Main Scripts During Postgraduate

3.绘图

1、单倍型分析

2、单倍型聚类

①NJ树

②contree

③豪哥的办法

④VCF2Dis

3、单倍型网络图

曼哈顿图系列

1、TASSEL结果画曼哈顿图

2、注释显著位点

3、R语言CMplot包绘制曼哈顿图

5、Candihap画图

6、单倍型群体结构可视化-haplostrips

https://bitbucket.org/dmarnetto/haplostrips/src/master/

7、fastphase和beagle原理相同-提取候选区段

8、fst

manhadun:https://www.plob.org/article/21662.html

图片组合

！计算核苷酸多样性（π）

画图1：https://cloud.tencent.com/developer/article/1593265

折线图

https://www.jianshu.com/p/7b6701ae593b

①文件格式

批量blast

1、GWAS前：文件准备工作（过滤按照第一次过滤，再进行填充）

1.加ID号

https://www.jianshu.com/p/2677dc3b1383 方法1

```

方法二
/data/apps/bcftools/1.9/bin/bcftools annotate --set-id +'%CHROM\_%POS' 938.raw.vcf.gz |/
```

```
gunzip
## add variance id to vcf file（看vcf有无ID，需要加上）
perl lecture06_07_add_id.pl 516sample.vcf combine_data.vcf

```

最终过滤版本

```
gunzip
## add variance id to vcf file（看vcf有无ID，需要加上）
perl lecture06_07_add_id.pl /w/00/g/g01/user305/Result/00.450_rawvcf/combine_450raw.vcf 
## filter by quality and depth（最小质量值、最小/最大深度、平均深度）
## 可以用le log查看
## plink对vcf过滤比较弱，如果要用plink先用vcftools进行过滤然后再用plink进行其他操作
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/TResul

开始
## keep only indels 分开indel和snp也可以使用gatk
## 要输出vcf文件一定要加上--recode
##vcftools --vcf filter1.vcf --keep-only-indels --recode --recode-INFO-all --out lecture
## keep only snps
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf filter.vcf --remove-indels --reco


#1.SNP过滤
## missing rate， maf，allele number filtering
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf snp_only_filter.vcf --max-missing
```

查看错误位点

```
## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf new.filter.vcf  --chr At1 --from-
```

填充

/w/00/g/g01/user305/Result/TResult/01.filter_method2_mer5.vcf/01_afilter

```
cat snp_bi.filter.vcf | perl -pe "s/\s\.:/\t.\/.:/g" | /data/apps/htslib/1.9/bin/bgzip -
```

2、课题组过滤

```
***重要***
每一步跑完，记得用bcftools输出一个stats文件，以便于掌握过滤之后位点的情况
/data/apps/bcftools/1.9/bin/bcftools stats new.filter.vcf.gz>new.filter.vcf.gz.stats
```

```
1.标记质量值小于30和indel5bp
/data/apps/bcftools/1.9/bin/bcftools filter -O z  -o genetalks-mihoutao.filter_variant.v
2.删除低质量和indel5bp
gzip -dc genetalks-mihoutao.filter_variant.vcf.gz |grep -v 'LOWQUAL'|grep -v 'SnpGap' |/
```

```

```

```
3.2比对个体
mkdir output.01.bam
$time -vo output.01.bam/Pd14008_04.01.bam.time bash -c "$bwa mem -t 16 -R '@RG\tID:Pd140
$time -vo output.01.bam/Pd14008_04.02.bam.time $java -Xmx4G -jar  $picard SortSam COMPRE
$time -vo output.01.bam/Pd14008_04.03.bam.time $java -Xmx30G -jar $picard MarkDuplicates
```

```
$time -vo output.01.bam/Pd14008_04.04.bam.time $java -Xmx4G -jar  $picard SortSam COMPRES
$time -vo output.01.bam/Pd14008_04.05.bam.depth.time bash -c "$samtools depth output.01.
```

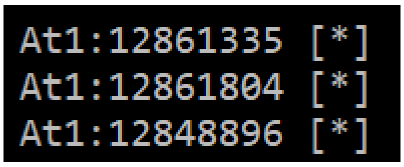
```
/data/apps/bcftools/1.9/bin/bcftools view -m2 -M2 -v snps genetalks-mihoutao.filter_vari

4.依据深度过滤（根据公司返回的报告）
过滤深度大于平均深度3倍的位点和小于平均深度1/3的位点
#####这里要注意，个体原始文件顺序和算深度的数据，个体顺序要一致，即脚本输出顺序与vcf文件里面的个
首先拿平均深度
过滤
注意输入输出都为gz文件
python filterstep.depth.py Indepth.txt Mp.cp.Low.Gap.2all.vcf.gz Mp.QUL30.indel5bp.alle2

```

```
5.个性化过滤
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf Mp.QUL30.indel5bp.alle2.depth.v
```

```
然后使用下面两个选项对vcf文件保留或者删除样品。
--keep <filename>   保留样品
--remove <filename>   删除样品
代码如下：
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/00.448
```

```
6.画structure
## missing rate and maf filtering
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/01.gx_firs
## make bed and LD pruning
/w/00/g/g01/user305/soft/plink/plink --vcf snp.int0.8.maf0.05.vcf --indep-pairwise 100 5

##extract prune in snps and recode to admixture format（转成ped格式）
```

```
/w/00/g/g01/user305/soft/plink/plink --bfile snp.int0.8.maf0.05 --extract snp.int0.8.maf


#课题组方法无法完整的画图，先得到群体结构的结果
for i in {1..15}; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666 prunData.
grep -h CV log*.out
```



```
At1:12861335 [*]
At1:12861804 [*]
At1:12848896 [*]
```

```
of A, C, T, G, or N characters at At1:10437381 [*]
```

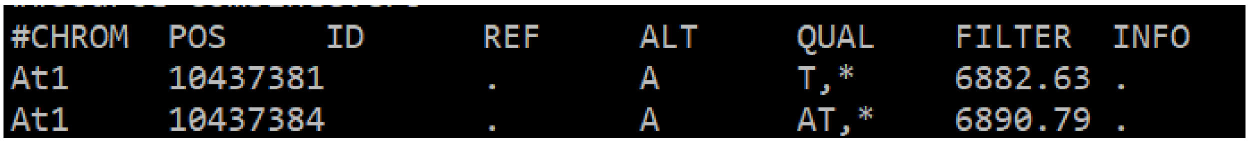
3.过滤2/3再beagle再过滤1/3

```
source ~/.profile
conda

#0.下载下来的其实是.gz文件，后续命令可以直接调用gz文件，也可以解压之后再调用，但必须每个都先做in
gatk IndexFeatureFile -I /w/00/g/g01/user305/Result/TResult/mer5.vcf/combine_data.vcf

## 过滤1：gatk VariantFiltration : --filter-expression 参考：https://www.omicsclass.com/a

gatk --java-options "-Xmx4g" VariantFiltration -R /w/00/g/g01/user305/data/ref/Tibet.fas


gzip all.raw.gatk.vcf


zcat all.raw.gatk.vcf.gz |awk '$0~/#/ || ($7 =="PASS"){print $0}' |gzip - > all.raw.gatk

#过滤2：vcfutils.pl
# -w INT    SNP within INT bp around a gap to be filtered [3]
# -W INT    window size for filtering adjacent gaps [10]

/data/apps/bcftools/1.9/bin/vcfutils.pl varFilter -w 5 -W 10 "gzip -dc all.raw.gatked.vc

```

```

```

```

# 2. 补缺失impute，使用Beagle软件进行补缺失
gzip combine_data.vcf

#zcat input.rmUnusedAlt.vcf.gz | sed -i "/\tAt1__9166711\t/d" input.rmUnusedAlt.vcf.gz> 


cat filt_done.vcf | perl -pe "s/\s\.:/\t.\/.:/g" >output.vcf
sed -e '/At1__10437381/d' filt_done.vcf > binput.vcf
sed -e '/At1__12861335/d' filt_done.vcf > binput.vcf
sed -e '/At1__12861804/d' filt_done.vcf > binput.vcf
sed -e '/At1__12848896/d' filt_done.vcf > binput.vcf
sed -e '/At1__10437381/d' filt_done.vcf > binput.vcf


https://zhuanlan.zhihu.com/p/109071456
#vcf转为 ped/map
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/01.filter_

## ped/map转换为bed/bim/fam
/w/00/g/g01/user305/soft/plink/plink --file tran --make-bed --out tran_test --allow-extr

/w/00/g/g01/user305/soft/plink/plink --bfile tran_test --list-duplicate-vars ids-only su
/w/00/g/g01/user305/soft/plink/plink --bfile tran_test --alleleACGT --snps-only just-acg
## bed/bim/fam 转为 vcf
/w/00/g/g01/user305/soft/plink/plink --bfile wbinput --export vcf --out bbinput --allow-



