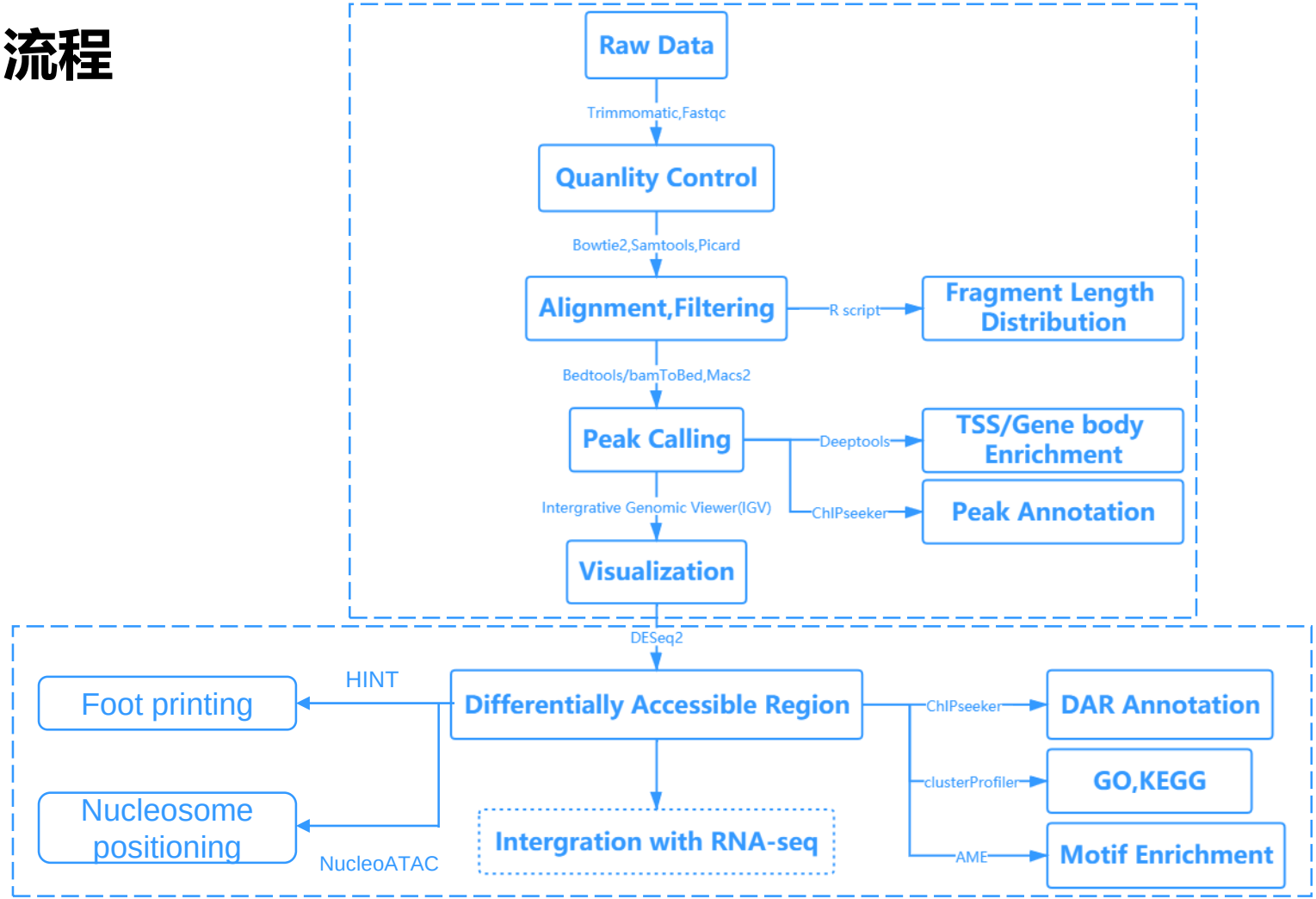
ATAC 数据分析

姜凌瀚 / 表观遗传事业部
jianglinghan@hotmail.com

Frasergen 菲沙基因

流程



质控

高级分析

Frasergen

去接头

工具：Trimmomatic

```
java -jar /public/home/aczhzv5pmn/software/Trimmomatic-0.39/trimmomatic-0.39.jar PE \
     -threads 8     \
     -summary /public/home/aczhzv5pmn/atac/01.QC/01.analysis/A1/A1.trim.stats \
     -validatePairs   \
     -baseout /public/home/aczhzv5pmn/atac/01.QC/01.analysis/A1/A1.fq.gz \
     /public/home/aczhzv5pmn/atac/rawdata/A_R1.fq.gz /public/home/aczhzv5pmn/atac/rawdata/A_R2.fq.gz \
     ILLUMINACLIP:/public/home/aczhzv5pmn/software/Trimmomatic-0.39/adapters/NexteraPE-PE.fa:2:30:10:8:true
MINLEN:8
```

