

Bioinformatics Report

Leb_G12

**Linux-based Essential Bioinformatics (LEB)
Completion Report**

Ma Xiaokai Zheng Hongxi Wu Yufeng Chen Chen

1 常用 linux 命令

• 服务器、存储空间相关

- **ssh**

- **ssh leb12a@117.78.18.116**

- **passwd *username***

- 更改密码

- **w**

- **w**

- 看服务器内谁登陆了，能看到 DNS、登陆时间等

- **w | less**

- 按回车往下，按 q 退出

- **who**

- 类似于 w

- **whoami**

- 我是谁？

- **exit**

- 退出服务器

- **top**

- 动态显示系统运行状态，包括处理器、内存使用情况以及进程，q 退出

- **df**

- 显示系统存储空间不同分区名称、容量和使用状态

- **du**

- 显示目录下所有子目录和文件占用空间，默认当前目录

- **-m**

- 按 MB 为单位 (还有 b、k 等)

- **free**

- 显示内存使用状态

- **ps**

- 显示进程信息

- **-u**

- 显示本用户当前进程信息

- **-ef**

- 显示所有用户当前进程详细信息

- **kill num**

- 终止编号为 num 的无用进程

- **正则表达式**

- 基本正则表达式 (BRE)

- .

- 匹配任意非空字符

- ^

- 开头锚定符

- \$

- 结尾锚定符

- []

- 匹配字符集中的某个字符或某一字符类

- 中括号中首位的 ^ 表示否定

- 中括号中非首位的 - 表示字符范围

- *

- 匹配某元素零至多次

- ERE

- ?

- 匹配前面的元素 0 或 1 次

- +

- 匹配前面的元素一至多次

- { }

- 匹配前面的元素指定次数

- {n} 匹配 n 次

- {n,m} 匹配 n-m 次 (n 与 m 可以缺失)

- **文件目录相关**

- **pwd**

- 打印当前工作目录

- **ls**

- 查看文件夹名字
- **-l**
 - 查看详细属性
 - 对于目录文件，第二列表示它的第一级子目录的个数
 - 此值要减 2 才等于该目录下的子目录的实际个数，因为含有./与../
- **-S**
 - 以大小排列
- **-t**
 - 以时间排列
- **-h**
 - 将大小转换为计算机大小单位，如 kb
- **-a**
 - 显示隐藏目录与隐藏文件
- *****
 - 如 **ls *.FASTA** 显示当前目录下所有以.FASTA 结尾的文件
 - 如 **ls HBA/HBA_*** 显示子目录 HBA 下所有以 HBA_ 起始的文件
- **-1**
 - 一行只写 1 个文件名
- **mkdir dirname**
 - 创建文件夹
- **-p**
 - 需要在创建上层目录，如目录早已存在则不当作错误
 - **mkdir -p seq/HBA**
 - 在当前目录下创建 seq 子目录，并在子目录 seq 下创建二级子目录 HBA
- **cd**
 - 进入文件夹
 - ..
 - 进入上层目录
- **cd** 后啥也不加进入根目录
- **cp source-filename target-filename**
 - 复制文件

- **-i**

- 若目标文件已存在，系统显示是否覆盖提示信息

- **-r**

- 递归，似乎用来复制文件夹

- **mv source-filename target-filename**

- 重命名文件

- **rm filename**

- 删除文件，默认不能删除目录

- 可以通过空格隔开文件名，删除多个文件

- **rm -i cat***

- 删除以 cat 起始的文件，每删除一个文件，系统提示是否确认删除

- **-r**

- 指示 rm 删除目录

- 并将参数中列出的全部目录和子目录均递归地删除

- **chmod**

- 修改文件属性，似乎对链接进行的修改会转移到原始文件上

- **文件属性**（可用 **ls -l** 查看）

- 第一个字母

- l: 链接

- d: 文件夹

- 2-10 分别代表 user、group、others 三种用户的权限

- r: 可读

- w: 可写

- x: 可执行

- **数字表示法**

- 7: rwx

- 6: rw-

- 5: r-x

- 4: r--

- 0: ---

- **符号表示法**

- u、g、o、a 表示影响对象
- a 为 all，为默认指定
- +、-、=代表增加权限、减少权限、只有指定权限可用
- **-w**
 - 去掉 w (可写)
 - 代码如 **chmod -w filename**
- **chmod 755 bin/***
 - 将子目录 bin 下所有文件设置为本用户可读可写可执行，其它用户可读可执行
- **chmod a-w keep/**
 - 取消子目录 keep 写权限，不能在该目录下创建文件或子目录
- **ln**
 - 建立链接，默认硬链接
 - 硬链接可认为是一个文件拥有两个文件名，只要还有其他的硬链接存在，这个文件就不会被真正删除
 - 而软链接则是系统新建一个链接文件，此文件指向其所要指的文件，类似于快捷方式
 - **ln -s filename linkname**
 - 建立软链接
 - 可链接文件或文件夹
 - 若软链接 test_chk_ln 指向一个文件夹
 - **rm -rf ./test_chk_ln/** 会删除文件夹下的所有内容，但是没有删除这个链接；
 - **rm -rf ./test_chk_ln** 则是仅删除这个软链接，不会删除下面的内容。
- **file**
 - 查看文件类型
- **find pathway option command**
 - 在命令列上第一个 - (), ! 之前的部分为 path
 - 默认在当前目录及所有嵌套子目录? 下查找子目录与文件
- **-name**
 - **find /etc -name '*.conf'**
 - 查找 /etc 目录下以 conf 结尾的文件，文件名区分大小写
- **-type**
 - 根据类型查找

- **f** 文件
 - **d** 目录
 - **c** 字符设备文件
 - **b** 块设备文件
 - **l** 链接文件
 - **p** 管道文件
- **-mtime**
 - **-n** n 天以内修改的文件
 - **+n** n 天以外修改的文件
 - **n** 正好 n 天 修改的文件
 - **-mindepth n**
 - 从第 n 级目录开始搜索
 - **-maxdepth n**
 - 至多搜索到第 n-1 级子目录
 - **-prune**
 - 排除某一目录，与**-path** 连用
 - **find . -path ./test -prune -o -type f**
 - 查找当前目录下的所有普通文件，但排除 test 目录
 - **-exec**
 - 对搜索到的文件执行特定的操作
 - 格式为： **-exec command {} \;**
 - {} 表示查询的结果
 - **find -name "*.FAS" -exec mv -v "{}" "{}TA" \;**
 - 将当前目录下所有文件的后缀.FAS 改为.FASTA
 - **-ok** 和 **-exec** 的功能一样，只是每次操作都会给用户提示
 - **-a**
 - 与
 - 默认情况下查询条件之间都是 与 的关系
 - **-o**
 - 或
 - **-not | !**

- 非

- **which / whereis / locate *filename***

- 寻找文件所在位置

- **Tree**

- 显示目录文件树形结构

- **文本文件操作命令**

- **cat**

- **cat *filename***

- 查看文件

- **-n**

- 查看文件，并显示行号

- **-A**

- 查看文件，并显示控制符，如制表键、行终止符

- **cat > *filename***

- 创建文本文档，必须有>，>后为文件名

- **Enter** 键后开始输入内容

- **Ctrl + C** 退出编写，似乎得换行，不然删除所在行

- **cat >> *filename***

- 在文末尾追加新的文本

- **cat *filename1 filename2***

- 垂直连接两个文件

- **nano *filename***

- 创建文本文档

- 编写完后，**Ctrl + O** 保存修改，**Ctrl + X** 退出编写

- **less *filename***

- 逐屏显示文件，**q** 终止显示

- 回车进一行，空格进一页

- **/ + 字符串 + ENTER** 搜索字符串，**n** 搜索下一个，**N** 搜索上一个

- **head *filename***

- 显示文件前 10 行

- **-n num**

- 显示前 num 行
- **-n -num**
 - 显示除最后 num 行外的其他行
- **tail filename**
 - 显示文件最后 10 行
 - **-n -num**
 - 显示最后 num 行
 - 似乎不加负号也是一样的
 - **-n +num**
 - 即从 num 行开始显示所有行
- **sort filename**
 - 按字母表顺序对文件中的行进行排序 (升序)
 - **-k num**
 - 基于第 num 字段对文件进行排序
 - 辅助使用 **-t** 来设定间隔符
 - **-u**
 - 不显示相同行
 - **-r**
 - 变为降序
 - **-o filename**
 - 输出至原文件
 - 若输出至其他文件，可以使用重定向 **sort filename > newfile**
 - 若输出至原文件，重定向会将原文件清空，此时使用 **-o**
- **cut filename**
 - **-f num (fields)**
 - 提取第 num 个字段
 - **-d** 以确立分隔符，似乎默认制表符
 - *num-num2*: 第 num-num2 个字段
 - *num, num2*: 第 num 和第 num2 个字段
 - **-c num (character)**
 - 提取第 num 个字符

- **paste** *filename1 filename2*
 - 用制表符水平连接多个文件（逐行合并），不限于两个
 - 若文件名为 `-`，则从标准输入中读取
 - `cat filename | paste - -`
 - 将文件中每两行合并到一行显示
 - `-s`
 - 合并同一输入文件的后继行
 - `-d`
 - 指定分隔相应行的定界符
 - 若合并多个，可以在引号内一起写下多个定界符，系统会在一行中循环使用
- **wc** *filename*
 - 统计并显示文件行数、由空格分隔的单词数和字符数，最后附加上文件名
 - `-l --lines`
 - 只显示行数。
 - `-w --words`
 - 只显示单词数
 - `-c --bytes --chars`
 - 只显示字符数
- **grep**
 - 在文件中搜索与指定正则表达式相匹配的行并将结果输出到标准输出
 - `-i`
 - 区分大小写
 - **grep “^>” 209HBA.fasta**
 - 检索并显示文件 209HBA.fasta 中以大于号起始的行数
 - **ls -1 /bin/ | grep “seq\$”**
 - 检索并显示/bin/目录中含以 seq 结尾的文件
 - `-c`
 - 不打印匹配结果，而是打印匹配的行数
- **sed**
- **其他常用命令**
 - **history**

- 显示最近用过的命令

- **alias**

- 设置命令的别名，仅作用于该次登陆的作业

- **alias rm= ‘rm -i’**

- **unalias rm**

- **man command**

- 查看命令的帮助手册

- **echo**

- 屏幕输出

- **touch**

- 生成空文件

- **Date**

- 显示系统日期

- **Cal**

- 显示日历

2 软件安装

• 压缩

- **gzip**

- 使用广泛的压缩程序，文件经它压缩过后，其名称后面会多出".gz"的扩展名
- **-d**
 - 解开压缩文件
- **-c**
 - 把压缩后的文件输出到标准输出设备
- **-l**
 - 列出压缩文件的相关信息，默认不解压
- **-r**
 - 递归处理，将指定目录下的所有文件及子目录一并处理
- **-v**
 - 显示指令执行过程

- **gunzip *filename***

- 解压缩程序，用于解开被 gzip 压缩过的文件
- **-c**
 - 把解压后的文件输出到标准输出设备
- **-l**
 - 列出压缩文件的相关信息
- **-r**
 - 递归处理，将指定目录下的所有文件及子目录一并处理
- **-v**
 - 显示指令执行过程

- **tar**

- 打包与解包，通常使用 **tar cvf** 与 **tar xvf**
- **-f**
 - 必选，不然找不到你的 tar 文件
- **-c**
 - 创建新的 tar 文件
- **-C *dir***

- 将压缩包解压到当前目录下的 `dir` 目录下

- **-x**

- 从档案中提取文件

- **-v**

- 显示指令执行过程

- **-r**

- 把要存档的文件追加到档案文件的末尾。例如用户已经作好备份文件，又发现还有一个目录或是一些文件忘记备份了，这时可以使用该选项，将忘记的目录或文件追加到备份文件中

- **-t**

- 列出档案的内容

- **-z, --gzip, --gunzip**

- 通过 gzip 命令压缩或解压档案

- **tar -czvf target-file docu**

- 将目录 `docu` 打包成 `target-file`，同时使用 gzip 进行压缩

- **tar -xzvf file**

- 将档案 `file` 还原为原目录，同时使用 gzip 进行解压缩

- **tar -tzvf file**

- 只查看档案 `file` 的文件列表，不进行解包

• 安装过程

- **configure**

- 配置，一般用来生成 Makefile，为下一步的编译做准备

- **--prefix docu**

- 将该软件安装在 `docu` 下面

- 执行文件就会安装在 /`docu/bin`

- 源文件就会安装在 /`docu/share`

- 如果没有指定路径那么都是走默认路径

- 可执行文件默认放在 /usr/local/bin

- 库文件默认放在 /usr/local/lib

- 配置文件默认放在 /usr/local/etc

- 其它的资源文件放在 /usr/local/share

- **make**

- 从 Makefile 或者 makefile 中读取指令，然后编译

- **make install**

- 安装，也从 Makefile 中读取指令，安装到指定的位置

- **make clean**

- 删除一些临时文件

- **环境变量配置（以 Blast 为例）**

- 更改环境变量路径需更改**.profile** 文件

- **.bashrc**

- 在该文件下创建别名可以在以后登录仍然有用

- **source**

- 激活，在更改.bashrc 后激活并运行该文件

- **.profile**

- 在每次登陆时只执行一次，而**.bashrc** 在每次执行脚本时都会执行一遍

- 若想将目标版本 BLAST 设定至环境变量

- 先将安装后的 blast_bin 拷贝到根目录的**./bin** 文件夹下，形成**./bin/blast_bin** 下含有一系列软件

- 在根目录的**.profile** 文件中注意到以下代码

- # set PATH so it includes user's private bin if it exists

- if [-d "\$HOME/bin"] ; then

- PATH="\$HOME/bin:\$PATH"

- fi

- 将其拷贝到最下方，然后更改原位置代码为

- # set BLAST 2.13.0 PATH TO blast_bin

- if [-d "\$HOME/bin/**blast_bin**"] ; then

- PATH="\$HOME/bin/**blast_bin**:\$PATH"

- fi

- /gpfs/share/home/1900012104/software/libpng-1.6.21/_install/bin

- 即在**.profile** 文件中增加 BLAST 软件的路径

- 保存退出后，**source** 运行**.profile** 文件，即添加成功

3 tmux

- 会话 session

- 创建与关闭

- **tmux new -s *session_name***

- 创建命名为 *session_name* 的 tmux 会话

- **tmux** 创建一个默认 tmux 会话

- **tmux kill-session -t *session_name***

- 关闭该会话， 默认当前会话

- 进入与退出

- **tmux a -t *session_name***

- 进入之前断开的命名为 *session_name* 的 tmux 会话

- 默认进入第一个会话

- **tmux detach**

- 断开当前会话连接， 同时下次还能接着用

- 快捷键 **Ctrl+B D**

- **tmux switch -t *session_name***

- 切换至目标会话

- 管理与命名

- **tmux ls**

- 查看目前操作了几个 session

- 快捷键 **Ctrl+B S**

- **tmux rename-session -t *old_session_name* *new_session_name***

- 重命名会话

- 快捷键 **Ctrl+B \$**

- 窗口 window

- **tmux new-window -n *window_name***

- 创建一个命名为 *window_name* 的新窗口

- 快捷键 **Ctrl+B C**

- **tmux kill-window -t *window_name***

- 删除名称为 *window_name* 的窗口

- 快捷键 `Ctrl+B &`
- **tmux select-window -t *window_name***
 - 快速进入指定的窗口当中
 - `Ctrl+B W` 显示窗口列表
 - 通过上下进行选择会话、窗口、分屏，然后回车进入指定的位置
 - `Ctrl+B N` 快速切换到下一个窗口。
 - `Ctrl+B P` 快速切换到上一个窗口。
 - `Ctrl+B 0~9` 选择 0 ~ 9 对应的窗口
- **tmux rename-window *window_name***
 - 重命名 window
 - 快捷键 `Ctrl+B ,`

• 窗格 pane

- **创建窗格**
 - **tmux split-window**
 - 创建水平分屏
 - `-h`
 - 创建垂直分屏
 - 快捷键 `Ctrl+B %` 创建垂直分屏
 - 快捷键 `Ctrl+B "` 创建水平分屏
- **切换窗格**
 - **tmux select-pane**
 - `-U -D -L -R`
 - 快捷键 `Ctrl+B 方向键` 切换不同窗格
 - 快捷键 `Ctrl+B ;` 切换上一窗格
 - 快捷键 `Ctrl+B O` 切换下一窗格
 - 快捷键 `Ctrl+B q`
 - 显示每个窗口的编号
 - 立即输入目标终端编号可切换到那个窗口
- **交换窗格**
 - **tmux swap-pane**
 - `-U` 向上移动

- -D 向下移动

- 快捷键 **Ctrl+B }** 交换上一窗格
- 快捷键 **Ctrl+B {** 交换下一窗格

- **调整窗格**

- 快捷键 **Ctrl+B Z** 放大当前窗格，再次操作缩小
- 快捷键 **Ctrl+B Ctrl+方向键** 移动当前窗格边缘
- 快捷键 **Ctrl+B Alt+方向键** 大幅移动当前窗格边缘
- 快捷键 **Ctrl+B T** 在当前窗格显示时钟，**Enter** 关闭

- **关闭窗格**

- **exit**
 - 输入 **exit** 杀死当前窗格
 - 可能快捷键 **Ctrl+D**
- **Ctrl+B x**
 - 关掉目标窗格

4 Conda

• 介绍

- Conda 是一个包管理器，其核心功能是包管理与环境管理。包管理与 pip 的使用类似，环境管理则允许用户方便地安装不同版本的 python 并可以快速切换
- Anaconda 是一个打包的集合器皿，里面预装好了 conda、某个版本的 python、众多 packages、科学计算工具等等，所以也称为 Python 的一种发行版
- Miniconda 只包含最基本的内容——python 与 conda，以及相关的必须依赖项，对于空间要求严格的用户，Miniconda 是一种更好的选择

• 下载与安装

- 进入清华镜像站：<https://mirrors.tuna.tsinghua.edu.cn/anaconda/miniconda/>
- 选择合适的版本，右键复制下载链接
- `wget -c /link`
- `bash sh_name` 或者直接 `./sh_name`
- 终端中输入以下命令，此后自动进入 base 环境
 - `source ~/.bashrc`
- 配置下载源

• 主要文件结构

- 在 conda 的安装路径中：
- **bin**
 - 存储 base 环境中安装的软件
- **pkgs**
 - 存储下载的所有包
- **env**
 - 存储新创建的环境的所有信息
 - env 目录下的所有文件夹名
 - `bin compiler_compat conda-meta include lib share ssl x86_64-conda_cos7-linux-gnu x86_64-conda-linux-gnu`
- conda 总目录下的所有文件夹名，不同的以粗体标识
 - `bin compiler_compat condabin conda-meta envs etc include lib LICENSE.txt man pkgs share shell ssl x86_64-conda_cos7-linux-gnu x86_64-conda-linux-gnu`