```

beagle填充REF field is not a sequence of A, C, T, G, or N characters at At1:10437381 - Google 搜索 ∨



```
#CHROM  POS       ID      REF     ALT     QUAL    FILTER  INFO
At1     10437381          .       A       T,*     6882.63 .
At1     10437384          .       A       AT,*    6890.79 .
```

提取指定区间

```
## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf combine_raw.vcf  --chr At1 --from

```

```

#下载件
wget https://faculty.washington.edu/browning/beagle/beagle.28Sep18.793.jar
# Pre-phasing
#该步骤为非必需步骤，对输入的VCF文件进行预phasing，理论上可以提高基因型推断质量
# 考虑换beagle最新款
-Xms2g -Xmx30g
/data/apps/jdk/jdk1.8.0_131/bin/java -Xms2g -Xmx30g -jar /w/00/g/g01/user305/soft/beagle


#注意：如果提示没有安装java，需要额外执行如下两句：
sudo apt update
sudo apt install default-jre     4373
```

```

#过滤3：vcftools
#--max-missing Exclude sites on the basis of the proportion of missing data
#(defined to be between 0 and 1, where 0 allows sites that are completely missing
#and 1 indicates no missing data allowed).

!!!!!#第3次过滤分离变异前（原始）
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf all.varFilter.vcf.gz --recode -

```

```
#分离indel与SNP 到不同的文件
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --remove-indels --recode --recode-INFO-a


zcat demo.filter.vcf.gz|wc -l
gunzip demo.filter.vcf
```

提取指定个体

```
然后使用下面两个选项对vcf文件保留或者删除样品。
--keep <filename>    保留样品
--remove <filename>    删除样品
代码如下：
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /data/apps/vcftools/vcftools-0.1.1

```

提取候选基因

```
awk '$3=="gene"' Tibet.gff3 |awk '$1=="At1"&& $4>20580546 && $5<20620546' > /w/00/g/g01/u

```

4.annovar注释

```
# 安装gffread和gtfToGenePred
#git clone https://github.com/gpertea/gffread
#下载annovar并解压缩: unzip zipped_file.zip -d unzipped_directory

#提取目标参考基因组
python /w/00/g/g01/user305/scripts/extract_seq.py


# build ANNOVAR index
/w/00/g/g01/user305/soft/gffread/gffread Tibet.gff3 -T -o Tibet.gtf

```

```
/w/00/g/g01/user305/soft/gtfToGenePred -genePredExt /w/00/g/g01/user305/soft/Annovar/tib

perl retrieve_seq_from_fasta.pl --format refGene --seqfile tibetdb/Tibet.fasta  tibetdb/


##为了避免别的脚本权限问题，干脆全部777
#chmod 777 * 成功解决权限问题。
#一定要在当前目录下进行
perl table_annovar.pl /w/00/g/g01/user305/Result/TResult/1.gx_first_mer5.vcf/05.gx_indoo




#测试之前成功的： perl table_annovar.pl /w/00/g/g01/user305/Result/TResult/first_filter_me
```

5.群体结构分析

https://zhuanlan.zhihu.com/p/109164405

structure（改50/10/0.2

```
2.1 利用plink按照LD衰减过滤vcf文件（每100kb保留50个snp,筛选标准是R2为0.2）
/w/00/g/g01/user305/soft/plink/plink --vcf out_417individual.vcf --indep-pairwise 100 50
/w/00/g/g01/user305/soft/plink/plink --vcf out_417individual.vcf --recode vcf-iid --extr




2.2 利用plink转换为structure可以识别的格式
/w/00/g/g01/user305/soft/plink/plink --bfile LD_filter --extract LD_filter.prune.in --ou
2.3 structure运行，linux只需这个命令（参数都在文件里面）
structure -m test_mainpara1_1 -e extraparams
```

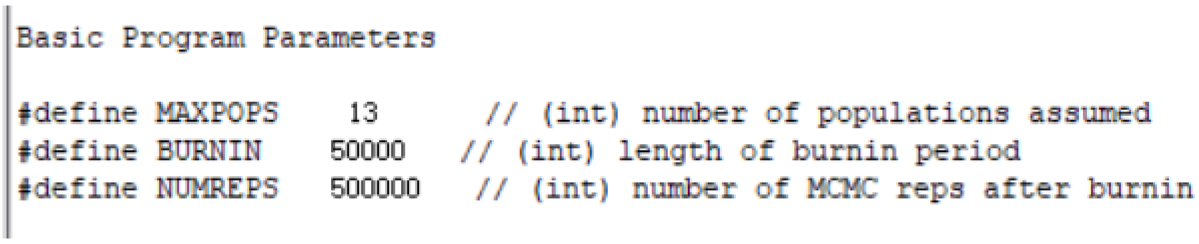
改

```
2.1 利用plink按照LD衰减过滤vcf文件（每100kb保留50个snp,筛选标准是R2为0.2）
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/3.HJ/01.29
/w/00/g/g01/user305/soft/plink/plink --bfile snp --indep-pairwise 50 10 0.2 --out snp.ld
/w/00/g/g01/user305/soft/plink/plink --bfile snp --extract snp.ldprun.prune.in --out pru
```

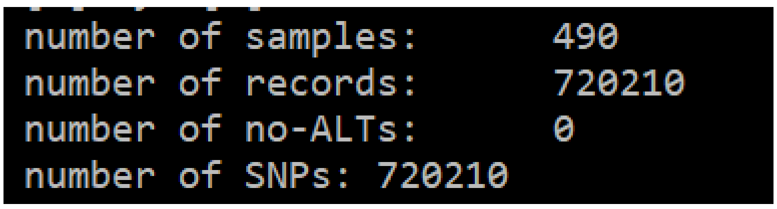
```



sed -i 's/^At//g' prunData.bim
for i in {1..25}; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666 prunData.

grep -h CV log*.out

```

~~**plink --bfile snp --maf 0.05 --geno 0.5 --indep-pairwise 100 10 0.05 --out snp.ldprun --allow-extra-chr**~~
~~# extract prun in snps~~
~~# prune出来的以后转成bed文件；record有很多种格式~~
~~plink --bfile snp --extract snp.ldprun.prune.in --out prunData --make-bed --allow-extra-chr~~
~~# vcf 转成structur~~
~~--recode12~~
~~#admixture接受plink格式，bed格式~~
~~/w/00/g/g01/user305/soft/plink/plink --vcf admixture.in.vcf --make-bed --out demo.admixture.in~~

配置文件



```
Basic Program Parameters

#define MAXPOPS     13       //  (int)  number of populations assumed
#define BURNIN     50000    //  (int)  length of burnin period
#define NUMREPS    500000    //  (int)  number of MCMC reps after burnin
```



```
number of samples:      490
number of records:      720210
number of no-ALTs:      0
number of SNPs:  720210
```

```
# 查看位点信息
/data/apps/bcftools/1.9/bin/bcftools stats demo.impute.vcf>demo.impute.vcf.stats

#outfile
/w/00/g/g01/user305/Result/TResult/mer5.vcf/2.third_filter/03.structure/str_1_1
#infile
/w/00/g/g01/user305/Result/TResult/mer5.vcf/2.third_filter/03.structure/prunData.recode.
```

```
#样本量
490
#标记数
720210