-threads：线程

-summary：输出统计文件

-validatePairs：验证双端reads

-baseout：输出fq文件

ILLUMINACLIP:<fastaWithAdaptersEtc>:<seed mismatches>:<palindrome clip threshold>:<simple clip threshold>:<minAdapterLength>:<keepBothReads>

```
A1_2U.fq.gz
A1_1U.fq.gz
A1.trim.stats
A1_2P.fq.gz
A1_1P.fq.gz
```

```
Input Read Pairs: 78246189
Both Surviving Reads: 78238863
Both Surviving Read Percent: 99.99
Forward Only Surviving Reads: 2319
Forward Only Surviving Read Percent: 0.00
Reverse Only Surviving Reads: 5002
Reverse Only Surviving Read Percent: 0.01
Dropped Reads: 5
Dropped Read Percent: 0.00
```

http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf

QC

工具：fastqc

```
fastqc -t 10 A1_1P.fq.gz A1_2P.fq.gz
```

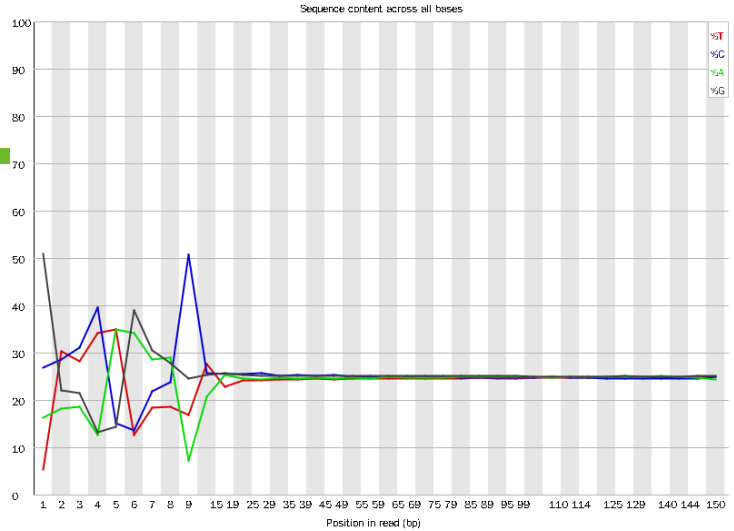
-t：线程数

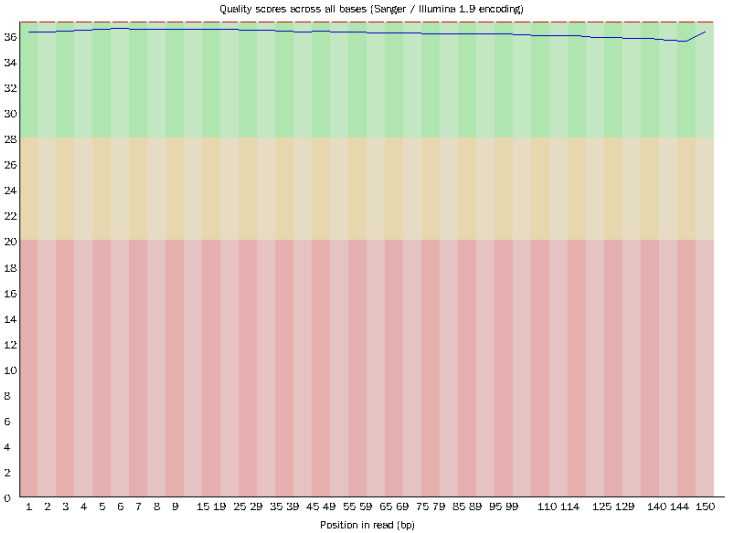
A1_1P.fq.gz：reads1

A1_2P.fq.gz：reads2

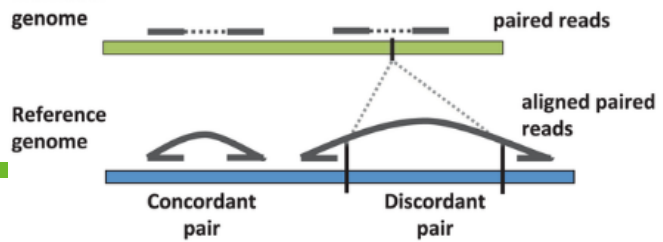
输出结果：

```
A1_2P_fastqc.zip
A1_2P_fastqc.html
A1_1P_fastqc.zip
A1_1P_fastqc.html
```





比对 - 1



discordant 示意

工具：bowtie2、samtools

```
bowtie2 -p 8 -q -I 10 -X 1000 --dovetail --no-unal --very-sensitive-local --
no-mixed --no-discordant -x hg18 -1 A1_1P.fq.gz -2 A1_2P.fq.gz 2>A1.alignment
.summary | samtools view -F 4 -u - | samtools sort -@ 6 -o A1.bam -
```

bowtie2:

-p：线程数

-q：表明 input 是 fastq 格式

-I：fragment 最小长度

-X：fragment 最大长度

--dovetail：允许 dovetail

--no-unal：不输出失败 align record

--very-sensitive-local：比对模式，敏感局部比对

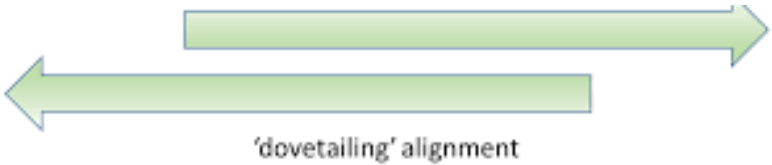
--no-mixed：不允许单端比对

--no-discordant：不允许discordant 比对

-x：参考基因组 index 前缀

A1_1P.fq.gz：reads1

A1_2P.fq.gz：reads2



samtools 参数：

-F 4: 不输出 flag 为 4 （未比对）的比对结果，各flag含义可在 Explain SAM Flags (broadinstitute.github.io) 查询

