A Protocol for m6A-seq Data Analysis

- Overview
  - Datasets
  - Contact
- M0. Check Databases
    - m6A_Atlas
    - RMDisease
    - WHISTLE
- M1. Data Preprocessing
  - Download GEO Data (SRA Toolkit)
    - Install and Config SRA Toolkit
    - Download Data from SRA
  - Quality Assessment (FastQC)
    - Install FastQC
    - Run FastQC
    - FastQC Results
  - Reads Trimming (Trim Galore)
    - Install Trim Galore
    - Adaptive Quality and Adapter Trimming
    - Outputs
      1. The text file
      2. The trimmed FastQ
  - Reads Alignment (HISAT2)
    - Install HISAT2
    - Install Samtools
    - Read Alignment
      1. Build indexes
      2. Run HISAT2 with Samtools
- M2. Differential Expression
  - Transcript Assembly and Quantification (StringTie)
    - Install StringTie
    - Run StringTie
    - Outputs
  - Differential Expression Analysis (Ballgown)
    - Differential Expression Analysis
    - Results
- M3. Site Detection - Host
  - Peak Calling (exomePeak2)
    - Installation
    - Peak Calling
  - Motif Discovery (Homer)
    - Install Homer
    - Find Motifs
  - Visualization of Reads (IGV)
    - Visualization of Aligned Reads
    - Visualization of Methylation Sites
      1. Install IGV
      2. Generate TDF
      3. Run IGV
  - RNA Annotation (RNAmod)
- M4. Site Detection - Virus
  - Peak Calling (exomePeak2)
  - Visualization of Reads (IGV)

- Visualization of Aligned Reads
- Visualization of Methylation Sites
  1. Generate TDF
  2. Run IGV

M5. Differential Methylation

- m6A-seq Data Analysis (exomePeak2)
- Distribution of m6A sites (MetaTX)
  - Visualization of the Distribution of Peaks
  - Report Isoform Probabilities
- Visualization of Reads (IGV)
- GO Enrichment Analysis (DAVID)
  - Prepare and Upload Gene Lists
  - Analyze Results
- RNA Annotation (RNAmod)

M6. Reference based analysis

- exomePeak2
  - Download and Convert Basic Site Information
  - Reference-based Analysis

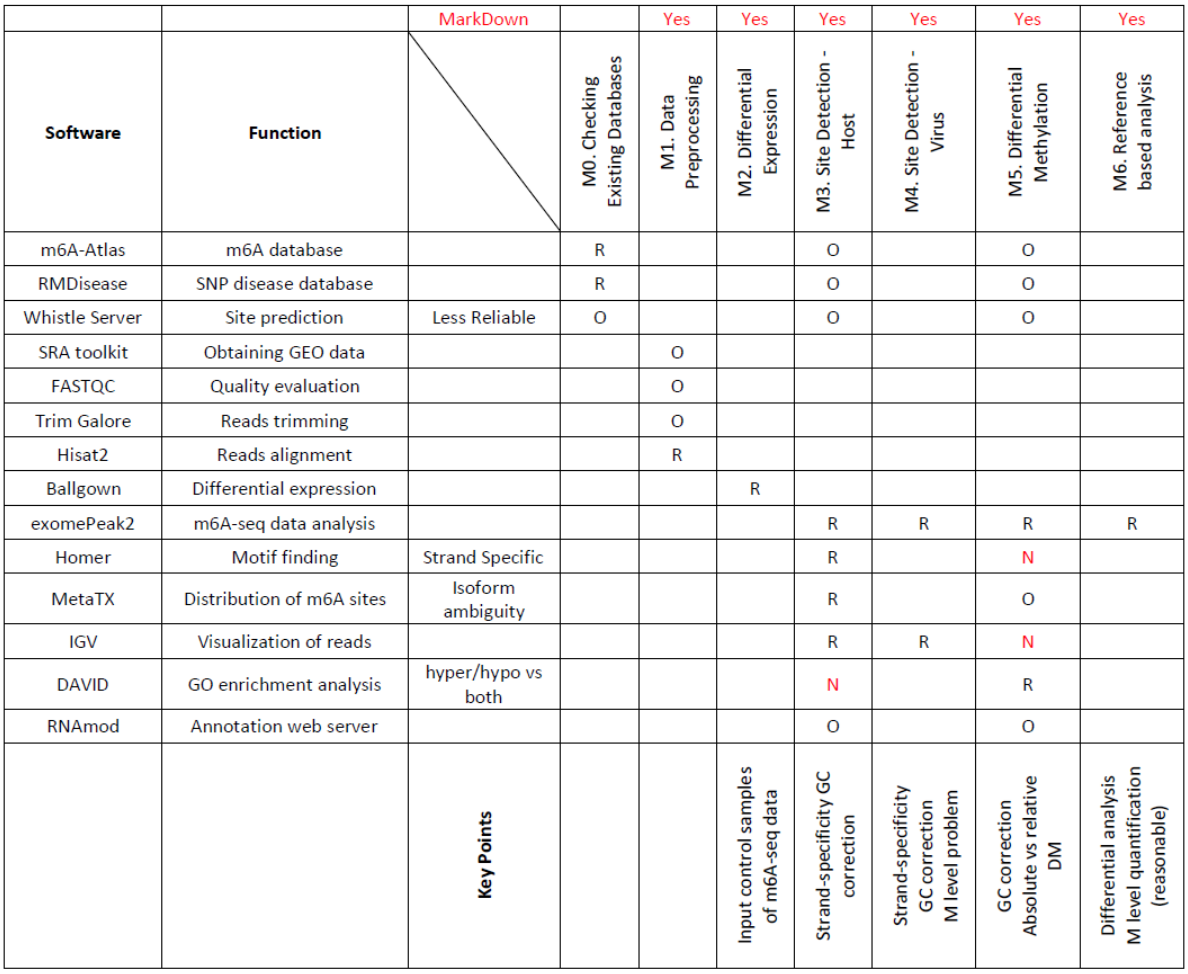
Appendix

- Construct BSgenome Object

Reference

Overview

This protocol serves as a step-by-step guide to m6A-seq data analysis. We include all the major steps involved in seven modules: **database checking, data preprocessing, differential expression analysis, site detection(host), site detection(virus),differential methylation analysis, and reference-based analysis**. We demonstrate the procedures by analyzing two published MeRIP-seq datasets from scratch based on UNIX shell and R system. The data information are summarized in the "Dataset" subsection. The software information and workflow are summarized in the following table.



		MarkDown		Yes	Yes	Yes	Yes	Yes	Yes
Software	**Function**		M0. Checking Existing Databases	M1. Data Preprocessing	M2. Differential Expression	M3. Site Detection - Host	M4. Site Detection - Virus	M5. Differential Methylation	M6. Reference based analysis
m6A-Atlas	m6A database		R			O		O	
RMDisease	SNP disease database		R			O		O	
Whistle Server	Site prediction	Less Reliable	O			O		O	
SRA toolkit	Obtaining GEO data			O					
FASTQC	Quality evaluation			O					
Trim Galore	Reads trimming			O					
Hisat2	Reads alignment			R					
Ballgown	Differential expression				R				
exomePeak2	m6A-seq data analysis					R	R	R	R
Homer	Motif finding	Strand Specific				R		N	
MetaTX	Distribution of m6A sites	Isoform ambiguity				R		O	
IGV	Visualization of reads					R	R	N	
DAVID	GO enrichment analysis	hyper/hypo vs both				N		R	
RNAmod	Annotation web server					O		O	
		Key Points			Input control samples of m6A-seq data	Strand-specificity GC correction	Strand-specificity GC correction M level problem	GC correction Absolute vs relative DM	Differential analysis M level quantification (reasonable)

Datasets

The datasets we are using for this protocol are from two studies described in Brandon et al (2018) [1] and Martin et al (2013) [2]. These datasets can be found and accessed on Gene Expression Omnibus (GEO) database.

- Expression profile of viral and cellular m6A epitranscriptomes in KSHV life cycle in Homo sapiens.
  - Accession code in GEO database: **GSE93676**
    - GSM2460344 - iSLK-uninf-input
      - SRR5978827
      - SRR5978828
      - SRR5978829
    - GSM2460345 - iSLK-uninf-ip
      - SRR5978834
      - SRR5978835
      - SRR5978836
    - GSM2460350 - iSLK-KSHV_BAC16-48hr-ip
      - SRR5978869
      - SRR5978870
      - SRR5978871
    - GSM2460351 - iSLK-KSHV_BAC16-48hr-input
      - SRR5179446
      - SRR5179447

        - SRR5179448
- Methylation profile of the Obesity-associated FTO gene in Mus musculus.
    - Accession code in GEO database: **GSE47217**
        - GSM1147014 - WT1_MeRIP
            - SRR866997
            - SRR866999
            - SRR867001
        - GSM1147015 - WT1_No IP
            - SRR866998
            - SRR867000
            - SRR867002
        - GSM1147020 - KO1_MeRIP
            - SRR866991
            - SRR866993
            - SRR866995
        - GSM1147021 - KO1_No IP
            - SRR866992
            - SRR866994
            - SRR866996

Contact

Jia Meng: Jia.Meng@xjtlu.edu.cn

Zhen Wei: Zhen.Wei01@xjtlu.edu.cn

M0. Check Databases

By checking the following databases before analyzing the datasets, we could have a basic understanding of the data in terms of estimated quantitative profiles, conservation, biological functions, annotation, and disease association of epi-transcriptome.

m6A_Atlas

m6A-Atlas is a comprehensive knowledgebase for unraveling the m6A epitranscriptome [3].

RMDisease

RMDisease is a database of genetic variants that affect RNA modifications, with implications for epitranscriptome pathogenesis [4].

WHISTLE

WHISTLE is a high-accuracy map of the human m6A epitranscriptome predicted using a machine learning approach [5].

M1. Data Preprocessing

Download GEO Data (SRA Toolkit)

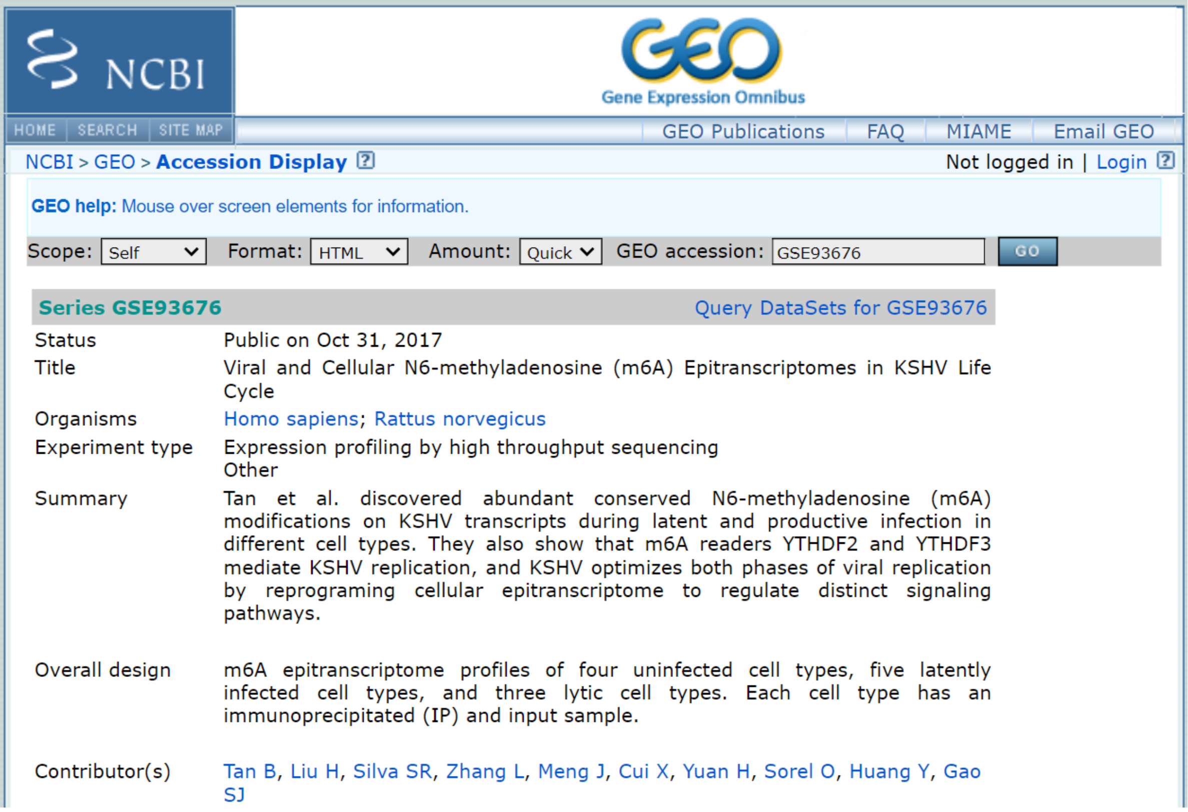
The Sequence Read Archive (SRA) is a publicly accessible archive for high throughput sequencing data. The SRA Toolkit from NCBI is a collection of tools for using data in the INSDC SRA. It takes the following steps to download data from SRA:

Install and Config SRA Toolkit

```
# Download and extract the latest version
$ wget --output-document sratoolkit.tar.gz https://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/2.10.9/sratoolkit.2.10.9-ubuntu64.tar.gz
$ tar -vxzf sratoolkit.2.10.9-ubuntu64.tar.gz

# Append the path to your PATH environment variable:
$ export PATH=$PATH:/path/to/sratoolkit.2.10.9-ubuntu64/bin

# Verify the installation
$ which fastq-dump
```

Download Data from SRA

1. Access the GEO summary page by searching "GSE93676" on GEO website.



NCBI > GEO > **Accession Display**

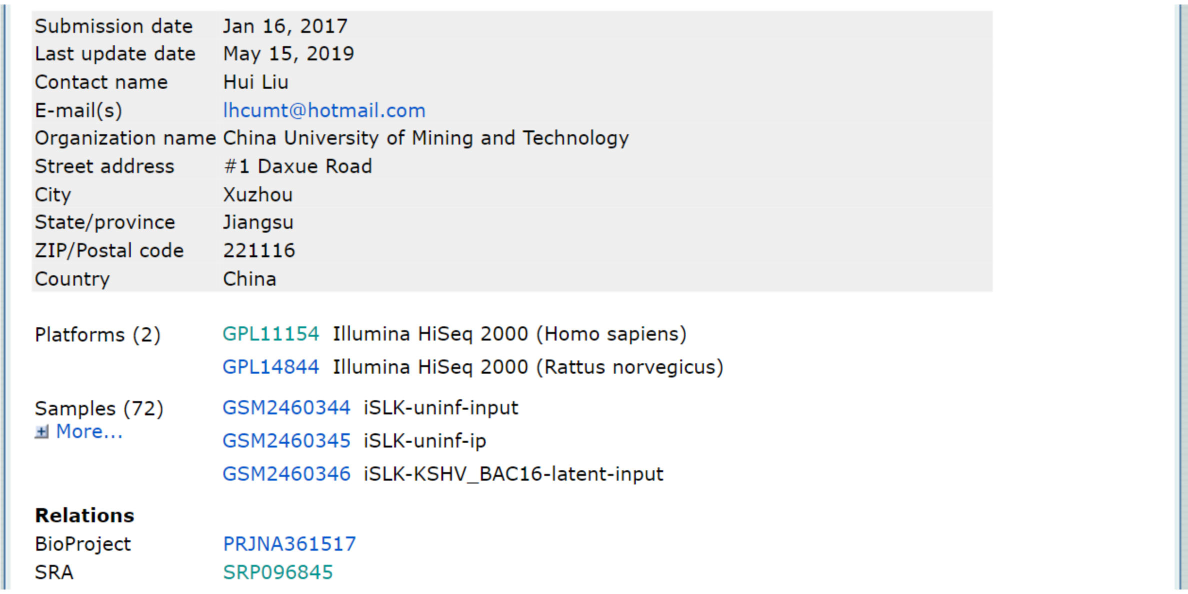
GEO help: Mouse over screen elements for information.