```

①自己研究的方法

Linkage disequilibrium based SNP pruning

`--indep-pairwise 50 5 0.5`

`--indep-pairwise 50 5 0.2`

a) consider a window of 50 SNPs

b) calculate LD between each pair of SNPs in the window

b) remove one of a pair of SNPs if the LD is greater than 0.5

c) shift the window 5 SNPs forward and repeat the procedure

```
## missing rate and maf filtering
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/mer5.vcf/3
## make bed and LD pruning
/w/00/g/g01/user305/soft/plink/plink --vcf snp.int0.8.maf0.05.vcf --indep-pairwise 100 5

##extract prune in snps and recode to admixture format（转成ped格式）
/w/00/g/g01/user305/soft/plink/plink --bfile snp.int0.8.maf0.05 --extract snp.int0.8.maf

#R需要先安装RSvgDeviece包
/data/apps/R/3.5.1/bin/R
>install.packages("RSvgDevice")
选的兰州

#for i in {1..30}; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666 /w/00/g/
```

```
#perl plotAdmixtureCV.pl /data/apps/R/3.5.1/bin/Rscript output output.cv.pdf    #要把所有
#课题组方法无法完整的画图，先得到群体结构的结果
for i in {1..30}; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666 /w/00/g/g
grep -h CV log*.out

30
```

```
##修改bim的名称,否则会报错
#sed -i 's/^At//g' /w/00/g/g01/user305/Result/TResult/mer5.vcf/3.1_second_filter/03.stru
#for K in {1..50}; do /data/apps/admixture/1.3.0/admixture --cv  /w/00/g/g01/user305/Resu
#grep -h CV log*.out
选择CV error最低的对应K值，.Q文件为要使用到的文件

#群体结构绘图
#先生成一个csv文件，然后再第一列添加上分组信息
paste <(cut -f1 prunData.nosex) prunData.10.Q |sed 's/^/G\t/g' |sed 's/\s/,/g'>plotQ.csv
#plotQ.csv可以上传到以下网址进行可视化：
http://omicsspeaks.com/strplot2/

```

K in {1..36}

②课题组方法

```
#计算LD
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf demo.impute.vcf.gz --ld-window-
删除LD位点
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf Mp.QUL30.indel5bp.alle2.depth.r


```

```
$bgzip = /data/apps/htslib/1.9/bin/bgzip
$vcftools = /data/apps/vcftools/vcftools-0.1.15/bin/vcftools
$plink = /data/apps/plink/20181202/plink
$admixture = /data/apps/admixture/1.3.0/admixture
```

```
$Rscript = /data/apps/R/3.5.1/bin/Rscript
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf demo.impute.vcf.gz --stdout --tl
```

1、过滤并生成bgzip文件

> 对于大型的VCF文件而言，如何快速访问其中的记录也是个难点。tabix可以对VCF文件构建索引，索引构建好之后，访问速度会快很多。tabix对VCF文件建立索引的用法如下
> 注意输入的VCF文件必须是使用bgzip压缩之后的VCF文件，生成的索引文件为view.vcf.gz.tbi， 后缀为.tbi。
> 构建好索引之后，可以快速的获取指定区域的记录，示例如下

https://www.jianshu.com/p/07640917764d **Tabix— bgzip操作**

```
#建立索引
tabix -p vcf view.vcf.gz
#1、获取位于11号染色体的SNP位点
tabix view.vcf.gz 11
#2、获取位于11号染色体上突变位置大于或者等于2343545的SNP位点
tabix view.vcf.gz 11:2343545
#3、获取位于11号染色体上突变位置介于2343540到2343596的SNP位点
tabix view.vcf.gz 11:2343540-2343596

```

```
1、重命名
perl vcfRenameChr.pl /data/apps/htslib/1.9/bin/bgzip variants.demo.impute.vcf.gz output/l
2、转换格式
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --gzvcf output/Lmar.rename.chr.vcf.gz --
/data/apps/plink/20181202/plink --noweb --file output/Lmar.rename.plink --recode12 --out
for i in 1 2 3 4 5 6 7 8 9; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666
perl plotAdmixture.pl /data/apps/R/3.5.1/bin/Rscript output/Lmar.rename.plink12.ped Lmar
perl plotAdmixtureCV.pl /data/apps/R/3.5.1/bin/Rscript output output.cv.pdf     #要把所有的



```

6、亲缘关系分析-GCTA

```
## convert to plink bed（GCTA接受bed格式，此处使用过滤后的vcf文件）
/w/00/g/g01/user305/soft/plink/plink --vcf demo.impute.vcf --make-bed --out snp --allow-


## gcta grm matrix
①输出亲缘关系矩阵②和③为文件前缀④常染色体的数目此处应该为5
/w/00/g/g01/user305/soft/gcta_1.91.6beta/gcta_1.91.6beta/gcta64 --make-grm-gz --out snp.


```

```
## convert to normal format
setwd("D:\\work_dir\\GWAS\\00.project\\000000.hifi\\01.gx\\02.leaves\\11.kinship")
library(reshape2)
tmp <- read.table(gzfile("snp.gcta.grm.gz"), header = F, stringsAsFactors = F)
ids <- read.table("snp.gcta.grm.id", header = F, stringsAsFactors = F)
tmp <- tmp[,c(1,2,4)]
result_matrix <- acast(tmp, V1~V2, value.var="V4", drop = F)
makeSymm <- function(m) {
  m[upper.tri(m)] <- t(m)[upper.tri(m)]
  return(m)
}

```

```
#转换为全矩阵
result_full <- makeSymm(result_matrix)
diag(result_full) <- 2  #这一步选做
result_df <- as.data.frame(result_full)
row.names(result_df) <- ids$V2
colnames(result_df) <- ids$V2
#生成的gcta.kinship.txt可以直接用作GWAS分析
write.table(result_df, file = "gcta.kinship.txt", row.names = T, col.names = NA, sep = "

```

```
## convert to plink bed（GCTA接受bed格式，此处使用过滤后的vcf文件）
/w/00/g/g01/user305/soft/plink/plink --vcf demo.impute.vcf --make-bed --out snp --allow-
```

```

##修改bim的名称,否则会报错
sed -i 's/^At//g' snp.bim
## gcta grm matrix（PCA分析用的--make-grm结果，而非gz后的，所以重新跑）
/w/00/g/g01/user305/soft/gcta_1.91.6beta/gcta_1.91.6beta/gcta64 --make-grm --out /w/00/g
## gcta pca（保留前20个PCA）
/w/00/g/g01/user305/soft/gcta_1.91.6beta/gcta_1.91.6beta/gcta64 --grm snp.gcta --pca 20
## snp.gcta.eigenval is eigenvalue file
## snp.gcta.eigenvec is eigenvector file


```

7、PCA绘制

```
## convert results to normal data frame
setwd("F:/works/developing/course/gwas/data/lecture08")
eigvec <- read.table("snp.gcta.eigenvec", header = F, stringsAsFactors = F)
colnames(eigvec) <- c("FID", "Sample", paste0("PC", 1:20))
write.table(eigvec[2:ncol(eigvec)], file = "gcta.eigenvector.xls", sep = "\t", row.names

```

```
eigval <- read.table("snp.gcta.eigenval", header = F)
pcs <- paste0("PC", 1:nrow(eigval))
eigval[nrow(eigval),1] <- 0
percentage <- eigval$V1/sum(eigval$V1)*100
eigval_df <- as.data.frame(cbind(pcs, eigval[,1], percentage), stringsAsFactors = F)

```

```
names(eigval_df) <- c("PCs", "variance", "proportion")
#str(eigval_df)为字符型，所以下面要换成数字型

eigval_df$variance <- as.numeric(eigval_df$variance)
eigval_df$proportion <- as.numeric(eigval_df$proportion)
write.table(eigval_df, file = "gcta.eigenvalue.xls", sep = "\t", quote = F, row.names =