-u：比对前XX个reads pair，若不设定，全部比对

-：接受标准流作为输入

sort: 排序功能

-@：压缩线程数

-o：输出文件

Frasergen

比对 - 2

输出文件

```
A1.alignment.summary
A1.bam
```

A1.alignment.summary：

```
43933854 reads; of these:
  43933854 (100.00%) were paired; of these:
    356593 (0.81%) aligned concordantly 0 times
    41586988 (94.66%) aligned concordantly exactly 1 time
    1990273 (4.53%) aligned concordantly >1 times
99.19% overall alignment rate
```

samtools: http://www.htslib.org/doc/samtools-view.html
Bowtie2: http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml#bowtie2-options-I

Frasergen

比对结果过滤

工具：samtools

```
samtools view -b -f 2 -q 30 -o A1.f2.q30.bam A1.bam
```

-b：输出为 bam 文件

-f：required-flags，“2”代表 “read mapped in proper pair”，

-q：比对质量 MAPQ 阈值

-o：输出文件

A1.bam：输入文件

samtools: http://www.htslib.org/doc/samtools-view.html
samtools flag: https://www.samformat.info/sam-format-flag

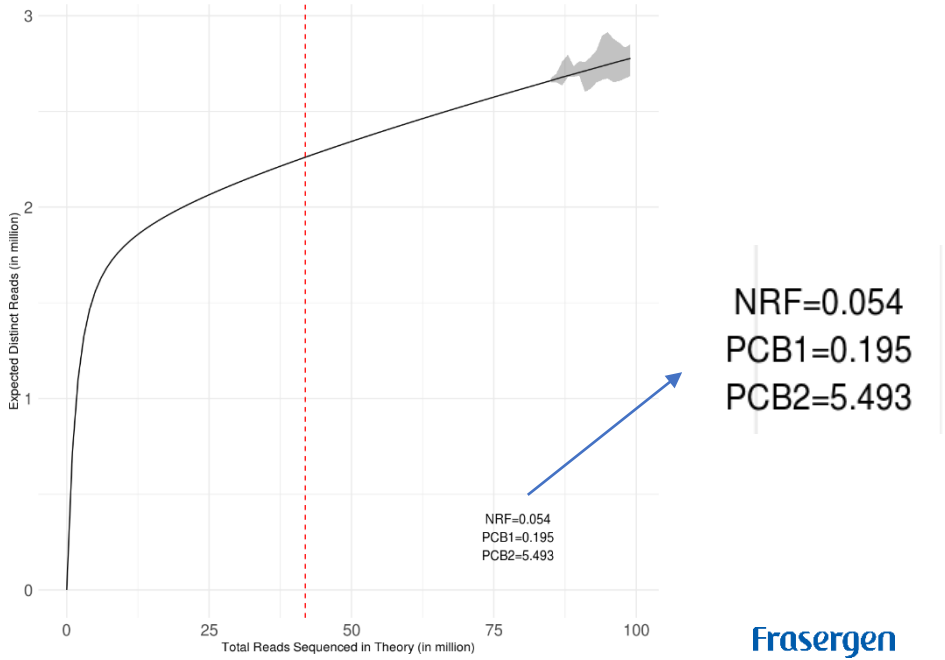
文库复杂度

工具：preseq

```
preseq lc_extrap -e 1e+8 -P -B -D -v -o A1.dat A1.f2.q30.bam 2>A1.log
Rscript preseq.R A1.dat A1.log A1
```

lc_extrap：预测未来实验的产量
-e：预测范围的最大值（右图）
-P：paired end
-B：指明输入文件为 sorted bam
-D：缺陷模式
-v：打印提示信息
-o：输出文件
A1.f2.q30.bam：输入文件

A1.dat：preseq 输出文件
A1.log：preseq log 文件
A1: 输出图片文件前缀



http://smithlabresearch.org/software/preseq/

Frasergen

去重

工具：Picard

CUT&Tag 底噪低，重复片段很有可能是真实片段。如果文库复杂度较高，可以不进行此步骤。

```
picard -Djava.io.tmpdir=./ MarkDuplicates PG=null VERBOSITY=ERROR QUIET=true
 CREATE_INDEX=false REMOVE_DUPLICATES=true INPUT=A1.f2.q30.bam OUTPUT=A1.f2.q
30.dedup.bam M=A1.pe.markduplicates.log
```

-Djava.io.tmpdir：临时文件夹　　INPUT：输入 bam 文件

MarkDuplicates：去重模式　　OUPUT：输出 bam 文件

REMOVE_DUPLICATES：去重与否　　M：log 文件

```
## METRICS CLASS    picard.sam.DuplicationMetrics
LIBRARY UNPAIRED_READS_EXAMINED READ_PAIRS_EXAMINED SECONDARY_OR_SUPPLEMENTARY_RDS  UNMAPPED_READS  UNPAIRED_
READ_DUPLICATES    READ_PAIR_DUPLICATES    READ_PAIR_OPTICAL_DUPLICATES    PERCENT_DUPLICATION ESTIMATED_LIBR
ARY_SIZE
Unknown Library 0   41880567    0   0   0   39950381    1214709 0.953912    1930186
```