Scope: Self Format: HTML Amount: Quick GEO accession: GSE93676

Series GSE93676 Query DataSets for GSE93676

Status	Public on Oct 31, 2017
Title	Viral and Cellular N6-methyladenosine (m6A) Epitranscriptomes in KSHV Life Cycle
Organisms	Homo sapiens; Rattus norvegicus
Experiment type	Expression profiling by high throughput sequencing Other
Summary	Tan et al. discovered abundant conserved N6-methyladenosine (m6A) modifications on KSHV transcripts during latent and productive infection in different cell types. They also show that m6A readers YTHDF2 and YTHDF3 mediate KSHV replication, and KSHV optimizes both phases of viral replication by reprograming cellular epitranscriptome to regulate distinct signaling pathways.
Overall design	m6A epitranscriptome profiles of four uninfected cell types, five latently infected cell types, and three lytic cell types. Each cell type has an immunoprecipitated (IP) and input sample.
Contributor(s)	Tan B, Liu H, Silva SR, Zhang L, Meng J, Cui X, Yuan H, Sorel O, Huang Y, Gao SJ

2. Find a link for "SRA" under the heading "Relations".



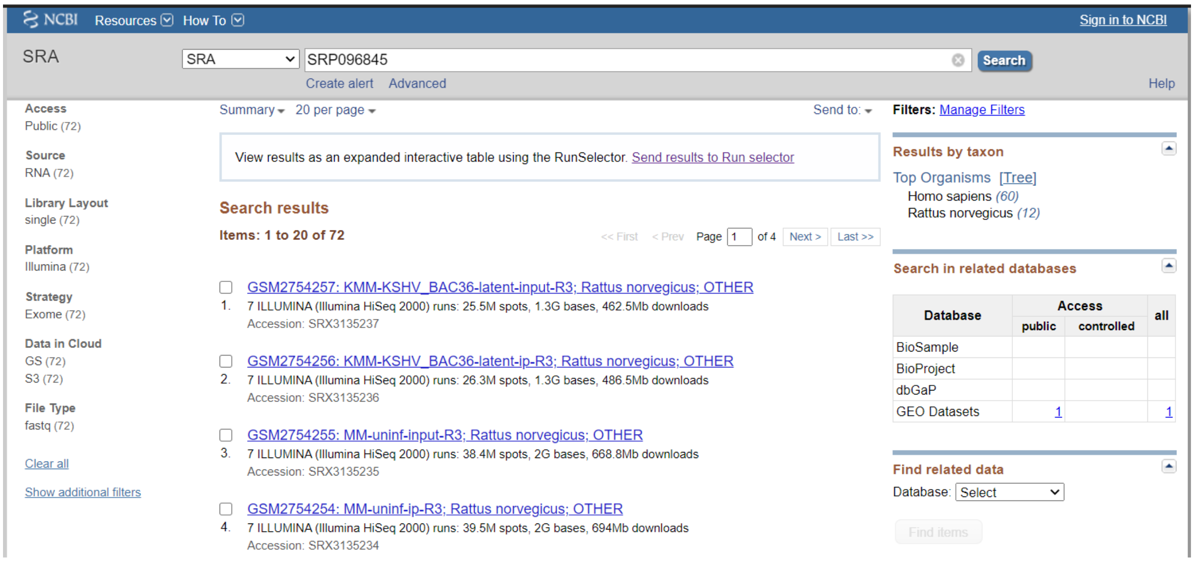
Submission date	Jan 16, 2017
Last update date	May 15, 2019
Contact name	Hui Liu
E-mail(s)	lhcumt@hotmail.com
Organization name	China University of Mining and Technology
Street address	#1 Daxue Road
City	Xuzhou
State/province	Jiangsu
ZIP/Postal code	221116
Country	China

Platforms (2)	GPL11154	Illumina HiSeq 2000 (Homo sapiens)
	GPL14844	Illumina HiSeq 2000 (Rattus norvegicus)
Samples (72) ⊞ More...	GSM2460344	iSLK-uninf-input
	GSM2460345	iSLK-uninf-ip
	GSM2460346	iSLK-KSHV_BAC16-latent-input

Relations

BioProject	PRJNA361517
SRA	SRP096845

3. Click on the link (SRP096845) which sends you to a page of all the biological samples with specific runs and files in this study.



SRA

SRA | SRP096845

Filters: Manage Filters

Access
Public (72)

Source
RNA (72)

Library Layout
single (72)

Platform
Illumina (72)

Strategy
Exome (72)

Data in Cloud
GS (72)
S3 (72)

File Type
fastq (72)

Clear all

Show additional filters

View results as an expanded interactive table using the RunSelector. Send results to Run selector

Search results

Items: 1 to 20 of 72

<< First < Prev Page 1 of 4 Next > Last >>

□ GSM2754257: KMM-KSHV_BAC36-latent-input-R3; Rattus norvegicus; OTHER
1. 7 ILLUMINA (Illumina HiSeq 2000) runs: 25.5M spots, 1.3G bases, 462.5Mb downloads
Accession: SRX3135237

□ GSM2754256: KMM-KSHV_BAC36-latent-ip-R3; Rattus norvegicus; OTHER
2. 7 ILLUMINA (Illumina HiSeq 2000) runs: 26.3M spots, 1.3G bases, 486.5Mb downloads
Accession: SRX3135236

□ GSM2754255: MM-uninf-input-R3; Rattus norvegicus; OTHER
3. 7 ILLUMINA (Illumina HiSeq 2000) runs: 38.4M spots, 2G bases, 668.8Mb downloads
Accession: SRX3135235

□ GSM2754254: MM-uninf-ip-R3; Rattus norvegicus; OTHER
4. 7 ILLUMINA (Illumina HiSeq 2000) runs: 39.5M spots, 2G bases, 694Mb downloads
Accession: SRX3135234

Results by taxon

Top Organisms [Tree]
Homo sapiens *(60)*
Rattus norvegicus *(12)*

Search in related databases

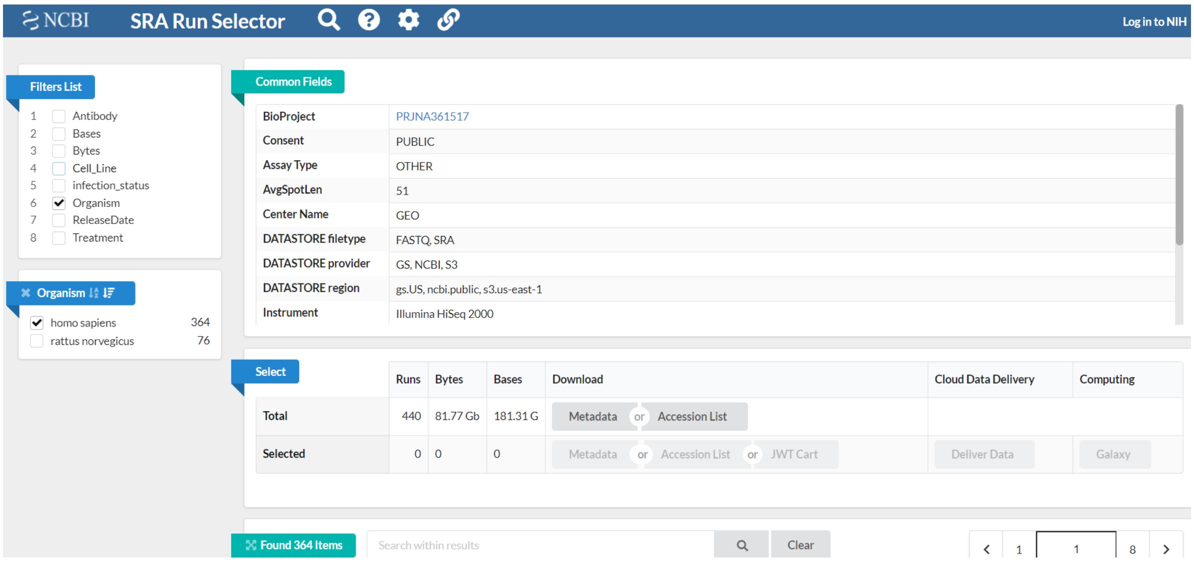
Database	Access		all
	public	controlled	
BioSample			
BioProject			
dbGaP			
GEO Datasets	1		1

Find related data

Database: Select

Find items

4. To find files of interest in one comprehensive list, navigate to the bottom of the page then click: "send to" > "Run Selector" > "go". Use "Filter List" to narrow down the choices.



Filters List

1 □ Antibody
2 □ Bases
3 □ Bytes
4 □ Cell_Line
5 □ infection_status
6 ☑ Organism
7 □ ReleaseDate
8 □ Treatment

Organism

☑	homo sapiens	364
□	rattus norvegicus	76

Common Fields

BioProject	PRJNA361517
Consent	PUBLIC
Assay Type	OTHER
AvgSpotLen	51
Center Name	GEO
DATASTORE filetype	FASTQ, SRA
DATASTORE provider	GS, NCBI, S3
DATASTORE region	gs.US, ncbi.public, s3.us-east-1
Instrument	Illumina HiSeq 2000

Select

	Runs	Bytes	Bases	Download	Cloud Data Delivery	Computing
Total	440	81.77 Gb	181.31 G	Metadata or Accession List		
Selected	0	0	0	Metadata or Accession List or JWT Cart	Deliver Data	Galaxy

Found 364 Items

Search within results

Clear

‹ 1 1 8 ›

5. Extract FastQ files from SRA-accession using SRA-Toolkit

```
#!/bin/bash
cd /path/to/raw_data/homo/
fetch_dump(){
prefetch $1
fastq-dump $1
}
export -f fetch_dump
for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
fetch_dump ${s}
done
```

```
#!/bin/bash
cd /path/to/raw_data/mm10/
fetch_dump(){
prefetch $1
fastq-dump $1
}
export -f fetch_dump
for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
fetch_dump ${s}
done
```

For paired-end data, you need to add `--split-files` option in the `fastq-dump` command.

```
fastq-dump --split-files SRR866997
```

Quality Assessment (FastQC)

It is always necessary to assess the quality of the sequence reads in FASTQ files from the sequencing facility. FastQC is a quality control application for high-throughput sequencing data. By using FastQC, we could be aware of any problems in raw sequence data before moving on to the next step.

Install FastQC

```
# FastQC requires a suitable 64-bit Java Runtime Environment (JRE) installed and
in the path. Check the version of Java:
$ java -version

# For Linux, download and extract the latest version of FastQC from [project
website]
(http://www.bioinformatics.babraham.ac.uk/projects/download.html#fastqc):
$ wget
http://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0.11.9.zip
$ unzip fastqc_v0.11.9.zip

# Append the path to your PATH environment variable:
$ export PATH=$PATH:/path/to/FastQC

# Verify installation:
$ fastqc -help
```

Run FastQC

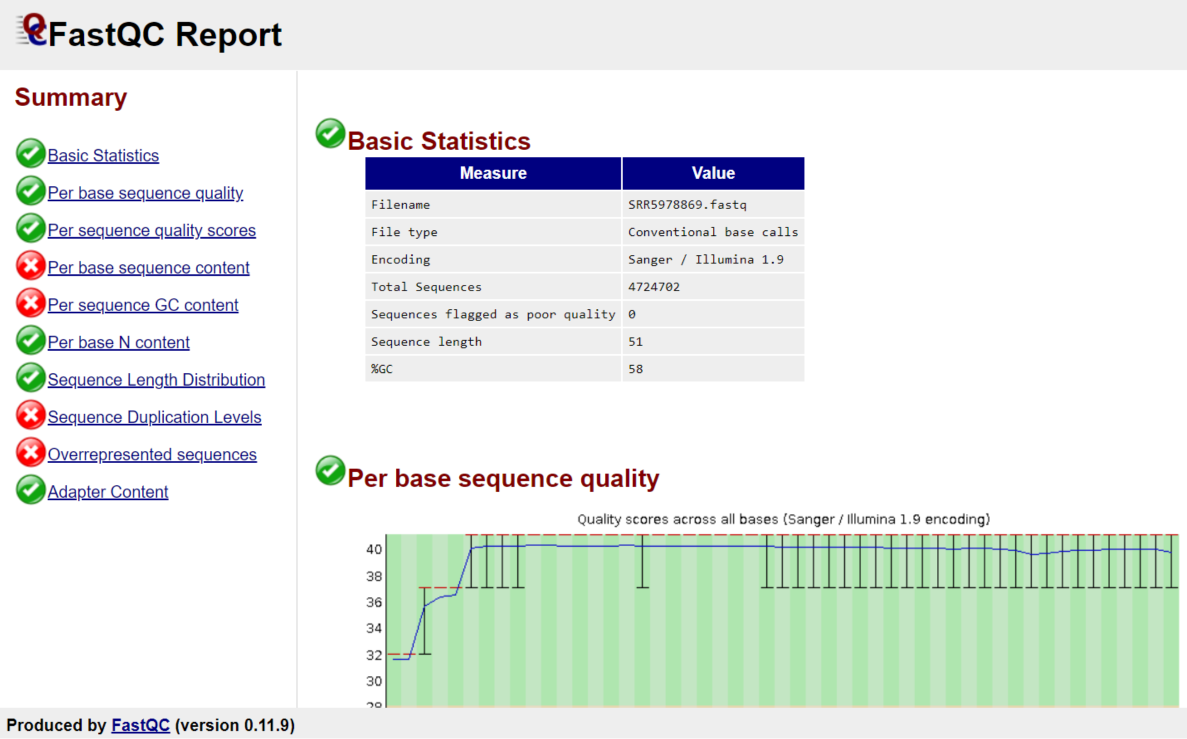
```
# Examine the quality of one FastQ file:
$ fastqc -o /path/to/fastqc_result/ /path/to/raw_data/homo/SRR5978869.fastq

# or examine the quality of multiple FastQ files:
$ fastqc -o /path/to/fastqc_result/ -t 6 /path/to/raw_data/homo/*.fastq
```

Note: `-o` (or `--outdir`) will create all output files in the specified output directory. `-t` specifies the number of files / threads that can be processed in parallel.

FastQC Results

FastQC produces two output files for each FastQ file: an HTML report ("SRR5978869_fastqc.html") and a packed file ("SRR5978869_fastqc.zip").

You could transfer the HTML file to local place by *FileZilla* (mac) or *WinSCP* (win), and open the file in browser. A screenshot of part of the HTML file is shown below.



FastQC Report

Summary

- ✓ Basic Statistics
- ✓ Per base sequence quality
- ✓ Per sequence quality scores
- ✗ Per base sequence content
- ✗ Per sequence GC content
- ✓ Per base N content
- ✓ Sequence Length Distribution
- ✗ Sequence Duplication Levels
- ✗ Overrepresented sequences
- ✓ Adapter Content

✓ Basic Statistics

Measure	Value
Filename	SRR5978869.fastq
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	4724702
Sequences flagged as poor quality	0
Sequence length	51
%GC	58

✓ Per base sequence quality

Quality scores across all bases (Sanger / Illumina 1.9 encoding)
40
38
36
34
32
30
28

Produced by FastQC (version 0.11.9)

Note that two of the most important analysis modules in FastQC are **“Per base sequence quality”** plot and the **“Overrepresented sequences”** table. The “Per base sequence quality” plot provides the distribution of quality scores across all bases at each position in the reads. The “Overrepresented sequences” table displays the sequences (at least 20 bp) that occur in more than 0.1% of the total number of sequences, which aids in identifying contamination. You could also refer to Analysis Modules in FastQC documentation for the interpretation of the HTML report.

The other output is a zip file for each sample. You could unpack the zip files and have a look at the summary.

```
# Unpack a .zip file in the result directory:
$ unzip SRR5978869_fastqc.zip

# or unpack .zip files in the result directory:
$ for zip in *.zip
do
unzip $zip
done

# To see the content of a single summary file:
$ cat SRR5978827_fastqc/summary.txt
```

```
# or cat all summary files into one text file and have a look at it:
$ cat */summary.txt > ~/all/fastqc_summaries.txt
$ cat ~/all/fastqc_summaries.txt
```

For paired-end data, since the two reads of the pair are generated separately, trying to get statistics like per base sequence quality on the combined forward and reverse reads would make no sense. In this case, you may check quality by inputting BAM files.

Reads Trimming (Trim Galore)

Trim Galore is a Perl wrapper around Cutadapt and FastQC to consistently apply adapter and quality trimming to FastQ files. We will use this tool for quality trimming, adapter trimming, and removing short sequences.

Install Trim Galore

Before installation, ensure that Cutadapt and FastQC are already installed.

```
# Check the version of cutadapt
$ cutadapt --version

# Check the version of FastQC
$ fastqc -v
```

Install the latest version of Trim Galore from Github or project website:

```
# Install Trim Galore
$ curl -fsSL https://github.com/FelixKrueger/TrimGalore/archive/0.6.6.tar.gz -o
trim_galore.tar.gz
$ tar xvzf trim_galore.tar.gz

# Verify installation
$ trim_galore -v
```

Adaptive Quality and Adapter Trimming

In this procedure, first, low-quality base calls are trimmed off from the 3' end of the reads before adapter removal. Next, adapter sequences from the 3’ end of reads are detected and removed by cutadapt. Lastly, trimmed short sequences (default: < 20bp) are filtered.

```
#!/bin/bash
trimGalore(){
trim_galore -o /path/to/trim_galore_result/ /path/to/raw_data/homo/$1.fastq
}
export -f trimGalore
for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
trimGalore ${s}
done
```

```
#!/bin/bash
trimGalore(){
trim_galore -o /path/to/trim_galore_result/ /path/to/raw_data/mm10/$1.fastq
}
export -f trimGalore
for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
trimGalore ${s}
done
```

For trimming paired-end data, you need to add a `--paired` option in the `trim_galore` command.

```
trim_galore --paired -o /path/to/trim_galore_result/ *_1.fastq *_2.fastq
```

Outputs

Trim Galore produced two output files for each FastQ file: one text file ("SRR5978869.fastq_trimming_report.txt") and a trimmed FastQ file ("SRR5978869_trimmed.fq").