```

2.GWAS关联分析

fast-lmm(51.19)

```
## convert vcf to tped format【先用plink将vcf格式转换为tped格式，不能接受vcf格式但是ped，bed
plink --vcf lecture09_sort.vcf --recode 12 transpose --output-missing-genotype 0 --out f
## create fastlmm input trait file in R【创造特有的表型文件】
setwd("F:/works/developing/course/gwas/data/lecture09")
tfam <- read.table("fastlmm_in.tfam", header = F, stringsAsFactors = F)
tr <- read.table("lecture09.trait.txt", header = T, check.names = F, stringsAsFactors = 
tr <- tr[match(tfam$V1, tr$`<Trait>`),] ## sort by tfam
tr[tr == -999] <- NA
tre <- cbind(tr[,1], tr)
for(i in 3:ncol(tre)){
  file_name <- paste0(“fastlmm.trait.", names(tre)[i], ".txt")
  write.table(tre[, c(1, 2, i)], file = file_name, col.names = F, row.names = F, sep = "
}
## run fast-lmm gwas with kinship only
fastlmmc -tfile fastlmm_in  -pheno fastlmm.trait.Trait04.txt -tfileSim fastlmm_in -simOu
## prepare covariant in R
setwd("F:/works/developing/course/gwas/data/lecture09")
tfam <- read.table("fastlmm_in.tfam", header = F, stringsAsFactors = F)
covs <- read.table("lecture09.Q.txt", header = T, check.names = F, stringsAsFactors = F,
covs <- covs[match(tfam$V1, covs$`<Trait>`), ]
covs <- cbind(covs[,1], covs[,1], covs[,2:ncol(covs)]) ## for fast-lmm
write.table(covs[, -ncol(covs)], file = "fastlmm.cov.txt", col.names = F, row.names = F,
## run fast-lmm gwas with kinship and Q-matrix
fastlmmc -tfile fastlmm_in -pheno fastlmm.trait.Trait04.txt -tfileSim fastlmm_in -simOut

```

EMMAX

```
## convert vcf to tped format
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/01.gx_firs
## kinship estimate using emmax-kin-intel64【软件自己算的kinship】
emmax-kin-intel64 emmax_in -v -d 10 -o emmax_in.BN.kinf
## create emmax input trait file in R【脚本同上】
setwd("F:/works/developing/course/gwas/data/lecture09")
tfam <- read.table("emmax_in.tfam", header = F, stringsAsFactors = F)
tr <- read.table("lecture09.trait.txt", header = T, check.names = F, stringsAsFactors = 
tr <- tr[match(tfam$V1, tr$`<Trait>`),] ## sort by tfam
tr[tr == -999] <- NA
tre <- cbind(tr[,1], tr)
for(i in 3:ncol(tre)){
  file_name <- paste0("emmax.trait.", names(tre)[i], ".txt")
  write.table(tre[, c(1, 2, i)], file = file_name, col.names = F, row.names = F, sep = "
}
## run emmax gwas with kingship only
emmax-intel64 -t emmax_in -o Trait04_emmax.k -p emmax.trait.Trait04.txt -k emmax_in.BN.k
## prepare covariant in R
setwd("F:/works/developing/course/gwas/data/lecture09")
tfam <- read.table("emmax_in.tfam", header = F, stringsAsFactors = F)
covs <- read.table("lecture09.Q.txt", header = T, check.names = F, stringsAsFactors = F,
covs <- covs[match(tfam$V1, covs$`<Trait>`), ]
covs <- cbind(covs[,1], covs[,1], rep(1, nrow(covs)), covs[,2:ncol(covs)]) ## for emmax
write.table(covs[, -ncol(covs)], file = "emmax.cov.txt", col.names = F, row.names = F, s
## run emmax gwas with kinship and Q-matrix
emmax-intel64 -t emmax_in -o Trait04_emmax.qk -p emmax.trait.Trait04.txt -k emmax_in.BN.

```

GEMMA

https://www.jianshu.com/p/372ec8585b49

https://cloud.tencent.com/developer/article/1677009

https://cloud.tencent.com/developer/article/1638988 !!!!!!

https://www.codenong.com/cs105808876/

https://cloud.tencent.com/developer/article/1593313

> 对此，Zhou等[29]提出了GEMMA方法，是EMMA的精确算法，计算速度比EMMA大大提高。近年来，快速运算得到长足发展，不断涌现新方法，如FaST-LMM[30]、FaST-LMM-Select[31]和BOLT-LMM[32]。最近，Wang等[33]将CMLM和FaST- LMM两种

> 算法整合，提出了运算速度更快的SU

```
## 下载GEMMA
wget -c https://github.com/genetics-statistics/GEMMA/releases/download/0.98.1/gemma-0.98
## 解压
gzip -d gemma-0.98.1-linux-static.gz
## 设置权限
chmod a+x gemma-0.98.1-linux-static
```

```
## 转换格式
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/01.gx_firs
/w/00/g/g01/user305/soft/plink/plink --file snp --make-bed --out gemma_input --allow-extr
```

```
接下来计算kinship矩阵。

/w/00/g/g01/user305/soft/gemma-0.98.1-linux-static -bfile gemma_input -gk 2 -o gemma
```

-bfile：输入Plink二进制格式文件的前缀。

-gk：指定生成的kinship矩阵类型。-gk 1 为centered matrix，-gk 2 为standardized matrix。

-o：输出文件前缀。

3.绘图

1、单倍型分析

```
1.先生成det文件找到强连锁区域
## whole genome block
## 需要加上no-pheno-req

/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/mer5.vcf/d
#该参数来源于--allow-extra-chr --blocks no-pheno-req --blocks-max-kb 1000 --geno 0.2 --bl
```

```
#基于选择信号和关联分析鉴定温带和热带玉米中优势单倍型和关联基因
```

```

## subset vcf and convert to Haploview format
## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf demo.impute.vcf  --chr At1 --from

2.beagle进行phase
/data/apps/jdk/jdk1.8.0_131/bin/java -Xms2g -Xmx8g -jar /w/00/g/g01/user305/soft/beagle.

20362400-20602400= 211.605kb

```

2、单倍型聚类

https://www.yuque.com/wusheng/gw7a9p/ukfrxg

①NJ树

```
/data/apps/plink/20181202/plink --vcf /w/00/g/g01/user305/Result/TResult/mer5.vcf/3.1_se

/data/apps/plink/20181202/plink --bfile plink --allow-extra-chr --distance-matrix

perl ~/script/NJTree/04.mdist2phylip.pl

fastme建树
/data/apps/fastme/2.1.5/bin/fastme -i plink.mdist.rename.phylip -m NJ -T 10 -o fastme.tr
```

②contree

```
#在正式运行phylip子程序之前，需要将VCF格式转换成phylip所需的格式。
# github中已经有开源的转换代码：edgardomortiz/vcf2phylip
# 查看文件权限是否是可执行的
## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf demo.impute.vcf  --chr At1 --from

python vcf2phy.py -i chr1_20304000-20604000.region.vcf


wget http://evolution.gs.washington.edu/phylip/download/phylip-3.69.tar.gz ./ #下载软件
tar zxvf phylip-3.697.tar.gz #解压
cd phylip-3.697/src
```

```
make -f Makefile.unx install
#以上几步即安装完软件，文件夹中的exe目录里为可执行程序
```

```
1、构建四个子程序的.par文件# seqboot.par
chr1.region.min4.phy
R #选择bootstrap
100 #设置bootstrap的值，即重复的replicate的数目，通常使用1000或者100，注意此处设定好后，后续两
Y #yes确认以上设定的参数
9 #设定随机参数，输入奇数值。


#2、 dnadist.par
seqboot.out
T  #选择设定Transition/transversion的比值
2.3628  #比值大小
M  #修改M值
D  #修改M值
100  #设定M值大小
2  #将软件运行情况显示出来
Y  #确认以上设定的参数

```

```
# 3、neighbor.par（注意不要有空格）
dnadist.out
M
100   #设定M值大小
9  #设定随机数，输入奇数值
Y  #确认以上设定的参数

#4、consense.par
nei.tree
Y #确认以上设定的参数
```

```
#/w/00/g/g01/user305/soft/phylip-3.697/exe
/w/00/g/g01/user305/soft/phylip-3.697/exe/seqboot < ./seqboot.par && mv ./outfile ./seqb
```

```
/w/00/g/g01/user305/soft/phylip-3.697/exe/dnadist < ./dnadist.par &&  mv ./outfile ./dna
/w/00/g/g01/user305/soft/phylip-3.697/exe/neighbor < ./neighbor.par && mv  ./outfile ./n
/w/00/g/g01/user305/soft/phylip-3.697/exe/consense < ./consense.par && mv ./outfile ./co

```

③豪哥的办法

```
/data/apps/iqtree/2.0.5/iqtree2 -s chr1_20304000-20604000.region.min4.phy -nt AUTO -m GT
```

④VCF2Dis

1、an Example of nj-tree with no boostrap

```
/w/00/g/g01/user305/soft/VCF2Dis-1.44/bin/VCF2Dis -InPut  /w/00/g/g01/user305/Result/TRe