部分log文件　　重复率

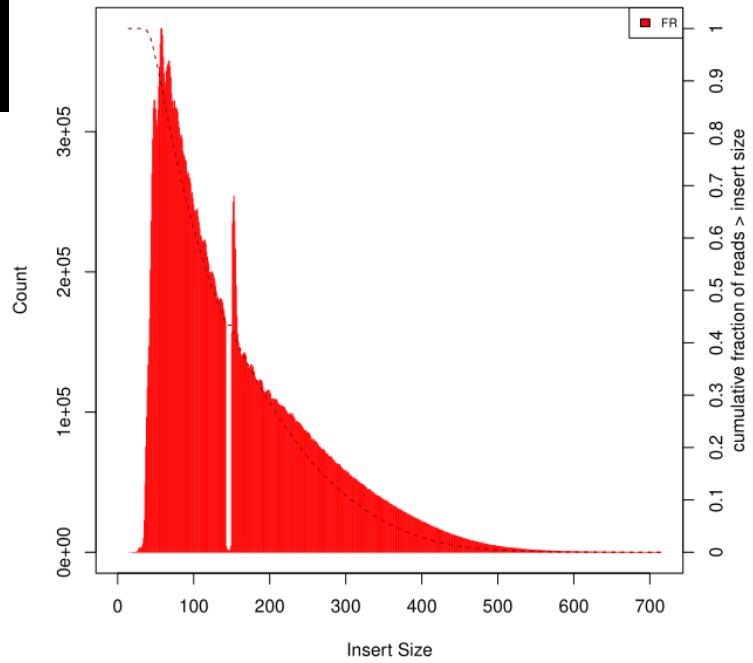
https://broadinstitute.github.io/picard/command-line-overview.html#MarkDuplicates

Frasergen

Fragment length distribution

工具：Picard

```
java -jar picard.jar CollectInsertSizeMetrics \
    I=ATAC-seq-0h-1.f2.q30.bam \
    O=insert_size_metrics.txt  \
    H=insert_size_histogram.pdf \
    M=0.5
```



https://gatk.broadinstitute.org/hc/en-us/articles/360037055772-CollectInsertSizeMetrics-Picard-

Call peak

工具：macs2

```
A1_peaks.xls
A1_control_lambda.bdg
A1_treat_pileup.bw
A1_treat_pileup.bedgraph
```

输出文件

```
GSIZE=$(bowtie2-inspect <bowtie 文库前缀>
| perl -ne 'BEGIN{$n=0} next if(/^>/);s/[^ATGC]//gi; $n+=length($_); END{prin
t int($n*0.85);}')

# narrowPeak calling
macs2 callpeak \
-t SAMPLE.filter.bam \                    # 输入 bam 文件
-f BAMPE \                                # paired-end reads 模式
-n SAMPLE \                               # 输出结果前缀
-g ${GSIZE} \                             # 参考基因组有效长度，数量级正确即可
-B \                                      # 生成 bedgraph 文件，用于可视化
--SPMR \                                  # signal per million reads, normalize
--keep-dup all                            # 保留所有 PCR 冗余

# broadPeak calling
只需再添加--broad 参数
```

因为前面已经用 picard 去过重了，所以无论是否设置此参数，对结果影响不大；如果未经 picard 去重，应当设置此参数，否则会造成较高的假阳性。

macs3 官方推荐的 call peak 参数：https://github.com/macs3-project/MACS

其他参数设置：

https://github.com/macs3-project/MACS/issues/145

Samarajiwa， 2017

关于 call peak 参数的讨论：https://mp.weixin.qq.com/s/W3Vo91uw_MEOBKVlMyEYTA

《表观遗传学常用技术手册》将更加详尽地讨论这个问题

Frasergen

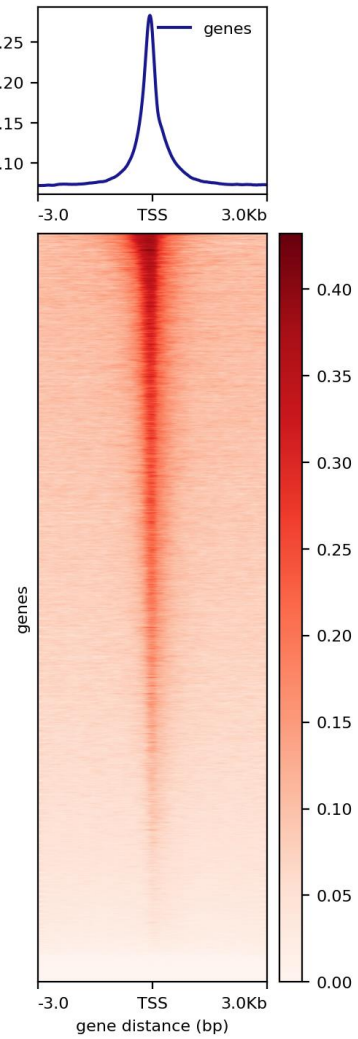
Peak 可视化

工具：deepTools

deepTools 是德国马普所开发的深度测序数据工具箱，可对测序数据进行多种处理和可视化。

```
computeMatrix reference-point --referencePoint TSS \
-p 6 \                                  # 线程数
-a 3000 \                               # 上游长度
-b 3000 \                               # 下游长度
-R <gff 文件中提取的 TSS 位置信息> \      # bed 或 gff 文件
-S A1_treat_pileup.bw \                 # macs2 生成的 bigwig
--skipZeros \                           # 跳过全 0 区间
-o A1_matrix_TSS.gz \                   # 输出文件名
--missingDataAsZero                     # 缺失数据设为 0
```

```
plotHeatmap \
--heatmapHeight 16\     # Heatmap 高度
--heatmapWidth 4  \     # Heatmap 宽度
--colorMap Reds \       # 配色方案
--legendLocation none \ # 图例
--samplesLabel A1 \     # 样品标签
-m A1_matrix.gz \       # computeMatrix 生成的矩阵文件
-o A1_enrichment.png    # 输出图片文件
```