1. The text file

The text file provides a summary of running parameters.

```
# To see the first few lines of the text file
$ head SRR5978869.fastq_trimming_report.txt
```

```
SUMMARISING RUN PARAMETERS
==========================
Input filename: SRR5978869.fastq
Trimming mode: single-endw
Trim Galore version: 0.6.4_dev
Cutadapt version: 3.2
Number of cores used for trimming: 1
Quality Phred score cutoff: 20
Quality encoding type selected: ASCII+33
```

2. The trimmed FastQ

The trimmed FastQ file can be used for further analysis.

Reads Alignment (HISAT2)

HISAT2 is a fast and sensitive alignment program for mapping next-generation sequencing reads to reference genome(s) [6]. We are going to use this tool to align the reads to hg19 genome, HHV8 genome, and mm10 genome, respectively.

Install HISAT2

```
# Download and extract the latest version
$ wget ftp://ftp.ccb.jhu.edu/pub/infphilo/hisat2/downloads/hisat2-2.0.4-Linux_x86_64.zip
$ unzip hisat2-2.0.4-Linux_x86_64.zip

# Append to PATH environment variable
$ export PATH=$PATH:/path/to/hisat2-2.0.4

# Verify installation
$ hisat2 --help
$ hisat2 --version
```

Install Samtools

Samtools is a suite of programs for interacting with high-throughput sequencing data [7]. SAM files produced by HISAT2 must be sorted and converted to BAM using samtools before running StringTie.

```
# Download and extract samtools
$ wget https://github.com/samtools/samtools/releases/download/1.11/samtools-1.11.tar.bz2 -O samtools-1.11.tar.bz2
$ tar -xjvf samtools-1.11.tar.bz2

# Append to PATH environment variable
$ export PATH=$PATH:/path/to/samtools-1.11

# Verify installation
$ samtools --version
```

Read Alignment

1. Build indexes

You can either download HISAT2 indexes from its website

```
$ wget https://genome-idx.s3.amazonaws.com/hisat/hg19_genome.tar.gz
$ tar -zxvf hg19_genome.tar.gz
```

or download reference sequence and gene annotation from Illumina iGenome before building index by `hisat2-build` command.

```
$ cd /path/to/homo/
$ hisat2-build -p 20 hg19_genome.fa genome

$ cd /path/to/HHV8/
$ hisat2-build -p 20 hhv8_sequence.fasta genome

$ cd /path/to/mm10/
$ hisat2-build -p 20 mm10_genome.fasta genome
```

`hisat2-build` generates eight `.ht2` files, from `genome.1.ht2` to `genome.8.ht2`, which we will use for alignment in the next step.

2. Run HISAT2 with Samtools

Getting sorted BAM and index:

```
#!/bin/bash
cd /path/to/homo/ # where storing index
hisat_samtool(){
hisat2 -x genome --summary-file “$1”.m6A.align_summary -p 5 -U
/path/to/trim_galore_result/“$1”_trimmed.fq | samtools view -Su |samtools sort -
o /path/to/homo_result/“$1”_sorted.bam
samtools index /path/to/homo_result/“$1”_sorted.bam
}
export -f hisat_samtool

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
hisat_samtool ${s}
done
mkdir alignment_summary
mv *.align_summary alignment_summary/
```

```
#!/bin/bash
cd /path/to/mm10/ # where storing index
hisat_samtool(){
hisat2 -x genome --summary-file “$1”.m6A.align_summary -p 5 -U
/path/to/trim_galore_result/“$1”_trimmed.fq | samtools view -Su |samtools sort -
o /path/to/mm10_result/“$1”_sorted.bam
samtools index /path/to/mm10_result/“$1”_sorted.bam
}
export -f hisat_samtool

for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
hisat_samtool ${s}
done
mkdir alignment_summary
mv *.align_summary alignment_summary/
```

```
#!/bin/bash
cd /path/to/hhv8/ # where storing index
hisat_samtool(){
hisat2 -x genome --summary-file “$1”.m6A.align_summary -p 5 -U
/path/to/trim_galore_result/“$1”_trimmed.fq | samtools view -Su |samtools sort -
o /path/to/hhv8_result/“$1”_sorted.bam
samtools index /path/to/hhv8_result/“$1”_sorted.bam
}
export -f hisat_samtool

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
hisat_samtool ${s}
done
mkdir alignment_summary
```

```
mv *.align_summary alignment_summary/
```

- Note that if the aligned results are going to assemble transcript with StringTie, a `--dta` option is necessary to include the tag `XS` to indicate the genomic strand that produced the RNA from which the read was sequenced. This is required by StringTie. The `hisat2-samtools` command should be modified as

  ```
  hisat2 -x genome --summary-file SRR5978827.m6A.align_summary -p 5 -U
  /path/to/trim_galore_result/SRR5978827_trimmed.fq --dta | samtools view -Su
  |samtools sort -o /path/to/homo_result/SRR5978827_sorted.bam
  ```

> `--dta/--downstream-transcriptome-assembly`
> Report alignments tailored for transcript assemblers including StringTie. With this option, HISAT2 requires longer anchor lengths for de novo discovery of splice sites. This leads to fewer alignments with short-anchors, which helps transcript assemblers improve significantly in computation and memory usage.

- Also note that for paired end data, you need to modify the `hisat2` command as follows to get SAM file

```
hisat2 -x genome --summary-file $s.m6A.align_summary -p 5 -1
/path/to/trim_galore_result/SRR5978827_trimmed_1.fq -2
/path/to/trim_galore_result/SRR5978827_trimmed_2.fq -S
/path/to/homo_result/SRR5978827.sam
```

or modify the `hisat2-samtools` combined command to directly get sorted BAM file

```
hisat2 -x genome --summary-file SRR5978827.m6A.align_summary -p 5 -1
/path/to/trim_galore_result/SRR5978827_trimmed_1.fq -2
/path/to/trim_galore_result/SRR5978827_trimmed_2.fq | samtools view -Su
|samtools sort -o /path/to/homo_result/SRR5978827_sorted.bam
```

M2. Differential Expression

Transcript Assembly and Quantification (StringTie)

StringTie is a highly efficient assembler for RNA-Seq alignments using a novel network flow algorithm [8]. It can simultaneously assemble and quantify expression levels for the features of the transcriptome in a Ballgown readable format. StringTie's output can be processed by specialized software like Ballgown (Alyssa et al. (2014)), Cuffdiff (Cole et al. (2010)) or other programs (DESeq2 (Anders & Huber (2010)), edgeR (Robinson et al. (2010)), etc).

The input SAM(BAM) file must be sorted by reference position. Every spliced read alignment in the input must contain the tag `XS` to indicate the genomic strand that produced the RNA from which the read was sequenced. These requirements are met by running HISAT2 with `--dta` option and samtools.

Install StringTie

```
# Download and extract StringTie
$ wget http://ccb.jhu.edu/software/stringtie/dl/stringtie-
2.1.4.Linux_x86_64.tar.gz
$ tar xvfz stringtie-2.1.4.Linux_x86_64.tar.gz

# Append to PATH environment variable
$ export PATH=$PATH:/path/to/stringtie-2.1.4.Linux_x86_64

# Verify installation
$ stringtie --version
```

Run StringTie

Run with the downloaded gene annotation:

```
#!/bin/bash
stringTie1(){
  stringtie /path/to/homo_result/"$1"_sorted.bam -p 20 -o
/path/to/stringtie_homo/"$1".gtf -G /path/to/hg19_annotation.gff
}
export -f stringTie1

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
stringTie1 ${s}
done
```

```
#!/bin/bash
stringTie1(){
  stringtie /path/to/mm10_result/"$1"_sorted.bam -p 20 -o
/path/to/stringtie_mm10/"$1".gtf -G /path/to/mm10_annotation.gff
}
export -f stringTie1

for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
stringTie1 ${s}
done
```

```
# Generate a non-redundant set of transcripts
$ cd /path/to/stringtie_homo/
$ stringtie --merge -G /path/to/hg19_annotation.gff -p 20 -o
homo_stringtie_merged.gtf homo_stringtie_list.txt

$ cd /path/to/stringtie_mm10/
$ stringtie --merge -G /path/to/mm10_annotation.gff -p 20 -o
mm10_stringtie_merged.gtf mm10_stringtie_list.txt
```

The text file contains all GTF files generated when assembling the read alignments.

```
SRR5978827.gtf
SRR5978828.gtf
......
```

Estimate transcript abundances and generate read coverage tables for Ballgown. Note that this is the only case where the `-G` option is not used with a reference annotation

```
#!/bin/bash
stringTie2(){
stringtie /path/to/homo_result/“$1”_sorted.bam -eB -p 20 -G
/path/to/stringtie_homo/homo_stringtie_merged.gtf -o
/path/to/stringtie_homo/“$1”.gtf
}
export -f stringTie2

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
mkdir $s
cd $s
stringTie2 ${s}
done
```

```
#!/bin/bash
stringTie2(){
stringtie /path/to/mm10_result/“$1”_sorted.bam -eB -p 20 -G
/path/to/stringtie_mm10/mm10_stringtie_merged.gtf -o
/path/to/stringtie_mm10/“$1”.gtf
}
export -f stringTie2

for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
mkdir $s
cd $s
stringTie2 ${s}
done
```

Note:

Arguments and Options	Description
-G	Use the reference annotation file (in GTF or GFF3 format) to guide the assembly process
-e	Limits the processing of read alignments to only estimate and output the assembled transcripts matching the reference transcripts given with the `-G` option (requires `-G`, recommended for `-B/-b`)
-B	enables the output of *Ballgown* input table files (*.ctab) containing coverage data for the reference transcripts given with the `-G` option
-b <path>	Same as -B option, but these files will be created in the provided directory <path> instead of the directory specified by the `-o` option
-p <int>	Specify the number of processing threads (CPUs) to use for transcript assembly. The default is 1

Outputs

1. StringTie's primary GTF output ("SRR5978869.gtf") contains details of the transcripts that StringTie assembles from RNA-Seq data.
2. Ballgown input table files ((1) e2t.ctab, (2) e_data.ctab, (3) i2t.ctab, (4) i_data.ctab, and (5) t_data.ctab) contain coverage data for all transcripts.
3. Merged GTF ("homo_stringtie_merged.gtf") is a uniform set of transcripts for all samples.

Differential Expression Analysis (Ballgown)

Ballgown is an R/Bioconductor package for flexible, isoform-level differential expression analysis of RNA-seq data [9]. Ballgown's data structures make it easy to use table-based packages like limma (Smyth (2005)), limma Voom (Law et al. (2014)), DESeq (Anders & Huber (2010)), DEXSeq (Anders et al. (2012)), or edgeR (Robinson et al. (2010)) for differential expression analysis.

Ballgown requires three pre-processing steps:

1. MeRIP-Seq reads should be aligned to a reference genome. (**HISAT2**)
2. A transcriptome should be assembled, or a reference transcriptome should be downloaded. (**StringTie**)
3. Expression for the features (transcript, exon, and intron junctions) in the transcriptome should be estimated in a Ballgown readable format. (**StringTie**)

Differential Expression Analysis

An example of the working directory:

```
stringtie_homo/
    SRR5978827/
        e2t.ctab
        e_data.ctab
        i2t.ctab
        i_data.ctab
        t_data.ctab
    SRR5978828/
```

```
        e2t.ctab
        e_data.ctab
        i2t.ctab
        i_data.ctab
        t_data.ctab
    ...
```

Create an R script `load_bg.R` for loading ballgown object:

```
library(ballgown)
bg <- ballgown(dataDir="/path/to/stringtie_homo", samplePattern='SRR',
meas='all')
save(bg, file='bg.rda')
```

Then run this script in terminal

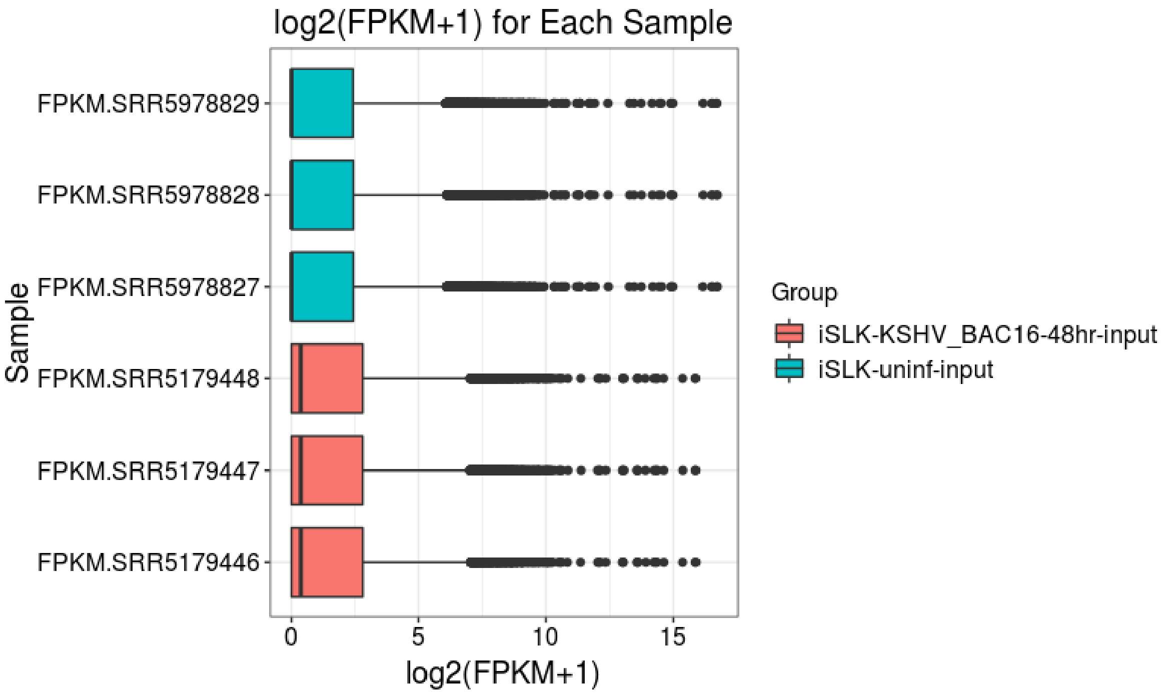
```
$ R CMD BATCH load_bg.R
```

Differential expression analysis with Ballgown:

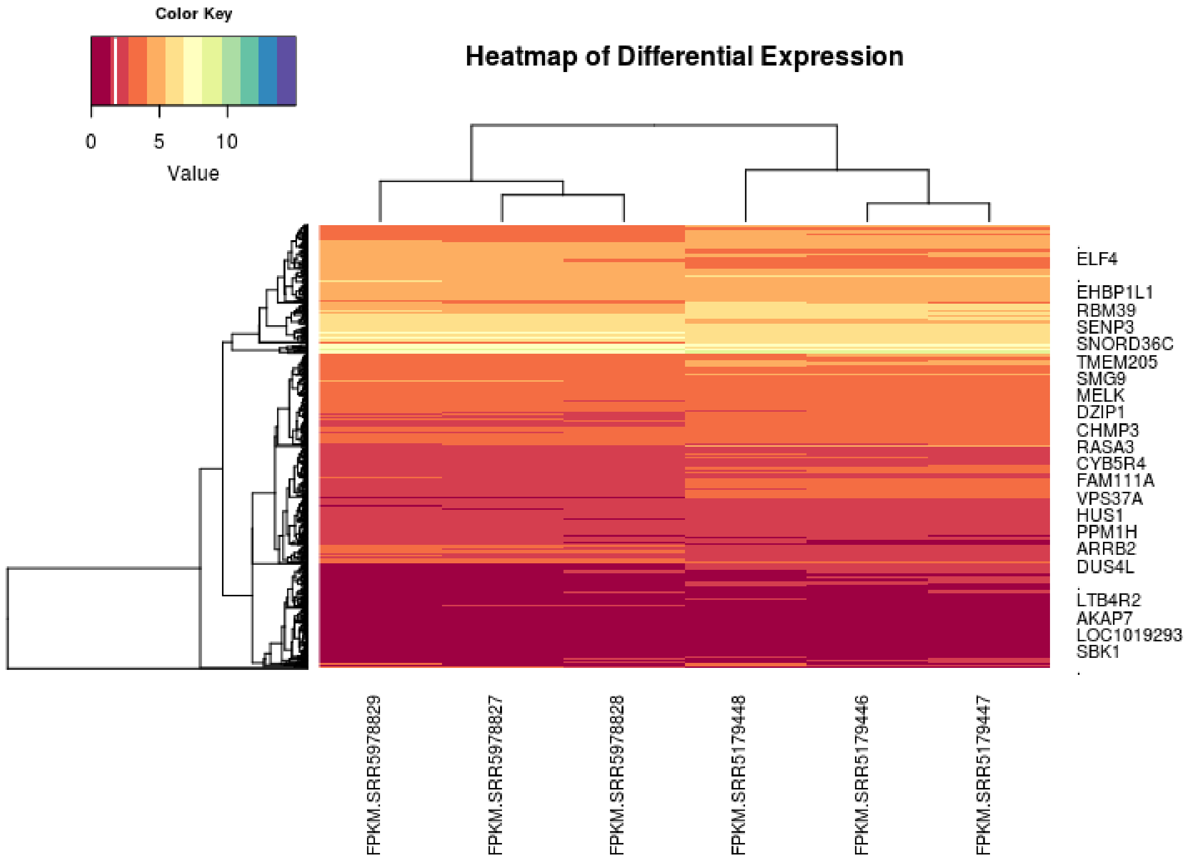
```
# Set directory
setwd("/path/to/ballgown/")

# Load R packages
library(ballgown)
library(genefilter)

# Load data
bg = ballgown(dataDir="/path/to/stringtie_homo", samplePattern='SRR',
meas='all')
# Save data for backup
save(bg, file='bg.rda')

# Load all attributes and gene names
bg_table = texpr(bg, 'all')
bg_gene_names = unique(bg_table[, 9:10])

# Get gene expression data frame
gene_expression = as.data.frame(gexpr(bg))

# Add pData specifying groups
group = c(rep("iSLK-KSHV_BAC16-48hr-input", 3), rep("iSLK-uninf-input", 3))
pData(bg) = data.frame(id=sampleNames(bg), group=group)

# Perform differential expression (DE) analysis with no filtering
results_transcripts = stattest(bg, feature="transcript", covariate="group",
getFC=TRUE, meas="FPKM")
results_genes = stattest(bg, feature="gene", covariate="group", getFC=TRUE,
meas="FPKM")
results_genes = merge(results_genes, bg_gene_names, by.x=c("id"),
by.y=c("gene_id"))

# Filter low-abundance genes.
bg_filt = subset(bg,"rowVars(texpr(bg)) > 1", genomesubset=TRUE)

# Load all attributes including gene name
```

```
bg_filt_table = texpr(bg_filt , 'all')
bg_filt_gene_names = unique(bg_filt_table[, 9:10])

# Perform DE analysis now using the filtered data
results_transcripts = stattest(bg_filt, feature="transcript", covariate="group",
getFC=TRUE, meas="FPKM")
results_genes = stattest(bg_filt, feature="gene", covariate="group", getFC=TRUE,
meas="FPKM")
results_genes = merge(results_genes, bg_filt_gene_names, by.x=c("id"),
by.y=c("gene_id"))

# Identify the significant genes with p-value < 0.05
sig_transcripts = subset(results_transcripts, results_transcripts$pval < 0.05)
sig_genes = subset(results_genes, results_genes$pval < 0.05)

# Visualization1: view the range of values and general distribution of FPKM
values
FPKM_dist = function(gene_expression) {
  log_fpkm = log2(gene_expression+1)
  fpkm_long = gather(data.frame(log_fpkm), "Sample", "log2(FPKM+1)")
  index_table = pData(bg)
  index_table$id = paste("FPKM.", index_table$id, sep='')
  colnames(index_table) = c("Sample", "Group")
  fpkm_long = left_join(fpkm_long, index_table, by = "Sample")
  p = ggplot(fpkm_long, aes(x=Sample, y=`log2(FPKM+1)`, fill = Group)) +
    geom_boxplot() + coord_flip() +
    labs(title = "log2(FPKM+1) for Each Sample") +
    theme_bw() +
    theme(plot.title = element_text( size=18, vjust=0.5, hjust=0.5),
          axis.title.x = element_text(size=15, vjust=0.5),
          axis.title.y = element_text( size=15, vjust=0.5),
          axis.text.x = element_text(size=12, colour="black"),
          axis.text.y = element_text(size=12, colour="black"),
          legend.text = element_text(size=12, colour="black"),
          legend.title = element_text(size=12, colour="black"),
          plot.caption = element_text(size=12, colour="gray75", face="italic",
hjust = 1, vjust = 1))
  return(p)
}
p1 = FPKM_dist(gene_expression)

# Visualization2: heat map of differential expression
heatmap = function(gene_expression, results_genes){
  library(RColorBrewer)
  Colors=brewer.pal(11,"Spectral")
  results_genes[,"de"] = log2(results_genes[,"fc"])
  sigpi = which(results_genes[,"pval"]<0.05)
  topn = order(abs(results_genes[sigpi,"fc"]), decreasing=TRUE)[1:25]
  topn = order(results_genes[sigpi,"qval"])[1:25]
  sigp = results_genes[sigpi,]
  sigde = which(abs(sigp[,"de"]) >= 2)
  sig_tn_de = sigp[sigde,]
  mydist=function(c) {dist(c,method="euclidian")}
  myclust=function(c) {hclust(c,method="average")}
  main_title="Heatmap of Differential Expression"
  par(cex.main=0.8)
  sig_genes_de=sig_tn_de[,"id"]
  sig_gene_names_de=sig_tn_de[,"gene_name"]
  data=log2(as.matrix(gene_expression[as.vector(sig_genes_de),])+1)
  p = heatmap.2(data,
                 hclustfun=myclust,
                 distfun=mydist,
                 na.rm = TRUE,
                 scale="none",
                 dendrogram="both",
                 margins=c(7,5),
                 Rowv=TRUE, Colv=TRUE,
                 symbreaks=FALSE, key=TRUE, symkey=FALSE,
                 density.info="none", trace="none",
                 main=main_title,
                 cexRow=0.8, cexCol=0.8,
                 labRow=sig_gene_names_de,
                 # col=rev(heat.colors(75)),
                 col=Colors)
  return(p)
}
p2 = heatmap(gene_expression, results_genes)
```

A barplot shows the range and general distribution of FPKM values:



A heat map shows the significant differential expressed genes among all samples:



Note that `pData` should hold a data frame of phenotype information for the samples in the experiment, and be added during ballgown object construction. It can also be added later.

Also note that you can remove all transcripts with a low variance across the samples before differential analysis.

Results

DE results for dataset GSE93676:

```
# To see the first few lines of significant genes and transcripts:
head(sig_transcripts)
#       feature  id           fc        pval      qval
# 4  transcript  40 1.237362e-02 0.047948681 0.7342077
# 13 transcript  68 3.643648e-13 0.041800485 0.7342077
# 35 transcript 133 2.283918e+17 0.009168513 0.7342077
# 51 transcript 167 6.634653e-10 0.034473953 0.7342077
# 70 transcript 244 2.241706e-16 0.048610761 0.7342077
# 76 transcript 276 2.637682e-18 0.002536443 0.7342077

head(sig_genes)
#              id feature           fc       pval      qval gene_name
# 1     100128071    gene 1.445238e-59 0.04327419 0.7012609   FAM229A
# 16       119710    gene 1.378979e-51 0.01122029 0.6171722  C11orf74
# 57         6289    gene 1.718444e-71 0.01229523 0.6171722      SAA2
# 74  MSTRG.10000    gene 1.942633e+58 0.01965612 0.6250552     NGLY1
# 97  MSTRG.10028    gene 5.764237e+05 0.04138674 0.6962953   PDCD6IP
# 141 MSTRG.10096    gene 1.998718e-66 0.02531592 0.6422143     PTH1R
```

DE results for dataset GSE47217:

```
head(sig_transcripts)
#        feature   id        fc        pval      qval
# 57  transcript  479 0.9339480 0.036290263 0.7220780
# 66  transcript  522 0.4784699 0.002714966 0.5205885
# 86  transcript  685 0.8718279 0.008432731 0.5794108
# 107 transcript  847 0.8360476 0.031839607 0.7176861
# 117 transcript  908 1.0784123 0.004445226 0.5428939
# 137 transcript 1051 0.4439950 0.036100181 0.7220780

head(sig_genes)
#             id feature        fc       pval      qval gene_name
# 1    MSTRG.1000    gene 1.0657124 0.04960470 0.5469039  Ppp1r15b
# 36 MSTRG.10096    gene 0.9003980 0.03641091 0.5417405     Lpin2
# 55 MSTRG.10153    gene 0.9036117 0.02278910 0.5227679     Birc6
# 56 MSTRG.10153    gene 0.9036117 0.02278910 0.5227679         .
# 85  MSTRG.1024    gene 1.2618979 0.03277910 0.5417405  Tmem183a
# 94  MSTRG.1027    gene 0.9601584 0.04558797 0.5458907   Adipor1
```

M3. Site Detection - Host

Peak Calling (exomePeak2)

exomePeak2 is an R/Bioconductor package which provides bias-aware quantification and peak detection for Methylated RNA immunoprecipitation sequencing data (MeRIP-Seq) [10]. We are going to use this package for peak calling to predict significant methylation sites.

Installation

```
if(!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("exomePeak2")
```

Peak Calling

```
library(exomePeak2)
set.seed(1)
root = "/path/to/homo_result"
setwd(root)

f1 = file.path(root, "SRR5978834_sorted.bam")
f2 = file.path(root, "SRR5978835_sorted.bam")
f3 = file.path(root, "SRR5978836_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5978827_sorted.bam")
f2 = file.path(root, "SRR5978828_sorted.bam")
f3 = file.path(root, "SRR5978829_sorted.bam")
INPUT_BAM = c(f1,f2,f3)
```

```
exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           genome = "hg19",
           library_type = "1st_strand",
           paired_end = FALSE)
```

```
library(exomePeak2)
set.seed(1)
root = "/path/to/mm10_result"
setwd(root)

f1 = file.path(root, "SRR866997_sorted.bam")
f2 = file.path(root, "SRR866999_sorted.bam")
f3 = file.path(root, "SRR867001_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR866998_sorted.bam")
f2 = file.path(root, "SRR867000_sorted.bam")
f3 = file.path(root, "SRR867002_sorted.bam")
INPUT_BAM = c(f1,f2,f3)

exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           genome = "mm10",
           paired_end = FALSE)
```

An output folder named `exomePeak2_output` will be created in the working directory containing. The most important two files "Mod.bed" and "Mod.csv" will be used in further analysis.

```
- exomePeak2_output
    - LfcGC.pdf
    - RunInfo.txt
    - Mod.bed
    - Mod.csv
    - Mod.rds
    - ADDInfo
        - ADDInfo_SizeFactors.csv
        - ADDInfo_GLM_allDesigns.csv
        - ADDInfo_ReadsCount.csv
        - ADDInfo_RPKM.csv
```

Motif Discovery (Homer)

Homer is a software for motif discovery and next-gen sequencing analysis [11]. We are going to use this tool to find enriched m6A motifs within peaks that has been found.

Install Homer

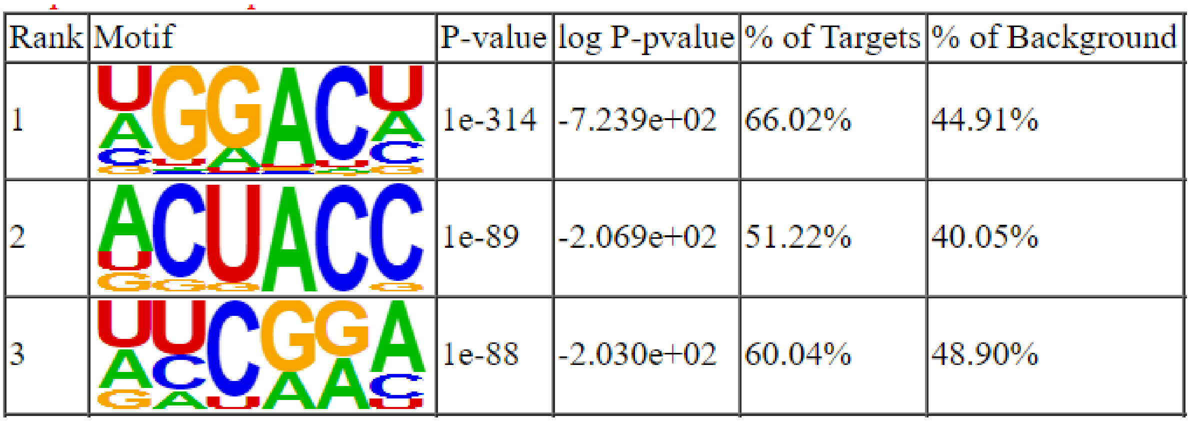
```
# Download and install following the instruction:
http://homer.ucsd.edu/homer/introduction/install.html
$ mkdir homer
$ cd homer
$ wget http://homer.ucsd.edu/homer/configureHomer.pl
$ perl configureHomer.pl -install

# Append to PATH environment variable
$ export PATH=$PATH:/path/to/homer/.//bin/

# Verify installation
$ findMotifs.pl
```

Find Motifs

```
$ findMotifsGenome.pl Mod.bed /path/to/hg19_genome.fa /path/to/MotifOutput -rna
-p 10 -len 5,6
```

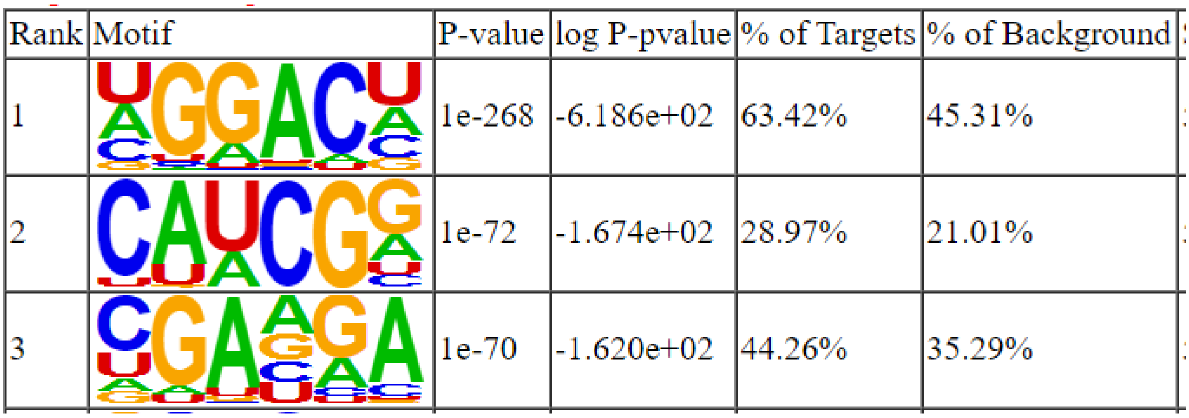
Note that Homer can analyze strand-specific genomic regions for motifs by running `findMotifsGenome.pl` with an `rna` option. Also note that filtering out the peaks longer than 1000bp can improve the reliability of motifs found by Homer.

The figure below shows the enriched motifs in peaks on hg19 transcripts.



Rank	Motif	P-value	log P-pvalue	% of Targets	% of Background
1	UGGACU	1e-314	-7.239e+02	66.02%	44.91%
2	ACUACC	1e-89	-2.069e+02	51.22%	40.05%
3	UUCGGA	1e-88	-2.030e+02	60.04%	48.90%

The figure below shows the enriched motifs in peaks on mm10 transcripts.