```

2、Run PHYLIP

```
#   After p_distance done , software PHYLIPNEW 3.69 (http://evolution.genetics.washingto
#  Install PHYLIPNEW:https://github.com/BGI-shenzhen/VCF2Dis/blob/master/Install.NJ.en.m
          PHYLIPNEW-3.69.650/bin/fneighbor  -datafile p_dis.matrix  -outfile tree.out1.

      #    3.2 Run  MEGA
      #    The MEGA6 (http://www.megasoftware.net/) was used to present the phylogenetic
```

3) an Example of nj-tree with boostrap

```
/w/00/g/g01/user305/soft/VCF2Dis-1.44/bin/VCF2Dis -InPut /w/00/g/g01/user305/Result/TResu
```

3、单倍型网络图

https://www.jianshu.com/p/f9371b782f05
https://www.cnblogs.com/chenwenyan/p/9877377.html
https://weibo.com/p/2304188741e6c10102x7aw?pids=Pl_Official_CardMixFeedv6__4&feed_filter=1 Popart使用方法
phylo_tools/phy2fasta.py at master · harmsm/phylo_tools · GitHub phy文件to fasta

method0:

python phy2vcf.py phased.min4.phy > input.fas

```
#!/usr/bin/env python
__description__ = \
"""
Converts the .phy file used as input for a phyml-ss calculation (with its
stockholm-style structural annotations) and converts it into a simple,
paml-friendly fasta file with no structural information.
"""
__author__ = "Michael J. Harms"
__date__ = "130426"
__usage__ = "
phy2fasta.py phy_file > fasta_output_file"

import sys

f = open(sys.argv[1],'r')
lines = f.readlines()
f.close()

lines = [l for l in lines[2:] if l.strip() != "" and not l.startswith("#")]

for l in lines:
    c = l.split()
    print ">%s\n%s" % tuple(c)
```

```
#用plink过滤低频位点
plink --vcf xxxx.vcf --maf 0.05 --geno 0.2 --recode vcf-iid -out xxxx-maf0.05 --allow-ex
#用plink筛选连锁区
plink --vcf xxxx-maf0.05.vcf --indep-pairwise 100 50 0.2 -out xxxx-maf0.05-LD --allow-ex
#用plink找到强连锁区
plink --noweb --bfile xxxx-maf0.05-LD --blocks no-pheno-req --allow-extra-chr
```

Method1:

```
## subset vcf and convert to Haploview format
## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/TResul

/data/apps/jdk/jdk1.8.0_131/bin/java -Xms2g -Xmx10g -jar /w/00/g/g01/user305/soft/beagle
#VCF转换phy
vcf2phy.py -i phased.vcf.gz
#有了phy文件拿到Win操作系统下，用Bioedit就可以变成fasta格式
```

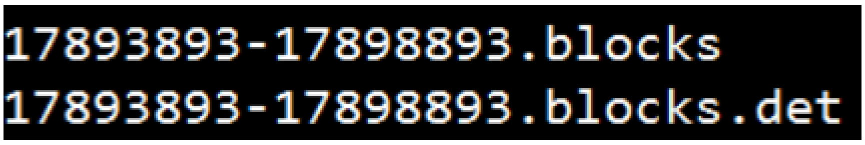
Method2:

1）将vcf转化为plink格式，假定输入的vcf文件名为：17893893-17898893.vcf

```
/vcftools --vcf 17893893-17898893.vcf --plink-tped --out 17893893-17898893
/plink --tfile 17893893-17898893 --recode --out 17893893-17898893
```

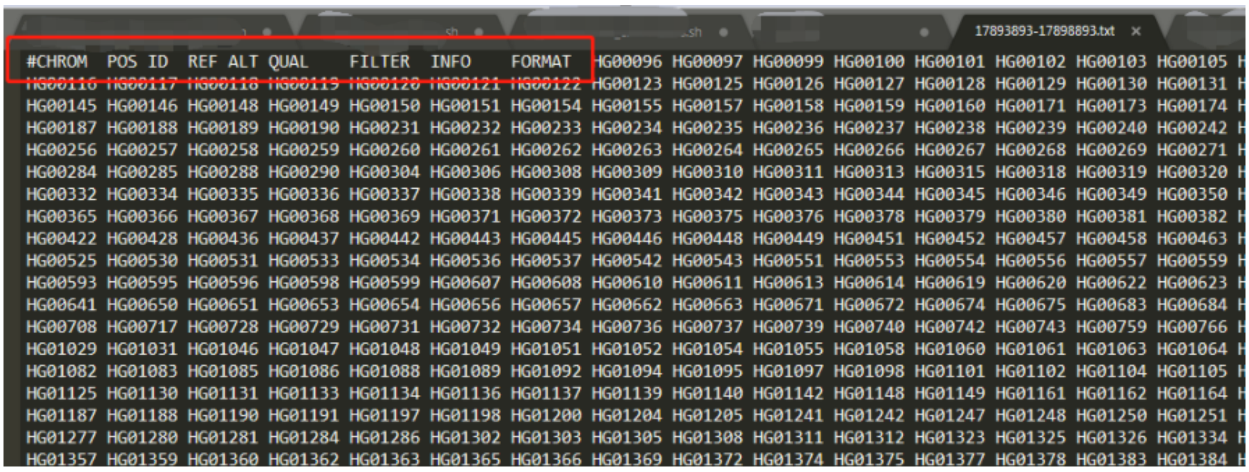
2) 用PLINK确定要研究的位点是否处于连锁的状态；生成blocks和blocks.det两种后缀格式文件；

```
/plink --file 17893893-17898893 --blocks no-pheno-req --out 17893893-17898893
```



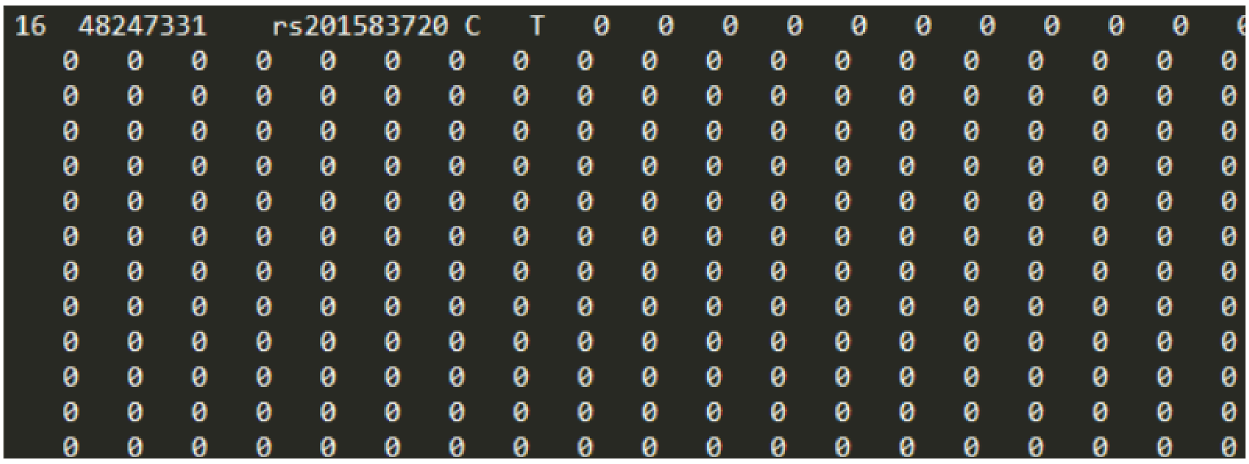
```
17893893-17898893.blocks
17893893-17898893.blocks.det
```