https://deeptools.readthedocs.io/en/develop/#

Ramirez et al. 2016, *NAR*

Peak 注释

ChIPseeker

安装

```
library(BiocManager)
BiocManager::install("ChIPseeker")
```

读入peak文件

```
peak <- readPeakFile(peakfile=peakFile, header=FALSE, as="GRanges")
```

读入注释文件

```
tx <- makeTxDbFromGFF(file=gff)
```

注释peak

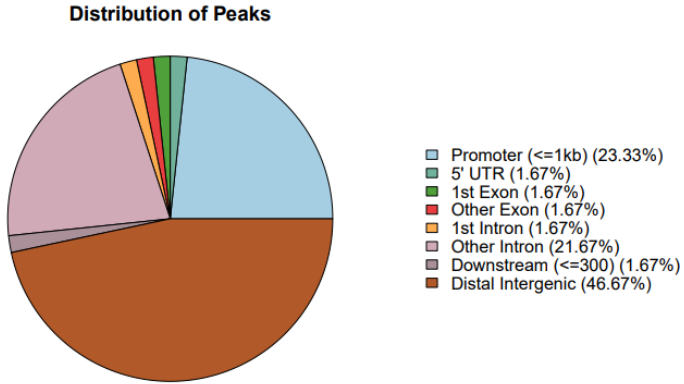
```
peakAnno <- annotatePeak(
              peak=peak,
              tssRegion = c(-3000, 3000),
              TxDb=tx,
              assignGenomicAnnotation=TRUE)
```

输出文件

```
peakAno.result <- as.data.frame(peakAnno@anno)
write.table(peakAno.result,
            file="sample_1.peakAnno.tsv",
            sep="\t",
            row.names=FALSE)
```

绘图

```
plotAnnoPie(peakAnno)
```



差异 peak - 1

工具：DiffBind

DiffBind是剑桥大学癌症研究中心的研究者开发的DP分析工具，对DP分析的各个步骤进行了细致的包装，使得用户能够以非常简洁的代码导入数据、count reads、normalize、可视化和存储数据。在此我们只介绍DP分析中用到的关键代码，其他更加丰富的内容请参阅 https://bioconductor.org/packages/release/bioc/vignettes/DiffBind/inst/doc/DiffBind.pdf。

安装DiffBind

DiffBind是R包，最新版本DiffBind需要R > 4.0，可用BiocManager安装。

```
> library(BiocManager)
> BiocManager::install("DiffBind")
```

差异 peak - 2

准备samplelist

SampleID	Condition	Replicate	Peaks	bamReads	PeakCaller
sample-1	A	1	sample-1_peaks.xls	sample-1.bam	macs
sample-2	A	2	sample-2_peaks.xls	sample-2.bam	macs
sample-3	B	1	sample-3_peaks.xls	sample-3.bam	macs
sample-4	B	2	sample-4_peaks.xls	sample-4.bam	macs

至少两个重复，否则会报错

SampleID：样品名；

Condition：组名；

Replicate：第几个重复；

Peaks：peak文件路径；

bamReads：样品的bam文件（经过质量筛选）

PeakCaller：与 Peaks有右图对应关系：

- “raw”: text file file; peak score is in fourth column
- “bed”: .bed file; peak score is in fifth column
- “narrow”: default peak.format: narrowPeaks file
- “macs”: MACS .xls file
- “swembl”: SWEMBL .peaks file
- “bayes”: bayesPeak file
- “peakset”: peakset written out using pv.writepeakset
- “fp4”: FindPeaks v4

在samplelist示例中，因为“Peaks”使用的是macs产生的xls文件，因此此列填“macs”。

差异 peak - 3

计算差异

导入上一步生成的 samplelist：

```
samples <- read.csv(samplelist, sep="\t")
```

将其转换为 dba 对象：

```
obj <- dba(sampleSheet=samples)
```

计数 peak 上的 reads 数：

```
count <- dba.count(obj)
```

normalize reads counts：

```
norm <- dba.normalize(
          count,
          method=DBA_DESEQ2,
          library=DBA_LIBSIZE_PEAKREADS,
          normalize=DBA_NORM_LIB,
          libFun=median)
```

不同于CUT&Tag 的 full lib size，此处我们设置的library 是 RiP，认为这样能较好地校正 Gain DAR的 bias，但这是有争议的。若不认同，可以与CUT&Tag 使用一样的参数。

构造比较分组，treat（B）在前，control（A）在后：

```
contrast <- dba.contrast(norm, contrast=c("Condition", B, A))
```

