in [2] for demonstration purpose.

```
> library("MiClip")
> test=MiClip(file=system.file("extdata/test.sam",package="MiClip"),mut.type="Del")
>
> # for paired-end data
> # test=MiClip(file="test.sam",paired=TRUE,suffix=c("F3","F5-RNA"))
```

This command returns a `MiClip` object for further analysis. Following sections will explain some of the available parameters in constructing this object. For detailed descriptions of all parameters, please refer to the *MiClip* manual.

One thing to note is that if you need to include the path name in the file name, the path name cannot start with anything like ”∼”. Namely, you must write the path name in full like ”/home/project/test.sam” rather than ”∼/project/test.sam”.

3.2 Read raw sequencing data and mutation data

The `MiClip.read` function calls some embeded perl scripts to form clusters (CLIP clusters) by overlapping reads and collect tag pile-up as well as mutation information from the input SAM file. This process will usually take a few minutes depending on the size of the file.

```
> test=MiClip.read(test) # read raw data

Identifying clusters finished!
Generating bin file finished!
```

3.3 Identify enriched bins

The `MiClip.enriched` function first collects tag pile-up information on a `step` bp basis (bins) and estimates the paramters for a two-poisson mixture model for the count values. Because we are running a truncated part of the real data for demonstration, so the model estimation will not be accurate. Then the first Hidden Markov Model will try to identify the enriched bins vs. non-enriched bins in CLIP clusters.

```
> test=MiClip.enriched(test,quiet=FALSE) # identify enriched regions

Initialization of the first HMM finished!
>>>>>
Iterations of the first HMM finished!
Viterbi algorithm of the first HMM finished!
```

The `empirical` parameter is devised to adjust model estimation in this step. The default for `empirical` is ”auto”, which lets the algorithm decides its value. User can set this value to roughly the minimal number of overlapping tags for a ”true” cluster according to user’s experience and experimental design. Larger value will lead to more conservative predictions.

3.4 Identify binding sites

The `MiClip.binding` function first concatenates neighboring enriched bins and then expands each chain of adjacent bins into single base pairs. Then `MiClip.binding` collects the tag pile-up and mutant pile-up information on each base for estimation of a mixture model of one zero-inflated binomial distribution and a binomial distribution. Then the second Hidden Markov Model is run to identify significant binding sites.

```
> test=MiClip.binding(test,quiet=FALSE) # identify binding sites

Initialization of the second HMM finished!
>>>
Iterations of the second HMM finished!
Viterbi algorithm of the second HMM finished!
```

The `model.cut` parameter is devised to adjust model estimation in this step. The default for `model.cut` is 0.2. User can set this value to roughly the minimal proportion of mutant tag vs. total tag on true binding sites according to user's experience and experimental design. Larger value will lead to more conservative predictions.

3.5 Screening for SNPs

The *MiClip* package builds in a function `MiClip.snp` for distinguishing true binding sites from possible SNPs. The control can be a sequenced sample that is not processed by the crosslinking step. Optionally, users can do quality screening on the fastq sequencing file before alignment to the reference genome. Then the `MiClip.snp` function will take the alignment file as input. Because there is no real control sample from the original study, we use part of the test.sam file as a fake control sample for demonstration.

```
> test=MiClip.snp(test,file=system.file("extdata/snp.sam",package="MiClip"),mut.type="Del")

Identifying clusters finished!
Generating bin file finished!
```