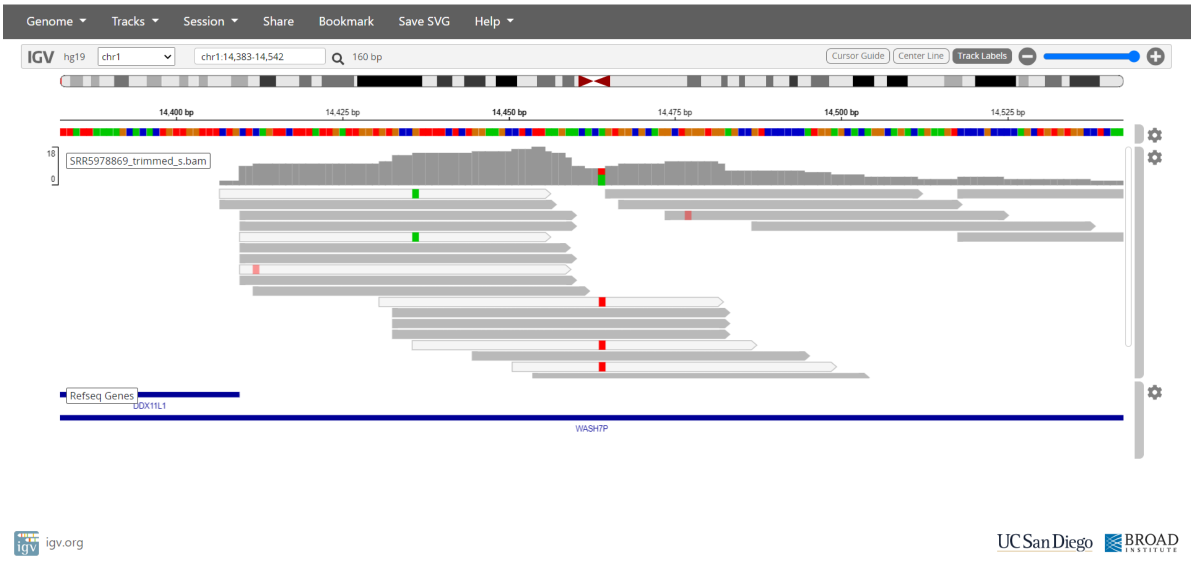
Rank	Motif	P-value	log P-pvalue	% of Targets	% of Background
1	UGGACU	1e-268	-6.186e+02	63.42%	45.31%
2	CAUCGG	1e-72	-1.674e+02	28.97%	21.01%
3	CGAAGA	1e-70	-1.620e+02	44.26%	35.29%

Visualization of Reads (IGV)

The Integrative Genomics Viewer (IGV) is a high-performance viewer that efficiently handles large heterogeneous data sets [13]. We are going to use this tool to visualize aligned reads on genome and the differential methylation sites.

Visualization of Aligned Reads

Upload BAM and its index (.bai) file to [IGV web application]() and zoom in to see the aligned reads.



Visualization of Methylation Sites

1. Install IGV

```
# Java of version 11 is required.
$ java -version

# Download and install from
http://software.broadinstitute.org/software/igv/download
$ wget https://data.broadinstitute.org/igv/projects/downloads/2.9/IGV_2.9.2.zip
$ unzip IGV_2.9.2.zip
$ cd IGV_2.9.2
$ nohup bash igv.sh

# Verify installation
$ igvtools version
```

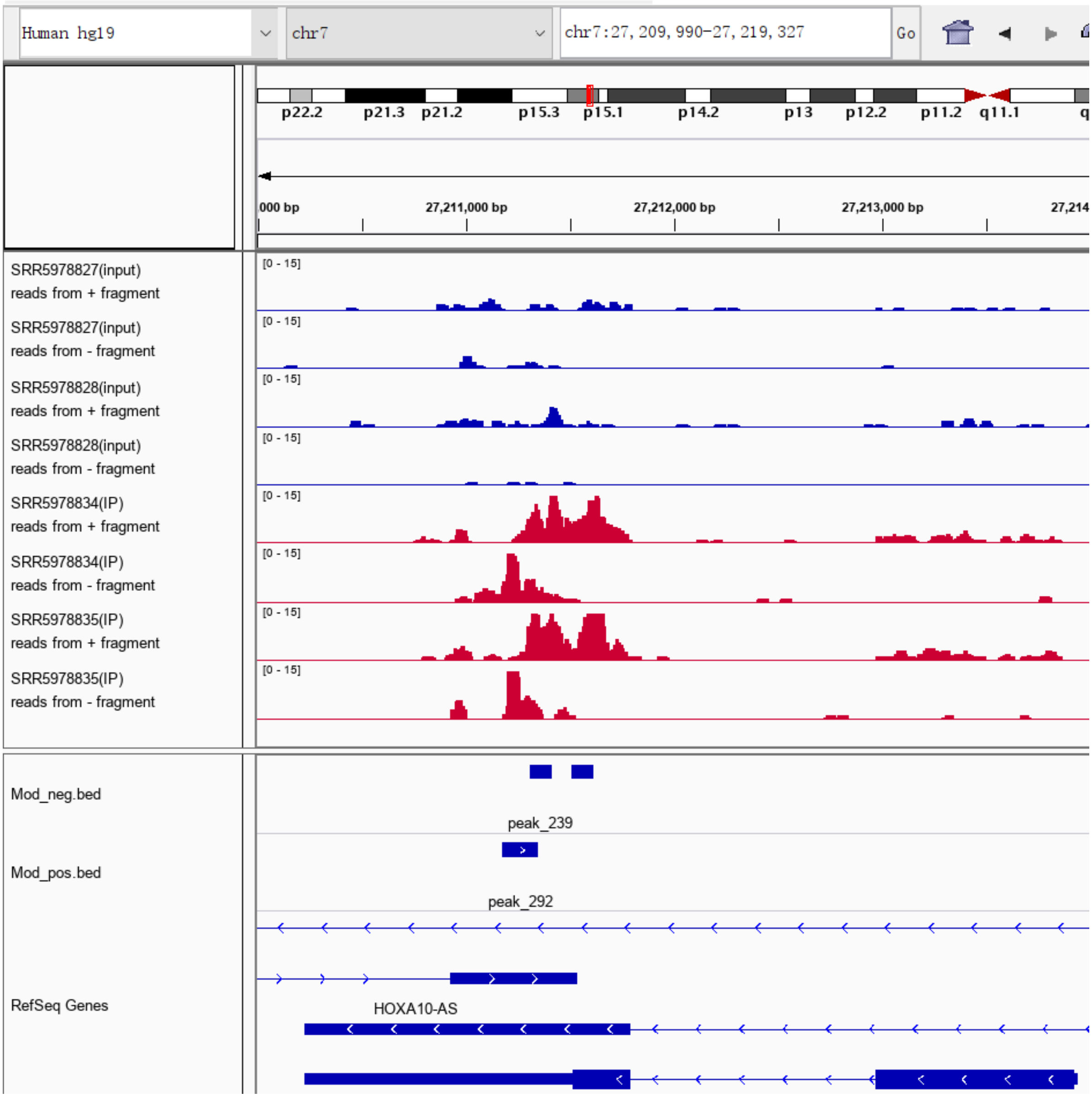
2. Generate TDF

```
#!/bin/bash
Data="/path/to/homo_result"
Output="/path/to/homo_igv"

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
igvtools count --strands read -z 5 -w 10 -e 0 $Data/${s}_sorted.bam
$Output/${s}.tdf homo_genome.fa
wait
done
```

```
#!/bin/bash
Data="/path/to/mm10_result"
Output="/path/to/mm10_igv"

for s in SRR866997 SRR866998 SRR866999 SRR867000 SRR867001 SRR867002 SRR866991
SRR866992 SRR866993 SRR866994 SRR866995 SRR866996
do
igvtools count --strands read -z 5 -w 10 -e 0 $Data/${s}_sorted.bam
$Output/${s}.tdf mm10_genome.fa
wait
done
```

3. Run IGV

The generated TDF files and BED file can then be visualized using IGV browser. As clearly shown in the figure below zooming in a region of an antisense gene , the exomePeak2 has the ability to distinguish methylation sites on specific strands.


Human hg19
chr7
chr7:27,209,990-27,219,327
Go
p22.2
p21.3
p21.2
p15.3
p15.1
p14.2
p13
p12.2
p11.2
q11.1
000 bp
27,211,000 bp
27,212,000 bp
27,213,000 bp
27,214
SRR5978827(input) reads from + fragment
SRR5978827(input) reads from - fragment
SRR5978828(input) reads from + fragment
SRR5978828(input) reads from - fragment
SRR5978834(IP) reads from + fragment
SRR5978834(IP) reads from - fragment
SRR5978835(IP) reads from + fragment
SRR5978835(IP) reads from - fragment
[0 - 15]
Mod_neg.bed
peak_239
Mod_pos.bed
peak_292
RefSeq Genes
HOXA10-AS

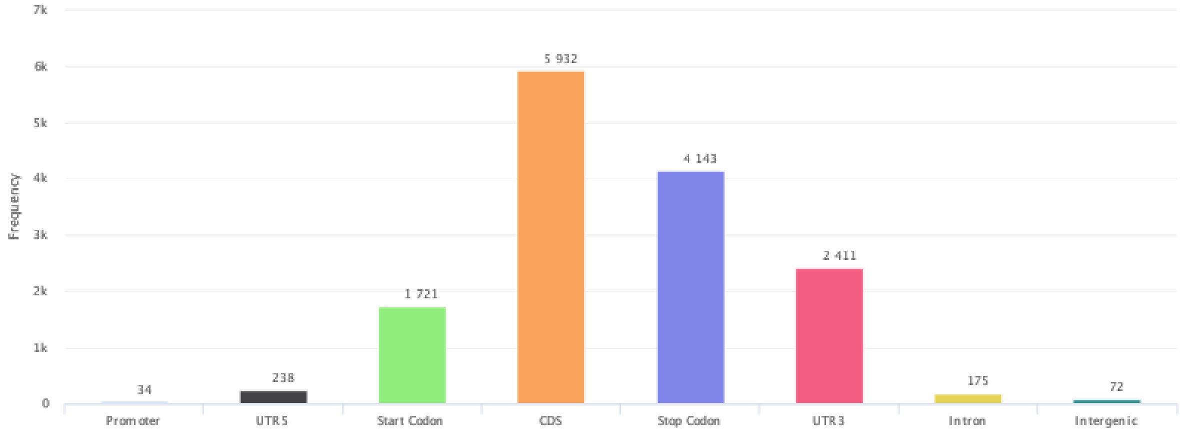
RNA Annotation (RNAmod)

RNAmod is a convenient web-based platform for the meta-analysis and functional annotation of modifications on mRNAs [14].

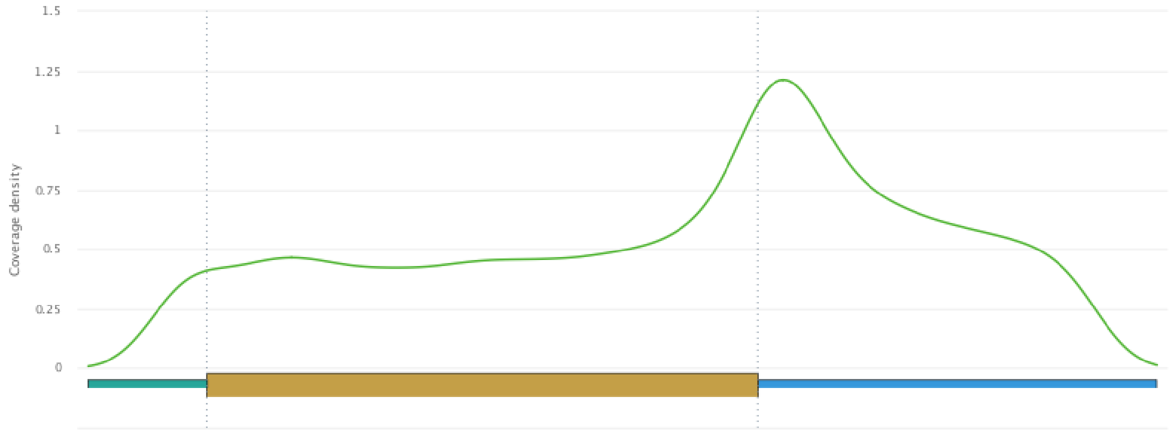
To obtain modification site information, click on "Single case" in the Home page, upload the BED file generated from exomePeak2 package, and submit the job. Then you will receive a job ID and will use it to query the job status and get the results in the result page. You can either view the results on browser or download the zip file to the local place.

Some of the figures about modification site information are displayed below:

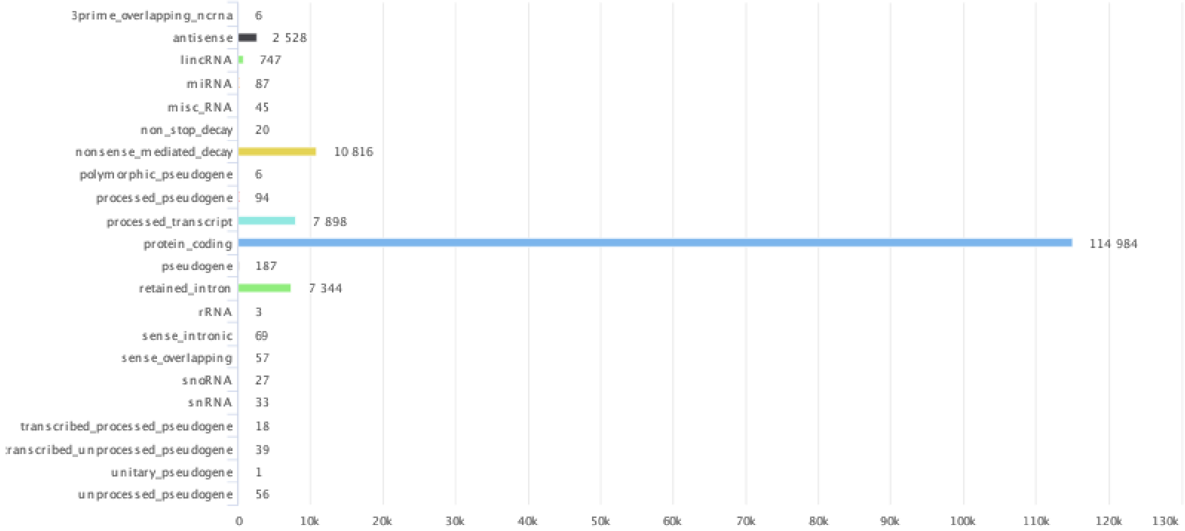
- Peaks gene features distribution:



- mRNA metagene plot



- Gene types distribution



M4. Site Detection - Virus

Peak Calling (exomePeak2)

We are going to use exomePeak2 for peak calling to find enriched m6A sites on HHV8 transcripts. The BED file as output could be used for further analysis.

```
library(exomePeak2)
set.seed(1)
root = "/path/to/hhv8_result"
setwd(root)
```

```
f1 = file.path(root, "SRR5978834_sorted.bam")
f2 = file.path(root, "SRR5978835_sorted.bam")
f3 = file.path(root, "SRR5978836_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5978827_sorted.bam")
f2 = file.path(root, "SRR5978828_sorted.bam")
f3 = file.path(root, "SRR5978829_sorted.bam")
INPUT_BAM = c(f1,f2,f3)

GENE_ANNO_GTF = file.path("/path/to/sequence.gff3")

exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           gff_dir = GENE_ANNO_GTF,
           library_type = "1st_strand",
           paired_end = FALSE)
```

An output folder named `exomePeak2_output` will be created in the working directory containing:

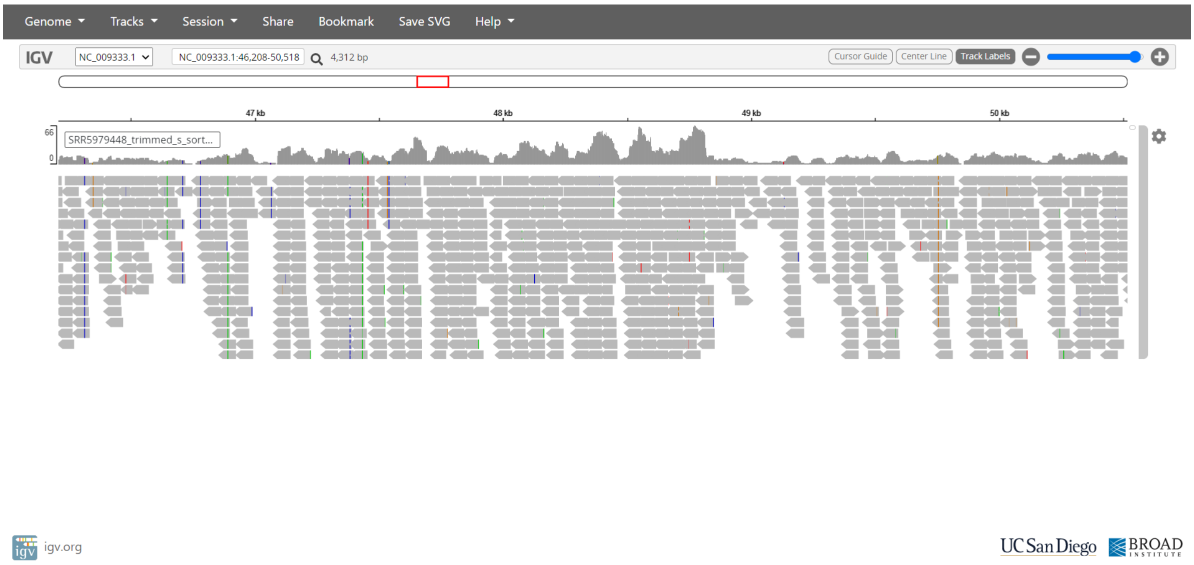
```
- exomePeak2_output
    - LfcGC.pdf
    - RunInfo.txt
    - Mod.bed
    - Mod.csv
    - Mod.rds
    - ADDInfo
        - ADDInfo_SizeFactors.csv
        - ADDInfo_GLM_allDesigns.csv
        - ADDInfo_ReadsCount.csv
        - ADDInfo_RPKM.csv
```

Visualization of Reads (IGV)

Visualization of Aligned Reads

Upload the following files to IGV web application.