- 常用命令

- **conda info**

- 查看 conda 基本信息

- 环境

- **conda create -n *ename* python=*n***

- 创建一个名字为 *ename*, python 的版本号为 *n* 的新环境

- **conda info -e**

- 查看已存在的环境名

- **conda activate *ename***

- 切换到名字为 *ename* 的环境, 貌似环境之间会嵌套

- **conda deactivate**

- 退出当前环境

- **conda remove -n *ename* --all**

- 删除环境

- 下载源

- **conda config --show**

- 显示所有设定参数信息

- **conda config --show channels**

- 显示下载源设定参数信息

- **conda config --add channels *channel_name***

- 新增下载源

- 常用源有 bioconda、conda-forge、r、

- <https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/>等

- 新增的源的优先级为最高

- **conda config --remove channels *channel_name***

- 删除下载源, 可以用来调整优先级

- 包

- **conda install *pname***

- 安装包 *pname*

- **-c *channel_name***

- 用来指定下载源

- **--use-local** *path*
 - 用来安装本地已下载的包
- **conda install -n** *ename* *pname*
- **conda remove** *pname*
 - 删除包 *pname*
 - **conda remove-n** *ename* *pname*
- **conda list**
 - 查看已安装的包的列表
- **conda update** *pname*
 - 更新包 *pname*
- **conda search** *pname*
 - 查找包及其所属的源

• 克隆、复制与移植

- **conda create -n** *new_ename* **--clone** *old_ename*
 - 将老环境克隆成新环境
- **conda env export >** [environment.yml](#)
 - 生成名为 [environment.yml](#) 的文件，包含了当前环境下所有的软件包
- **conda env create -f** [environment.yml](#)
 - 根据 yml 文件准确复制原环境
- **conda-pack 打包**
 - 下载 conda-pack 包
 - **conda install -c conda-forge conda-pack**
 - 打包环境
 - **conda pack**
 - **-n** *ename*
 - 打包名为 *ename* 的环境
 - **-o** *filename.tar.gz*
 - 输出文件名为 *filename.tar.gz*
 - **-p** *path*
 - 输出路径为 *path*
 - 不能与-o 同时使用，若需更改文件名可在 *filename* 前加路径

- 在 `env` 下创建目录，名为 `new_ename`

- `tar -xzvf filename.tar.gz -C new_ename`

- 将打包 `filename.tar.gz` 文件解压到当前文件夹下的 `new_ename` 文件夹下

- 解压完毕后即移植完成

- 一般 `base` 环境无法被打包

5 BASH 基础

• 壳程序 (Shell)

• 定义

- 能够操作应用程序的软件，如 bash
- 包括图形用户界面模式的软件可称为广义的 shell

• Bourne Again Shell (bash)

- Bourne shell 的增强版本
- `/etc/shells` 文件储存系统上合法的 shell
- `/etc/passwd` 文件保存每个账号登录后取得的默认的 shell
- `~/.bash_history` 文件记录前一次登录前所执行过的命令，该次登录的命令缓存在内存中

• 登录

• 登录类型

- **login shell**
 - 取得 bash 时经过了完整的登录流程，需要输入账号与密码
 - 读取 `/etc/profile` 和用户的个人配置文件两个文件
 - 个人配置文件依次读取下列文件中的第一个：`~/.bash_profile`、`~/.bash_login`、`~/.profile`
- **non-login shell**
 - 取得 bash 时不重复登录的动作，如 `bash` 命令
 - 仅读取 `~/.bashrc` 文件

• 文件

• `/etc/profile`

- 系统整体的设置，是每个用户登录 bash 时一定会读取的配置文件
- 主要设置 PATH、MAIL、USER、HOSTNAME、HISTSIZE、umask
- 并依序调用：

• `/etc/profile.d/*.sh`

- 一系列文件，包括 bash 界面的颜色、语系、别名等
- 可在该目录下自行建立文件，用于设置所有用户的共享别名

• `/etc/locale.conf`

- 由 `/etc/profile.d/lang.sh` 调用，决定 bash 默认使用的语系
- `/usr/share/bash-completion/completions/*`

- 由`/etc/profile.d/bash-completion.sh`调用，用于命令的选项、参数补齐
- `~/.bash_profile`
 - 与`~/bash_login`、`~/.profile`等价
 - 设置 PATH 变量
 - 因为`/etc/profile`已经调用设置了 PATH 变量，这里是用冒号加进去的
 - 调用存在的`~/.bashrc`
- `~/.bashrc`
 - 规范安全的别名等使用者的个人设置
 - 调用`/etc/bashrc`去进行整体的环境配置
- `/etc/bashrc`
 - CentOS 特有的文件，能根据 UID 不同设置 umask 和 PS1，并调用`/etc/profile.d/*.sh`的设置
 - 华为云没有该文件，似乎直接把这个功能写在`~/.bashrc`中了
- `~/.bash_logout`
 - 记录注销 bash 后，系统离开前应做的命令

- 命令

- `type`
 - 查询命令是否为 Bash shell 的内置命令
 - 不加任何参数时，type 会显示是外部命令还是 bash 内置命令
- `-t`
 - 以 file (外部命令)、alias (命令别名名称)、builtin (bash 内置) 显示来源
- `-p`
 - 接外部命令时才显示完整文件名
- `-a`
 - 会由 PATH 定义的路径中，把所有含 name 的命令都列出来，包含 alias

- 命令查找顺序

- 以相对/绝对路径执行命令
- 由 alias 找到该命令执行
- 由 bash 内置命令执行
- 通过\$PATH 变量的顺序查找到第一个命令执行

- 变量

- **echo \$var_name**

- 读出变量的值，值前面得加\$
- 也可以 **echo \${var_name}**

- **设置与更改**

- **var_name=value**

- 用=连接变量名与变量内容
 - 变量名由英文字母与数字构成，首字符不能为数字

- **unset var_name**

- 删除变量

- **read var_name**

- 读取来自键盘输入的变量进入 *var_name*
 - -p
 - 后接提示字符
 - -t
 - 后接等待秒数

- **变量类型：默认字符串**

- **declare var_name**

- 将 *var_name* 定义成不同的指定类型
 - -a: 数组类型
 - -i: 整数类型
 - -x: 环境变量
 - -r: 只读，不能更改与 unset
 - +[aixr]: 取消设定的属性
- bash 中的运算默认最多只能到达整数形态

- **变量内容**

- 变量内容不能有空格，但可以有特殊字符
 - 若有空格，可以使用' 或" "结合起来，也可使用 \ 协助转义
 - 若有特殊字符，可以使用' '结合起来，也可使用 \ 协助转义
 - 双引号内的特殊字符可以保持原有的特性，单引号内的特殊字符仅为纯文本

- **内部执行**

- 在命令执行的过程中，如果需要借助其他命令的输出作为协助，除了管道符之外还可以使用
 - `command`
 - 反单引号
 - `$(command)`

- **变量内容的变更**

- **扩增内容**

- 若在变量内部使用其他变量或自身，可以通过`$var_name`、 `${var_name}`、`"$var_name"`
 - 推荐使用后两个，否则会造成名称混淆

- **删除**

- `${var_name##word}`
 - 若变量 `var_name` 从头开始符合关键词 `word`，则删除符合的最短数据
- `${var_name###word}`
 - 若变量 `var_name` 从头开始符合关键词 `word`，则删除符合的最短数据

- **例**

- (base) leb12a@bbt:~\$ **echo \$cut**
- **#cuttd**
- (base) leb12a@bbt:~\$ **echo \${cut##*t}**
- **d**
- (base) leb12a@bbt:~\$ **echo \${cut#"##*t}**
- **td**
- 如上，删除 `cut` 变量的从#到 t 的片段，且为最短的#cut
- *代表通配符，能够匹配 0 至无穷个字符
- 特殊字符使用双引号引住，这样就不会识别成##从后删除

- `${var_name%word}`
 - 若变量 `var_name` 从尾开始符合关键词 `word`，则删除符合的最短数据
- `${var_name%%word}`
 - 若变量 `var_name` 从尾开始符合关键词 `word`，则删除符合的最长数据

- **替换**

- `${var_name//word1/word2}`
 - 将变量 `var_name` 从头开始第一个 `word1` 替换成 `word2`

- `${var_name//word1/word2}`
 - 将变量 `var_name` 所有 `word1` 替换成 `word2`

- **环境变量与自定义变量**

- 子进程继承父进程的环境变量，但不继承自定义变量
- **export var_name**
 - 使变量变为环境变量
- **env**
 - 输出目前 shell 环境中所有环境变量
 - **HOME**
 - 根目录
 - 推测：~似乎代表着\$HOME
 - **SHELL**
 - 目前环境使用的 shell 是哪个程序，linux 默认/bin/bash
 - **HISTSIZE**
 - 曾经执行过的命令能保存多少条
 - `~/.bash_history` 文件记录前一次登录前所执行过的命令，该次登录的命令缓存在内存中
 - **PATH**
 - 执行文件查找的路径，依序查询
 - 不同目录间以：分隔
 - **LANG**
 - 储存语系数据
 - **locale**
 - 查看自己语系变量的设置
 - **-a**
 - 查看 Linux 所支持的语系
- **set**
 - 显示所有变量，含环境变量、操作界面变量、自定义变量
 - **PS1**
 - 命令提示字符的格式
 - **\$**

- 目前 shell 的进程号，即 PID

- ?

- 上个执行命令的返回值
- 成功返回 0，错误返回错误代码

- **终端环境设置**

- \Enter

- 转义 Enter 字符以换行输入，需要紧贴着

- **光标设置**

- Ctrl + u

- 从光标向前删除命令串

- Ctrl + k

- 从光标向后删除命令串

- Ctrl + a

- 将光标移动至整个命令串最前面

- Ctrl + e

- 将光标移动至整个命令串最后面

- **bash 默认组合键**

- Ctrl+C

- 终止目前的命令

- Ctrl+Z

- 暂停目前的命令

- Ctrl+D

- 输入结束、EOF

- Ctrl+M

- 回车

- Ctrl+S

- 暂停屏幕的输出

- Ctrl+Q

- 恢复屏幕的输出

- **按键设置**

- stty

- **-a**
 - 列出目前环境所有按键列表
 - ^代表 **Ctrl**
- **stty** *comm key*
 - *key*代表需要更改为的按键
 - *comm*为进行的操作
 - **intr**
 - 发送中断信号给目前正在运行的程序
 - **quit**
 - 发送 quit 信号给目前正在运行的程序
 - **erase**
 - 向后删除字符
 - 如果按下回车出现^H 或^?，可使用 **stty erase ^H** 或者 **stty erase ^?**命令恢复
 - ^?命令一般无法用键盘打出，只是 **Backspace** 的记号
 - **eof**
 - 结束输入
 - **kill**
 - 删除命令行上的所有文字

• 数据流

- **重定向：类似于指针**
 - **>**
 - 默认为 **1>**，将标准输出重定向**覆盖**至指定的文件或设备上
 - **>>**
 - 默认为 **1>>**，将标准输出重定向**累加**至指定的文件或设备上
 - **2>**
 - 将标准错误输出重定向**覆盖**至指定的文件或设备上
 - **2>>**
 - 将标准错误输出重定向**累加**至指定的文件或设备上
 - **> /dev/null** 或 **2> /dev/null**
 - 黑洞设备 /dev/null，吸收所有导向的信息

- **tee file**

- 将 STDIN 同时输入至 *file* 和 STDOUT 中，用于管道与双重管道
- **-a**
 - 以累加的方式输入 *file*

- **2>&1**

- 将标准错误输出重定向至标准输出所指向的位置

- **1>&2**

- 将标准输出重定向至标准错误输出所指向的位置

- **>/dev/null 2>&1**

- 标准输出和错误输出都进了“黑洞”

- **2>&1 >/dev/null**

- 是标准输出进了黑洞，错误输出打印到屏幕

- **<**

- 将键盘输入改由文件输入

- **<<**

- 后接结束字符，可以自定义，多个字符需用引号

- **相关性的命令执行**

- **com1 && com2**

- 若 *com1* 执行完毕且为正确 ($$?=0$)，开始执行 *com2*
- 若 *com2* 执行完毕且为错误 ($$?\neq0$)，不执行 *com2*，同时将 $$?\neq0$ 传递到后项

- **com1 || com2**

- 若 *com1* 执行完毕且为正确 ($$?=0$)，不执行 *com2*，同时将 $$?=0$ 传递到后项
- 若 *com2* 执行完毕且为错误 ($$?\neq0$)，开始执行 *com2*

6 shell 脚本

• 执行方法

- 在子进程中执行：
 - 直接使用绝对路径、相对路径或放在 PATH 下面执行，**shell.sh** 需要具备 rx 的权限
 - 以 bash 程序来执行：**bash** 或 **sh** 命令，**shell.sh** 需要具备 r 的权限
 - /bin/sh 就是/bin/bash（链接文件）
- 在父进程中执行：
 - **source** 执行脚本，在原本的 bash 内生效

• shebang 行

- **#!/bin/bash**
 - 通常出现在 linux 的 shell 脚本第一行，作为解释行，告诉解释器 shell 的执行方式
 - 第一个字符非#：表示这是一个 bash 脚本。
 - 第一个字符是#，但第二个字符非!，表示这是一个 csh 脚本。
 - 第一个字符是#，且第二个字符是!，则调用其后指定的程序来执行以下的这个脚本
 - 如果用 **bash** 命令指定解释器，则无需 shebang 行

• 默认变量

- **\$0**
 - 脚本文件名
- **\$1、\$2...**
 - 后接的参数，第一个为 1，以此类推
 - 似乎不用在写程序时选定参数个数
- **\$#**
 - 输入的参数总个数
- **\$@**
 - 代表所有变量独立的连起来，每个变量以双引号括着
- **\$***
 - 代表所有变量连起来，以分隔符隔开，默认为空格
- **shift num**
 - 把前 num 个变量删掉，默认 num 为 1
 - 之后的变量向前递补

- **exit num**

- 程序中断，返回 *num* 给 \$? 变量

- **数值运算**

- **var=\$(())**

- 小括号内可以加上空格符

- **declare -i result=\$vara..\$varb**

- 运算式中不能加入空格

- **bc**

- **命令行模式**

- 输入 **bc** 进入计算器模式，输入 **quit** 退出

- **管道模式**

- 接在双引号括着的字符串后面

- **echo "15+5" | bc**

- **scale**

- 设置小数位

- **\$ echo 'scale=2; (2.777 - 1.4744)/1' | bc**

- **1.30**

- **ibase**

- 设置 input 进制

- **echo "ibase=2;111" |bc**

- **7**

- **obase**

- 设置 output 进制

- **判断式**

- **test**

- 执行结果不会展示任何信息，但可以通过 \$?、&&、|| 展现结果

- **文件**

- **-e**: 该文件名是否存在

- **-f**: 该文件名是否存在且为文件

- **-d**: 该文件名是否存在且为文件夹

- **权限**

- **-r**: 该文件名是否存在且可读
- **-w**: 该文件名是否存在且可写
- **-x**: 该文件名是否存在且可执行
- **-s**: 该文件名是否存在且非空

- **文件比较**

- **-nt**: file1 是否比 file2 新
- **-ot**: file1 是否比 file2 旧
- **-ef**: file1 是否和 file2 是同一个文件 (指向一个 node)

- **整数比较**

- **-eq**: 两数值相等
- **-ne**: 两数值不等
- **-gt**: n1 大于 n2
- **-lt**: n1 小于 n2
- **-ge**: n1 大于等于 n2
- **-le**: n1 小于等于 n2

- **字符串判定**

- **-z**: 字符串是否为空字符串
- **-n**: 字符串是否不为空字符串
- **==**: 两字符串是否相等
- **!=**: 两字符串是否不相等

- **多重条件判定**

- **-a**: 同时成立
- **-o**: 任意一个成立
- **!**: 反相

- **[]**

- 几乎与 **test** 相同，常用于 **if**
- 判断符号内的所有组件需要以空格分隔
- 变量最好以双引号括起来，以防里面有空格
- 常数最好以引号括起来

- **函数**

- **格式**

- **function** *fname*() {
 - ...
 - }
- 一般放在最前
 - 内置变量格式与 shell 脚本类似，\$0 为函数名，之后为后接变量

• 条件判断

- 条件判断式为中括号或多个中括号以 && 或 || 相连

• if-then

- **if** 条件判断式; **then**

- ...

- **fi**

• 复杂情况

- **if** 条件判断式; **then**

- ...

- **elif** 条件判断式; **then**

- ...

- **else**

- ...

- **fi**

• case

- 判断变量 var 内容为 A 或 B 的情况

- 每个内容建议括双引号，后接关键字)

- 类别结尾以;;处理

- *)代表其他值

- **case** \$var **in**

- "A")

- ...

- ;;

- "B")

- ...

- ;;

- *)
- ...
- ;;
- esac

- **循环**

- **while**
 - **while** 条件判断
 - **do**
 - ...
 - **done**
- **until**
 - **until** 条件判断
 - **do**
 - ...
 - **done**
- **for**
 - **for var in con1 con2 con3 ..**
 - **do**
 - ...
 - **done**
 - 可以使用{a..d}或者\$(seq 1 100)等方式创建连续
- **for**
 - **for ((初始值; 限制值; 复制运算))**
 - **do**
 - ...
 - **done**
 - 类似于 c, **for ((i=1; i<3; i++))**

- **调试**

- **sh**
 - 默认执行脚本
 - **-n**: 不执行脚本, 仅查询语法问题

- **-v**: 执行脚本前输出脚本内容
- **-x**: 将使用到的内容输出到屏幕

7 Github

• 版本控制系统 (VCSs)

- 版本控制是一种记录一个或若干文件内容变化，以便将来查阅特定版本修订情况的系统
- 例子 1：版本控制就是你写了一个文件，过了几天后，你对其进行修改，但是你希望把原始的版本保留，同时保留修改后的，这其实就是一个版本控制
- 例子 2：想象一下你正在对某个重要的图片进行调整，例如你想看看只调饱和度的效果，但是你还希望看看只调整对比度的效果，这个时候，你就需要保存这一幅图片没修改、只调整饱和度和只调整对比度这三个修订版本，但是当问题变得复杂，你需要看看多种处理组合对图片的影响，并且对每种操作及操作后的图片进行记录保存，这其实就是一个版本控制的过程
- 一般而言，很多人习惯用复制整个项目目录的方式来保存不同的版本，或许还会改名加上备份时间以示区别。这么做唯一的好处就是简单，但是特别容易犯错。因此，就有人开发了各种版本控制系统 (Version Control Systems, VCSs) 。
- 最流行的一种**本地端的版本控制系统**叫做 RCS，现今许多计算机系统上都还看得到它的踪影。RCS 的工作原理是在硬盘上保存补丁集（补丁是指文件修订前后的变化）；通过应用所有的补丁，可以重新计算出各个版本的文件内容。
- **集中化的版本控制系统 (CVCS)**，诸如 CVS、Subversion 以及 Perforce 等，都有一个单一的集中管理的服务器，保存所有文件的修订版本，而协同工作的人们都通过客户连到这台服务器，取出最新的文件或者提交更新。
- 但很显然的一点是当中央服务器出现故障。那么在故障的这段时间内，谁都无法上传更新，无法协同工作。如果中心数据库所在的磁盘发生损坏，又没有做恰当备份，毫无疑问你将丢失所有数据——包括项目的整个变更历史，只剩下人们在各自机器上保留的单独快照。
- **分布式版本控制系统 (DVCS)** 像 Git、Mercurial、Bazaar 以及 Darcs 等，客户端并不只提取最新版本的文件快照，而是把代码仓库完整地镜像下来，包括完整的历史记录。这么一来，任何一处协同作用的服务器发生故障，事后都可以用任何一个镜像出来的本地仓库恢复。因为每一次的克隆操作，实际上都是一次对代码仓库的完整备份。

