MeRIP-seq/m^{6}A-seq data analyses pipeline

Summary
We generated whole-transcriptome level *in vitro* sense mRNA (IVT RNA) which has the similar gene signature with the endogenous extracted samples but without any modifications. Since MeRIP-seq/m^{6}A-seq is the most extensively used in epitranscriptome studies, we incorporated the IVT RNA library into the well-defined MeRIP-seq/m^{6}A-seq protocol and explored potential false positives if existed. This file provided a pipeline for analyzing MeRIP-seq/m^{6}A-seq datasets.

Requirements:
Unix/Linux based operating system (tested with CentOS release 6.10)
cutadapt (tested with version 2.5)
fastqc (tested with version 0.11.5)
trimmomatic (tested with version 0.36)
hisat2 (tested with version 2.1.0)
samtools (tested with version 1.6)
bedtools (tested with version 2.29.2)
macs2 (tested with version 2.1.2)
exomePeak2 (tested with version 1.3.7)

Reference genome:
GRCh37 (human), mm10 (mouse)

For home-made datasets of human HEK293T cell line and mESCs:

1. Cutadapt and quality control

rawdata cutadapt
```
cutadapt -a AGATCGGAAGAGC -A AGATCGGAAGAGC -m 15 -j 3
```
cleandata fastqc
```
fastqc -t 16 clean_data.fq
```

2. Mapping and alignment filter

```
hisat2 -p 16 –rna-strandness RF -x hisat2_ref -1 R1.fq -2 R2.fq | samtools view -@ 16 -bS -f 2 -F
256 -q 30 | samtools sort -@ 16 -o bam_file
```

3. Peak calling

3.1 Peak calling using Macs2

```
##human
macs2 callpeak -t IP_bam -c Input_bam -B --SPMR --keep-dup all -n name --outdir dir -f BAM -g
298509897 --nomodel --extsize 150 -q 0.05 --fe-cutoff 2
##mouse
macs2 callpeak -t IP_bam -c Input_bam -B --SPMR --keep-dup all -n name --outdir dir -f BAM -g
105516336 --nomodel --extsize 150 -q 0.05 --fe-cutoff 2
```

3.2 Peak calling exomePeak2

gtf = “Homo_sapiens.GRCh37.75.gtf” or “mm10.gtf”
result = exomePeak2(gff_dir=gtf, bam_ip=ip_bam, bam_input=input_bam, save_dir=Dir,paired_end=T,parallel=T,p_cutoff=0.00001,log2FC_cutoff=1,fragment_length=150)

4. Extract intersecting peaks between replicates and different methods
4.1 Intersecting peaks between replicates using Macs2
bedtools intersect -a rep1 -b rep2 -f 0.5 -F 0.5 -e > macs2_intersect.bed

4.2 Intersecting peaks between replicates using exomePeak2
## python script exomePeak2_intersect.py was used to obtain intersecting peaks.
bedtools intersect -a rep1 -b rep2 -s -split -wo > tmp_file
python exomePeak2_intersect.py tmp_file | cut -f 1,2,3,4,5,6,7,8,9,10,11,12 | sort | uniq > exomePeak2_intersect.bed

5. Peak calibrating
5.1 Peak calibrating using Macs2
false_positive peaks:
bedtools intersect -a IVT.bed -b mRNA.bed -wa -f 0.5 -F 0.5 -e | sort | uniq > false_positive.bed
calibrated m6A peaks:
bedtools intersect -a mRNA.bed -b IVT.bed -v -wa -f 0.5 -F 0.5 -e | sort | uniq > calibrated.bed
IVT unique peaks:
bedtools intersect -a IVT.bed -b mRNA.bed -v -wa -f 0.5 -F 0.5 -e | sort | uniq > IVT_uniq.bed

5.2 Peak calibrating using exomePeak2
bedtools intersect -a IVT_bed -b mRNA_bed -s -split -wo > tmp_file
false_positive peaks:
python exomePeak2_intersect.py tmp_file | cut -f 1,2,3,4,5,6,7,8,9,10,11,12 | sort | uniq > false_positive.bed
calibrated m6A peaks:
python exomePeak2_intersect.py tmp_file | cut -f 16 | awk 'BEGIN{OFS=FS="\t"}ARGIND==1{a[$0]=FNR}ARGIND==2{if(!($4 in a)){print $0}}' - mRNA.bed | sort | uniq > calibrated.bed
IVT unique peaks:
python exomePeak2_intersect.py tmp_file | cut -f 4 | awk 'BEGIN{OFS=FS="\t"}ARGIND==1{a[$0]=FNR}ARGIND==2{if(!($4 in a)){print $0}}' - IVT.bed | sort | uniq > IVT_uniq.bed