- HHV8 virus genome (.fa) and its index (.fai) to "Genome".
- BAM and its index (.bai) file to "Track".



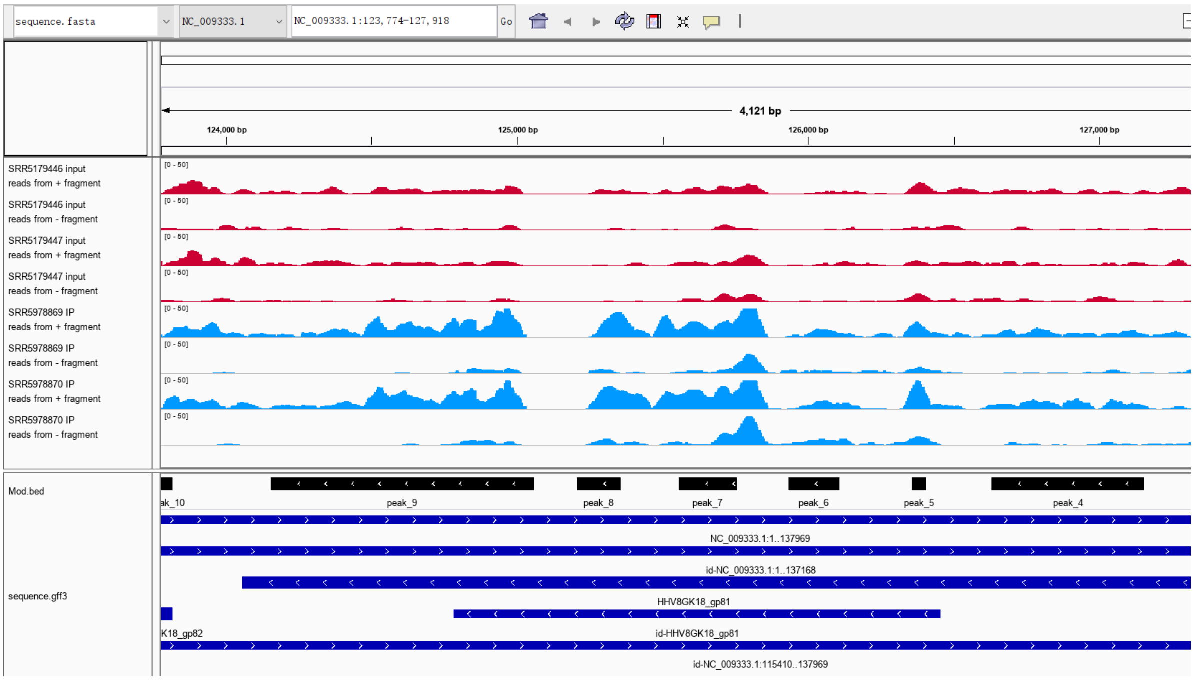
Visualization of Methylation Sites

1. Generate TDF

```
#!/bin/bash
Data="/path/to/hhv8_result"
Output="/path/to/hhv8_igv"

for s in SRR5978827 SRR5978828 SRR5978829 SRR5978834 SRR5978835 SRR5978836
SRR5978869 SRR5978870 SRR5978871 SRR5179446 SRR5179447 SRR5179448
do
igvtools count -z 5 -w 10 -e 0 $Data/${s}_sorted.bam $Output/${s}.tdf
hhv8_genome.fa
wait
done
```

2. Run IGV

The generated TDF files and BED file can then be visualized using IGV browser.


sequence.fasta
NC_009333.1
NC_009333.1:123,774-127,918
Go
4,121 bp
124,000 bp
125,000 bp
126,000 bp
127,000 bp
SRR5179446 input reads from + fragment
SRR5179446 input reads from - fragment
SRR5179447 input reads from + fragment
SRR5179447 input reads from - fragment
SRR5978869 IP reads from + fragment
SRR5978869 IP reads from - fragment
SRR5978870 IP reads from + fragment
SRR5978870 IP reads from - fragment
[0 - 50]
Mod.bed
ak_10
peak_9
peak_8
peak_7
peak_6
peak_5
peak_4
sequence.gff3
NC_009333.1:1..137969
id-NC_009333.1:1..137168
HHV8GK18_gp81
K18_gp82
id-HHV8GK18_gp81
id-NC_009333.1:115410..137969

M5. Differential Methylation

m6A-seq Data Analysis (exomePeak2)

Here we are going to use exomePeak2 for peak calling and differential methylation analysis. The BED file as output will be used for further analysis.

```
library(exomePeak2)
set.seed(1)
root = "/path/to/homo_result"
setwd(root)

f1 = file.path(root, "SRR5978834_sorted.bam")
f2 = file.path(root, "SRR5978835_sorted.bam")
f3 = file.path(root, "SRR5978836_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5978827_sorted.bam")
f2 = file.path(root, "SRR5978828_sorted.bam")
f3 = file.path(root, "SRR5978829_sorted.bam")
INPUT_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5179446_sorted.bam")
f2 = file.path(root, "SRR5179447_sorted.bam")
f3 = file.path(root, "SRR5179448_sorted.bam")
TREATED_INPUT_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5978869_sorted.bam")
f2 = file.path(root, "SRR5978870_sorted.bam")
f3 = file.path(root, "SRR5978871_sorted.bam")
TREATED_IP_BAM = c(f1,f2,f3)

exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           bam_treated_input = TREATED_INPUT_BAM,
           bam_treated_ip = TREATED_IP_BAM,
           genome = "hg19",
           library_type = "1st_strand",
           paired_end = FALSE)
```

```
library(exomePeak2)
set.seed(1)
root = "/path/to/mm10_result"
setwd(root)

f1 = file.path(root, "SRR866997_sorted.bam")
f2 = file.path(root, "SRR866999_sorted.bam")
f3 = file.path(root, "SRR867001_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR866998_sorted.bam")
f2 = file.path(root, "SRR867000_sorted.bam")
f3 = file.path(root, "SRR867002_sorted.bam")
INPUT_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR866992_sorted.bam")
f2 = file.path(root, "SRR866994_sorted.bam")
f3 = file.path(root, "SRR866996_sorted.bam")
TREATED_INPUT_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR866991_sorted.bam")
f2 = file.path(root, "SRR866993_sorted.bam")
f3 = file.path(root, "SRR866995_sorted.bam")
TREATED_IP_BAM = c(f1,f2,f3)

exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           bam_treated_input = TREATED_INPUT_BAM,
           bam_treated_ip = TREATED_IP_BAM,
           genome = "mm10",
           library_type = "1st_strand",
           paired_end = FALSE)
```

An output folder named `exomePeak2_output` will be created in the working directory containing:

```
- exomePeak2_output
    - LfcGC.pdf
    - RunInfo.txt
    - DiffMod.bed
    - DiffMod.csv
    - DiffMod.rds
    - ADDInfo
        - ADDInfo_SizeFactors.csv
        - ADDInfo_GLM_allDesigns.csv
        - ADDInfo_ReadsCount.csv
        - ADDInfo_RPKM.csv
```

Distribution of m6A sites (MetaTX)

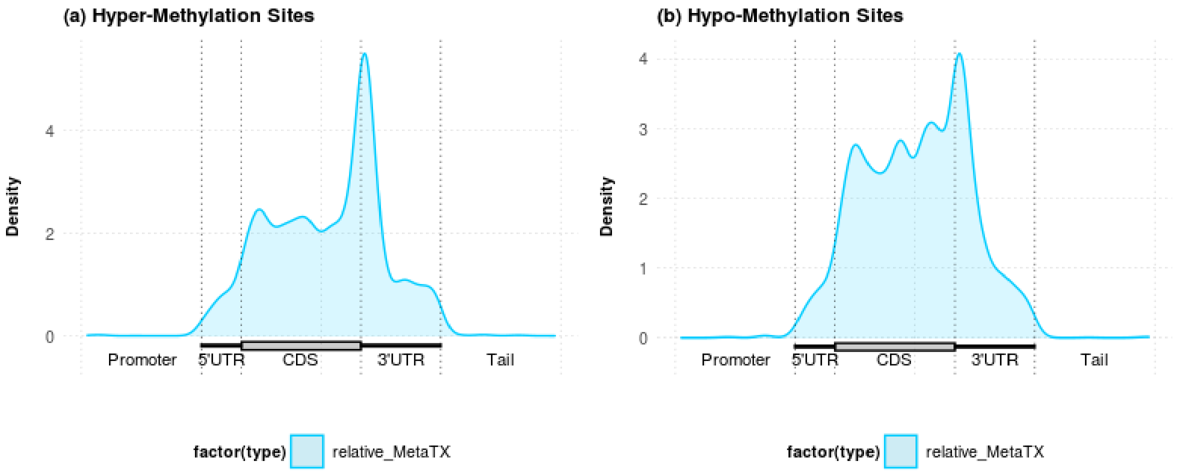
We are going to use MetaTX to visualize the distribution of methylation sites.

Visualization of the Distribution of Peaks

The following code produces separate figures of the distribution of hyper-methylation sites and hypo-methylation sites.

```r
# Load libraries
library(MetaTX)
library(rtracklayer)
library(readr)
library(bedr)
library(genomation)
library(GenomicRanges)

# Import BED file from exomePeak2
file = "/path/to/exomePeak2_output_peakcalling_1strand/Mod.bed"
file1 = "/path/to/exomePeak2_output_peakcalling_1strand/Mod.csv"
data_csv = read_csv(file1)
gr_obj =  import(file)

# Separate by hyper and hypo methylation sites
data1 = gr_obj[which(data_csv$DiffModLog2FC > 0),]
data2 = gr_obj[which(data_csv$DiffModLog2FC < 0),]

# convert to bed files
df1 = data.frame(seqnames=seqnames(data1),
                  starts=start(data1),
                  ends=end(data1),
                  names=elementMetadata(data1)$name,
                  scores=elementMetadata(data1)$score,
                  strands=strand(data1))
df2 = data.frame(seqnames=seqnames(data2),
                  starts=start(data2),
                  ends=end(data2),
                  names=elementMetadata(data2)$name,
                  scores=elementMetadata(data2)$score,
                  strands=strand(data2))
write.table(df1, file="Mod_metaTX_diff_pos.bed", quote=F, sep="\t", row.names=F, col.names=F)
write.table(df2, file="Mod_metaTX_diff_neg.bed", quote=F, sep="\t", row.names=F, col.names=F)

# Import separated bed files
file_pos = "Mod_metaTX_diff_pos.bed"
file_neg = "Mod_metaTX_diff_neg.bed"
gr_obj_pos = import(file_pos)
gr_obj_neg = import(file_neg)
gr_obj_pos = resize(gr_obj_pos, width = 1, fix = "center")
gr_obj_neg = resize(gr_obj_neg, width = 1, fix = "center")

# Download information about mRNA components
txdb = TxDb.Hsapiens.UCSC.hg19.knownGene
cds_by_tx0_1 = cdsBy(txdb, "tx")
fiveUTR_tx0_1 = fiveUTRsByTranscript(txdb,use.names=FALSE)
```

```
threeUTR_tx0_1 = threeUTRsByTranscript(txdb,use.names=FALSE)

# Map peaks to the RNA model
remap_results_m6A_1 = remapCoord(features = gr_obj_pos, txdb = txdb, num_bin =
10, includeNeighborDNA = TRUE, cds_by_tx0 = cds_by_tx0_1, fiveUTR_tx0 =
fiveUTR_tx0_1,
                                  threeUTR_tx0 = threeUTR_tx0_1)
# Plot 1
p1 =  metaTXplot(remap_results_m6A_1,
                  num_bin              = 10,
                  includeNeighborDNA   = TRUE,
                  relativeProportion   = c(1, 3, 2, 3),
                  title  = '(a) Hyper-Methylation Sites',
                  legend = 'relative',
                  type = 'relative'
)

# Map peaks to the RNA model
remap_results_m6A_2 = remapCoord(features = gr_obj_neg, txdb = txdb, num_bin =
10, includeNeighborDNA = TRUE, cds_by_tx0 = cds_by_tx0_1, fiveUTR_tx0 =
fiveUTR_tx0_1,
                                  threeUTR_tx0 = threeUTR_tx0_1)
# Plot 2
p2 =  metaTXplot(remap_results_m6A_2,
                  num_bin              = 10,
                  includeNeighborDNA   = TRUE,
                  relativeProportion   = c(1, 3, 2, 3),
                  title  = '(b) Hypo-Methylation Sites',
                  legend = 'relative',
                  type = 'relative'
)

# Plot 1 + 2
ggdraw() +
  draw_plot(p1, 0, 0, .5, 1) +
  draw_plot(p2, .5, 0, .5, 1)
```



Report Isoform Probabilities

MetaTX also provides a function for returning the probabilities of a particular feature being located on different isoforms.

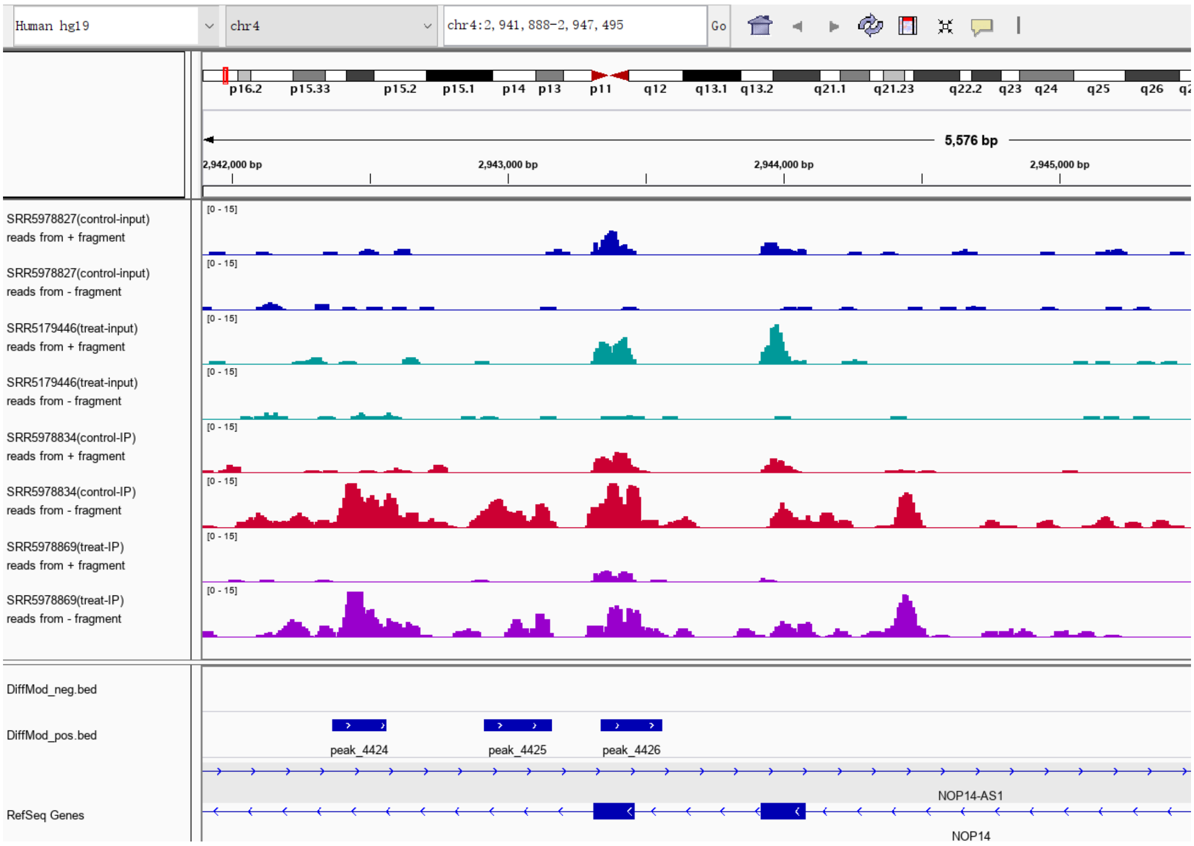
```
isoform_probs <- isoformProb(remap_results_m6A_1, num_bin = 10,
includeNeighborDNA = TRUE, lambda = 2)
write.csv(isoform_probs, "isoform_probs.csv")
```

Here are the first few rows of the outputs.

index_trans (double)	index_methyl (double)	seqnames (character)	methyl_pos (double)	strand (character)	trans_ID (double)	isoform_prob (double)
1	1	chr19	58867266	-	70456	0.00000000
2	1	chr19	58867266	-	70457	0.51380127
3	1	chr19	58867266	-	70458	0.48619873
4	2	chr20	43269048	-	72132	0.00000000
5	3	chr18	25532067	-	65378	0.43630632
6	3	chr18	25532067	-	65379	0.56369368
7	4	chr3	101396111	+	14200	1.00000000
8	5	chrX	119387453	+	76492	0.45666710
9	5	chrX	119387453	+	76493	0.47472665

Visualization of Reads (IGV)

Here we use the IGV tool to visualize reads and peaks comparing control and experimental groups. This task can be simply done by uploading the generated TDF files and BED file onto the IGV browser.