• Git 概述

- Git 它不仅仅是个版本控制系统，它也是个内容管理系统 (CMS)，工作管理系统等
- 它是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件
- 作为一个开源的分布式版本控制系统，它被用于敏捷高效地处理任何或小或大的项目
- 与常用的版本控制工具 CVS, Subversion 等不同，它采用的分布式版本库的方式，不必服务器端软件支持，使源代码的发布和交流极其方便
- **Git 和其它版本控制系统（包括 Subversion 及其他近似工具）的主要差别在于 Git 对待数据的方式**

- 从概念上来说，其它大部分系统以文件变更列表的方式存储信息，这类系统（CVS、Subversion、Perforce、Bazaar 等等）将它们存储的信息看作是一组基本文件和每个文件随时间逐步累积的差异（它们通常称作基于差异的版本控制）
- Git 不按照以上方式对待或保存数据。反之，Git 更像是把数据看作是对小型文件系统的一系列快照
- 在 Git 中，每当你提交更新或保存项目状态时，它基本上就会对当时的全部文件创建一个快照并保存这个快照的索引。为了效率，如果文件没有修改，Git 不再重新存储该文件，而是只保留一个链接指向之前存储的文件。**Git 对待数据更像是一个快照流。**

• 三种文件状态

- 已提交 (**committed**)
 - 数据已经安全地保存在本地数据库中
- 已修改 (**modified**)
 - 修改了文件，但还没保存到数据库中
- 已暂存 (**staged**)
 - 对一个已修改文件的当前版本做了标记，使之包含在下次提交的快照中

• 安装 Git

- sudo dnf install git-all**
- git clone git://git.kernel.org/pub/scm/git/git.git**
 - 使用 git 来获取 git 的更新

• 配置 Git

- 初次运行 Git 前需要定制 Git 环境
- Git 自带一个 git config 的工具来帮助设置控制 Git 外观和行为的配置变量，存储在
 - /etc/gitconfig** 文件：包含系统上每一个用户及他们仓库的通用配置
 - ~/.gitconfig** 文件：只针对当前用户
 - 当前使用仓库的 Git 目录中的**.git/config**：针对该仓库
 - 每一个级别会覆盖上一级别的配置
- git config --list --show-origin**
 - 查看 Git 当前仓库的所有的配置以及它们所在的文件
- 设置用户名和邮件地址**
 - git config --global user.email "..."**
 - git config --global user.name "..."**
 - 针对特定项目使用不同的用户名与邮件地址时，可以在项目目录下运行没有 --global 选项的命令来配置

• 获取 Git 仓库

- 在已存在的目录中初始化仓库

- **git init**

- 在该目录中创建一个名为 `.git` 的子目录，含有初始化的 Git 仓库中所有的必须文件

- 克隆现有的仓库

- **git clone <url> name**

- 在当前目录下创建一个名为 `name` (可选) 的目录，在这个目录下初始化一个 `.git` 文件夹

• 记录更新

- **git status**

- 查看哪些文件处于什么状态，展示当前目录下任何处于未跟踪与修改状态的新文件

- Changes to be committed: 这行下面的文件为已暂存状态

- **-s**

- 缩短状态命令的输出

- ?? : 未跟踪文件

- A : 新添加到暂存区中的文件

- M : 修改过的文件

- 左栏指明了暂存区的状态，右栏指明了工作区的状态

- **git add file**

- 开始跟踪 `file` 文件，或者把已跟踪的文件放到暂存区

- 也可用于合并时把有冲突的文件标记为已解决状态

- 如果是目录的路径，该命令将递归地跟踪该目录下的所有文件

- **git commit -m "..."**

- 提交放在暂存区域的快照

- **-m** 后写提交说明，也可不加**-m** 在 vim 编辑器中输入提交说明

- **-a** 可自动 add 所有已经跟踪过的文件

- **--amend** 可以新的提交覆盖上一次提交，用于提交忘记交的文件

- **git rm file**

- 移除文件，下一次提交时，该文件不再纳入版本管理

- **-f**

- 强制删除之前修改过或已经放到暂存区的文件，防止误删尚未添加到快照的数据

- **--cached**

- 从暂存区域移除，但仍然希望保留在当前工作目录中

- **git reset HEAD *file***

- 从暂存区中取消这次修改的文件，用于撤销，将两次更改分两次 commit

- **git mv *file_from* *file_to***

- 重命名文件

- 若用其他方式改名，注意先 rm 后 add

- **git log**

- 按时间先后顺序列出所有的提交，最近的更新排在最上面

- **git checkout -- *file***

- 撤消对文件的修改，还原成上次提交时的样子

- **远程仓库**

- **git remote**

- 列出指定的每一个远程服务器的简写

- **-v**

- 显示需要读写远程仓库使用的 Git 保存的简写与其对应的 URL

- **add *name* <*url*>**

- 添加一个新的远程 Git 仓库，简写为 *name*, 链接为 *url*

- **show <*remote*>**

- 列出远程仓库的 URL 与跟踪分支的信息

- **rename *name* *newname***

- 重命名远程仓库 *name* 的简写

- **remove <*remote*>**

- 移除一个远程仓库，删除和这个远程仓库相关的远程跟踪分支以及配置信息

- **git fetch <*remote*>**

- 访问远程仓库，从中拉取所有你还没有的数据

- **git pull**

- 自动抓取远程分支后合并到当前分支，通常会从最初克隆的服务器上抓取数据并自动尝试合并到当前所在的分支

- **git push <*remote*> <*branch*>**

- 推送到上游，本地远程分支名不同用冒号隔开

ChIP-seq 流程笔记

1. 数据来源

参考基因组及注释文件

- 常用的参考基因组下载地址有：
 - Ensembl: <ftp://ftp.ensembl.org/pub>
 - EnsemblGenomes: <ftp://ftp.ensemblgenomes.org/pub/>
 - NCBI: <ftp://ftp.ncbi.nih.gov/genomes/>
 - UCSC: <ftp://hgdownload.soe.ucsc.edu/goldenPath>
- 如打开<https://ftp.ensembl.org/pub/>, 操作流程如下
 - 找到编号最大最新的release文件夹, 选择想要下载的数据格式fasta/gtf/gff3
 - 找到目的动物名, 点进去下载目的文件
 - 参考基因组类型
 - TOPOLEVEL: 包含所有染色体序列、未组装到染色体序列和用N填充的单倍型/补丁区域
 - PRIMARY ASSEMBLY: 用于序列比对的最完善的基因组, 去除了单倍型/补丁区域
 - 若没有'primary_assembly'文件, 'toplevel'文件具有相同的效用
 - toplevel版本会包含haplotype信息, 多余的信息会增加比对工具的工作
 - rm_masked: 将所有重复区和低复杂区用N代替, 比对时不会有reads比对到这些区域。因为它造成了信息的丢失, 导致uniquely比对到masked基因组上的reads实际上可能不是unique的; 还会在允许错配的情况下, 本来来自重复区的reads比对到基因组的其它位置。
 - soft-masked: 把所有重复区和低复杂区的序列用小写字母标出。主要的比对软件, 比如BWA、bowtie2等都忽略这些soft-mask, 直接把小写字母当做大写字母比对, 所以使用soft-masked基因组的比对效果和使用unmasked基因组的比对效果是相同的
 - 若下载的为染色体的.fa文件, 可使用 `cat *.fa > merge.fa` 合并至一个fasta文件
 - gtf类型
 - .chr.gtf.gz 只包含主要染色体 (有MT) 的注释
 - .gtf.gz 包含主要染色体以及一些没有定位的染色体片段上的注释信息
 - .chr_patch_hapl_scaff.gtf.gz 除了上面的, 还有patches和ALT contig的注释
 - .ab initio后缀的GTF文件包括了Genscan和其他工具预测得到的注释信息

ChIP-seq测序结果

- 阴性对照
 - input: 样本经过超声, 但是没有进行ChIP, 包含样本超声后总DNA。开头进行电泳, 检测超声效果。同时也可以与最后ChIP样本进行比较, 判断ChIP的效果
 - 如果用同一引物进行PCR, ChIP组和input组亮度差不多, 说明ChIP效率高, 样本中所有的目的基因片段都被ChIP下来了, 反之说明效率低, 实验条件有待改进
 - IgG: 用普通的IgG做为抗体, 理论上不会ChIP下来任何DNA片段, 但是由于非特异结合, 或者实验过程中, 没有发生结合的DNA清除不完全, 可能也会出现条带

- 通常使用IgG和input两种control结合，推荐使用多个不同的IgG作为对照，也有只用input作为对照的
- 阳性对照
 - 一般用anti-RNA polymerase II抗体，因为RNA polymerase II是通用转录因子，在所有细胞中都能结合基因的核心启动子区，因此理论上，ChIP后PCR会有条带。一般阳性对照不进行测序
- 本文使用的示例数据存放在chipseq文件夹中，其中又包含human_genome和ctcf_chip两个文件夹，前者存放基因组数据，后者存放测序数据

2. 质控与过滤

FastQC

- 参考<https://zhuanlan.zhihu.com/p/20731723>
- FastQC是一款基于Java的软件，一般都是在linux环境下使用命令行运行，它可以快速多线程地对测序数据进行质量评估

```
# 基本格式

# fastqc [-o output dir] [--(no)extract] [-f fastq|bam|sam] [-c contaminant file]
seqfile1 .. seqfileN

# -o --outdir FastQC生成的报告文件的储存路径，生成的报告的文件名是根据输入来定的
# --extract 生成的报告默认会打包成1个压缩文件，使用这个参数是让程序不打包
# -t --threads 选择程序运行的线程数
# -c --contaminants 污染物选项，输入的是一个文件，格式是Name [Tab] Sequence，里面是可能的污染序列，如果有这个选项，FastQC会在计算时候评估污染的情况，并在统计的时候进行分析，一般用不到
# -a --adapters 也是输入一个文件，文件的格式Name [Tab] Sequence，储存的是测序的adpater序列信息，如果不输入，目前版本的FastQC就按照通用引物来评估序列时候有adapter的残留
# -q --quiet 安静运行模式，一般不选这个选项的时候，程序会实时报告运行的状况。
```

- 示例

```
fastqc -t 20 -o fastqc_result/ ctcf_chip/ctcf_chip_rep1_r1.fastq.gz

# fastqc_result文件夹已提前创建来存放fastqc结果，可同步做多个测序文件的质控
# 若想存储日志文件，可在命令后加上> fastqc_result/fastqc.log 2>&1
# 若想后台运行，可在命令后加上 &
```

运行后，出现 `ctcf_chip_rep1_r1_fastqc.html` `ctcf_chip_rep1_r1_fastqc.zip` 两个文件

- html文件解读
 - **基本信息** Basic Statistics

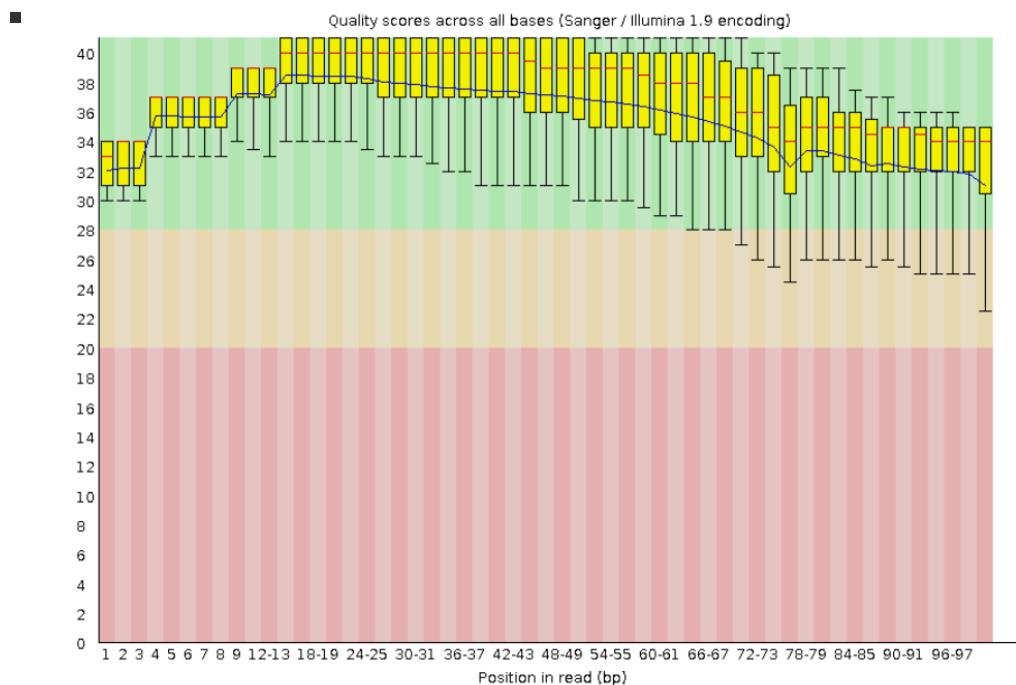


Basic Statistics

Measure	Value
Filename	ctcf_chip_rep1_r1.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	41173691
Sequences flagged as poor quality	0
Sequence length	101
%GC	43

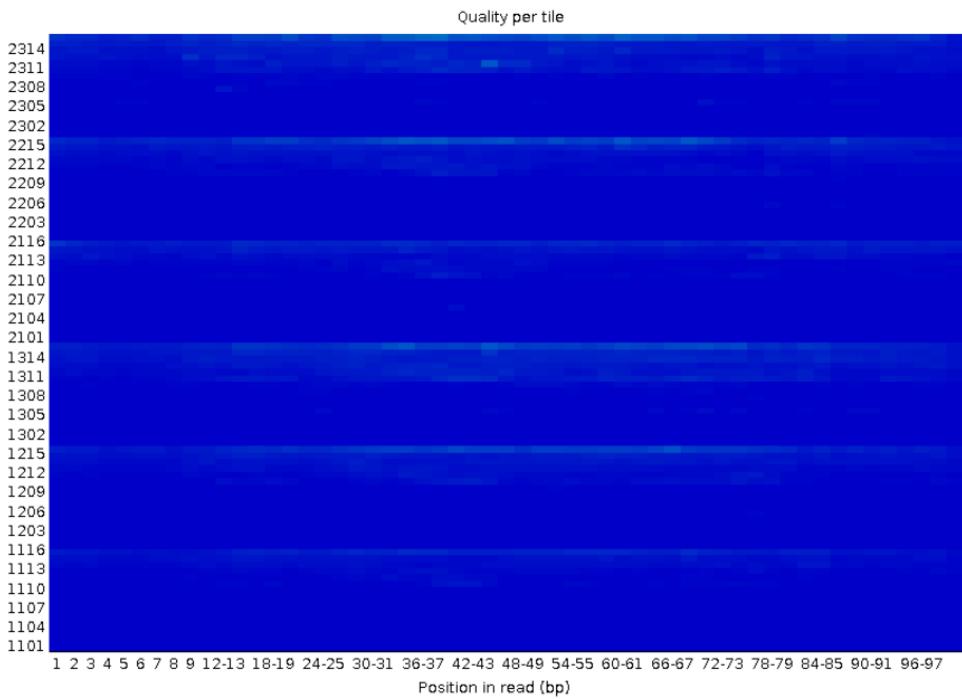
- # Encoding 指测序平台的版本和相应的编码版本号，这个在计算Phred反推error P的时候有用
- # Total Sequences 记录了输入文本的reads的数量
- # Sequence length 是测序的长度
- # %GC 是需要重点关注的一个指标，这个值表示的是整体序列中的GC含量，这个数值一般是物种特异的，比如人类细胞就是42%左右

- 序列测序质量统计 Per base sequence quality



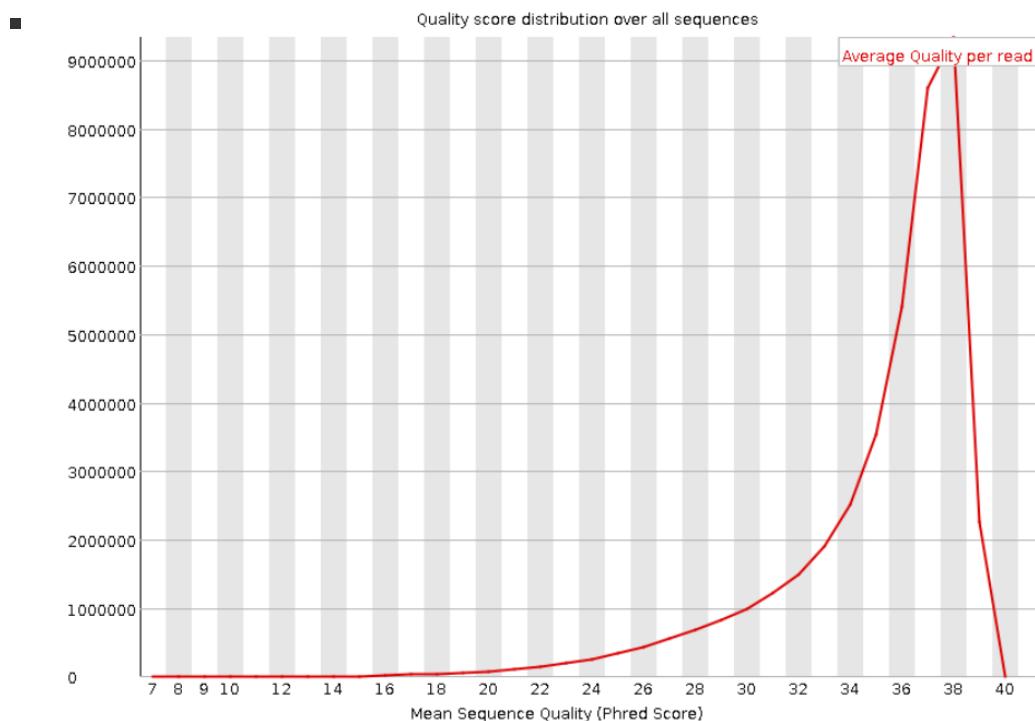
- # 横轴是测序序列的碱基位置
- # 纵轴是质量得分， $Q = -10 \times \log_{10}(\text{error } P)$ 即20表示1%的错误率，30表示0.1%
- # boxplot为该位置的所有序列的测序质量的统计
- # 蓝色的细线是各个位置的平均值的连线
- # 一般要求此图中，所有位置的10%分位数大于20，也就是我们常说的Q20过滤
- # Warning 报警 如果任何碱基质量低于10，或者是任何中位数低于25
- # Failure 报错 如果任何碱基质量低于5，或者是任何中位数低于20

- 每个tile测序的情况 Per tile sequence quality



- # 纵轴为tail的Index编号
这个图主要是为了防止，在测序过程中，某些tail受到不可控因素的影响而出现测序质量偏低
蓝色代表测序质量很高，暖色代表测序质量不高，如果某些tail出现暖色，可以在后续分析中把该tail测序的结果全部都去除