4 Output of *MiClip*

4.1 Output format

`MiClip.binding` returns a `MiClip` object which normally comprises of three data frames.

```
> enriched=test$enriched # test will contain at least three data frames
> sites=test$sites
> clusters=test$clusters
> head(enriched) # view these data frames

  region_id   chr   start     end strand tag enriched probability
1         1 chr18 3341965 3341969      +   2    FALSE       1.000
2         1 chr18 3341970 3341974      +   3    FALSE       0.999
3         1 chr18 3341975 3341979      +   3    FALSE       0.999
4         1 chr18 3341980 3341984      +   3    FALSE       0.999
5         1 chr18 3341985 3341989      +   3    FALSE       0.999
6         1 chr18 3341990 3341994      +   3    FALSE       0.999

> head(sites)

  region_id sub_region_id strand   chr     pos tag mutant sites probability
1         7             1      + chr18 3399930   3      0 FALSE       0.989
2         7             1      + chr18 3399931   3      3  TRUE       0.996
3         7             1      + chr18 3399932   3      0 FALSE       0.995
4         7             1      + chr18 3399933   3      0 FALSE       0.999
5         7             1      + chr18 3399934  12      0 FALSE       1.000
6         7             1      + chr18 3399935  12      0 FALSE       1.000
    SNP
1 FALSE
2  TRUE
```

```
3 FALSE
4 FALSE
5 FALSE
6 FALSE
```

```
> head(clusters)
```

```
  region_id   chr strand   start     end enriched sites   SNP
1         1 chr18      + 3341965 3342004    FALSE FALSE FALSE
2         2 chr18      + 3362790 3362829    FALSE FALSE FALSE
3         3 chr18      + 3391135 3391169    FALSE FALSE FALSE
4         4 chr18      + 3395450 3395484    FALSE FALSE FALSE
5         5 chr18      + 3396420 3396454    FALSE FALSE FALSE
6         6 chr18      + 3399855 3399889    FALSE FALSE FALSE
```

```
> head(enriched[enriched$enriched,]) # view enriched bins
```

```
   region_id   chr   start     end strand tag enriched probability
49         7 chr18 3399930 3399934      +   5     TRUE       0.749
50         7 chr18 3399935 3399939      +  12     TRUE       1.000
51         7 chr18 3399940 3399944      +  12     TRUE       1.000
52         7 chr18 3399945 3399949      +  11     TRUE       1.000
53         7 chr18 3399950 3399954      +   9     TRUE       1.000
54         7 chr18 3399955 3399959      +   8     TRUE       1.000
```

```
> head(sites[sites$sites,]) # view binding sites
```

```
     region_id sub_region_id strand   chr     pos tag mutant sites probability
2            7             1      + chr18 3399931   3      3  TRUE       0.996
396         82            12      + chr18 4673570   7      3  TRUE       0.973
699        101            21      + chr18 4681128   7      2  TRUE       0.677
910        107            26      + chr18 4682164  14      4  TRUE       0.952
1021       111            28      + chr18 4682610  12      7  TRUE       1.000
1525       218            43      + chr18 5650699   6      3  TRUE       0.983
       SNP
2     TRUE
396  FALSE
699  FALSE
910  FALSE
1021  TRUE
1525 FALSE
```

```
> head(clusters[clusters$enriched,]) # view clusters with enriched bins
```

```
   region_id   chr strand   start     end enriched sites   SNP
7          7 chr18      + 3399910 3399969     TRUE  TRUE  TRUE
9          9 chr18      + 3405685 3405769     TRUE FALSE FALSE
15        15 chr18      + 3421220 3421359     TRUE FALSE FALSE
23        23 chr18      + 3426755 3426809     TRUE FALSE FALSE
27        27 chr18      + 3430955 3430994     TRUE FALSE FALSE
36        36 chr18      + 3435015 3435064     TRUE FALSE FALSE
```

```
> head(clusters[clusters$sites,]) # view clusters with binding sites

    region_id   chr strand   start     end enriched sites   SNP
7           7 chr18      + 3399910 3399969     TRUE  TRUE  TRUE
82         82 chr18      + 4673520 4673639     TRUE  TRUE FALSE
101       101 chr18      + 4681105 4681189     TRUE  TRUE FALSE
107       107 chr18      + 4681985 4682249     TRUE  TRUE FALSE
111       111 chr18      + 4682585 4682639     TRUE  TRUE  TRUE
218       218 chr18      + 5650685 5650729     TRUE  TRUE FALSE
```

`enriched` is the output of the first Hidden Markov Model. `region_id` is the id number for each cluster. `chr`, `strand`, `start` and `end` specify the genomic location of each bin. `tag` is the rounded average tag count in each bin. `enriched` and `probability` are the inference results.

`sites` is the output of the second Hidden Markov Model. `region_id` is the id number for the cluster which each base resides in. `sub_region_id` is the id number of the concatenated segment. Sometimes one enriched cluster has multiple modes, so it may be cut into two or more segments. `chr`, `strand` and `pos` specify the genomic location of each base. `tag` is the tag count and `mutant` is the mutant count on each base. `sites` and `probability` are the inference results.