Human hg19
chr4
chr4:2,941,888-2,947,495
Go
p16.2
p15.33
p15.2
p15.1
p14
p13
p11
q12
q13.1
q13.2
q21.1
q21.23
q22.2
q23
q24
q25
q26
5,576 bp
2,942,000 bp
2,943,000 bp
2,944,000 bp
2,945,000 bp
SRR5978827(control-input) reads from + fragment
SRR5978827(control-input) reads from - fragment
SRR5179446(treat-input) reads from + fragment
SRR5179446(treat-input) reads from - fragment
SRR5978834(control-IP) reads from + fragment
SRR5978834(control-IP) reads from - fragment
SRR5978869(treat-IP) reads from + fragment
SRR5978869(treat-IP) reads from - fragment
[0 - 15]
DiffMod_neg.bed
DiffMod_pos.bed
peak_4424
peak_4425
peak_4426
NOP14-AS1
RefSeq Genes
NOP14

GO Enrichment Analysis (DAVID)

The **D**atabase for **A**nnotation, **V**isualization and **I**ntegrated **D**iscovery (**DAVID**) is a website providing a comprehensive set of functional annotation tools for investigators to understand biological meaning behind large list of genes [15]. We are going to use this tool to discover enriched gene functions.

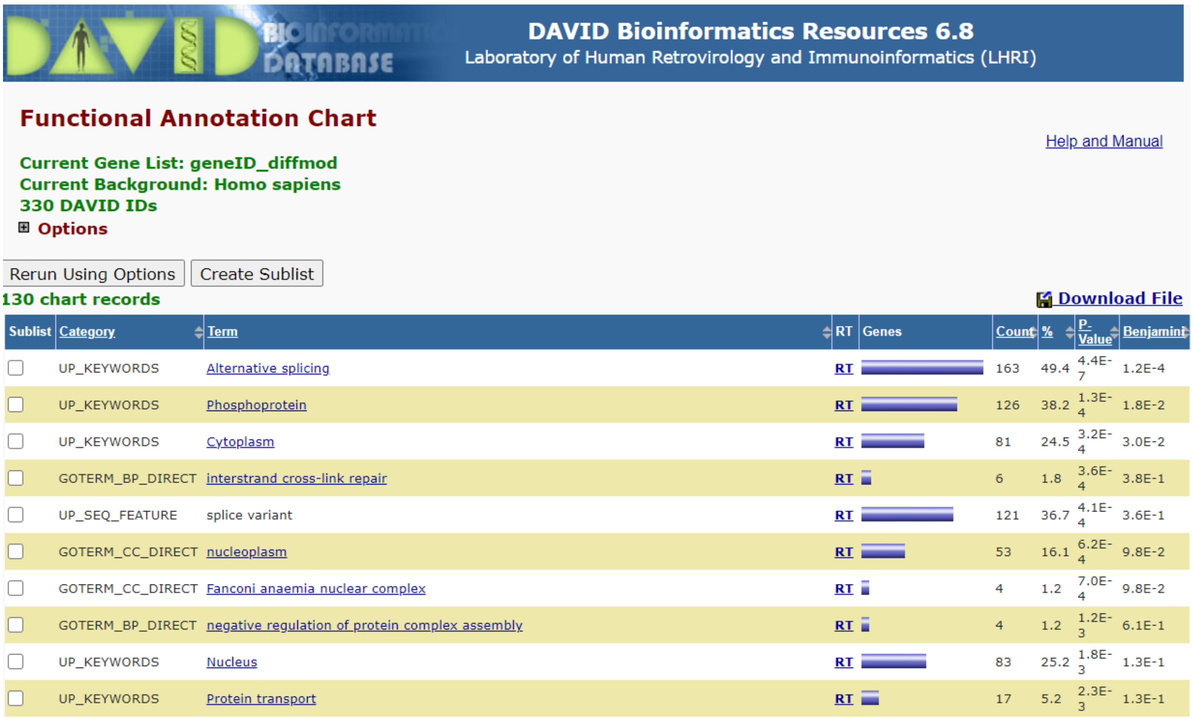
Prepare and Upload Gene Lists

From the output file ("DiffMod.csv") generated from exomePeak2, we remove duplicated values in "geneID" column and copy all the unique IDs to a txt file ("geneID_diff.txt").

Upload txt files to DAVID website with Identifier as "ENTREZ_GENE_ID", species as "Homo sapiens", and "Gene List" selected. Submit all lists and wait for results.

Analyze Results

Open "Functional Annotation Chart" and click on "Download File" to download the txt file containing results.



DAVID Bioinformatics Resources 6.8
Laboratory of Human Retrovirology and Immunoinformatics (LHRI)

Functional Annotation Chart

Help and Manual

Current Gene List: geneID_diffmod
Current Background: Homo sapiens
330 DAVID IDs
⊞ Options

Rerun Using Options | Create Sublist

130 chart records

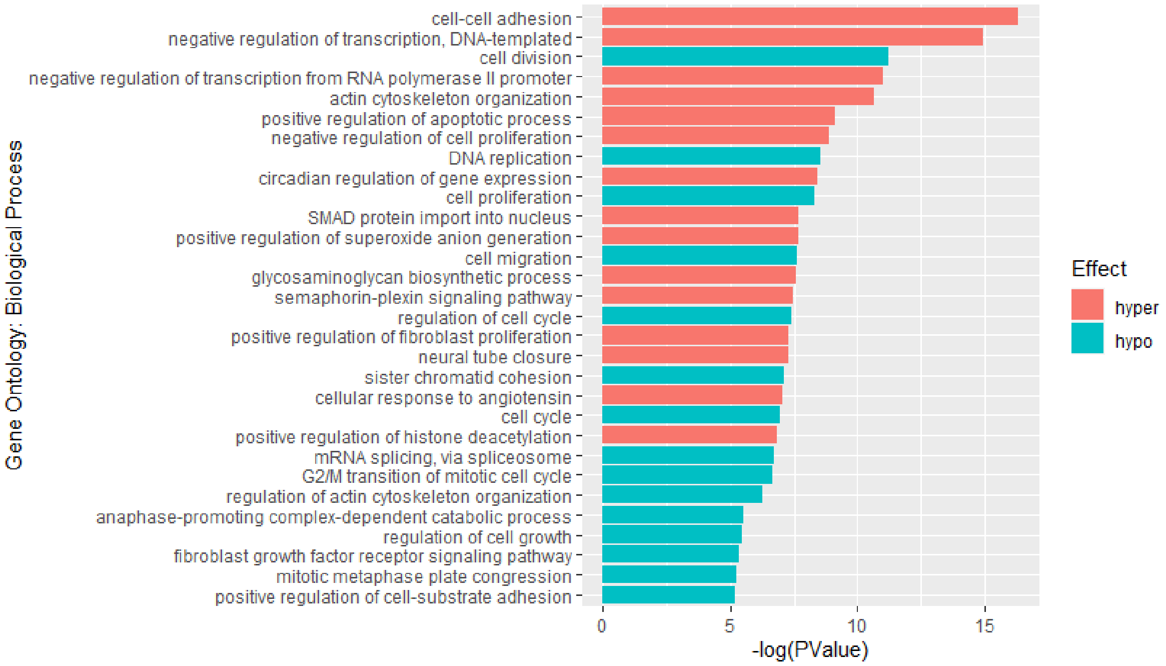
Download File

Sublist	Category	Term	RT	Genes	Count	%	P-Value	Benjamini
□	UP_KEYWORDS	Alternative splicing	RT		163	49.4	4.4E-7	1.2E-4
□	UP_KEYWORDS	Phosphoprotein	RT		126	38.2	1.3E-4	1.8E-2
□	UP_KEYWORDS	Cytoplasm	RT		81	24.5	3.2E-4	3.0E-2
□	GOTERM_BP_DIRECT	interstrand cross-link repair	RT		6	1.8	3.6E-4	3.8E-1
□	UP_SEQ_FEATURE	splice variant	RT		121	36.7	4.1E-4	3.6E-1
□	GOTERM_CC_DIRECT	nucleoplasm	RT		53	16.1	6.2E-4	9.8E-2
□	GOTERM_CC_DIRECT	Fanconi anaemia nuclear complex	RT		4	1.2	7.0E-4	9.8E-2
□	GOTERM_BP_DIRECT	negative regulation of protein complex assembly	RT		4	1.2	1.2E-3	6.1E-1
□	UP_KEYWORDS	Nucleus	RT		83	25.2	1.8E-3	1.3E-1
□	UP_KEYWORDS	Protein transport	RT		17	5.2	2.3E-3	1.3E-1

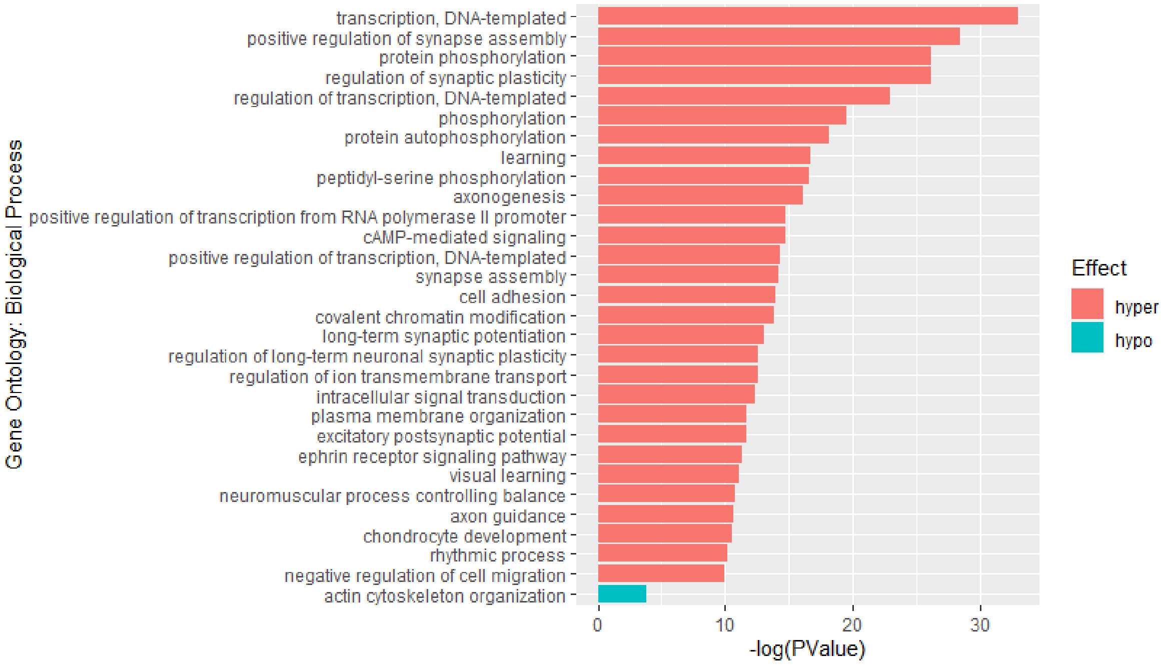
Import txt file into R, analyze results and display in figures.

```
library(readr)
library(dplyr)
chart = read_tsv("chart_diff.txt")

# generate a figure for differential expressed genes
generateFigure = function(chart, num, term = "GOTERM_BP_DIRECT"){

  p = selectPvalue(chart)
  frame = as.data.frame(chart %>%
    filter(Category == term) %>%
    select(c("Term", "%", "PValue")) %>%
    rename("Ratio"=`%`) %>%
    mutate(Term = as.factor(gsub("^.*?~", "",Term)),
           Ratio = Ratio / 100))[1:30,] # make sure no less than 30 terms in
total

  fig2 = frame %>%
    ggplot(aes(x=reorder(Term, -PValue),y=-log(PValue),fill = Ratio)) +
    geom_bar(stat="identity") +
    coord_flip() +
    xlab("Gene Ontology: Biological Process")

  return(fig2)
}

generateFigure(chart, 30, term)


# generate a figure for hyper/hypo genes
chartp = read_tsv("pos_david_mm.txt")
chartn = read_tsv("neg_david_mm.txt")
chartp = chartp %>% mutate(Effect = "hyper")
chartn = chartn %>% mutate(Effect = "hypo")
charts = rbind(chartp, chartn)

generateFigure_hyper_hypo = function(charts, term = "GOTERM_BP_DIRECT"){

  print(charts %>%
          filter(Category == term & Effect == 'hyper') %>%
          select(c("Term", "%", "PValue", "Effect")) %>%
          rename("Ratio"=`%`) %>%
          mutate(Term = as.factor(gsub("^.*?~", "",Term)),
                 Ratio = Ratio / 100) %>% nrow())

  frame_p = as.data.frame(charts %>%
    filter(Category == term & Effect == 'hyper') %>%
    select(c("Term", "%", "PValue", "Effect")) %>%
    rename("Ratio"=`%`) %>%
    mutate(Term = as.factor(gsub("^.*?~", "",Term)),
           Ratio = Ratio / 100))[1:29,]

  print(charts %>%
          filter(Category == term & Effect == 'hypo') %>%
          select(c("Term", "%", "PValue", "Effect")) %>%
          rename("Ratio"=`%`) %>%
          mutate(Term = as.factor(gsub("^.*?~", "",Term)),
                 Ratio = Ratio / 100) %>% nrow())

  frame_n = as.data.frame(charts %>%
     filter(Category == term & Effect == 'hypo') %>%
     select(c("Term", "%", "PValue", "Effect")) %>%
     rename("Ratio"=`%`) %>%
     mutate(Term = as.factor(gsub("^.*?~", "",Term)),
            Ratio = Ratio / 100))[1,]

  frame = rbind(frame_p,frame_n)

  fig2 = frame %>%
    ggplot(aes(x=reorder(Term, -PValue),y=-log(PValue),fill = Effect)) +
    geom_bar(stat="identity") +
    coord_flip() +
    xlab("Gene Ontology: Biological Process")

  return(fig2)
}

generateFigure_hyper_hypo(charts)
```

Enriched biological processes regulated by differential methylated genes on hg19 genome.



Enriched biological processes regulated by differential methylated genes on mm10 genome.