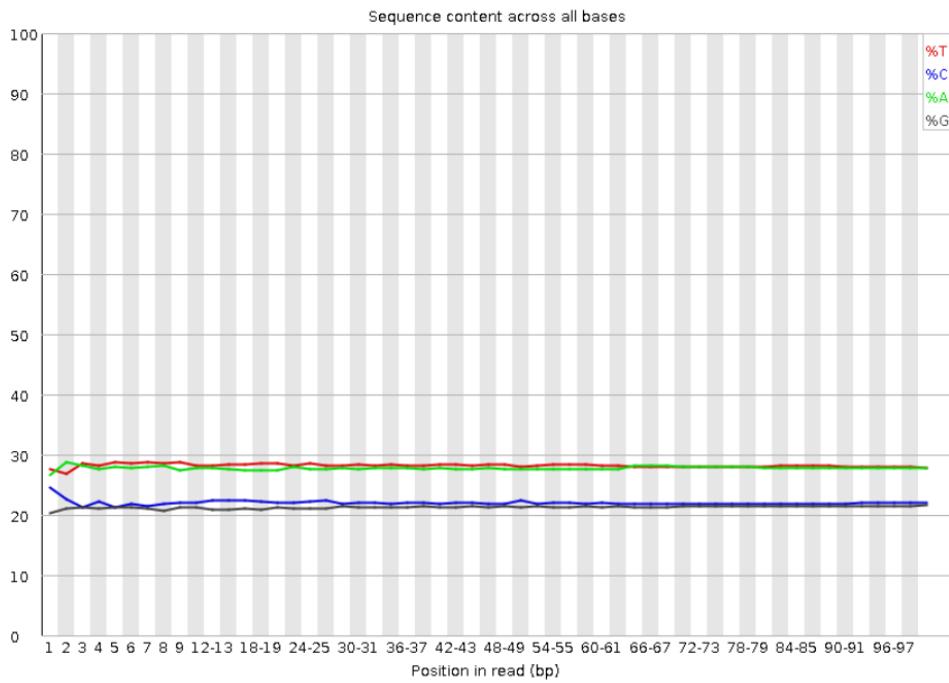
○ 每条序列的测序质量统计 Per sequence quality scores



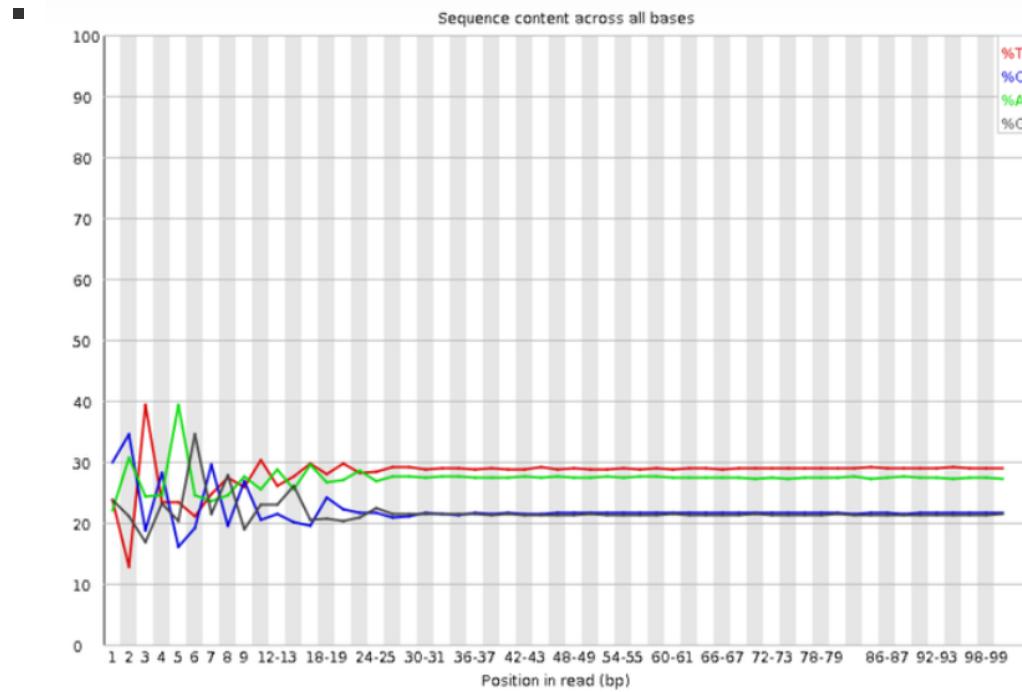
- # 横轴是单个reads每个位置Q之的平均值就是这条reads的质量值
纵轴是每个值对应的reads数目

○ GC 含量统计 Per base sequence content

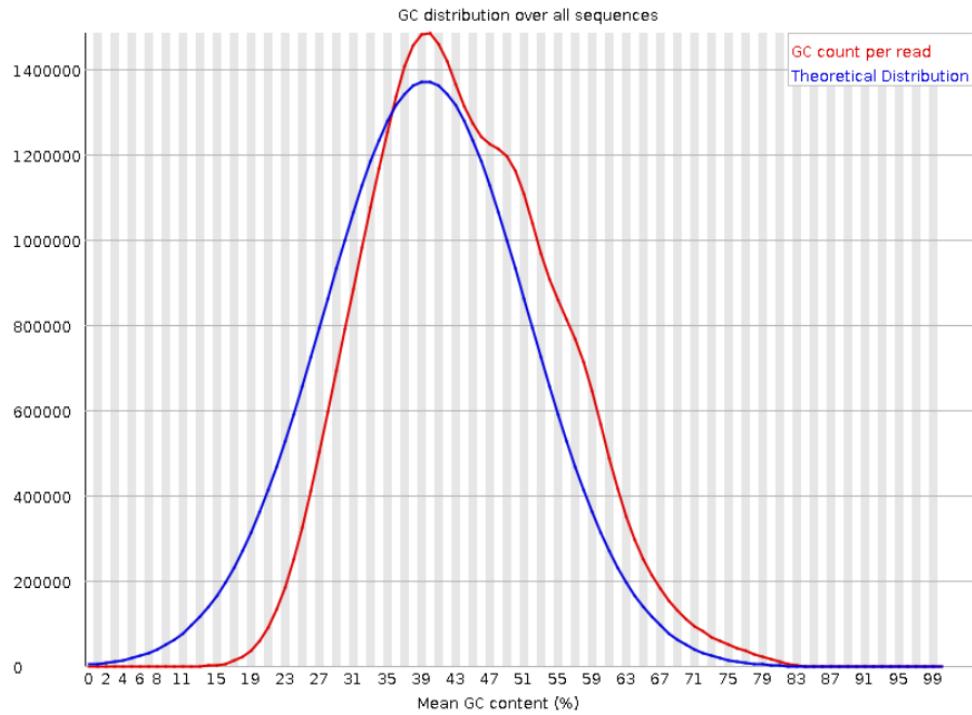




- 图中四条线代表A T C G在每个位置平均含量，理论上来说，A和T应该相等，G和C应该相等
- # 一般测序的时候，刚开始测序仪状态不稳定，很可能出现下图的情况。像这种情况，即使测序的得分很高，也需要cut开始部分的序列信息，可以cut前面5bp

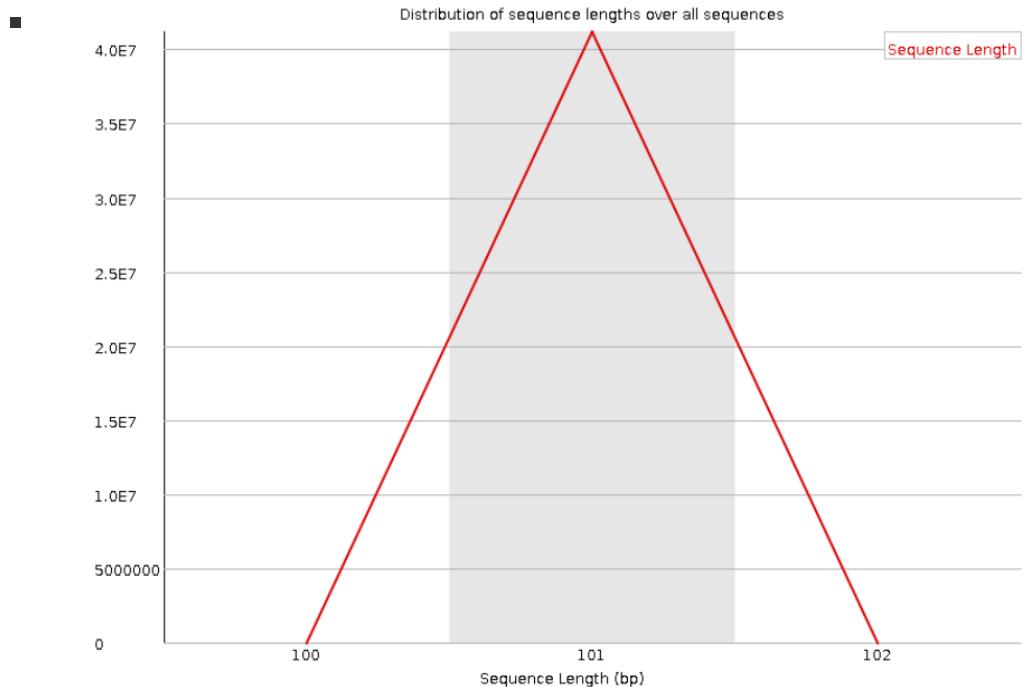


- 序列平均GC含量分布图 Per sequence GC content



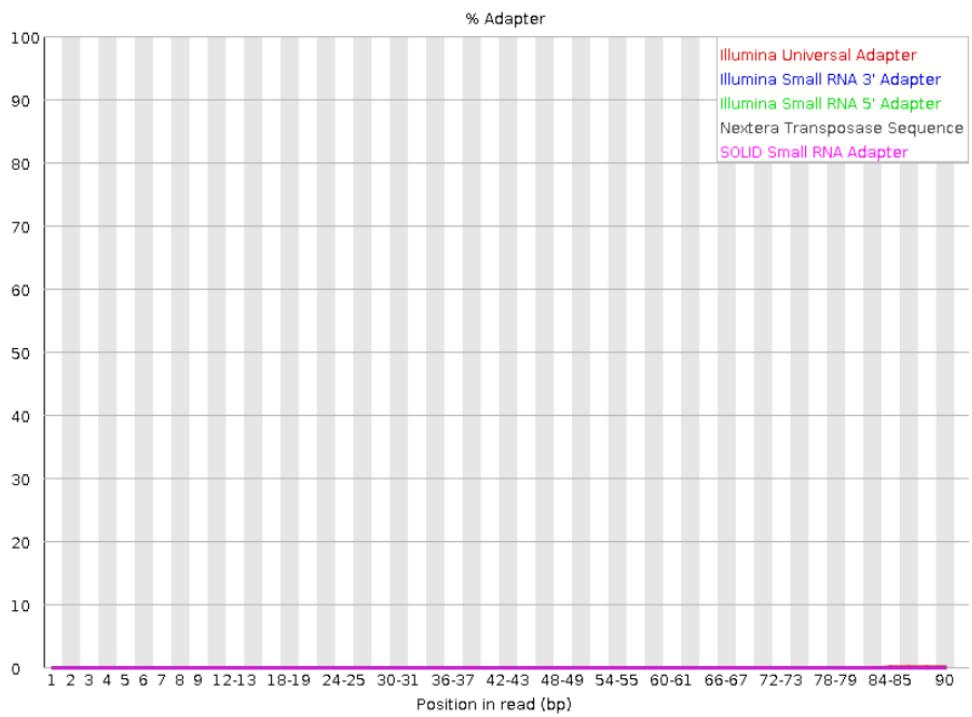
- # 蓝色的线是程序根据经验分布给出的理论值，红色是真实值，两个应该比较接近才比较好
- # 当红色的线出现双峰，是混入了其他物种的DNA序列，之后比对不上的序列会被扔掉，影响不大
- # GC content 偏离正态分布在ChIP-Seq里是正常的

○ 序列测序长度统计 Sequence Length Distribution



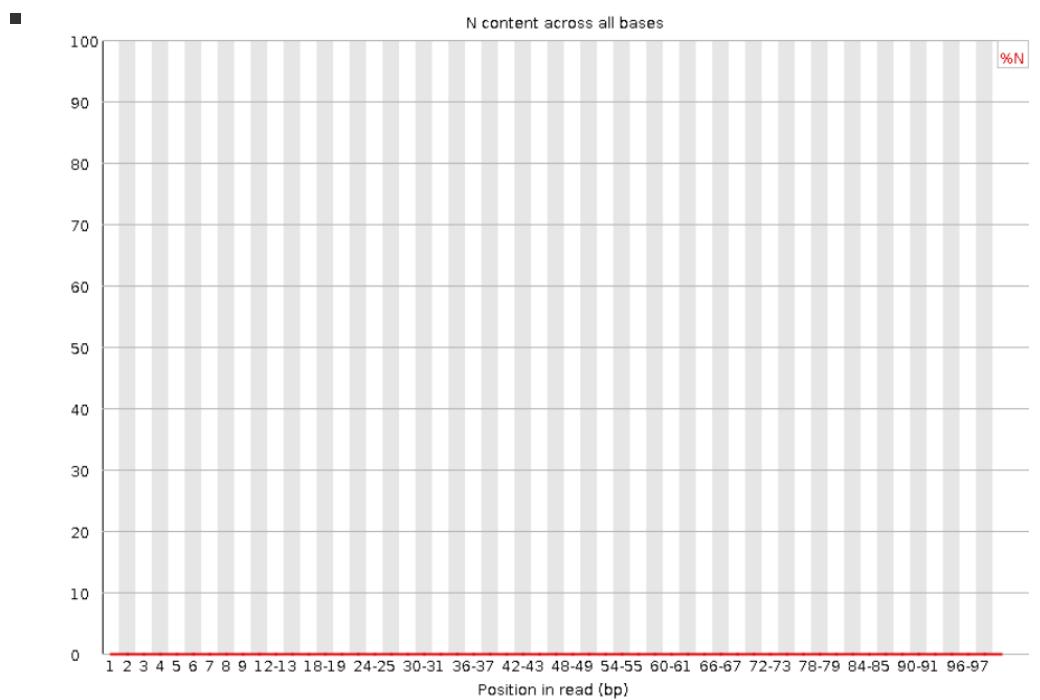
- # 每次测序仪测出来的长度在理论上应该是完全相等的，但是总会有一些偏差
- # 比如此图中，101bp是主要的，但是还是有少量的100和102bp的长度，不过数量比较少，不影响后续分析
- # 当测序的长度不同时，如果很严重，则表明测序仪在此次测序过程中产生的数据不可信

○ 序列Adapter Adapter Content



- # 如果-a选项没有内容，则默认使用图例中的四种通用adapter序列进行统计

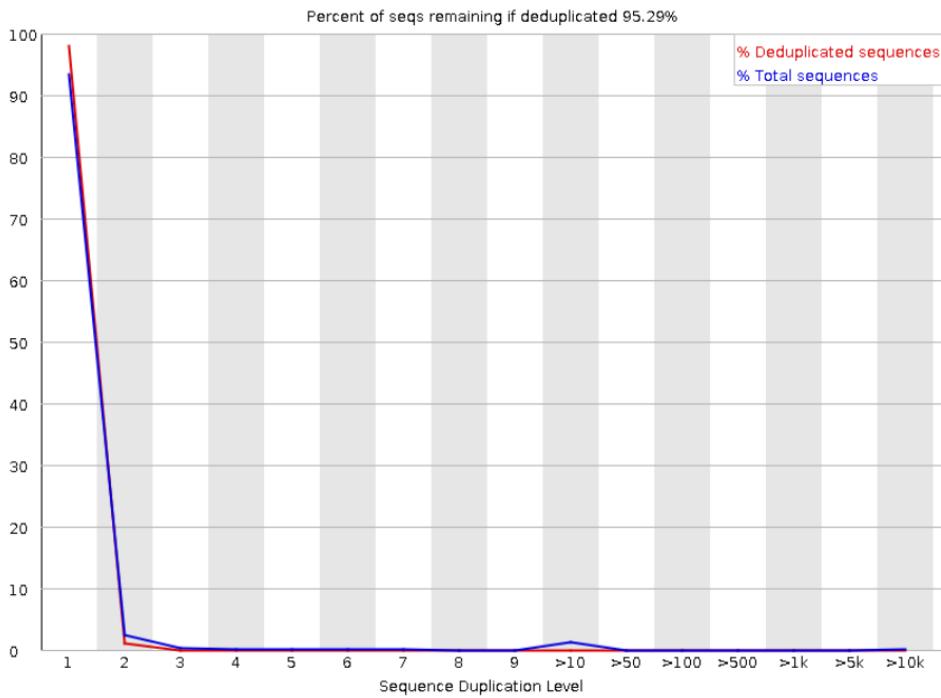
◦ 每个碱基N含量 Per base N content



- # 如果测序仪不能很有把握的确定一个碱基类型，那么通常会用N来代替这个位置可能的碱基
- # 上图反应了每个位置的N的比例。在测序中出现一些N是很正常的，尤其是在读长的末端。
- # 如果N的比例高达几个百分比，就需要引起我们的注意

◦ 序列重复水平 Sequence Duplication Levels





- # 在一个理想的测序文库中，大多数的序列应该只出现一次
如果多次重复出现，那么可能意味着存在一定程度的富集偏倚（如PCR过度扩增等）
蓝线是表示测序文库中所有序列的重复次数分布情况，红线是去重之后的分布情况
正常情况下，蓝线和红线的峰值都应在坐标轴做左端
如果出现了过多的重复序列，那么峰值会变低，曲线变平。可能意味着存在测序文库污染或者严重技术偏倚导致过多的重复序列。

○ 过表达序列表 Overrepresented sequences

- **! Overrepresented sequences**

Sequence	Count	Percentage	Possible Source
GATCGGAAGAGCACACGTCTGAACTCCAGTCACATCACCGATCTCGTATGC	153150	0.37196082323540053	TruSeq Adapter, Index 1 (100% over 50bp)
- # 展示长度至少20bp，数量占总数0.1%以上的reads碱基组成，它可以帮助判断污染（比如：载体、接头序列）
可以帮助我们判断GC表报警的来源，如果是已知的载体或者接头，它会列出来；如果不是，可以复制序列去blast。

MultiQC

```
# 基本格式

# multiqc[OPTIONS] <analysis directory>
# -o, --outdir TEXT 报告输出路径
# -n, 输出文件名称, 默认multiqc_report
```

- General Statistics

-

Sample Name	% Dups	% GC	M Seqs
ctcf_chip_input_r1	5.5%	42%	26.8
ctcf_chip_input_r2	3.6%	42%	26.8
ctcf_chip_rep1_r1	4.7%	43%	41.2
ctcf_chip_rep1_r2	4.1%	43%	41.2
ctcf_chip_rep2_r1	4.4%	43%	39.2
ctcf_chip_rep2_r2	3.9%	43%	39.2

- # %Dups: 重复reads的比例，越小越好
- # %GC: GC含量占总碱基的比例，比例越小越好
- # Length: 测序长度
- # M Seqs: 总测序量（单位: millions）