计算差异：

```
analyze <- dba.analyze(contrast)
```

Frasergen

差异 peak - 4

报告展示保存结果

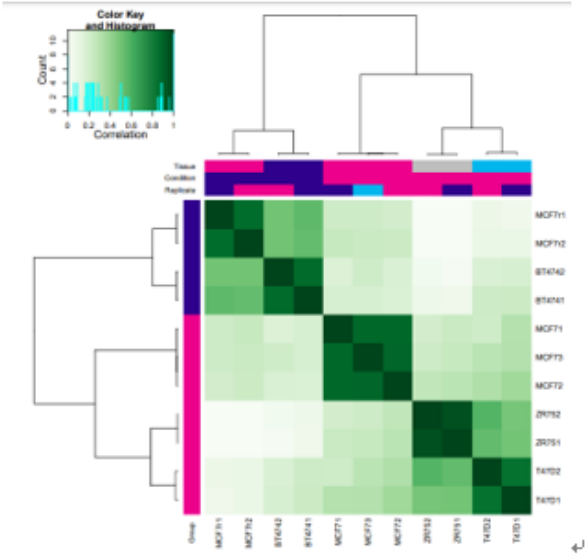
报告结果，下图展示的命令输出了所有差异，可自行调整 p 值和 Fold Change 阈值进行筛选：

```
res <- dba.report(analyze, th=1)
```

特别值得说明的是，DiffBind 提供了丰富的作图功能，可以方便地绘制 venn 图、heatmap 图、PCA 图、MA 图、火山图、箱线图等。

如，使用如下命令可绘制所有样品的相关性热图：

```
dba.plotHeatmap(norm)
```



DiffBind 绘制的所有样品相关性热图（示意）

DiffBind 还可以非常方便地保存和导入中间变量：

保存：

```
dba.save(norm, file="All", dir=outdir)
```

导入：

```
norm <- dba.load("All")
```

Frasergen

富集分析 - 1

工具：clusterProfiler

clusterProfiler是一个R包，第一版于2012年发行，截至2022年已经更新至4.0版本。clusterProfiler提供了丰富的gene set富集分析、通路富集分析函数，以及丰富的作图函数，我们这里只介绍通路富集的部分，更多详细的使用说明参见 https://guangchuangyu.github.io/software/clusterProfiler/documentation/。

```
> library(BiocManager)
> BiocManager::install("clusterProfiler")
```

Frasergen

富集分析 - 2

准备输入文件

基因和通路的映射表，第一列基因，第二列通路，tab分隔：

```
ENSG00000176399 GO:0007610
ENSG00000213585 GO:0007610
ENSG00000139436 GO:0007610
ENSG00000138071 GO:0007610
```

GO Term的描述信息：

```
GO:0000001      mitochondrion inheritance
GO:0000002      mitochondrial genome maintenance
GO:0000003      reproduction
GO:0019952      reproduction
GO:0050876      reproduction
```

待富集基因列表：

```
ENSG00000285866
ENSG00000263301
ENSG00000235659
ENSG00000105137
ENSG00000236654
ENSG00000232132
ENSG00000189052
ENSG00000166352
```

Frasergen

富集分析 - 3

读入基因通路映射表：

```
gene2go <- read.table(
             ACC2GO,
             stringsAsFactor=FALSE,
             sep="\t",
             quote="")
```

读入GO Term的描述信息：

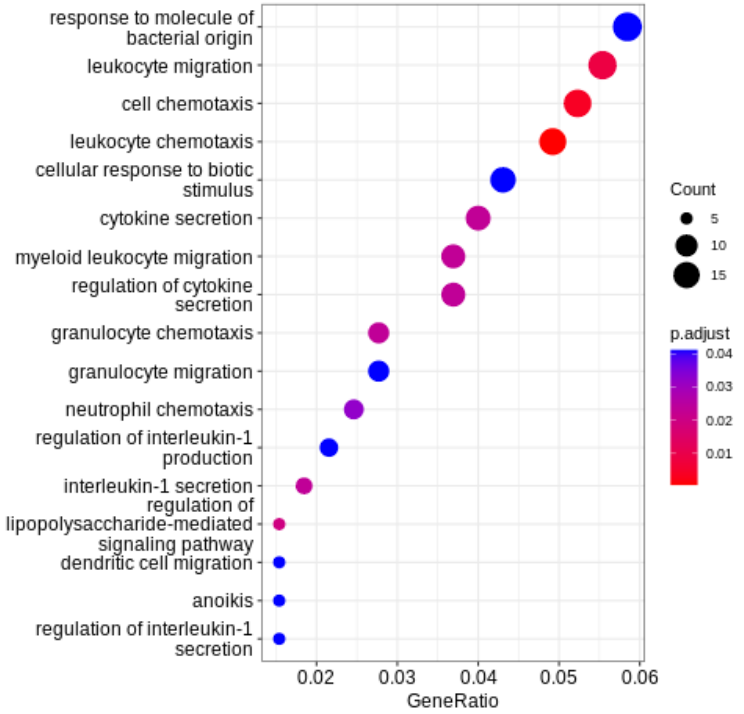
```
GO_DB <- read.table(
           "GO_list.tsv",
           stringsAsFactor=FALSE,
           sep="\t",
           quote="",
           comment="")
```