3) 提取这35个位点作为接下来的单倍型网络构建；去掉vcf的头文件，保存为txt格式，见如下图，17893893-17898893.txt：



```
#CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  HG00096 HG00097 HG00099 HG00100 HG00101 HG00102 HG00103 HG00105 H
HG00116 HG00117 HG00118 HG00119 HG00120 HG00121 HG00122 HG00123 HG00125 HG00126 HG00127 HG00128 HG00129 HG00130 HG00131 H
HG00145 HG00146 HG00148 HG00149 HG00150 HG00151 HG00154 HG00155 HG00157 HG00158 HG00159 HG00160 HG00171 HG00173 HG00174 H
HG00187 HG00188 HG00189 HG00190 HG00231 HG00232 HG00233 HG00234 HG00235 HG00236 HG00237 HG00238 HG00239 HG00240 HG00242 H
HG00256 HG00257 HG00258 HG00259 HG00260 HG00261 HG00262 HG00263 HG00264 HG00265 HG00266 HG00267 HG00268 HG00269 HG00271 H
HG00284 HG00285 HG00288 HG00290 HG00304 HG00306 HG00308 HG00309 HG00310 HG00311 HG00313 HG00315 HG00318 HG00319 HG00320 H
HG00332 HG00334 HG00335 HG00336 HG00337 HG00338 HG00339 HG00341 HG00342 HG00343 HG00344 HG00345 HG00346 HG00349 HG00350 H
HG00365 HG00366 HG00367 HG00368 HG00369 HG00371 HG00372 HG00373 HG00375 HG00376 HG00378 HG00379 HG00380 HG00381 HG00382 H
HG00422 HG00428 HG00436 HG00437 HG00442 HG00443 HG00445 HG00446 HG00448 HG00449 HG00451 HG00452 HG00457 HG00458 HG00463 H
HG00525 HG00530 HG00531 HG00533 HG00534 HG00536 HG00537 HG00542 HG00543 HG00551 HG00553 HG00554 HG00556 HG00557 HG00559 H
HG00593 HG00595 HG00596 HG00598 HG00599 HG00607 HG00608 HG00610 HG00611 HG00613 HG00614 HG00619 HG00620 HG00622 HG00623 H
HG00641 HG00650 HG00651 HG00653 HG00654 HG00656 HG00657 HG00662 HG00663 HG00671 HG00672 HG00674 HG00675 HG00683 HG00684 H
HG00708 HG00717 HG00728 HG00729 HG00731 HG00732 HG00734 HG00736 HG00737 HG00739 HG00740 HG00742 HG00743 HG00759 HG00766 H
HG01029 HG01031 HG01046 HG01047 HG01048 HG01049 HG01051 HG01052 HG01054 HG01055 HG01058 HG01060 HG01061 HG01063 HG01064 H
HG01082 HG01083 HG01085 HG01086 HG01088 HG01089 HG01092 HG01094 HG01095 HG01097 HG01098 HG01101 HG01102 HG01104 HG01105 H
HG01125 HG01130 HG01131 HG01133 HG01134 HG01136 HG01137 HG01139 HG01140 HG01142 HG01148 HG01149 HG01161 HG01162 HG01164 H
HG01187 HG01188 HG01190 HG01191 HG01197 HG01198 HG01200 HG01204 HG01205 HG01241 HG01242 HG01247 HG01248 HG01250 HG01251 H
HG01277 HG01280 HG01281 HG01284 HG01286 HG01302 HG01303 HG01305 HG01308 HG01311 HG01312 HG01323 HG01325 HG01326 HG01334 H
HG01357 HG01359 HG01360 HG01362 HG01363 HG01365 HG01366 HG01369 HG01372 HG01374 HG01375 HG01377 HG01378 HG01383 HG01384 H
```

4）准备17893893-17898893_singstring.txt文件，该文件其实就是去掉 0|0，0|1，1|0，1|1的 “|” 。



```
16  48247331    rs201583720 C   T   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

曼哈顿图系列

1、TASSEL结果画曼哈顿图

```
awk '$1=="Relativedryweight" {print $3, $4, $7}' MLM_result.txt > input_file.txt
sed 's/^AT//g' input_file.txt > new_file.txt
```

2、注释显著位点

https://zhuanlan.zhihu.com/p/96653959

```
library(ggrepel)
#准备数据
data <- Snp_pos %>%
# 添加高亮和注释信息：snpsOfInterest中的rs编号和P值大于6的点
 mutate( is_highlight=ifelse(SNP %in% snpsOfInterest, "yes", "no")) %>%
 mutate( is_annotate=ifelse(-log10(P)>6, "yes", "no"))
# 绘图
p1 <- ggplot(data, aes(x=BPcum, y=-log10(P))) +
   geom_point( aes(color=as.factor(CHR)), alpha=0.8, size=1.3) +
   scale_color_manual(values = rep(c("grey", "skyblue"), 22 )) +
   scale_x_continuous( label = X_axis$CHR, breaks= X_axis$center ) +
   scale_y_continuous(expand = c(0, 0) ) +
   # 添加高亮点
   geom_point(data=subset(data, is_highlight=="yes"), color="orange", size=2) +
   # 添加高亮label，且防止重叠
   geom_label_repel( data=subset(data, is_annotate=="yes"), aes(label=SNP), size=2) +
   theme_bw() +
theme(
     legend.position="none",
     panel.border = element_blank(),
     panel.grid.major.x = element_blank(),
     panel.grid.minor.x = element_blank()
  )
```

3、R语言CMplot包绘制曼哈顿图

https://cloud.tencent.com/developer/article/1481862

```

```

5、Candihap画图

https://mp.weixin.qq.com/s/1leAY8a-wYAFISFHgmu93Q
https://github.com/xukaili/CandiHap/tree/master/Sanger_ab1_Linux

```

perl  /w/00/g/g01/user305/soft/CandiHap/vcf2hmp.pl  /w/00/g/g01/user305/Result/TResult/m
```

6、单倍型群体结构可视化-haplostrips

https://bitbucket.org/dmarnetto/haplostrips/src/master/

```
#命令行下使用：win+R / CMD
haplostrips -v chr22.1000genomes.example.vcf.gz -R -V chr22.archaic_hominids.example.vcf
```

7、fastphase和beagle原理相同-提取候选区段

https://mp.weixin.qq.com/s/YwXq_MA4SEXJhNzm-lF5-g
https://mp.weixin.qq.com/s/BT6wHdIA70ltuBusxzD-wA

```
#1.解压缩
tar -xzvf  phase.2.1.1.linux.tar.gz
##直接使用beagle的结果
6.画structure
## missing rate and maf filtering
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/mer5.vcf/3
## make bed and LD pruning
/w/00/g/g01/user305/soft/plink/plink --vcf /w/00/g/g01/user305/Result/TResult/mer5.vcf/2

```

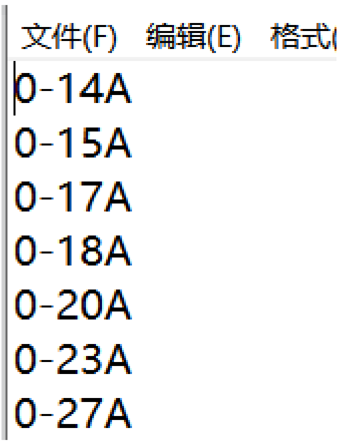
```
##extract prune in snps and recode to admixture format（转成ped格式）
/w/00/g/g01/user305/soft/plink/plink --bfile snp.int0.8.maf0.05 --extract snp.int0.8.maf


#课题组方法无法完整的画图，先得到群体结构的结果
for i in {1..10}; do /data/apps/admixture/1.3.0/admixture --cv -j30 --seed=666 /w/00/g/g
grep -h CV log*.out
```

8、fst(10k windows,2k 步长)

https://www.jianshu.com/p/7b6701ae593b
https://www.jianshu.com/p/b73a8d6233be
https://www.jianshu.com/p/98e56862347f

```
##对每一个SNP变异位点进行计算
vcftools --vcf output.10.06filterSNPrmOutGroup.hwe0.0001.vcf --weir-fst-pop Cultivar01.t

##按照区域来计算
vcftools --vcf output.10.06filterSNPrmOutGroup.hwe0.0001.vcf --weir-fst-pop Cultivar01.t
```

给的极端值txt文件只包含ID



0-14A
0-15A
0-17A
0-18A
0-20A
0-23A
0-27A

```
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/TResul

##图1
library(ggplot2)
data<-read.table("p_1_2_bin.windowed.weir.fst",header=T)
sc3 = subset(data,CHROM=="At1")
p <- ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p + theme_bw()
```

```

```

```
##图2
library(ggplot2)
data<-read.table("test.out.weir.fst",header=T)
sc3 = subset(data,CHROM=="Gm01")
p <- ggplot(sc3,aes(x=POS,y=WEIR_AND_COCKERHAM_FST)) + geom_point(size=0.5, colour="blue
p + theme_bwt1
```