Cutadapt

- 质量控制
 - https://blog.csdn.net/weixin_43569478/article/details/108079222
 - <https://blog.csdn.net/hanli1992/article/details/82970028>
 - <https://www.jianshu.com/p/190ab5bdd6c6>
 - 对于NGS数据分析而言，第一步都是进行质量控制，质量控制包括去除adapter序列，去除低质量序列等内容
 - 在文库构建阶段，为了能够上机测序，会在插入片段两端添加adapter序列
 - 当目的测序片段较短，测序读长超过了插入片段长度时，就会读取到adapter序列
- 注意事项
 - 由于测序错误率的原因，测序得到的adapter序列会和原本的adapter序列存在几个碱基的误差，所以去除adapter序列时必须允许碱基的错配
 - 由于插入片段的长度在一定范围内变化，而adpter序列出现在两端的位置，所以测序读到的adapter序列可能只是原本adapter的部分序列
 - 针对目前主流的双端测序数据，adapter序列都是出现在3'端，R1序列的3'端可能出现3'adapter 序列，R2端序列的3'端会出现5'端adpter的反向互补序列
 - 无论是R1端还是R2端，其5'端都不会出现adapter，因为测序反应是直接从插入片段开始的。对于双端数据，只需要分别对R1和R2序列去除3'端adapter序列就可以了

```
# 基本格式
cutadapt -a ADAPTER [options] [-o output.fastq] input.fastq

# 双端测序格式
cutadapt -a ADAPT1 -A ADAPT2 [options] -o out1.fastq -p out2.fastq
in_read1.fastq in_read2.fastq

# -j 设置线程数
# -n --times 去除几次接头，默认为1
# -e 去除的序列与adaptor相比，missmatch率低于该值，则认为是adaptor，一般设置为0.1（错配最大碱基数为adaptor长度*0.1）
# -O --overlap 当与adaptor overlap的碱基数大于等于该值时，才进行去除，默认为3
# -a read1 3'的adaptor
# -A read2 3'的adaptor
# -o read1的输出文件
# -p read2的输出文件
# -m --minimum-length 切除接头后的序列长度最小值
# -- discard-trimmed #添加该参数，去掉检测到接头的序列（默认只剪切掉接头序列及之后）
```

```
# -- pair-filter=(any|both) any: read1 和 read2任何一个检测到接头均舍弃; both: 两
read全部检测到接头才同时舍弃

# -q [5'CUTOFF,]3'CUTOFF --quality-cutoff 在去接头前先将低于此数值的bases去除，一
般设为30（计算方法与直接过滤不同，参见本节第一个链接最后部分）
# --quality-base N 最大质量值，默认33，较老的illumina测序需设置成64
```

- 示例

```
cutadapt -j 24 -n 1 -e 0.1 -o 3 -q 25 -m 55 \
-a AGATCGGAAGAGCACACGTCTGAACTCCAGTCA -A AGATCGGAAGAGCGTCGTAGGGAAAGAGTGT \
-o filtered_reads/fixed_ctcf_rep1_read1.fastq.gz -p
filtered_reads/fixed_ctcf_rep1_read2.fastq.gz \
ctcf_chip/ctcf_chip_rep1_r1.fastq.gz ctcf_chip/ctcf_chip_rep1_r2.fastq.gz \
>filtered_reads/cutadapt.log 2>&1 &
```

- 结果解析

- Summary

```
==== Summary ===

Total read pairs processed: 41,173,691
  Read 1 with adapter: 1,435,700 (3.5%)
  Read 2 with adapter: 1,219,854 (3.0%)

== Read fate breakdown ==
Pairs that were too short: 2,929,400 (7.1%)
Pairs written (passing filters): 38,244,291 (92.9%)

■ Total basepairs processed: 8,317,085,582 bp
  Read 1: 4,158,542,791 bp
  Read 2: 4,158,542,791 bp
  Quality-trimmed: 351,144,737 bp (4.2%)
    Read 1: 96,446,694 bp
    Read 2: 254,698,043 bp
  Total written (filtered): 7,622,685,259 bp (91.7%)
    Read 1: 3,825,324,333 bp
    Read 2: 3,797,360,926 bp
```

- Total reads processed: 读取到的reads数
- Total basepairs processed: 总碱基数

- 每样品开头部分

```
==== First read: Adapter 1 ===

Sequence: AGATCGGAAGAGCACACGTCTGAACTCCAGTCA; Type: regular 3'; Length: 33; Trimmed: 1435700 times

Minimum overlap: 3
No. of allowed errors:
  1-9 bp: 0; 10-19 bp: 1; 20-29 bp: 2; 30-33 bp: 3

Bases preceding removed adapters:
  A: 26.0%
  C: 22.0%
  G: 21.2%
  T: 17.5%
  none/other: 13.3%
```

- 第二段代表着在设置的e值下，每匹配到的adaptor长度所对应的错配最大值
- Bases preceding removed adapters显示在切除的adaptor前的一个碱基的比例，若某碱基严重偏多，可能是预设的adaptor没给全

- Overview of removed sequences

- Overview of removed sequences

length	count	expect	max.err	error counts
3	794640	643338.9	0	794640
4	185334	160834.7	0	185334
5	52625	40208.7	0	52625
6	23109	10052.2	0	23109
7	17375	2513.0	0	17375
8	12779	628.3	0	12779
9	14527	157.1	0	13850 677
10	12088	39.3	1	11163 925
...				
38	2995	0.0	3	1204 1554 200 37
39	5176	0.0	3	1723 3056 348 49
40	6185	0.0	3	1688 4062 405 30
41	4329	0.0	3	2673 1466 171 19
...				

- 第一列代表着匹配到adaptor的长度，设置-O为3，所以从3开始
- count代表该长度匹配到的reads个数
- expect代表按随机期望可能出现的reads个数
- max.err代表该长度允许的最大错配个数
- error counts代表0-max.err中，每种错配出现的次数
- 如这次过滤在4329个reads中删除掉了长度为41的adaptor序列，最多允许3个错配，有2673次完全匹配、1466次错配1个碱基、171次错配2个碱基、19次错配3个碱基

3. 比对

不同比对软件比较

- 参考<https://www.jianshu.com/p/601469194b5e>
- 基于BWT算法的比对软件
 - BWA为较早的DNA-seq比对工具
 - bowtie适用于将长度为50bp以下的reads比对到基因组DNA序列上
 - bowtie2擅长比对50-100bp长的reads，长度甚至长达1000。适合比对那些比较长的基因组
 - BWT算法原理可参考<https://www.jianshu.com/p/17ded3dd8d5b>、<https://www.jianshu.com/p/f94b2631681c>
 - bowtie和bowtie2是两个不同类型的比对工具，各有优点
 - bowtie2先将seed序列比对到基因组中，再利用NW或SW算法作比对，支持gap，但是seed序列不能有gap
 - 貌似都不能跨内含子进行转录组数据分析
- Tophat和Tophat2
 - Tophat2是Tophat2的升级版
 - Tophat只可以调用bowtie，而Tophat2不仅可以调用bowtie2（默认），还可以更改设置调用bowtie
 - 首先使用bowtie/bowtie2对序列进行比对，对于那些没有比对上的，会考虑其跨外显子的可能性，将reads劈开重新比对

- HISAT2
 - 不仅支持RNA-seq的比对还支持DNA-seq比对
 - 宣称速度更快，内存占用率更小，准确率更高
 - junction正确率最高，但是灵敏度相对较低
 - STAR
 - 灵敏度更高，但是会有许多包含soft-clip的低质量比对
 - unique mapping比例最高，它对于双端测序的reads，要么全部比对上，要么全部抛弃，不会像像TopHat和HISAT2一样只比对上某一个reads
 - HISAT-3N
 - 3N意为3 nucleotides，专为**核苷酸转换测序技术**而设计，基于 HISAT2 实现
 - 用于任何碱基转换测序reads的mapping，包括BS-seq、SLAM-seq、TAB-seq、oxBS-seq、TAPS、scBS-seq、scSLAM-seq等
 - 《Gaining comprehensive biological insight into the transcriptome by performing a broad-spectrum RNA-seq analysis》
认为比较好的RNA-seq组合方式为HISAT2-FeatureCounts-DESeq2
 - 《基因组数据分析》对于RNA-seq的组合方式为STAR-FeatureCounts-DESeq2
 - 龙星课程对于RNA-seq的组合方式为STAR+RSEM
 - 《Linux生物信息技术》对于ChIP-seq的组合方式为
-

bowtie 2

- 参考<https://blog.csdn.net/u011262253/article/details/79833969>、<https://www.jianshu.com/p/f74270606416>、<https://www.jianshu.com/p/c1886d345cf1>
- 构建索引
 - 在测序产生的reads片段与参考基因组进行比对时，为了比对更加快速，需要建立索引（相对于目录性质），帮助比对软件更快速的找到目标区域

```
# 基本格式
bowtie2-build [options]* <reference_in> <bt2_index_base>

# <reference_in> index的参考fasta文件
# <bt2_base> 生成的index文件的前缀
# -f 参考基因组为fasta序列（默认）
# --threads 线程数
# --usage 帮助界面
```

- 示例

```
bowtie2-build -f --threads 8 human_genome/hg38_human.fa
myindex/hg38_human_myindex \
>myindex/build_index.log 2>&1 &
```

- 在目标文件夹生成了六个index文件

```

leb12a@bbt-enhance:~/chipseq$ ll -h myindex/
total 3.9G
drwxr-xr-x 2 leb12a leb 4.0K Jun  8 15:48 .
drwxr-xr-x 5 leb12a leb 4.0K Jun  8 15:05 ../
-rw-r--r-- 1 leb12a leb 62K Jun  8 15:48 build_index.log
-rw-r--r-- 1 leb12a leb 938M Jun  8 15:27 hg38_human_myindex.1.bt2
-rw-r--r-- 1 leb12a leb 701M Jun  8 15:27 hg38_human_myindex.2.bt2
-rw-r--r-- 1 leb12a leb 8.9K Jun  8 15:07 hg38_human_myindex.3.bt2
-rw-r--r-- 1 leb12a leb 701M Jun  8 15:07 hg38_human_myindex.4.bt2
-rw-r--r-- 1 leb12a leb 938M Jun  8 15:48 hg38_human_myindex.rev.1.bt2
-rw-r--r-- 1 leb12a leb 701M Jun  8 15:48 hg38_human_myindex.rev.2.bt2

```

- 也可在<https://bowtie-bio.sourceforge.net/bowtie2/manual.shtml#how-is-bowtie-2-different-from-bowtie-1>上下载目的基因组的index

- 比对

```

# 基本格式
bowtie2 [options]* -x <bt2-idx> -1 <m1> -2 <m2> [-S <sam>]

# -x 参考基因组index文件的路径及前缀
# -1 read1 fastq文件
# -2 read2 fastq文件
# -S 输出的SAM文件位置
# -p 线程数
# --no-unal 在输出结果中删除那些没有被比对的查询序列；如果未指定该选项，将输出经比对的序列文件和未被比对的查询序列文件

# --end-to-end 端对端比对模式，将整个查询序列与参考序列进行比对，更精准（默认）
# --local 局部比对模式，reads和参考基因组不必完全相同，为了达到最大可能的对齐分数，可能从reads末端省略一些字符
# 根据数据特征选择比对模式，ChIP-seq一般序列准确，使用end-to-end
# --fast 跳过预处理步骤，直接开始比对；如果输入的序列数据不需要再次进行过滤和质量控制，则可以使用该选项

# -I/--minins 插入片段最小长度(0)
# -X/--maxins 插入片段最大长度 (500)
# --fr/--rf/--ff 设定和谐比对的方向(--fr)
# --no-mixed 阻止默认动作：如果bowtie2找不到一对reads的成对比对，就会尝试以非配对的方式比对这一对中的两条reads
# --no-discordant 阻止默认动作：如果一对reads不能和谐比对(满足-I, -X, --fr/--rf/--ff的条件)到参考序列上，则搜寻其不和谐比对(两条reads都能独一无二地比对到参考序列上，但是不满足-I, -X, --fr/--rf/--ff的条件)
#-k 默认设置下，bowtie2搜索出了一个read不同的比对结果，并报告其中最好的比对结果。而在该模式下，bowtie2最多搜索出一个read <int>个比对结果，并将这些结果按得分降序报告出来。
#-a 和-k参数一样，但不限制搜索的结果数目，并将所有的比对结果都按降序报告出来。基因组含很多重复序列时会导致程序运行缓慢。

```

- 示例

```

bowtie2 -p 8 -x myindex/hg38_human_myindex -1
filtered_reads/fixed_ctcf_rep1_read1.fastq.gz \
-2 filtered_reads/fixed_ctcf_rep1_read2.fastq.gz -S ctcf_rep1_mapped.sam \
\
>bowtie2_map.log 2>&1 &

```

- 标准输出

```

leb12a@bbt-enhance:~/chipseq$ cat map/bowtie2_map.log
38244291 reads; of these:
  38244291 (100.00%) were paired; of these:
    841199 (2.20%) aligned concordantly 0 times
    32372026 (84.65%) aligned concordantly exactly 1 time
    5031066 (13.16%) aligned concordantly >1 times
    ...
    841199 pairs aligned concordantly 0 times; of these:
      217865 (25.90%) aligned discordantly 1 time
    ...
    623334 pairs aligned 0 times concordantly or discordantly; of these:
      1246668 mates make up the pairs; of these:
        710829 (57.02%) aligned 0 times
        248634 (19.94%) aligned exactly 1 time
        287205 (23.04%) aligned >1 times
  99.07% overall alignment rate

```

- 第一部分：总共38244291条reads参加比对，其中2.2%没合理比对上，13.16%合理比对上多次
- 第二部分：在那2.2%的reads中，又有25.9%的reads双端比对上了，但是不合理
- 第三部分：在那2.2%的reads中的剩余部分中未能双端比对的，进行单端比对的结果

- SAM文件格式

- 参考https://blog.csdn.net/weixin_69556916/article/details/130488775

- 头部区

```

■ @HD      VN:1.0  SO:unsorted
@SQ      SN:chr10       LN:133797422
@SQ      SN:chr11       LN:135086622
@SQ      SN:chr12       LN:133275309
...
@SQ      SN:chrX LN:156040895
@SQ      SN:chrY LN:57227415
@PG      ID:bowtie2      PN:bowtie2      VN:2.4.4
CL:"/usr/bin/bowtie2-align-s --wrapper basic-0 -p 8 -x
myindex/hg38_human_myindex -s ctcf_rep1_mapped.sam -1
filtered_reads/fixed_ctcf_rep1_read1.fastq.gz -2
filtered_reads/fixed_ctcf_rep1_read2.fastq.gz"

```

- @HD 第一行
 - VN: 格式版本
 - SO: 比对排序的类型，samtools软件在进行行排序后不能自动更新bam文件的SO值，而picard却可以
- @SQ 参考序列
 - SN: 参考序列名
 - LN: 参考序列长度
- @PG 比对所使用的软件、版本及程序
- 主体部分：11个主列、1个可选列

```

HNH75M1:468:C83LBACXX:7:1101:3223:2248 83      chr17    62931776
        42      101M    =      62931610      -267
ATTTTTTGAGACTGAGTCTCACTATCACCCAGGCTGGAGTGCTGTGGCACGATGTCGGCTCACTG
CAACCTCCGCCCTCCCAGGTTAACGTGATTCTCC      ?
DDDDDCDCC@:CC>CDC@:4CCC>DB93BBDCDDCDDC>ADBDCCDDECDFFFHHIIIIHCFGGIG
F<JJJJJIIGGIJJJJJJIIHFHHGFFFFCCC AS:i:0 xs:i:-53 xn:i:0 xm:i:0
xo:i:0 xg:i:0 nm:i:0 md:z:101 ys:i:0 yt:z:CP
HNH75M1:468:C83LBACXX:7:1101:3223:2248 163      chr17    62931610
        42      98M    =      62931776      267
ATAGGCACACGCCACCATGCCAGCTAATTTTGTTAGTAGAGACG
GGATTTCCCCATGTTGCCAGGCTGGTCTCGAACTCCTGACCTCAGG ?1=B?BBAH?
ADDGGIIDBHIIAGHIEHIIIIIDDHIIIIIFGGEIIIBFGHGHIGHIIHHHHEDFFFFCCEAB;
@@CC:ABBCCCCCCC@>AC AS:i:0 xs:i:-26
xn:i:0 xm:i:0 xo:i:0 xg:i:0 nm:i:0 md:z:98 ys:i:0
yt:z:CP
HNH75M1:468:C83LBACXX:7:1101:3863:2381 77      *      0      0
*      *      0      0
TGATGGCAATGGTGGGGATACTGGTGATAGTGGTGTGGTGGTGGTGGTGGTGGTGGTATTGGT
GGTGGTGGTGATAATGGTGTGGTGATGGTTTGG
CCCFHHGH<EGGIGIDGHIJCGGHIICGFGIFHIJ?@F?
DH<FH@C=CAGAHDEEHACFFFFBCC9?B';=385:;>@CD(5+8?(<>@CD3:??<< YT:Z:UP

```

1. QNAME 比对的序列名称: 如 HNH75M1:468:C83LBACXX:7:1101:3223:2248

2. FLAG 值, 为以下各项相加

- 1 (1) 该read是成对的paired reads中的一个
- 2 (10) paired reads中每个都正确比对到参考序列上
- 4 (100) 该read没比对到参考序列上
- 8 (1000) 与该read成对的matepair read没有比对到参考序列上
- 16 (10000) 该read其反向互补序列能够比对到参考序列
- 32 (100000) 与该read成对的matepair read其反向互补序列能够比对到参考序列
- 64 (1000000) 在paired reads中, 该read是与参考序列比对的第一条
- 128 (1000000) 在paired reads中, 该read是与参考序列比对的第二条
- 256 (100000000) 该read是次优的比对结果
- 512 (1000000000) 该read没有通过质量控制
- 1024 (10000000000) 由于PCR或测序错误产生的重复reads
- 2048 (100000000000) 补充匹配的read

3. RENAME 比对上的参考序列名, 这里为染色体名

- 如果这列是 *, 可以认为这条read没有比对上的序列, 则这一行的第4、5、8、9列都是0, 第6, 7列与该列是相同的表示方法

4. POS 表示read比对到RENAME这条序列的最左边的位置

- 如果该read能够完全比对到这条序列则这个位置是read的第一个碱基比对的位置
- 如果该read的反向互补序列比对到这条序列, 则这个位置是read的反向互补序列的第一个碱基比对的位置

5. MAPQ 比对质量

- mapping的错误率的-10log10, 越大mapping率越高
- 值为255表示mapping值是不可用的, 如果是unmapped read则MAPQ为0

6. CIGAR string 比对结果信息

- 可以理解为reads mapping到第三列序列的mapping状态
- 如37M1D2M1I, 意思是37个匹配, 1个参考序列上的删除, 2个匹配, 1个参考序列上的插入

标记字符	内容描述
M	匹配(包含完全匹配和单碱基错配)
I	序列插入 (包含潜在Insertion变异)
D	序列删除 (包含潜在Deletion变异)
N	跳过参考序列, 常见于RNA-Seq数据的比对
S	软跳过 (soft clip), 跳过read中的部分序列, 不会改变read的长度
H	硬跳过 (hard clip), 直接剪切掉read中分布序列, 会改变read的长度
P	padding, 简单来说和N类似也是read比对时中间跳过参考序列的部分区域
=	完全匹配 (很少用)
X	序列错配 (很少用)
B	BAM_CBACK, 这是一个曾经适用于CG测序技术特点的标记 (华大GDP@生信字典未采用) 作用和N类似, 跳过一定长度的参考序列

7. MRNM 这条reads第二次比对上的参考序列名

- =表示参考序列与reads一模一样, *表示没有完全一模一样的参考序列

8. MPOS 表示与该reads对应的mate pair reads的比对位置

- 如果这对pair-end reads比对到同一条reference序列上, 在sam文件中reads的id出现2次
- Read1比对的第4列等于Read2比对的第8列, Read1比对的第8列等于Read2比对的第4列

9. ISIZE 序列模板长度

- 如果R1端的read和R2端的read能够mapping到同一条Reference序列上, 则该列的值表示第8列减去第4列加上第6列的值
- R1端和R2端相同id的reads第九列值相同, 但该值为一正一负

10. SEQ 表示序列片段的序列信息

11. QUAL reads碱基质量值

12. 多种型号的标记

samtools

• sort

- 参考<https://zhuanlan.zhihu.com/p/573630960>
- 对bam文件进行排序

```
# 基本格式
samtools sort [options...] [in.bam]

# -m 设置每个线程所能用的最大内存
# -@ 指定线程数
# -O 指定输出格式(SAM, BAM, CRAM)
# -n 按read名进行排序 (默认情况下按照参考序列的名字顺序与坐标顺序排列
# -o 输出文件路径
```

• view

- 针对bam文件进行相关操作

- # 基本格式
samtools view [options] <in.bam>|<in.sam>|<in.cram> [region ...]