富集：

```
res <- enricher(
         gene=genelist,
         pvalueCutoff=0.05,
         TERM2GENE=gene2go[,c(2,1)],
         TERM2NAME=GO_DB,
         pAdjustMethod="BH")
```

绘图：

```
dotplot(res, showCategory=20)
```



Motif 富集 - 1

工具：AME

AME(Analysis of Motif Enrichment)是**MEME套件**（https://meme-suite.org/meme/index.html）中的一个模块，该套件由包括华盛顿大学、内华达大学、UCSD等多个机构的研究人员合作开发，提供了多种motif分析功能，包括motif discovery、motif enrichment、motif scanning、motif比较等多种功能。AME是其中用于motif enrichment的模块。

安装AME

a) 创建环境，并安装MEME

```
conda create -n meme -c conda-forge -c bioconda meme
```

b) 激活环境，查看AME所在路径

```
source activate meme
$which ame
/public/frasergen/PUB/software/meme/meme-5.3.3/bin/ame
```

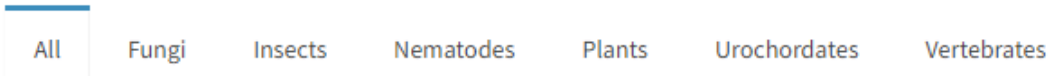
c) 将AME所在路径加入环境变量

```
export PATH=/public/frasergen/PUB/software/meme/meme-5.3.3/bin:$PATH
```

Motif 富集 - 2

下载PFM(Position Frequency Matrix)数据

从JASPAR（https://jaspar.genereg.net/downloads/）下载按照下图分类的motif位置频率矩阵数据。



其中有专门适用于MEME的格式：

Frasergen

Motif 富集 - 3

```
ame -o out_dir\
    -control merge_peak.fa\
    <DAR.fa>\
    <pfm>
```

-o：输出路径；

-control：背景DNA序列；

DAR.fa：差异开放区域的DNA序列；

pfm：某JASPAR类别（Vertebrates、Plants等）的motif的位置频率矩阵，格式如下：

```
MOTIF MA0266.1 ABF2
letter-probability matrix: alength= 4 w= 7 nsites= 100 E= 0
 0.180000  0.390000  0.250000  0.180000
 0.090000  0.090000  0.020000  0.800000
 0.000000  1.000000  0.000000  0.000000
 0.000000  0.000000  0.000000  1.000000
 1.000000  0.000000  0.000000  0.000000
 0.000000  0.000000  1.000000  0.000000
 0.940594  0.019802  0.019802  0.019802
URL http://jaspar.genereg.net/matrix/MA0266.1
```

Frasergen

足迹分析 - 1

工具：HINT-ATAC

HINT是RGT（Regulatory Genomics Toolbox ）套件的一个模块，该套件由德国RWTH大学医院计算生物学和生物信息学研究小组研发。除了HINT之外，RGT还包含motif分析、ChIP-Seq 差异peak caller（ODIN、THOR）等套件。

HINT-ATAC是HINT的一个子命令，专门用于ATAC足迹分析，与之相应的，有HINT-DNase，专门用于DNase-seq的足迹分析。

https://www.regulatory-genomics.org/hint/introduction/

足迹分析 - 2

1. 安装 HINT-ATAC

HINT-ATAC 整合在 RGT 中，需要安装整个 RGT 套件。RGT 主要使用 python 语言开发，可直接使用 pip 安装。

1. 安装 RGT 之前创建一个名为 hint 的 conda 环境，并安装其依赖的 python 包：

```
conda create -n hint -c conda-forge -y python=3.9 cython numpy scipy
```

如果自己的环境中已有这些依赖包，可跳过此步骤。

2. 激活环境，安装 RGT

```
source activate hint
```

```
pip install RGT
```

3. 将 rgt-hint 所在路径加入环境变量

使用 which 查看 rgt-hint 所在路径：

```
$which rgt-hint
/public/frasergen/PUB/software/Anaconda/anaconda3-3d/bin/rgt-hint
```

将其加入环境变量：

```
export PATH=/public/frasergen/PUB/software/Anaconda/anaconda3-3d/bin:$PATH
```

此步骤非必须。若不执行此步骤，后续使用 HINT 时需要激活环境，可根据需求选择是否跳过。

4. 配置 rgtdata

安装 RGT 时，会自动在家目录下创建如下文件夹：

```
~/rgtdata
```

HINT 运行时需要用到序列文件（fa）、注释文件（gtf、bed）、chromsize 等文件都需要存储在此文件夹中，才能被 HINT 检索到。具体配置方法见使用部分。

如果想更改 rgtdata 的存储路径，可将文件夹复制到目标路径下，然后更改如下环境变量：

```
export RGTDATA=/public/frasergen/3D/share/rgtdata
```

足迹分析 - 3

2. 准备数据

在进行足迹分析以前，要先行准备数据文件夹，并将各文件路径写入配置文件 data.config 中。下面举例说明。

data.config 示例如下：

```
[hg19]
genome: hg19/genome_hg19.fa
chromosome_sizes: hg19/chrom.sizes.hg19
gene_regions: hg19/genes_Gencode_hg19.bed
annotation: hg19/gencode.v19.annotation.gtf
gene_alias: hg19/alias_human.txt
```