```
library(ggplot2)
data<-read.table("p_1_2_bin.windowed.weir.fst",header=T)
sc3 = subset(data,CHROM=="At1")
p1 = ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p1 + theme_bw()

sc3 = subset(data,CHROM=="At2")
p2 = ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p2 + theme_bw()


sc3 = subset(data,CHROM=="At3")
p3 = ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p3 + theme_bw()


sc3 = subset(data,CHROM=="At4")
p4 = ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p4 + theme_bw()

sc3 = subset(data,CHROM=="At5")
p5 = ggplot(sc3,aes(x=BIN_END/1000000,y=WEIGHTED_FST)) + geom_point(size=0.5, colour="bl
p5 + theme_bw()
install.packages("ggpubr")
library(ggpubr)
ggarrange(p1, p2, p3, p4, p5, ncol = 2, nrow = 3)
```

manhadun:https://www.plob.org/article/21662.html

1、下面脚本需要改变染色体的名称

```
sed "1d" p_1_2_bin.windowed.weir.fst|awk '{if($5<0)print $1"\t"$2"\t0";else print $1"\t"$

### 加载要使用的R包
library(qqman)
library(Cairo)
 #z注意修改染色体编号
Fstfile<-read.table("TM_4500vsYJ_800_20k_5k_plot.txt", header=F, stringsAsFactors=F)
SNP<-paste(Fstfile[,1],Fstfile[,2],sep = ":")
Fstfile=cbind(SNP,Fstfile)
colnames(Fstfile)<-c("SNP","CHR","POS","Fst")
outfile<-"TM_4500vsYJ_800_20k_5k"
filePNG<-paste(outfile,"manhattan.png",sep=".")
CairoPNG(file=filePNG,width=1500,height=500)
colorset<-c("#FF0000","#FFD700","#2E8B57","#7FFFAA","#6495ED","#0000FF","#FF00FF")

### 调用qqman,画出全部染色体
manhattan(Fstfile,chr="CHR",bp="POS",p="Fst",snp="SNP", col=colorset,logp=F,suggestiveli
 dev.off()
### 只绘制单条染色体
if (FALSE){
    manhattan(subset(Fstfile,CHR=="10"),chr="CHR",bp="POS",p="Fst",snp="SNP", col=colors
}

dev.off()

```

```
library("qqman")
mydata=read.csv("p_c_t_bin.windowed.weir.fst",header = T,sep=",")
manhattan(mydata)
manhattan(mydata,main="p_c_t",ylim=c(0,1),cex=0.8,cex.axis=0.8,col = c("gray75","red","b
###  main="GWAS-Manhattan"这个命令是给图片取名字; highlight = GL这个命令是标记出高亮的snp的点
suggestiveline = F, genomewideline = 4.596这两个命令是设定阈值用的可以按照自己的需要设定, 如果
col = c("gray75","red","black")这个命令就是规定曼哈顿图的颜色的, 可以根据自己的需要而定。
qq(mydata$P)
```

图片组合

```
# 绘制第一个图像
> p1 = ggplot(cars, aes(x = speed, y = dist)) +
    geom_line(size = 1) +
    labs(x = "speed", y = 'dist')

# 绘制第二个图像
> p2 = ggplot(cars, mapping = aes(x = speed, y = dist)) +
    geom_point(size = 1, col = 'darker') +
    labs(x = "speed", y = 'dist')

# 整合两张图
> ggarrange(p1, p2, ncol = 2, nrow = 1)
```

！计算核苷酸多样性（π）

> 非常简单，把某群体的全部样本ID放在一个TXT文件里就行，注意要每一行一个ID。
>
> 计算核苷酸多样性（π）

```
#局部和整体
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/flower

/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/TResul
```

画图1：https://cloud.tencent.com/developer/article/1593265

箱线图

```
bb<-read.table("../../vcf_handling/Fish_Populations/benthic_pi.windowed.pi",header=T)
ll<-read.table("../../vcf_handling/Fish_Populations/littoral_pi.windowed.pi",header=T)
dim(bb)
head(bb)
bb$indiv<-"benthic"
dim(ll)
head(ll)
ll$indiv<-"littoral"
df<-rbind(bb,ll)
colnames(df)
dim(df)
library(ggplot2)
```

```
options(scipen=200)
ggplot(df,aes(x=indiv,y=PI,fill=indiv))+
  geom_boxplot()+theme_bw()
```

折线图

```
ggplot(df,aes(x=BIN_START,y=PI,group=indiv,color=indiv))+
  geom_line()+theme_bw()+
  theme(legend.position = "top",
        legend.title = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title = element_blank())
```

https://www.jianshu.com/p/7b6701ae593b

```
#读入数据;
dt1<- read.delim("pi.txt",sep="\t", header = T, check.names = F)
# 加载ggplot2包;
library(ggplot2)
#定义染色体位置。
#我分析的物种有8条染色体，我将所有的染色体都串起来作图，因此需要标出每条染色体的中间位置在哪。
br = c(440000, 1370000, 2410000, 3515000, 4610000,5800000,7095000,8399791)
la = c("1","2","3","4","5","6","7","8")
#自定义图表主题，对图表做精细调整;
mytheme<-theme(panel.grid.major =element_blank(),
               panel.grid.minor = element_blank(),
               panel.background = element_blank(),
               panel.border = element_blank(),
               axis.line.y = element_line(color = "black"),
               axis.line.x = element_line(color = "black"),
               #axis.title.x = element_text(size = rel(1.2)),
               axis.title.y = element_text(size = rel(1.2)),
               axis.text.y = element_text(size=rel(1.2),color="black"),
               #axis.text.x = element_text(size=rel(1.2),color="black"),
               plot.margin=unit(x=c(top.mar,right.mar,bottom.mar,left.mar),units="inche
#绘制
pa<-ggplot(data=dt1, mapping = aes(x=No,y=pop1))+geom_line(color=dt1$Color1,size=1)
#设置x轴范围，避免点的溢出绘图区;
```

```
pa<-pa+scale_x_continuous(limits = c(-1000, 9230000),breaks = br,labels = la)
#设置y轴范围
pa<-pa+scale_y_continuous(limits = c(-0.0005,0.01),breaks = c(0,0.002,0.004,0.006,0.008,(
#设置图例、坐标轴、图表的标题;
pa<-pa+labs(y="Pi (10e-3)",x=NULL)
#自定义图表主题，对图表做精细调整;
pa<-pa+mytheme
#出图
pa

```

9、0|0转换-聚类热图

https://www.cnblogs.com/Mao1518202/p/11269595.html 最重要的热图指南

RectChr 之多样品某一区域 genotype/gene/单倍型 热度图 - 知乎 (zhihu.com)

https://github.com/BGI-shenzhen/RectChr/tree/main/Example

方法一:

https://mp.weixin.qq.com/s/8tchKCRVsBs5f5fQtR-d1A ! ! ! !

方法二：plink

https://mp.weixin.qq.com/s/hX0XjOpTugTEIMEI3rz1_g

使用–bfile 、 --file 和 --tfile读取文件类型不一样:

–bfile 读取二进制文件，bed、bim和fam格式

–file 读取文本文件，ped和map格式

```

## 用vcftools提取指定区间
/data/apps/vcftools/vcftools-0.1.15/bin/vcftools --vcf /w/00/g/g01/user305/Result/TResult

方法一:
#对数据进行清洗，将其转化为0,1,2的形式
#转ped、map格式

/w/00/g/g01/user305/soft/plink/plink --vcf chr4_10kb.vcf --recode --out region_10k --allo
```

```
#首先，使用plink命令，将基因型数据转化为012的raw格式:
/w/00/g/g01/user305/soft/plink/plink --file region_10k --recodeA --allow-extra-chr


方法二:
/w/00/g/g01/user305/soft/plink/plink --vcf chr1_20304000-20604000.region.vcf --recode --
## ped/map转换为bed/bim/fam
/w/00/g/g01/user305/soft/plink/plink --file region_300k --make-bed --out snp_region_300k



```

extract_sample

```
#!/bin/bash
for i in `cat hap1.txt`
do
grep $i trait.txt >> result1.txt

done
```

```
file1（id）file2（id+数据）
grep -wFf file1 file2
```

https://www.jianshu.com/p/5b6bdbebcd61 重要热图
https://ww2.mathworks.cn/help/matlab/ref/matlab.graphics.chart.heatmapchart-properties.html 帮助文档