For public datasets:
Extended Table 1 provided detailed information of each public dataset.
1. Cutadapt and quality control
Paired-end:
java -jar trimmomatic-0.36.jar PE -threads 16 -phred33 -trimlog trim.logfile fq_R1 fq_R2 R1.clean.fq.gz R1.trim.fq.gz R2.clean.fq.gz R2.trim.fq.gz ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 SLIDINGWINDOW:5:15 LEADING:5 TRAILING:5 MINLEN:15

Single-end:

```
java -jar trimmomatic-0.36.jar SE -threads 16 -phred33 -trimlog trim.logfile fq clean.fq.gz ILLUMINACLIP:TruSeq3-PE.fa:2:30:10 SLIDINGWINDOW:5:15 LEADING:5 TRAILING:5 MINLEN:15
fastqc -t 16 clean_data.fq
```

2. Mapping and alignment filter

Paired-end:

```
hisat2 -p 16 -x hisat2_ref -1 R1.fq -2 R2.fq | samtools view -@ 16 -bS -f 2 -F 256 -q 30 | samtools sort -@ 16 -o bam_file
```

Single-end:

```
hisat2 -p 16 -x hisat2_ref -U clean_reads.fq | samtools view -@ 16 -bS -F 256 -q 30 | samtools sort -@ 16 -o bam_file
```

3. Peak calling

Note: data with replicates were called peaks separately

3.1 Peak calling using Macs2

```
##human
macs2 callpeak -t IP_bam -c Input_bam -B --SPMR --keep-dup all -n name --outdir dir -f BAM -g 298509897 --nomodel --extsize 100 -q 0.05 --fe-cutoff 2
##mouse
macs2 callpeak -t IP_bam -c Input_bam -B --SPMR --keep-dup all -n name --outdir dir -f BAM -g 105516336 --nomodel --extsize 100 -q 0.05 --fe-cutoff 2
```

3.2 Peak calling using exomePeak2

```
gtf = “Homo_sapiens.GRCh37.75.gtf” or “mm10.gtf”
```

Paired-end:

```
result = exomePeak2(gff_dir=gtf, bam_ip=ip_bam, bam_input=input_bam, save_dir=Dir,paired_end=T,parallel=T,p_cutoff=0.00001,log2FC_cutoff=1,fragment_length=100)
```

Single-end:

```
result = exomePeak2(gff_dir=gtf, bam_ip=ip_bam, bam_input=input_bam, save_dir=Dir,paired_end=F,parallel=T,p_cutoff=0.00001,log2FC_cutoff=1,fragment_length=100)
```

4. Extract intersecting peaks between replicates

4.1 Intersecting peaks between replicates using Macs2

Two replicates：

```
bedtools intersect -a rep1 -b rep2 -f 0.5 -F 0.5 -e > reps_intersect.bed
```

Three replicates：

```
bedtools intersect -a rep1 -b rep2 -f 0.5 -F 0.5 -e | bedtools intersect -a stdin -b rep3 -f 0.5 -F 0.5 -e > reps_intersect.bed
```

4.2 Intersecting peaks between replicates using exomePeak2

Two replicates：

```
bedtools intersect -a rep1 -b rep2 -s -split -wo > tmp_file
```

```
python exomePeak2_intersect.py tmp_file | cut -f 1,2,3,4,5,6,7,8,9,10,11,12 | sort | uniq > reps_intersect.bed
```

Three replicates：

```
bedtools intersect -a rep1 -b rep2 -s -split -wo > tmp_file
python exomePeak2_intersect.py tmp_file | cut -f 1,2,3,4,5,6,7,8,9,10,11,12 | sort | uniq > two_reps_intersect.bed
bedtools intersect -a two_reps_intersect.bed -b rep3 -s -split -wo > tmp_file2
python exomePeak2_intersect.py tmp_file2 | cut -f 1,2,3,4,5,6,7,8,9,10,11,12 | sort | uniq > three_reps_intersect.bed
```