-b 输出BAM格式文件（默认.sam）
-S 输入SAM格式的文件（默认.bam）
-h 输出的SAM文件带header（默认不带）
-c 输出（过滤后）比对的数量
-o 输出文件名

-f 输出带有指定FLAG的结果
-F 输出不带指定FLAG的结果
-q 指定MAPQ的下限
-@ 指定线程数

- 示例

- 按照染色体坐标将ctcf_rep1_mapped.sam排序并输出成.bam格式

```
 samtools sort -O BAM -o map/ctcf_rep1_mapped_sort.bam -@ 16 -m 4G  
map/ctcf_rep1_mapped.sam
```

- 可以通过管道命令查看.bam文件

```
 samtools view map/ctcf_rep1_mapped_sort.bam | cut -f 2-9 | head
```

- 噢一下过滤前的reads数

```
 samtools view -c -@ 16 map/ctcf_rep1_mapped_sort.bam  
76488582
```

- 浅浅过滤一下

```
 samtools view -b -q 30 -@ 16 -f 1 -f 2 -o map/ctcf_rep1_filtered.bam  
map/ctcf_rep1_mapped_sort.bam
```

- 看看过滤后的reads数

```
 samtools view -c -@ 16 map/ctcf_rep1_filtered.bam  
68203018
```

- bowtie2默认只输出比对得分最好的比对结果，所以不用去除multiple mapping

picard

- 参考<https://www.jianshu.com/p/645aaa48a8f0>、<https://www.jianshu.com/p/ca3b581085ba>
- 去重
 - 来自一个DNA片段的两个拷贝，可能会锚定在两个bead上，经过测序得到的这两条read，就是PCR duplication

- illumina测序时照相机错误的将一个簇识别为多个簇，也会导致产生完全一样的reads
- 不同的序列在进行PCR扩增时，扩增的倍数应该是相同的。但是由于聚合酶的偏好性，PCR扩增次数过多的情况下，会导致一些序列持续扩增，因此需要进行去重
- 一般来说，ChIP-seq、Call SNP、WGBS、ATAC-seq需要去重
- RNA-seq、RRBS、Targeted-seq不去重
- 到底要不要去除重复，可以通过质量检测报告判断这些常规要去重的数据最后要不要进行后期的除重工作
- 暴力的方法：根据实验建库的方式，若去除的都去重（ChIP/ATAC al et），靶向的，酶切的建库均不去重（RRBS, Target-BS）但消耗计算资源和运行时间
- 去重的工具有samtools、Picard、sambamba等，一般使用Picard
- Picard的MarkDuplicates方法可以识别duplicates。该工具仅对duplicates做一个标记，只在需要的时候对reads进行去重
- 不仅考虑reads的比对位置，还会考虑其中的插入错配等情况以及比对方向，重视reads的5'端信息

基本格式

```
MarkDuplicates [arguments]
```

-I 输入的BAM、SAM、CRAM文件，需要按坐标排序

-M 指定输出的matrix log文件

-O 指定输出bam文件

--REMOVE_DUPLICATES 如果为true，则将检测出的PCR duplication直接去除。默认值:false。

- 示例

```
java -jar /home/public/picard/picard.jar MarkDuplicates --REMOVE_DUPLICATES
true \
-I map/ctcf_rep1_filtered.bam \
-O deduplication/ctcf_rep1_dedup.bam -M deduplication/deduplication.log \
>deduplication/picard.log 2>&1 &
```

使用java调用picard工具包中的MarkDuplicates工具

```
samtools view -c -@ 16 deduplication/ctcf_rep1_dedup.bam
67911634
```

4. Peak calling

macs2

- 参考<https://www.jianshu.com/p/69744f961b97>、<https://www.jianshu.com/p/9c20435f8fd3>、<https://mp.weixin.qq.com/s/rN7ged0wepNRTXNG7MEgpQ>
- macs2会通过设置一个通常为1000bp的滑动窗口，以滑动窗口为单位来统计该区段上的reads富集成的peaks，而后使用泊松分布进行统计学检验

#安装软件

```

conda install -c bioconda macs2

#基本用法
macs2 callpeak -t TFILE [TFILE ...] [-c [CFILE ...]] ...

# -t 处理组的文件，若有多个使用空格隔开
# -c 对照组的文件，若有多个使用空格隔开
# -n 输出结果的前缀
# -f 输入文件的类型，默认自动检测。BAMPE为双端测序的BAM文件，软件会根据算法处理双峰
# -g 有效基因组大小，可输入数字类似2.7e9，或hs人类、mm小鼠、ce线虫、dm果蝇
# -q q值的cutoff
# -s 二代测序读长，MACS会用前面10个序列进行推测
# --outdir 输出文件夹
# --min-length peak的最短长度，大于该值才视作peak
# --max-gap 两个相邻peaks的最大距离，若近于该值则合并为一个peak
# -B 使用bedGraph存储累积值和local lambda
# --broad 进行broad peak（组蛋白修饰常用），一般为narrow peak

# --nomodel 不需要MACS去构建模型
# --extsize 将read从5'至3'衍生至等长片段，如果转录因子已知结合200bp则设置--extsize 200
# --shift 绝对的偏移值，会先于extsize前对read进行延伸

```

- 其中-g的值可以通过去掉基因组中含N的碱基自行计算，但好像不能去除重复的序列

```

#下载ucsc-facount软件
conda install -c bioconda ucsc-facount

#计算基因组大小，用Total的len-N得到有效基因组长度
faCount human_genome/hg38_human.fa

```

- 示例

```

macs2 callpeak -t deduplication/ctcf_rep1_dedup.bam \
-c ctcf_chip/mapping/ctcf_chip_input_filtered_dedup.bam \
-f BAMPE -g hs -n CTCF_ChIP_rep1 -B -q 0.01 --outdir callpeak/ \
> callpeak/peak_calling.log 2>&1 &

```

- 生成如下系列文件

```

(chipseq) leb12a@bbt-enhance:~/chipseq$ ls callpeak/
CTCF_ChIP_rep1_control_lambda.bdg  CTCF_ChIP_rep1_peaks.xls
CTCF_ChIP_rep1_treat_pileup.bdg
CTCF_ChIP_rep1_peaks.narrowPeak      CTCF_ChIP_rep1_summits.bed
peak_calling.log

```

- NAME_peaks.xls**

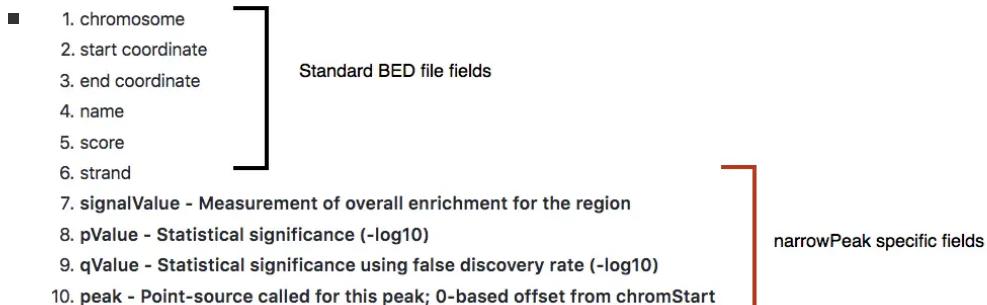
chr	start	end	length	abs_summit	pileup	-LOG10(pvalue)	fold_enrichment	-LOG10(qvalue)	name
chr1	778618	779016	399	778803	46	19.3663	4.95501	15.7628	CTCF_ChIP_rep1_peak_1
chr1	912845	913104	260	913017	24	9.91395	4.21341	6.82797	CTCF_ChIP_rep1_peak_2
chr1	920977	921268	292	921181	37	9.67286	3.14044	6.60726	CTCF_ChIP_rep1_peak_3
chr1	924770	925246	477	925062	40	17.9592	5.16207	14.4193	CTCF_ChIP_rep1_peak_4
chr1	925566	926213	648	925724	42	16.441	4.53331	12.9776	CTCF_ChIP_rep1_peak_5
chr1	931534	931793	260	931735	29	9.03264	3.44276	6.00705	CTCF_ChIP_rep1_peak_6
chr1	937995	938600	606	938301	77	35.2117	5.84606	30.9839	CTCF_ChIP_rep1_peak_7
chr1	939851	940165	315	940071	32	14.1543	4.84269	10.814	CTCF_ChIP_rep1_peak_8
chr1	941156	941507	352	941229	37	13.8942	4.25593	10.5646	CTCF_ChIP_rep1_peak_9
chr1	951402	951720	320	951520	20	15.2204	5.50701	11.0253	CTCF_ChIP_rep1_peak_10

- 染色体
- peak起始位点
- peak终止位点
- peak的长度
- peak顶点坐标
- 堆积信号值及其-log10(pvalue)
- 峰顶处reads相对于local background的富集倍数及其-log10(pvalue)

- NAME_peaks.narrowPeak

- (chipseq) 1eb12a@bbt-enhance:~/chipseq\$ head callpeak/CTCF_ChIP_rep1_peaks.narrowPeak

chr1	778617	779016	CTCF_ChIP_rep1_peak_1	157 .	4.95501
19.3663	15.7628	185			
chr1	912844	913104	CTCF_ChIP_rep1_peak_2	68 .	4.21341
9.91395	6.82797	172			
chr1	920976	921268	CTCF_ChIP_rep1_peak_3	66 .	3.14044
9.67286	6.60726	204			
chr1	924769	925246	CTCF_ChIP_rep1_peak_4	144 .	5.16207
17.9592	14.4193	292			
chr1	925565	926213	CTCF_ChIP_rep1_peak_5	129 .	4.53331
16.441	12.9776	158			
chr1	931533	931793	CTCF_ChIP_rep1_peak_6	60 .	3.44276
9.03264	6.00705	201			
chr1	937994	938600	CTCF_ChIP_rep1_peak_7	309 .	5.84606
35.2117	30.9839	306			
chr1	939850	940165	CTCF_ChIP_rep1_peak_8	108 .	4.84269
14.1543	10.814	220			



- NAME_summits.bed

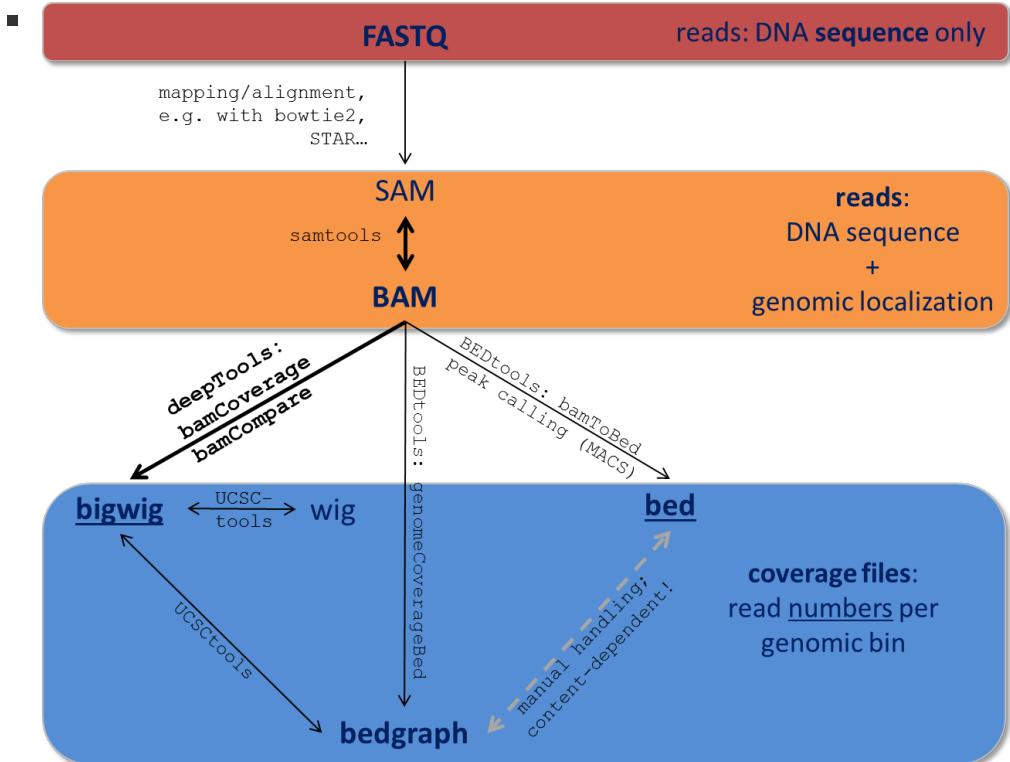
- 包含了每个peaks峰顶的位置信息，可用于motif和binding sites的预测
- 第五列score与"NAME_peaks.narrowPeak"的第8列（峰顶处的-log10pvalue）一致

- (chipseq) 1eb12a@bbt-enhance:~/chipseq\$ head callpeak/CTCF_ChIP_rep1_summits.bed

chr1	778802	778803	CTCF_ChIP_rep1_peak_1	15.7628
chr1	913016	913017	CTCF_ChIP_rep1_peak_2	6.82797
chr1	921180	921181	CTCF_ChIP_rep1_peak_3	6.60726
chr1	925061	925062	CTCF_ChIP_rep1_peak_4	14.4193
chr1	925723	925724	CTCF_ChIP_rep1_peak_5	12.9776
chr1	931734	931735	CTCF_ChIP_rep1_peak_6	6.00705
chr1	938300	938301	CTCF_ChIP_rep1_peak_7	30.9839
chr1	940070	940071	CTCF_ChIP_rep1_peak_8	10.814

- NAME_treat_pileup.bdg

- bdg/bg/bedGraph文件用于存放区间的坐标轴信息和相关评分(score)的文件，用于存储稀疏，不连续的数据。本文件记录了处理组的堆积信号；可以转化为bigwig文件（二进制，有索引，更高效）。
- 与BED文件类似，它只包含4列，第4列必须是一个分数
- NAME_control_lambda.bdg
 - 记录了由对照组中估计得到的局部偏好；可以转化为bigwig文件。
- 其他常见的文件格式：
- bigWig
 - bw/bigwig是bedGraph或wig文件的二进制格式
 - 包含间隔和相关分数的坐标。分数可以是任何东西，例如平均读取覆盖率
 - 比如将基因组分成许多的bin，存储这个bin中的碱基覆盖率



ChIP质量控制

- 可使用deeptools工具中的plotFingerprint软件
- 按照bin的大小，plotFingerprint将基因组分割后统计每个bin中所有的reads数目，而后从小到大排列
- 绘制一张以x轴为累积bin的百分比，y轴为累积reads的百分比的图
- 对于Input来说，如果它的信号是均匀随机分布在整个基因组上，那么它绘制出来的结果图应该就是一条对角线
- 对于实验组来说，如果它的信号是不均匀随机分布在整个基因组上，在某些区域有很强的富集程度，那么在x轴前部分的bin中累积read相对较低，而最后一小部分bin中的累积read则会出现陡升

```

#基本用法
plotFingerprint -b treatment.bam control.bam -plot fingerprint.png

# -b 指定bam文件，多个文件用空格隔开
# --labels 样本标签，用空格隔开，默认文件名
# -plot -o 输出文件名，可根据后缀自行判断(png, eps, pdf, svg)
# -p 线程数
# 其他参数可在https://www.jianshu.com/p/7e28c38aacd3进一步了解

```

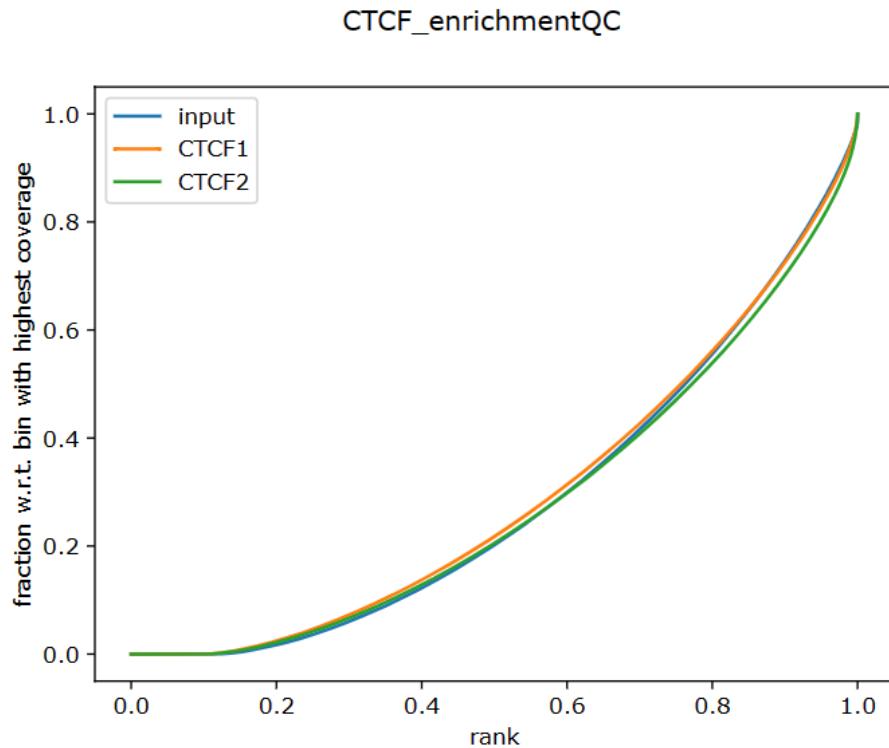
- 示例

```

plotFingerprint -b ctcf_chip/mapping/ctcf_chip_input_filtered_dedup.bam \
ctcf_chip/mapping/ctcf_chip_rep1_filtered_dedup.bam \
ctcf_chip/mapping/ctcf_chip_rep2_filtered_dedup.bam \
--labels input CTCF1 CTCF2 --plotFileFormat svg --plotTitle
CTCF_enrichmentQC \
-o plotFP/CTCF_enrichment2.svg -p 16 &

```

- 结果



- 好像不太乐观
 - 在最开始的一些bin中没有reads。我们可以根据图中前面没有reads的bin所占百分比有一个大概了解
 - 有些broad peak的ChIP-seq数据，如果input和IP差异不大，我们也不能说该IP是失败的，还是要具体情况具体分析

5. 与TES、TTS的共定位情况

TSS bed文件的准备

- 与<https://genome.ucsc.edu/cgi-bin/hgTables>下载Bed文件（基因组的5'UTR信息）
- 实际上，bed文件可以存储其他任意信息，比如可以将reads与高甲基化区域做一个共定位分析

Table Browser

Use this tool to retrieve and export data from the Genome Browser annotation track database. You can limit retrieve sequence covered by a track. [More...](#)

Select dataset

clade: Mammal genome: Human assembly: Dec. 2013 (GRCh38/hg38)
group: Genes and Gene Predictions track: NCBI RefSeq
table: UCSC RefSeq (refGene) data format description

Define region of interest

region: genome position chr2:25,160,915-25,168,903 lookup define regions
identifiers (names/accessions): paste list upload list

Optional: Subset, combine, compare with another track

filter: create
subtrack merge: create
intersection: create
correlation: create

Retrieve and display data

output format: BED - browser extensible data Send output to Galaxy GREAT
output filename: hg38_uscs_refseq.bed (leave blank to keep output in browser)
file type returned: plain text gzip compressed

get output summary/statistics

Output refGene as BED

Include custom track header:

name= tb_refGene
description= table browser query on refGene
visibility= pack
url=