`clusters` is the summary of results for all clusters. `region_id` is the id number for each cluster. `chr`, `strand`, `start` and `end` specify the genomic range. `enriched` specifies whether a cluster is found to have at least one enriched bin, and `sites` specifies whether a cluster is found to have at least one significant binding site.

4.2 SNPs

If we further process the `MiClip` object with `MiClip.snp`, a data frame `snps` will be added to the `MiClip` object. It contains information of the possible SNP sites extracted from the control experiment file. Also, a column will be added to `sites` specifying whether a binding site could actually be a SNP and another column will be added to `clusters` specifying whether a cluster contains a least one binding site which could actually be a SNP.

```
> snps=test$snps
> head(snps) # Inferred possible SNP sites are contained in this data frame

       chr strand     pos tag mutant
242  chr18      + 3399931   3      3
2314 chr18      + 4383748   3      3
2695 chr18      + 4394004   4      4
3939 chr18      + 4674843   3      3
5811 chr18      + 4682610  12      7
6153 chr18      + 4948777   4      4

> head(sites[sites$SNP,]) # In this dataset, three possible SNPs are found

     region_id sub_region_id strand   chr     pos tag mutant sites probability
2            7             1      + chr18 3399931   3      3  TRUE       0.996
1021       111            28      + chr18 4682610  12      7  TRUE       1.000
5146       745           139      - chr18 6227630   6      4  TRUE       0.999
      SNP
```

```
2    TRUE
1021 TRUE
5146 TRUE
```

```
> head(clusters[clusters$SNP,])
```

```
    region_id   chr strand   start     end enriched sites  SNP
7           7 chr18      + 3399910 3399969     TRUE  TRUE TRUE
111       111 chr18      + 4682585 4682639     TRUE  TRUE TRUE
745       745 chr18      - 6227590 6227664     TRUE  TRUE TRUE
```

4.3 `MiClip.sum`

MiClip provides a summary function `MiClip.sum`. Because we are using a very small toy sample data, the results presented are not realistic.

```
> MiClip.sum(test)
```

```
For identifying enriched regions
# of clusters: 1110
# of identified enriched clusters: 170
# of bins: 10624
# of bins in each state:
FALSE  TRUE
 9334  1290
Statistics of probability
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.2590  0.9990  1.0000  0.9763  1.0000  1.0000
Average tag count of enriched bin: 8
Average tag count of not enriched bin: 2


For identifying binding sites
# of enriched clusters: 170
# of sub enriched clusters: 179
# of enriched clusters with identified binding sites: 22
# of bases: 6365
# of bases in each state:
FALSE  TRUE
 6342    23
Statistics of probability
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.5380  1.0000  1.0000  0.9991  1.0000  1.0000
Average tag count of binding site: 7
Average mutant count of binding site: 3
Average tag count of not binding site: 8
Average mutant count of not binding site: 0
```

`MiClip.sum` gives the basic statistics on the results of the two rounds of Hidden Markov Model.

5 Session Info

```
> sessionInfo()

R version 2.15.0 (2012-03-30)
Platform: x86_64-redhat-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=C                 LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats4    splines   stats     graphics  grDevices utils     datasets
[8] methods   base

other attached packages:
[1] MiClip_1.2   VGAM_0.9-0   moments_0.13

loaded via a namespace (and not attached):
[1] tools_2.15.0
```

References

[1] Macias, S., et al. (2012) DGCR8 HITS-CLIP reveals novel functions for the Microprocessor. *Nat Struct Mol Biol* 19(8): p. 760-6.

[2] Chi SW, Zang JB, Mele A, Darnell RB. (2009) Argonaute HITS-CLIP decodes microRNA-mRNA interaction maps. *Nature* 2009 Jul 23;460(7254):479-86. Epub 2009 Jun 17.

[3] Lohse M, Bolger AM, Nagel A, Fernie AR, Lunn JE, Stitt M, Usadel B. (2012) RobiNA: a user-friendly, integrated software solution for RNA-Seq-based transcriptomics. *Nucleic Acids Res* 2012 Jul;40(Web Server issue):W622-7.