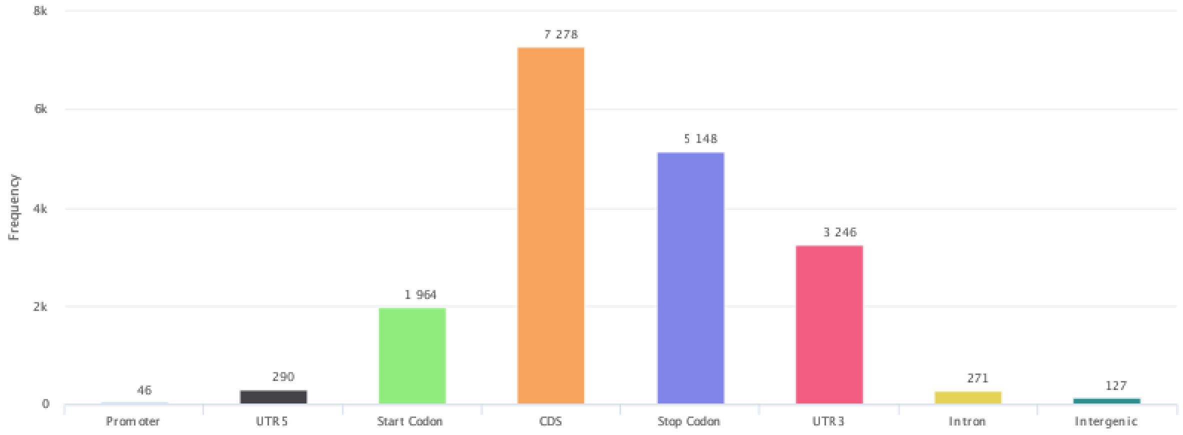


RNA Annotation (RNAmod)

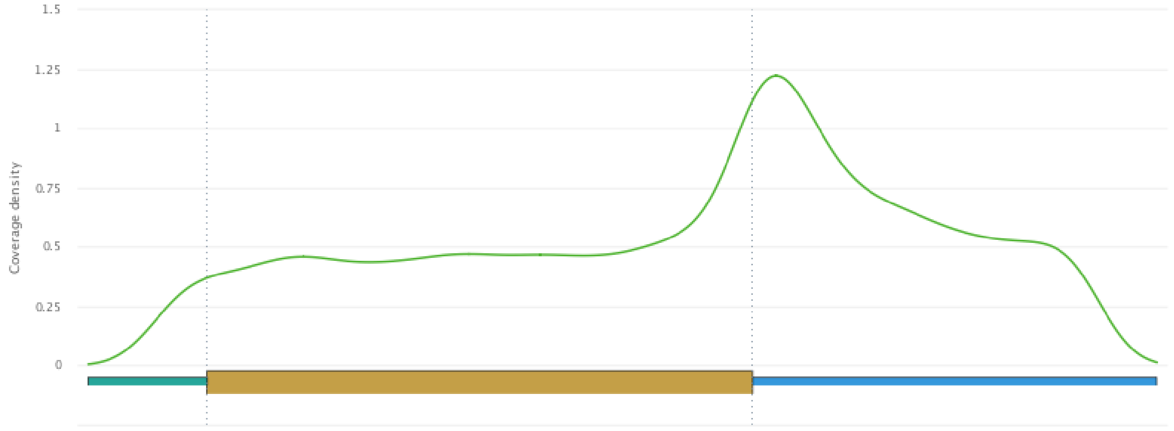
To obtain modification site information, click on "Single case" in the RNAmod Home page, upload the BED file generated from exomePeak2 package, and submit the job. Then you will receive a job ID and will use it to query the job status and get the results in the result page. You can either view the results on browser or download the zip file to the local place.

Some of the figures about modification site information are displayed below:

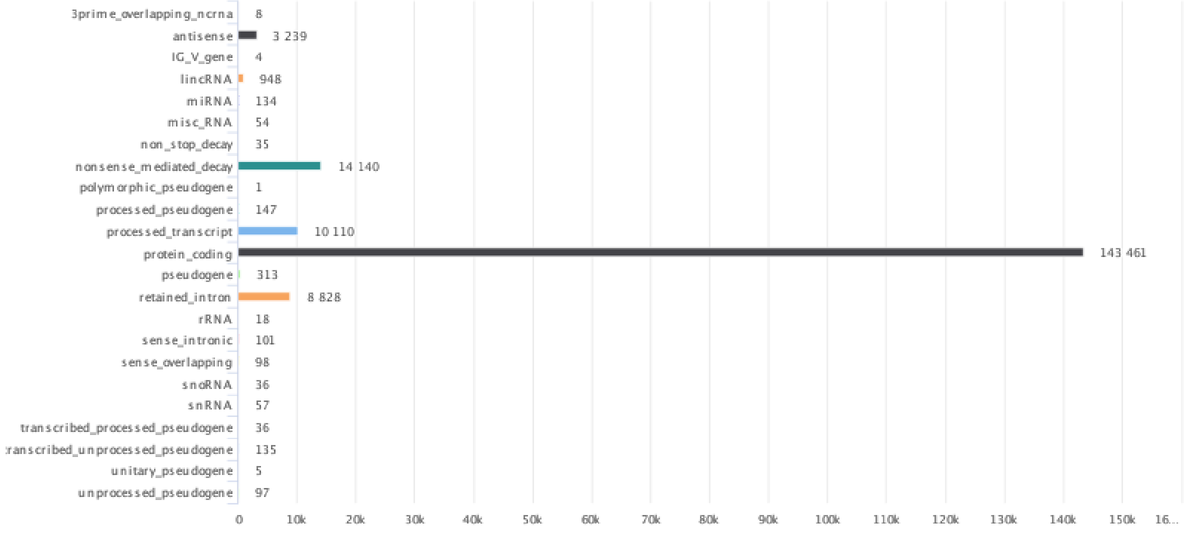
- Peaks gene features distribution:



- mRNA metagene plot



- Gene types distribution



M6. Reference based analysis

exomePeak2

Here we use exomePeak2 to conduct reference based quantification and differential analysis.

Download and Convert Basic Site Information

Download single base RNA modification annotation of m6A on human genome from m6A-Atlas database, which should be tabular data in a txt file. Convert tabular data to genomic ranges by:

```
library(readr)
library(GenomicRanges)
# Downloaded from m6A-Atlas database
my.file="/path/to/home/tangyujiao/big/download/m6A_H.sapiens_basical_information
.txt"

# Load txt file
m6A_basic_info <- read_table2(my.file, col_names = FALSE)
m6A_basic_info <- m6A_basic_info[, c(1:11)]
colNames<- c("ID","chr","start","end", "num1","strand", "LOC", "ENS","RNA",
"Gene", "seq")
colnames(m6A_basic_info) <- colNames

# Concert to Grange object
mod_annot <- makeGRangesFromDataFrame(m6A_basic_info,
                         keep.extra.columns=FALSE,
                         ignore.strand=FALSE,
                         seqinfo=NULL,
                         seqnames.field="chr",
                         start.field="start",
                         end.field="end",
                         strand.field="strand",
                         starts.in.df.are.0based=FALSE)

# Save Grange to rds
saveRDS(mod_annot, "/path/to/mod_annot.rds")
```

Reference-based Analysis

```
library(exomePeak2)
set.seed(1)
root = "/path/to/homo_result"
setwd(root)

f1 = file.path(root, "SRR5978834_sorted.bam")
f2 = file.path(root, "SRR5978835_sorted.bam")
f3 = file.path(root, "SRR5978836_sorted.bam")
IP_BAM = c(f1,f2,f3)

f1 = file.path(root, "SRR5978827_sorted.bam")
f2 = file.path(root, "SRR5978828_sorted.bam")
f3 = file.path(root, "SRR5978829_sorted.bam")
INPUT_BAM = c(f1,f2,f3)

f2 = "/path/to/mod_annot.rds"
MOD_ANNO_GRANGE <- readRDS(f2)
exomePeak2(bam_ip = IP_BAM,
           bam_input = INPUT_BAM,
           genome = "hg19",
           library_type = "1st_strand",
           paired_end = FALSE,
             mod_annot = MOD_ANNO_GRANGE)
```

An output folder named `exomePeak2_output` will be created in the working directory containing:

```
- exomePeak2_output
    - LfcGC.pdf
    - RunInfo.txt
    - Mod.bed
    - Mod.csv
    - Mod.rds
    - ADDInfo
        - ADDInfo_SizeFactors.csv
        - ADDInfo_GLM_allDesigns.csv
        - ADDInfo_ReadsCount.csv
        - ADDInfo_RPKM.csv
```

Appendix

Construct BSgenome Object

exomePeak2 requires BSgenome object of matched species. However, when there is no available BSgenome object, we have to manually construct one. Therefore, this page provides instructions for constructing BSgenome object from scratch by using *Zea mays* (Corn) genome as an example.

1. Download Genome FASTA

   Download *Zea mays* (Corn) reference genome "AGPv4" from Illumina iGenome, and find genome fasta file.

   ```
   $ wget http://igenomes.illumina.com.s3-website-us-east-
   1.amazonaws.com/Zea_mays/Ensembl/AGPv4/Zea_mays_Ensembl_AGPv4.tar.gz
   $ tar -xvzf Zea_mays_Ensembl_AGPv4.tar.gz
   ```

2. Convert fasta to 2bit file format

   Download "faToTwoBit" tool by

   ```
   $ rsync -aP \
      rsync://hgdownload.soe.ucsc.edu/genome/admin/exe/linux.x86_64/faToTwoBit
   ./
   ```

   Convert file format

   ```
   $ mkdir
   /home/zhen.di/corn/Zea_mays/Ensembl/AGPv4/Sequence/WholeGenomeFasta/preparat
   ion/
   $ cd
   /home/zhen.di/corn/Zea_mays/Ensembl/AGPv4/Sequence/WholeGenomeFasta/preparat
   ion/
   $ cp ../genome.fa zmAGPv4.fa
   $ faToTwoBit zmAGPv4.fa zmAGPv4.2bit
   ```

3. Extract chromosome names

```
$ less -S zmAGPv4.fa | grep ">" |awk '{print $1}' | sed 's/^>//g' >
zmAGPv4.chromName.txt
```

Find chromosome name by

```
$ cat zmAGPv4.chromName.txt
```

4. Prepare seed file "BSgenome.Zmays.Ensemble.zmAGPv4-seed" in the same directory

```
Package: BSgenome.Zmays.Ensemble.zmAGPv4
Title: Genome sequences for Zea mays (Ensemble AGPv4)
Description: A BSgenome package containing the full genome sequences for Zea
mays (Maize) as provided by Ensemble (B73 AGPv4, Sept. 2020) and stored in
Biostrings objects.
Version: 1.0
organism: Zea mays
common_name: Maize
provider: Ensemble
provider_version: zmAGPv4
release_date: Sept. 2020
release_name: Maize Genome Sequencing B73 RefGen_v4.0
#source_url: ftp://ftp.ensemblgenomes.org/pub/plants/release-
49/fasta/zea_mays/dna/
circ_seqs: c("Mt","Pt")
organism_biocview: Zea_mays
BSgenomeObjname: Zmays
seqs_srcdir:
/home/zhen.di/corn/Zea_mays/Ensembl/AGPv4/Sequence/WholeGenomeFasta/preparat
ion/
seqfiles_suffix: .fa
seqnames: c("10", "1", "2", "3", "4", "5", "6", "7", "8", "9", "MT", "Pt")
seqfile_name: zmAGPv4.2bit
```

Please make sure the information included in the seed file is correct by referring to BSgenomeForge doc and description.

5. Construct BSgenome data package

```
# Construct BSgenome data package in R
library(rtracklayer)
library(BSgenome)
setwd("/home/zhen.di/corn/Zea_mays/Ensembl/AGPv4/Sequence/WholeGenomeFasta/p
reparation")
forgeBSgenomeDataPkg("BSgenome.Zmays.Ensemble.zmAGPv4-seed")

## The construction completes successfully with messages:

# Creating package in ./BSgenome.Zmays.Ensemble.zmAGPv4
# Loading
'/home/zhen.di/corn/Zea_mays/Ensembl/AGPv4/Sequence/WholeGenomeFasta/prepara
tion//zmAGPv4.2bit' ... DONE
# Writing sequences to
'./BSgenome.Zmays.Ensemble.zmAGPv4/inst/extdata/single_sequences.2bit' ...
DONE
```

6. Construct R package

```
$ R CMD build BSgenome.Zmays.Ensemble.zmAGPv4
```

7. Check whether the tar.gz file has been successfully created by

```
$ R CMD check BSgenome.Zmays.Ensemble.zmAGPv4_1.0.tar.gz
```

   You should receive messages showing everything is OK.

8. Install package into R

```
$ R CMD INSTALL BSgenome.Zmays.Ensemble.zmAGPv4_1.0.tar.gz
```

9. Construct TxDb object

```
library(AnnotationHub) # help to build the annotation object
library(biomaRt)
library(GenomicFeatures)
# library(exomePeak2)
library(rtracklayer)
library(BSgenome)
library(BSgenome.Zmays.Ensemble.zmAGPv4)

maize_txdb<- makeTxDbFromBiomart(biomart = "plants_mart",dataset =
"zmays_eg_gene",host = "http://plants.ensembl.org")
saveDb(maize_txdb, file="maize_v4.sqlite")
maize_txdb <- loadDb("maize_v4.sqlite")
```

10. Use TxDb object in exomePeak2 function

```
exomePeak2(bam_ip = c("IP_1.bam","IP_2.bam"),
          bam_input = c("Input_1.bam","Input_2.bam"),
          txdb = maize_txdb,
          bsgenome = BSgenome.Zmays.Ensemble.zmAGPv4,
          paired_end = TRUE)
```

Reference

[1] B. Tan, H. Liu, S. Zhang, S. R. da Silva, L. Zhang, J. Meng et al., "Viral and cellular N(6)-methyladenosine and N(6),2'-O-dimethyladenosine epitranscriptomes in the KSHV life cycle," (in eng), Nat Microbiol, vol. 3, no. 1, pp. 108-120, 2018, doi: 10.1038/s41564-017-0056-8. [paper]

[2] M. E. Hess, S. Hess, K. D. Meyer, L. A. Verhagen, L. Koch, H. S. Brönneke et al., "The fat mass and obesity associated gene (Fto) regulates activity of the dopaminergic midbrain circuitry," (in eng), Nat Neurosci, vol. 16, no. 8, pp. 1042-8, Aug 2013, doi: 10.1038/nn.3449. [paper]

[3] Y. Tang, K. Chen, B. Song, J. Ma, X. Wu, Q. Xu et al., "m6A-Atlas: a comprehensive knowledgebase for unraveling the N6-methyladenosine (m6A) epitranscriptome," Nucleic Acids Res, vol. 49, no. D1, pp. D134-D143, 2020, doi: 10.1093/nar/gkaa692. [paper]

[4] K. Chen, B. Song, Y. Tang, Z. Wei, Q. Xu, J. Su et al., "RMDisease: a database of genetic variants that affect RNA modifications, with implications for epitranscriptome pathogenesis," Nucleic Acids Res, vol. 49, no. D1, pp. D1396-D1404, 2020, doi: 10.1093/nar/gkaa790. [paper]

[5] K. Chen, Z. Wei, Q. Zhang, X. Wu, R. Rong, Z. Lu et al., "WHISTLE: a high-accuracy map of the human N6-methyladenosine (m6A) epitranscriptome predicted using a machine learning approach," Nucleic Acids Res, vol. 47, no. 7, pp. e41-e41, 2019, doi: 10.1093/nar/gkz074. [paper]

[6] D. Kim, J. M. Paggi, C. Park, C. Bennett, and S. L. Salzberg, "Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype," Nature Biotechnology, vol. 37, no. 8, pp. 907-915, 2019/08/01 2019, doi: 10.1038/s41587-019-0201-4. [paper]

[7] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer et al., "The Sequence Alignment/Map format and SAMtools," (in eng), Bioinformatics, vol. 25, no. 16, pp. 2078-9, Aug 15 2009, doi: 10.1093/bioinformatics/btp352.[paper]

[8] M. Pertea, G. M. Pertea, C. M. Antonescu, T.-C. Chang, J. T. Mendell, and S. L. Salzberg, "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads," Nature Biotechnology, vol. 33, no. 3, pp. 290-295, 2015/03/01 2015, doi: 10.1038/nbt.3122. [paper]

[9] A. C. Frazee, G. Pertea, A. E. Jaffe, B. Langmead, S. L. Salzberg, and J. T. Leek, "Flexible isoform-level differential expression analysis with Ballgown," bioRxiv, p. 003665, 2014, doi: 10.1101/003665. [paper]

[10] Zhen Wei (2020). exomePeak2: Bias Awared Peak Calling and Quantification for MeRIP-Seq. R package version 1.0.0. [bioc]

[11] S. Heinz, C. Benner, N. Spann, E. Bertolino, Y. C. Lin, P. Laslo et al., "Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities," Molecular cell, vol. 38, no. 4, pp. 576-589, 2010. [paper]

[12] Y. Wang, K. Chen, Z. Wei, F. Coenen, J. Su, and J. Meng, "MetaTX: deciphering the distribution of mRNA-related features in the presence of isoform ambiguity, with applications in epitranscriptome analysis," Bioinformatics, 2020, doi: 10.1093/bioinformatics/btaa938. [paper]

[13] H. Thorvaldsdóttir, J. T. Robinson, and J. P. Mesirov, "Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration," (in eng), Brief Bioinform, vol. 14, no. 2, pp. 178-92, Mar 2013, doi: 10.1093/bib/bbs017. [paper]

[14] Q. Liu and R. I. Gregory, "RNAmod: an integrated system for the annotation of mRNA modifications," Nucleic Acids Res, vol. 47, no. W1, pp. W548-W555, 2019, doi: 10.1093/nar/gkz479. [paper]

[15] X. Jiao, B. T. Sherman, D. W. Huang, R. Stephens, M. W. Baseler, H. C. Lane et al., "DAVID-WS: a stateful web service to facilitate gene/protein list analysis," (in eng), Bioinformatics (Oxford, England), vol. 28, no. 13, pp. 1805-1806, 2012, doi: 10.1093/bioinformatics/bts251.[paper]