Create one BED record per:

Whole Gene
 Upstream by 200 bases
 Exons plus 0 bases at each end
 Introns plus 0 bases at each end
 5' UTR Exons
 Coding Exons
 3' UTR Exons
 Downstream by 200 bases

Note: if a feature is close to the beginning or end of a chromosome and upstream/downstream

get BED cancel

- 下载下来后

```
(base) leb12a@bbt-enhance:~/chipseq$ less -S plotHP/hg38_uscs_refseq_5UTR.bed | head
chr1 201283451 201283702 NM_000299_utr5_0_0_chr1_201283452_f 0 +
chr1 67127240 67127257 NM_001276351_utr5_2_0_chr1_67127241_r 0 -
chr1 67131141 67131227 NM_001276351_utr5_1_0_chr1_67131142_r 0 -
chr1 67134929 67134970 NM_001276351_utr5_0_0_chr1_67134930_r 0 -
chr1 201283505 201283702 NM_001005337_utr5_0_0_chr1_201283506_f 0 +
chr1 67127240 67127257 NM_001276352_utr5_2_0_chr1_67127241_r 0 -
chr1 67131141 67131227 NM_001276352_utr5_1_0_chr1_67131142_r 0 -
chr1 67134929 67134970 NM_001276352_utr5_0_0_chr1_67134930_r 0 -
chr1 8656297 8656441 NM_001042681_utr5_1_0_chr1_8656298_r 0 -
chr1 8817159 8817640 NM_001042681_utr5_0_0_chr1_8817160_r 0 -
```

该文件存储每个基因的5'UTR的起止信息、正负链

bamCoverage

- 该工具将reads比对作为输入（BAM文件），并生成覆盖轨迹（bigWig或bedGraph）作为输出
- 覆盖率根据每个bin的读取次数来计算，其中bin是定义大小的短连续计数窗口。
- bamCoverage通过缩放因子、每百万次映射读取的每千碱基读取数(RPKM)、每百万次映射读取的计数(CPM)、每百万次映射读取的箱数(BPM)和1倍深度(每基因组覆盖的读取数，RPGC)提供归一化
- 可参考<https://www.weinformatics.cn/c732ce362e/>

```
# 基本格式
bamCoverage -b reads.bam -o coverage.bw

# -b 输入的bam文件
# -o 输出文件路径
# -of 输出文件格式，默认bigwig，可选bedgraph
# --binSize 设置bin的碱基长度，默认50bp
# --effectiveGenomeSize 有效基因组长度，可在下连接中查看详情
#似乎去除多重比对的结果更适合用khmer program计算而不是faCounts

#https://deeptools.readthedocs.io/en/latest/content/feature/effectiveGenomes
ize.html
# --normalizeUsing 可选择多种归一化方式，help中有详细计算公式与介绍
# --extendReads/-e 延伸reads长度至指定大小
#对于双端配对测序，会将中间也纳入计算；对于单端测序与非配对则延伸至指定/程序预估值
#不建议RNA-seq使用这个参数
# --numberOfProcessors/-p 线程数（咋感觉设了后还是一个核跑）
# -centerReads 中心化reads，可获得更陡峭的峰
```

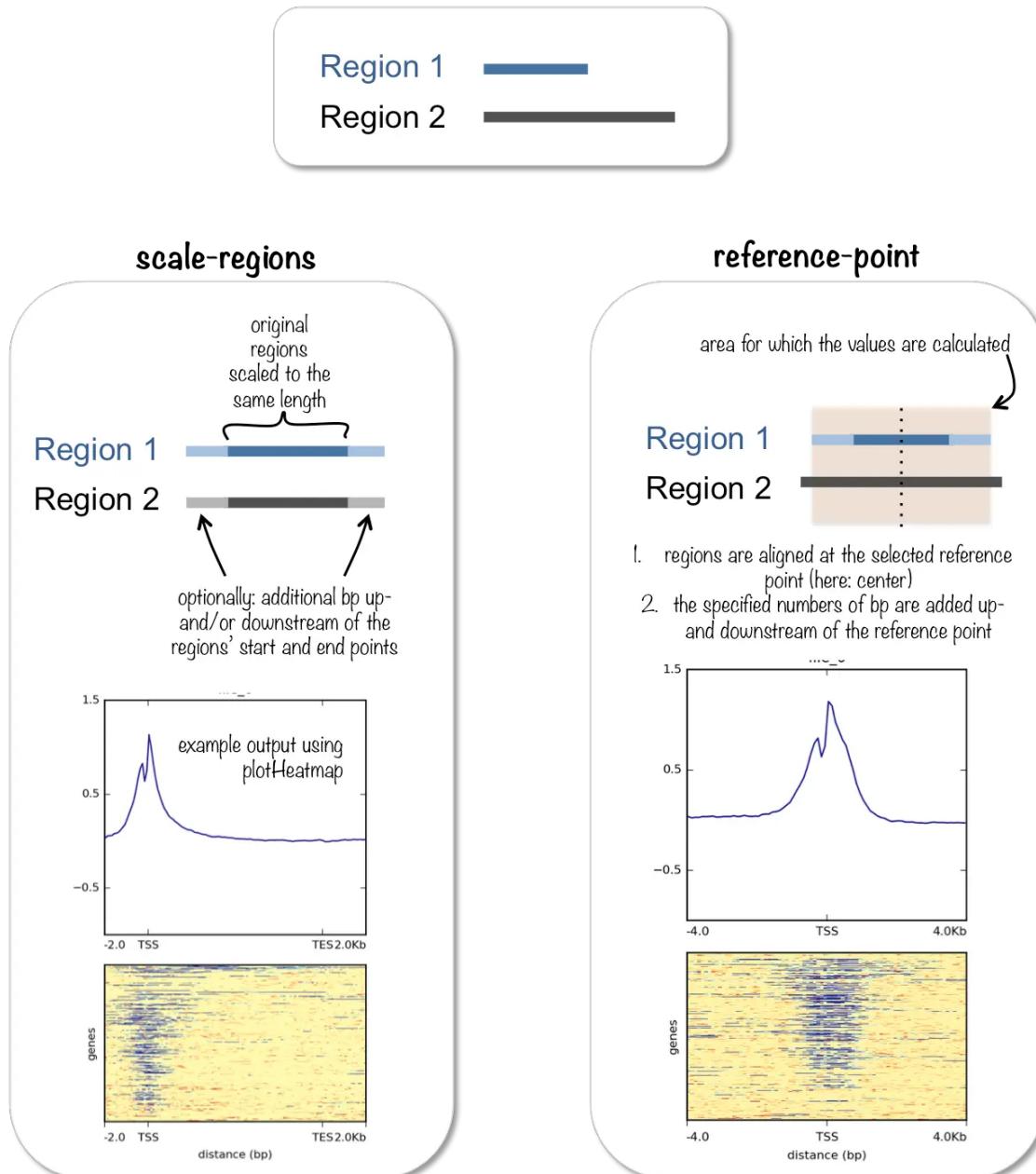
- 示例

```
bamCoverage -b deduplication/ctcf_rep1_dedup.bam -o
plotHP/ctcf_rep1_dedup.bw --binsize 10 \
--normalizeUsing RPGC --effectiveGenomeSize 2913022398 -e -p 16 &
```

输出略多，还得存个log

computeMatrix

- computeMatrix 提供以下两种用法以不同的参考系计算 ChIP-seq 的信号
 - scale-regions：以一段区域为参考系，将所有参考的基因组区域缩放至指定的区域长度，计算该区域内的 ChIP-seq 信号，缩放机制可以参照<https://www.jianshu.com/p/ab2bb3f55d6f>
 - reference-point：以某一坐标为参考系，取上下游一定长度作为窗口计算区域内的 ChIP-seq 信号值



- 以 reference-point 为例计算 TSS 两边情况

```

# 基本格式
computeMatrix reference-point -S <biwig file(s)> -R <bed file> -a 3000 -b
3000

# -S 包含要绘制的分数的bigwig文件。多个文件之间用空格隔开
# -R BED或GTF格式的文件，其中包含要绘制的区域。可输入多个文件或在文件内用#隔开以分组，使其
平行画图
# -o 输出文件
# -p 线程数
# -a 起始位点下游的距离（after）
# -b 起始位点上游的距离（before）
# --sortRegions 矩阵中信号排列顺序
# --samplesLabel 矩阵中样品命名
# --skipZeros 不包含分数为零的区域（默认包含）
# --referencePoint 选择参考点：TSS（默认）、TES、center

```

- 示例

```

computeMatrix reference-point -S
ctcf_chip/deeptools/ctcf_chip_input_filtered_dedup.bw \
ctcf_chip/deeptools/ctcf_chip_rep1_filtered_dedup.bw \
ctcf_chip/deeptools/ctcf_chip_rep2_filtered_dedup.bw \
-R plotHP/hg38_uscs_refseq_5UTR.bed -a 2500 -b 2500 --samplesLabel input
ctcf1 ctcf2 \
--sortRegions descend -o plotHP/input_rep1_rep2_TSS_Matrix.gz -p 16 \
>plotHP/computeMatrix1.log 2>&1 &

```

plotHeatmap

- 主要用来画热图并包含聚类功能，作图会把之前computeMatrix时候提交的多个bed文件分开作图

```

# 基本格式
plotHeatmap [--matrixFile MATRIXFILE] --outFileName OUTFILENAME

# -m 输入文件
# -o 输出文件 可选png、eps、pdf、svg，后缀自动识别
# --dpi 分辨率，默认200
# -T/plotTitle 图的名称
# --regionsLabel 重新定义区域名称，否则为输入文件名
# --yAxisLabel y轴标题
# --zMin --zMax 指定colorbar范围
# --colorMap --colorList 修改颜色
# --whatToShow 默认是在热图和热图颜色条的顶部包含摘要或概要图

```

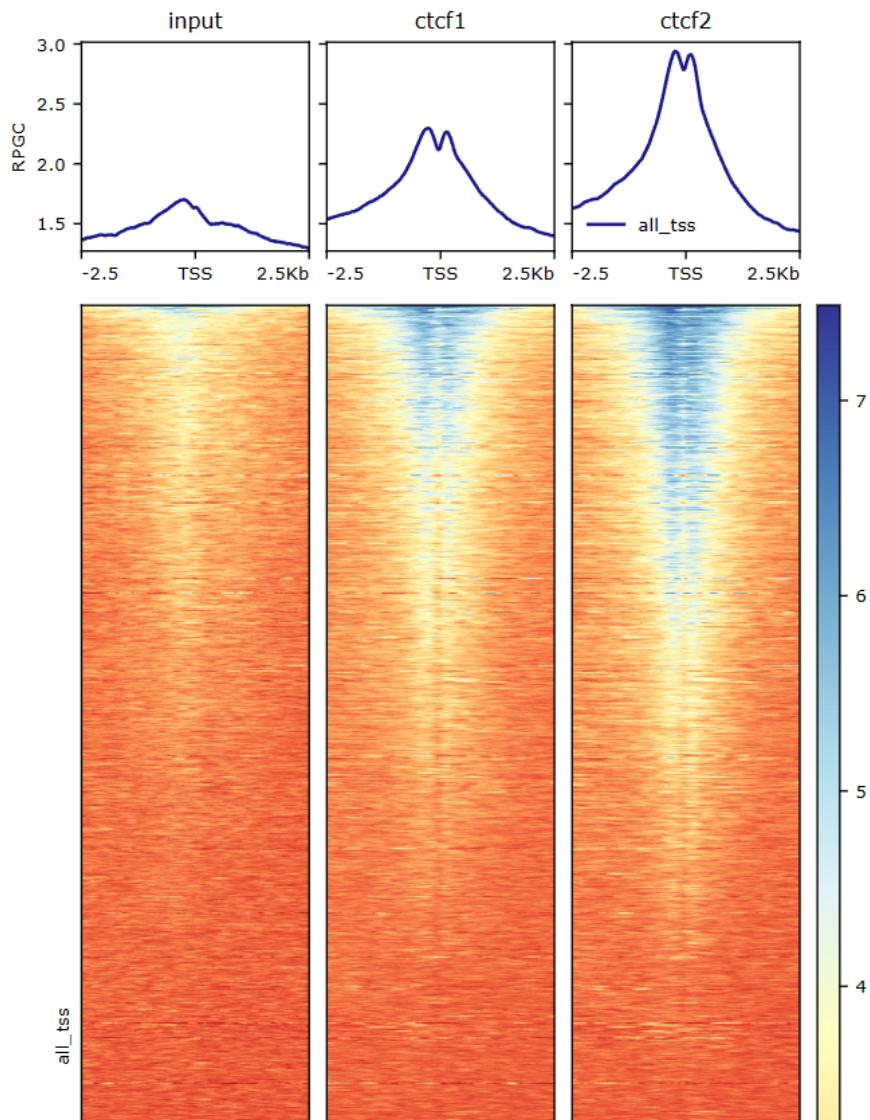
- 示例

```

plotHeatmap -m plotHP/input_rep1_rep2_TSS_Matrix.gz -o input_rep1_rep2.svg \
--yAxisLabel RPGC --regionsLabel all_tss

```

- 部分结果



- 怎么画整个基因区域的搞不定了

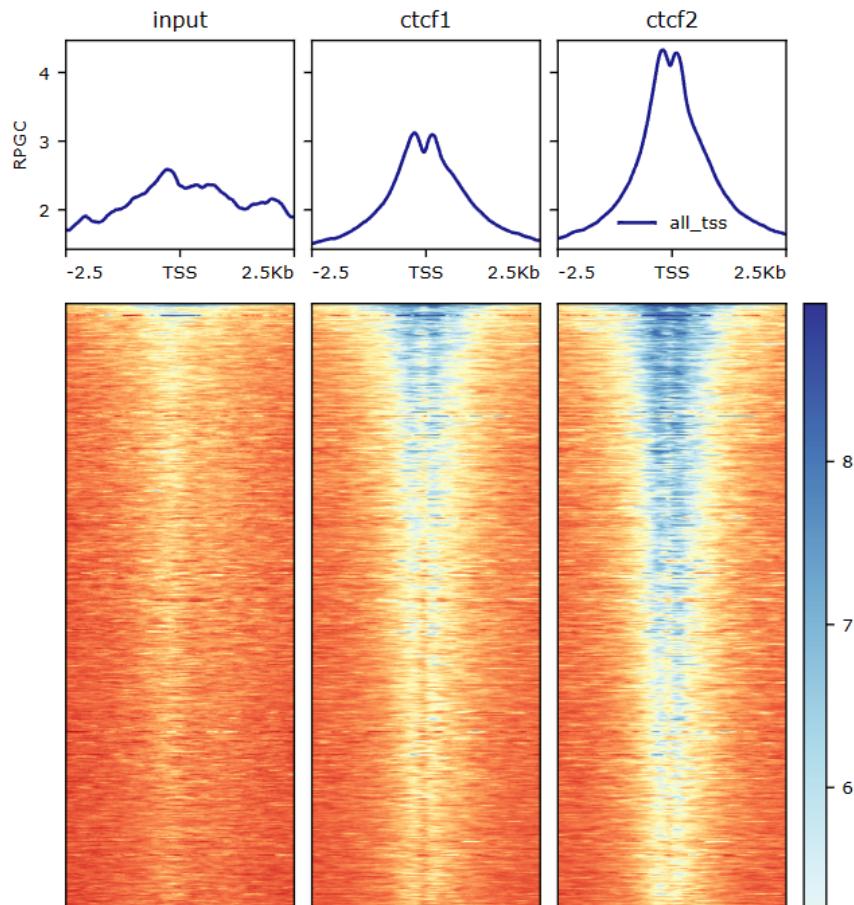
- 参考https://mp.weixin.qq.com/s/VHMaMEkz4pSmUc_Xab8Srg
- 先下载一个gtf文件试试

```
wget
https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/genes/hg38.ncbiRefSeq.gtf.gz
```

- 试试画的图和bed文件相比的情况

```
computeMatrix reference-point -S \
> ctcf_chip/deeptools/ctcf_chip_input_filtered_dedup.bw \
> ctcf_chip/deeptools/ctcf_chip_rep1_filtered_dedup.bw \
> ctcf_chip/deeptools/ctcf_chip_rep2_filtered_dedup.bw \
> -R plotHP/hg38.ncbiRefSeq.gtf -a 2500 -b 2500 --samplesLabel input
      ctcf1 ctcf2 \
> --sortRegions descend -o plotHP/input_rep1_rep2_TSS_Matrix.gz -p 16 \
> >plotHP/computeMatrix2.log 2>&1 &

plotHeatmap -m plotHP/input_rep1_rep2_TSS_Matrix.gz -o
re_input_rep1_rep2.svg \
--yAxisLabel RPGR --regionsLabel all_tss -T re_TSS
```



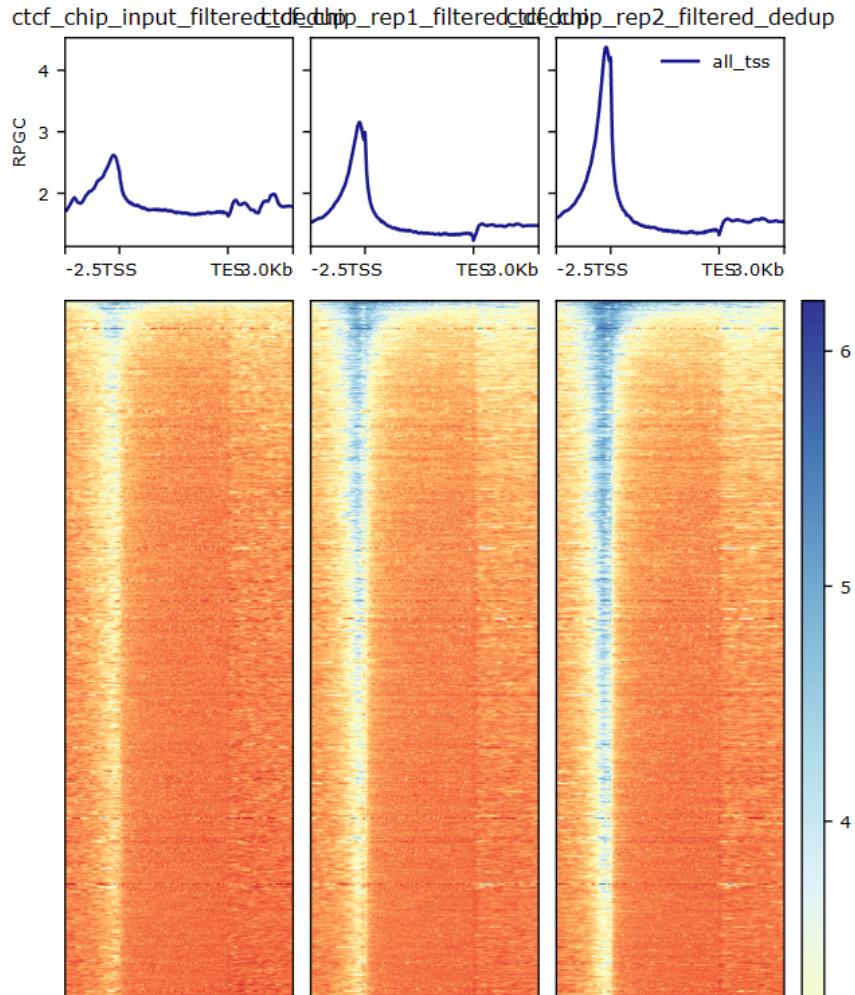
- 和之前一张图还是有点区别的，形状大抵相似，但是RPGC的值相对而言大了一点
- 还有个区别是时间久了点，能跑就行..
- 接下来试试scale-regions画整个基因的结合

```

○ computeMatrix scale-regions -S \
  ctcf_chip/deeptools/ctcf_chip_input_filtered_dedup.bw \
  ctcf_chip/deeptools/ctcf_chip_rep1_filtered_dedup.bw \
  ctcf_chip/deeptools/ctcf_chip_rep2_filtered_dedup.bw \
  -R plotHP/hg38.ncbiRefSeq.gtf \
  --beforeRegionStartLength 2500 \
  --regionBodyLength 5000 \
  --afterRegionStartLength 3000 \
  --skipzeros -o plotHP/scale_region_input_rep1_rep2_TSS_Matrix.gz \
  -p 10 >plotHP/computeMatrix3.log 2>&1 &

plotHeatmap -m plotHP/scale_region_input_rep1_rep2_TSS_Matrix.gz -o
TSS_TES.svg \
--yAxisLabel RPGC --regionsLabel all_tss -T TSS_TES