[hg19]：下面罗列的数据所在的文件夹；

genome：参考基因组的 DNA 序列，fa 格式；

chromosome_sizes：染色体 size 文件，每行两列——染色体编号、染色体 size，以 tab 分隔；

gene_regions：注释信息，bed 格式；

annotation：注释信息，gtf 格式；

gene_alias：基因多种命名方式，一行一个基因，每列一种命名方式，用 tab 分隔；

足迹分析 - 4

3. 使用 HINT-ATAC

```
rgt-hint footprinting --atac-seq\
                      --paired-end\
                      --organism=hg19\
                      --output-prefix=sample_1\
                      --output-location=out_dir\
                      <sample_bam>\
                      <peak_bed>
```

--atac-seq：指明用于分析的数据来自 atac-seq；

--pair-end：指明 sample_bam 来自双端测序；

--output-prefix：输出文件前缀，通常填样品名；

--output-location：输出路径；

<sample_bam>：样本 bam 文件；

<peak_bed>：peak bed 文件；

输出文件：

```
sample_1.bed
sample_1.info
```

sample_1.bed：转录因子足迹区域的 bed 文件；

sampe_1.info：输入和输出的一些统计信息（如下图）；

```
Number of reads: 57739022
Number of peaks: 134024
Number of tag counts within peaks: 19951111.0
Number of footprints: 700421
```

核小体定位

```
nucleoatac run --bed sample_1.bed\
               --bam sample_1.bam\
               --fasta hg19.fasta\
               --out sample_1\
               --cores 10
```

--bed：样本的 bed 注释；

--bam：样本的比对文件；

--fasta：参考基因组的序列文件；

--out：输出前缀；

--cores：允许使用的 CPU 数目；

输出文件：

```
sample_1.fragmentsizes.txt                    sample_1.occ.bedgraph.gz
sample_1.ins.bedgraph.gz                      sample_1.occ_fit.eps
sample_1.nfrpos.bed.gz                        sample_1.occ_fit.txt
sample_1.nuc_dist.eps                         sample_1.occ.lower_bound.bedgraph.gz
sample_1.nuc_dist.txt                         sample_1.occpeaks.bed.gz
sample_1.nucleoatac_signal.bedgraph.gz        sample_1.occ.upper_bound.bedgraph.gz
sample_1.nucleoatac_signal.smooth.bedgraph.gz sample_1_peaks.bed
sample_1.nucmap_combined.bed.gz               sample_1.VMat
sample_1.nucpos.bed.gz                        sample_1.VMat.eps
sample_1.nucpos.redundant.bed.gz
```

文件较多，取其中较重要的加以说明，详细说明参见 https://nucleoatac.readthedocs.io/en/latest/nucleoatac/：

sample_1.occpeaks.bed.gz：低分辨率核小体占位 bed 文件；

sample_1.nucpos.bed.gz：高分辨率核小体占位 bed 文件；

sample_1.nucmap_combined.bed.gz：合并高低分辨率核小体占位的 bed 文件；

sample_1.nfrpos.bed.gz：nucleosome free region 的 bed 文件；

https://nucleoatac.readthedocs.io/en/latest/nucleoatac/

ATAC-Seq 信号强度与基因表达水平的关系

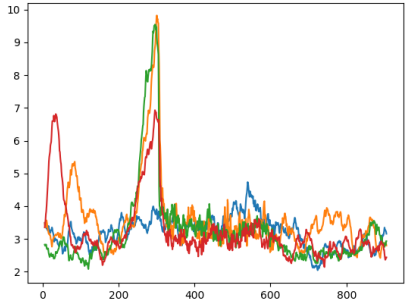
1. 用 computeMatrix 计算 ATAC 的reads 在 TSS 附近的分布矩阵
2. 把基因按照表达量分组
3. 按照基因分组把矩阵分块

```
computeMatrix scale-regions \
    -p 6 \
    -S A1_treat_pileup.bw \
    -R ref/Mus.genebody.bed \
    -b 3000 \
    -a 3000 \
    -m 3000 \
    -o A1.genebody.matrix.gz \
    --skipZeros
```


```
@{"upstream":[3000],"downstream":[3000],"body":[3000],"bin size":[10],"ref point":[null],"verbose":false,"bi
n avg type":"mean","missing data as zero":false,"min threshold":null,"max threshold":null,"scale":1,"skip ze
ros":true,"nan after end":false,"proc number":6,"sort regions":"keep","sort using":"mean","unscaled 5 prime"
:[0],"unscaled 3 prime":[0],"group_labels":["genes"],"group_boundaries":[0,138102],"sample_labels":["M3_trea
t_pileup"],"sample_boundaries":[0,900]}
chr1    3073252 3074322 gene:ENSMUSG00000102693 1000.0  +       0.028680        0.028680        0.028680
        0.028680        0.028680        0.014340        0.014340        0.014340        0.014340        0.01
4340        0.014340        0.014340        0.014340        0.014340        0.012906        0.000000
0.001434        0.014340        0.014340        0.014340        0.014340        0.014340        0.025812
```


4. 每块矩阵列求和，绘在一张图上即可

```
f = "A1.genebody.matrix.gz"
df = pd.read_csv(f, nrows=200, sep="\t", comment="@", header=None)
data = df.iloc[:, 6:]
lst = np.split(data, 4, axis=0)  # 随机划分四个基因集
for i in lst:
    i.sum().plot()
plt.savefig("test.png")
```



Thanks