```

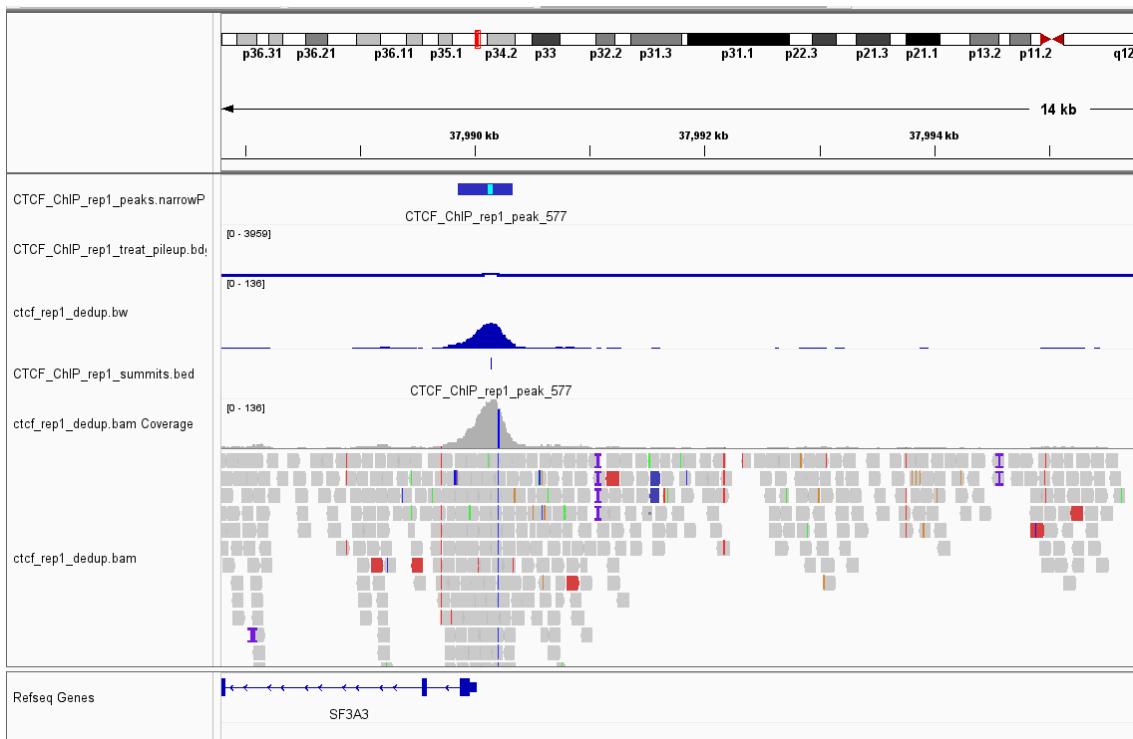


- 还不错，并且和之前的reference-point相比，虽说位置可能不大一致（因为scale过），但高度还是差不多的

6. Peak注释与motif search

IGV

- The intergrative Genomics Viewer是生物信息学中常用的结果可视化软件，只是用来看看reads都定位到哪里
- 可以同时观看一个参考基因组与比对到上面的多种测序数据，如.bw、.bam、.bed、.narrowpeak等
- 注意参考基因组的选择一定与之前比对的参考基因组一致
- 右键测序文件名，通过Change Track Height调整高度，Set Data Range调整展示值范围



- 上图是一个示例，可以看到
 - .narrowpeak显示了call Peak发现的峰值信息
 - .bw是我用bamCoverage软件生成的bw文件，体现了碱基覆盖率
 - .bam文件是比对完后生成的bam文件，注意要排序后生成index才能进IGV显示
 - .bed文件显示的是峰顶信息，放大后看到只是一个碱基的长度
 - .bdg文件是和.narrowpeak一起生成的，不知道为什么是平的..

meme

- MEME-ChIP是一款综合性的motif分析软件，集成了MEME、DREME、CentriMo、Tomtom、FIMO、SpaMo
- 适用于分析数据量较大的序列上的motif信息。首先通过MEME和DREME两款软件预测de novo motif，然后利用CentriMo识别在序列的中心区域显著富集的motif，同时采用Tomtom软件将预测到的de novo motif与指定数据库的已知motif进行比对，确定二者的相似度。最后利用FIMO软件预测motif在输入序列上的结合位点

```
# 基本格式
meme-chip [options] <primary sequence file>

# -o 输出文件夹，文件夹已存在时失败
# -oc 输出文件夹，覆盖原有文件
# -db Tomtom和CentriMo的目标数据库，若不存在不运行这俩软件
# -meme-p 线程数
# -dna -rna 设置测序文件格式，默认dna
```

- 示例
 - 生成所需序列文件

```
awk -v OFS="\t" '{$6=$2-50;$7=$3+49;print $1,$6,$7,$4}'  
callpeak/CTCF_ChIP_rep1_summits.bed > callpeak/CTCF_ChIP_rep1_motif.bed
```

即生成CTCF_ChIP_rep1_motif.bed文件，由染色体名、峰前50个碱基坐标、峰后49个碱基坐标、峰名组成

- 使用bedtools getfasta从参考基因组上取下目的序列

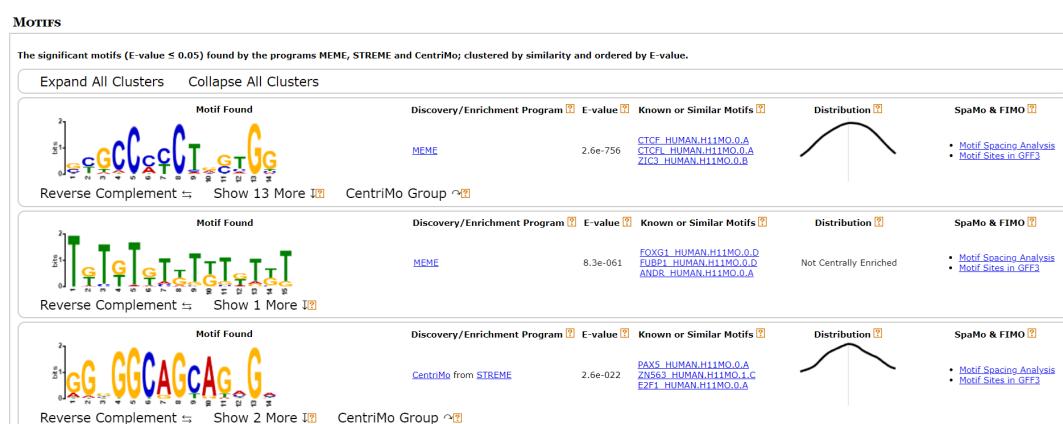
```
# 基本格式  
bedtools getfasta [OPTIONS] -fi <fasta> -bed <bed/gff/vcf>  
  
# -fi 输入fasta文件  
# -fo 输出fasta文件, 默认STDOUT  
# -bed 记录从fi中提取的范围文件, 有BED/GFF/VCF
```

```
bedtools getfasta -fi human_genome/hg38_human.fa -bed  
callpeak/CTCF_ChIP_rep1_motif.bed > callpeak/CTCF_ChIP_rep1_motif.fasta
```

- ## ○ 运行

```
meme-chip -db  
./human_genome/./meme/db/motif_databases/HUMAN/HOCOMOCOV11_full_HUMAN_  
mono_meme_format.meme \  
> -meme-p 24 -dna ./callpeak/CTCF_ChIP_rep1_motif.fasta
```

- 对于人类的motif，似乎选择这个full比较好
 - 在本文件夹下的memechip.out生成了许多文件，选取html文件打开，看到



- 存储了模体信息、预测来源、显著度、相似模体、定位相对于submmits峰值的分布等等信息
 - 若未指定db，输出的模体数量会少一点（不进行CentriMo），且没有与其他motif的比较信息

homer

- Annotation的过程主要分为两个部分
 - 首先确定到最近的TSS的距离，并将峰值分配给该基因
 - 第二步确定峰/区域中心所占区域的基因组注释
 - 提供基本注释：包括一个峰是否在TSS(转录起始位点)、TTS(转录终止位点)、外显子(编码)、5' UTR外显子、3' UTR外显子、Intronic或Intergenic，这是许多研究人员感兴趣的常见注释。
 - 提供详细标注：同样考虑重复元素和CpG岛
 - 优先级为
 - TSS (by default defined from -1kb to +100bp)
 - TTS (by default defined from -100 bp to +1kb)
 - CDS Exons
 - 5' UTR Exons
 - 3' UTR Exons
 - **CpG Islands
 - **Repeats
 - Introns
 - Intergenic
 - 并不能更改TSS区域的定义(-1kb到+100bp)，但可以自行通过按“到最近TSS的距离”列对EXCEL中的注释输出进行排序

- configureHomer.pl

- 可通过该软件安装、更新homer至configureHomer.pl所在文件夹
- 可查看目前已安装包的信息

```
perl /home/public/homer/configureHomer.pl --list
```

- 可安装数据集

```
perl /home/public/homer/configureHomer.pl -install danRer11
```

- annotatePeaks.pl

```
# 基本用法
annotatePeaks.pl <peak file | tss> <genome version> [additional options...]

#前两个文件一定要为<peak file | tss> <genome version>
# <peak file | tss> 为峰值信息，可为HOMER flies或6列的BED (chr、start、end、
PeakID、not used、Strand)
# <genome version> 基因组信息，如hg38
# -gtf <gtf filename> 如果指定了一个GTF文件，HOMER将使用GTF文件中的TSS来确定到最近的
TSS的距离并定义TSS/TTS/外显子/内含子，但会使用original HOMER annotation文件进行详细
注释，因为GTF没有repeats序列信息
# 默认输出至STDOUT，所以需要重定向
```

- 示例

- 注释

```
annotatePeaks.pl callpeak/CTCF_ChIP_rep1_peaks.narrowPeak hg38 >
annotation/peak_annotation.txt
```

- 输出的文本文件中包含：

A	B	C	D	E	F	G	H	I	J
PeakID (cmd=annotatePeaks.pl)	Chr	Start	End	Strand	Peak Score	Focus R	Annotation	Detailed Annotation	Distance to TSS
CTCF_ChIP_rep1_peak_6296	chr17	18625542	18626012	+	2068	NA	promoter-TSS (NR_036647)	promoter-TSS (NR_036647)	-160
CTCF_ChIP_rep1_peak_1170	chr1	155017382	155017930	+	1669	NA	TTS (NM_144622)	TTS (NM_144622)	15026
CTCF_ChIP_rep1_peak_3665	chr12	53675899	53676466	+	1561	NA	promoter-TSS (NM_001369757)	promoter-TSS (NM_001369757)	-99
CTCF_ChIP_rep1_peak_1005	chr1	114346387	114346929	+	1434	NA	Intergenic	Intergenic	164545
CTCF_ChIP_rep1_peak_6730	chr17	49569723	49570527	+	1408	NA	intron (NR_103773, intron 1 of 2)	CpG	3939
CTCF_ChIP_rep1_peak_1666	chr1	225474868	225475472	+	1383	NA	Intergenic	CpG	-46315
CTCF_ChIP_rep1_peak_6757	chr17	50397382	50397868	+	1382	NA	promoter-TSS (NM_018509)	promoter-TSS (NM_018509)	-102
CTCF_ChIP_rep1_peak_2462	chr10	117216531	117217155	+	1285	NA	Intergenic	CpG	19354
CTCF_ChIP_rep1_peak_8032	chr19	33777793	33778354	+	1265	NA	Intergenic	MIRISINE MIR	-18797
CTCF_ChIP_rep1_peak_8129	chr19	38402605	38403439	+	1231	NA	promoter-TSS (NM_174905)	promoter-TSS (NM_174905)	-113
CTCF_ChIP_rep1_peak_1720	chr1	228139826	228140271	+	1218	NA	promoter-TSS (NM_001242839)	promoter-TSS (NM_001242839)	-36
CTCF_ChIP_rep1_peak_2264	chr10	94362628	94363280	+	1181	NA	promoter-TSS (NM_022451)	promoter-TSS (NM_022451)	-15
CTCF_ChIP_rep1_peak_8085	chr19	35947373	35947818	+	1166	NA	TTS (NM_024509)	TTS (NM_024509)	480
CTCF_ChIP_rep1_peak_5129	chr15	77633323	77633874	+	1155	NA	promoter-TSS (NM_032808)	promoter-TSS (NM_032808)	-686
CTCF_ChIP_rep1_peak_6302	chr17	19362805	19363256	+	1150	NA	promoter-TSS (NM_001321214)	promoter-TSS (NM_001321214)	-292
CTCF_ChIP_rep1_peak_12386	chr5	140042909	140043826	+	1139	NA	promoter-TSS (NM_013981)	promoter-TSS (NM_013981)	-68
CTCF_ChIP_rep1_peak_7721	chr19	10416435	10417239	+	1125	NA	promoter-TSS (NM_001243121)	promoter-TSS (NM_001243121)	64
CTCF_ChIP_rep1_peak_8924	chr2	73177296	73177794	+	1123	NA	Intergenic	CpG	-25029

I	J	K	L	M	N	O	P	Q	R	S
Detailed Annotation	Distance to TSS	Nearest PromoterID	Entrez ID	Nearest Unigene	Nearest RefSeq	Nearest Ensembl	Gene Name	Gene Alias	Gene Description	Gene Type
promoter-TSS (NR_036647)	-160	NR_036647	284047	Hs440012	NR_036647	ENSG00000154874	CDC14B	-	coiled-coil domain cont; pseudo	
TTS (NM_144622)	15026	NM_001256455	51043	Hs729279	NM_001252406	ENSG00000160685	ZBTB7B	CKR0XTHPOK ZBTB zinc finger and BTB dom protein-coding		
promoter-TSS (NM_001369757)	-99	NR_163137	517	Hs524464	NM_005176	ENSG00000135390	ATP5MC2	ATP5A ATPG2	ATP synthase membrane protein-coding	
Intergenic	164545	NM_015906	51592	Hs26837	NM_015906	ENSG00000197323	TRIM33	ECTOPIC T(RFG)TF	tripartite motif contains protein-coding	
CpG	3939	NR_103773	10028886	Hs720309	NR_103773	ENSG00000249906	LOC100288866	-	uncharacterized LOC100 ncRNA	
CpG	-46315	NM_194442	3930	Hs435166	NM_002296	ENSG00000143815	LBR	C14SR DHCR14B LMB	lamin B receptor	protein-coding
promoter-TSS (NM_018509)	-102	NM_018509	55379	Hs370927	NM_018509	ENSG00000108829	LRRC59	PRO185 p34	leucine rich repeat conta	protein-coding
CpG	19354	NM_181840	338567	Hs449650	NM_181840	ENSG00000186795	KCNK18	K2p18.1 MGR13 TRE	potassium two pore domain protein-coding	
MIRISINE MIR	-18797	NM_001129995	79047	Hs221873	NM_024076	ENSG00000153885	KCTD15	-	potassium channel tetra	protein-coding
promoter-TSS (NM_174905)	-113	NM_001351675	147965	Hs355162	NM_174905	ENSG00000130244	FAM98C	-	family with sequence sim	protein-coding
promoter-TSS (NM_001242839)	-36	NM_001242839	2987	Hs37933	NM_000858	ENSG00000143774	GUK1	GMK	guanylate kinase 1	protein-coding
promoter-TSS (NM_022451)	-15	NM_022451	64318	Hs74899	NM_022451	ENSG00000173145	NOCL	AD24 C10orf117 FAD	NOC3 like DNA replicativ	protein-coding
TTS (NM_024509)	480	NM_136525	10537238	Hs641592	NR_136525	ENSG00000105372383	-	-	uncharacterized LOC105 ncRNA	
promoter-TSS (NM_032808)	-686	NM_032808	84893	Hs656765	NM_032808	ENSG00000169783	LINGO1	LERN1 LRRN6A MT	leucine rich repeat and	protein-coding
promoter-TSS (NM_001321214)	-292	NM_001330149	27077	Hs304967	NM_015681	ENSG00000108641	B9D1	B9 EPB9 BTS27 MK	B9 domain containing 1	protein-coding
promoter-TSS (NM_013981)	-68	NM_013983	9542	Hs408515	NM_004883	ENSG00000158458	NRG2	DON1 HRG2 NTAK	neuregulin 2	protein-coding
promoter-TSS (NM_001243121)	64	NM_001243121	5141	Hs89901	NM_006202	ENSG00000065989	PDE4A	DPDE2 PDE4 PDE46	phosphodiesterase 4A	protein-coding
CpG	-25029	NM_001134462	344022	Hs694384	NM_001134462	ENSG00000214513	NOTO	-	notochord homeobox	protein-coding

- Peak ID
- Chromosome
- Peak start position
- Peak end position
- Strand
- Peak Score
- FDR/Peak Focus Ratio/Region Size
- Annotation (i.e. Exon, Intron, ...)
- Detailed Annotation (Exon, Intron etc. + CpG Islands, repeats, etc.)
- Distance to nearest RefSeq TSS
- Nearest TSS: Native ID of annotation file
- Nearest TSS: Entrez Gene ID
- Nearest TSS: Unigene ID
- Nearest TSS: RefSeq ID
- Nearest TSS: Ensembl ID
- Nearest TSS: Gene Symbol
- Nearest TSS: Gene Aliases
- Nearest TSS: Gene description
- Additional columns depend on options selected when running the program.

- 而后提取EntrezID简单做一个GO就结束了

```
setwd("C:/Users/17364/Desktop")
library(clusterProfiler)
library(org.Hs.eg.db)
library(enrichplot)
library(ggplot2)

gene_list <- read.table("peak_annotation.txt", header = T, sep = "\t",
fill = T)
gene_list <- gene_list$Entrez.ID
```

```

gene_list <- na.omit(gene_list)
gene_list <- as.character(gene_list)

GO <- enrichGO(gene = gene_list,
               OrgDb = org.Hs.eg.db,
               keyType = "ENTREZID",
               pvalueCutoff = 0.01,
               qvalueCutoff = 0.01,
               ont = "all")

pdf(file = "GO_result.pdf", width = 8, height = 10)
dotplot(GO, x = "GeneRatio", color = "p.adjust", showCategory = 10, split
= "ONTOLOGY") +
  facet_grid(ONTOLOGY~., scale = 'free')
dev.off()

```