```
setwd("D:\\work_dir\\10.40KB_picture")

install.packages("gplots")
```

```
test<- read.table("test1.xls", head=T)

library(pheatmap)
library(grid)
library(ComplexHeatmap)

data = as.matrix(test)
is.matrix(data)

Heatmap(data, cluster_rows = T, cluster_columns = F,show_row_names = F, show_column_names

pheatmap(data, cluster_cols = F, cluster_rows = T, fontsize = 1)
```

先将dataframe换为矩阵

https://blog.csdn.net/weixin_41938903/article/details/84983804

```
# R读入csv文件
data1<-read.csv('d:/ko_wt.csv')  #参数默认为TRUE
# 将dataframe转化为matrix, 目前找到了2种方法
①
y = data.metrix(data1)
②
y <- apply(data1, as.numeric)
```

简单自己的总结

```
BiocManager::install("ComplexHeatmap")
setwd("D:\\pheatmap")
test<- read.csv("test1.csv", head=T)

library(pheatmap)
library(grid)
library(ComplexHeatmap)

data1 = as.matrix(data)
is.matrix(data1)

Heatmap(data1,  cluster_rows = F, cluster_columns = F,show_row_names = F, show_column_na
```

complexheatmap-https://www.jianshu.com/p/06705a7812ee

```
#设置工作路径
getwd()
setwd("D:\\pheatmap")
test<- read.table("genotype_transpose.csv", head=T)
rownames(test)<-test[,1]
test1<-test[,c(1:490)]

library(pheatmap)
pheatmap(test1)
?pheatmap
vignette()
devtools::install_github('satijalab/seurat-data')
library(SeuratData)
InstallData("pbmc3k")
BiocManager::install("ComplexHeatmap")

```

```
最后用 awk 进行矩阵转置:
awk '{i=1;while(i <= NF){col[i]=col[i] $i " ";i=i+1}} END {i=1;while(i<=NF){print col[i]
awk -F, '{for(i=1;i<=NF;i=i+1){a[NR,i]=$i}}END{for(j=1;j<=NF;j++){str=a[1,j];for(i=2;i<=


```

热图2https://mp.weixin.qq.com/s/lNswPaJP1P4caEnAE_X--A

1.准备

```
rm(list=ls()) # 清除环境变量
setwd("C:/Rdata") # 设置工作目录
getwd() # 查看当前的工作目录
```

2.安装包

```
1.2 安装和加载R包
安装我们绘图需要用的包，如果已安装，直接加载就好了。
install.packages("pheatmap") # 安装包
install.packages("ggplot2") # 安装包
library(pheatmap) # 加载包
library(ggplot2) # 加载包
```

3.查看数据

```
data <- read.table("test.FPKM.txt",  # 读取的数据文件名称，这里文件是放在工作目录下
                   header=T, # 数据集第一行为变量名
                   row.names=1, # 第一列为行名
                   sep="\t") # 指定分隔符号
dim(data) # 查看变量有多少行多少列
```

4.绘制热图

```
先绘制个基本款的热图。
p <- pheatmap(data)

3.1 热图横向标准化
设置标准化方向scale，对其横向标准化。
p <- pheatmap(data, scale="row")
```

```
3.2 图形外观调整
设置边框为白色，去掉横向、纵向聚类。
p <- pheatmap(data, scale="row",
            border="white", # 设置边框为白色
            cluster_cols = F, # 去掉横向、纵向聚类
            cluster_rows = F)
```

```
3.3 去除横纵坐标标签
去掉横纵坐标中的id。
p <- pheatmap(data,scale="row",
              border="white",  # 设置边框为白色
              cluster_cols = F, # 去掉横向、纵向聚类
              cluster_rows = F,
              show_rownames = F, #去掉横、纵坐标id
              show_colnames = F)
```

```
3.4 修改图例
3.4.1 不显示右上角图例
p <- pheatmap(data,scale="row",
              border="white",  # 设置边框为白色
              cluster_cols = F, # 去掉横向、纵向聚类
              cluster_rows = F,
              show_rownames = F, #去掉横、纵坐标id
              show_colnames = F,    #此处参数可以选择对横纵坐标去聚类
              legend = F) # 去掉图例
```

```
3.4.2 设置图例范围
p <- pheatmap(data,scale="row",
              border="white",  # 设置边框为白色
              cluster_cols = F, # 去掉横向、纵向聚类
              cluster_rows = F,
              show_rownames = F, #去掉横、纵坐标id
              show_colnames = F,
              legend = T, # 添加图例
              legend_breaks=c(-1,0,1)) # 设置图例范围

# 也可以设置legend_breaks=c(-2,0,2)试试
```

5、颜色改变——使用heatmap

https://www.jianshu.com/p/06705a7812ee
https://www.jianshu.com/p/325107a11492
https://mp.weixin.qq.com/s/mESHydcbzyMc2p4rpBE-MA 格式转换，在微信公众号可以找到

```
#格式判断和格式转换
is.matrix(data)
data1 = as.matrix(data)
is.matrix(data1)
```

基本显示

```r
ht <- Heatmap(mat)
draw(ht)
```

颜色的改变

- 在热图当中如果需要改变颜色的话，**一定要使用**circlize::colorRamp2()来对颜色进行赋值。这个函数接受两个参数:**颜色分割的位置**以及**相应分割点上的颜色**。最后我们的Heatmap当中通过col来制定颜色

PS：colorRamp2默认的使用的配色方案是LAB的。如果想使用RGB的则可以通过其中的space参数进行修改。

```r
library(circlize)
col_fun <- colorRamp2(c(-2, 0, 2), c("navy", "white", "firebrick3"))
Heatmap(mat, col = col_fun)
```

聚类

http://showteeth.tech/posts/31202.html

https://www.yuque.com/bioitee/complexheatmap-reference/dimension-names

基本设置

无监督的聚类属于热图的可视化的一个重要组成部分。

1. cluster_rows/columns来设置**是否进行聚类**
2. show_column/row_dend设置**是否显示聚类树**(会发生聚类)
3. column/row_dend_side设置**聚类图绘制的位置**
4. column/row_dend_height设置**聚类树的高度**

```r
Heatmap(mat, cluster_rows = F, show_column_dend = T,
        column_dend_side = "bottom",  column_dend_height = unit(2, "cm"))
```

去掉横纵坐标

```r
Heatmap(data1, cluster_rows = F, cluster_columns = F,show_row_names = F, show_column_nam
```

```r

```

```
3
```

注释

```
ha = HeatmapAnnotation(foo = anno_points(matrix(runif(20), nc = 2),
    pch = 1:2, gp = gpar(col = 2:3)))
```

条形块

```
# color mapping for -log10(pvalue)
pvalue_col_fun = colorRamp2(c(0, 2, 3), c("green", "white", "red"))
ha = HeatmapAnnotation(
    pvalue = anno_simple(-log10(pvalue), col = pvalue_col_fun, pch = pch),
    annotation_name_side = "left")
ht = Heatmap(matrix(rnorm(100), 10), name = "mat", top_annotation = ha)
```

如果想颠倒横纵坐标画图，可以直接使用t(matrix)进行转置。

http://showteeth.tech/posts/31202.html

R语言画热图时图例（legend）过多超出画图边界

https://cloud.tencent.com/developer/article/1628471

三、解决方案——将整个矩阵转化为数值型变量

```
test=apply(test,2,as.numeric)
Warning message:
In apply(test, 2, as.numeric) : NAs introduced by coercion
```

```
是rt1的expression列不是数值型向量
> rt1 <- as.data.frame(rt1)
> rt1[,1] <- as.numeric(rt1[,1])
添加这两句进去就行
```

热图注释

https://www.jianshu.com/p/8ed637722b0b

```
1
```

曼哈顿图优化

数据准备

```
awk '{print $2, $3, $4, $7}' input1.txt > input_qqman.txt
sed -i "s/^At//g" input_qqman.txt
```

①qqman

```
#安装方法:
#====设置安装源为清华大学TUNA镜像站点=====================
options("repos" = c(CRAN="https://mirrors.tuna.tsinghua.edu.cn/CRAN/"))
options(BioC_mirror="https://mirrors.tuna.tsinghua.edu.cn/bioconductor")
install.packages('qqman')
#==========================================================

library(qqman)
setwd('F:/PlantTech/training/GWAS/qqman')
#输入文件格式如下:
#  SNP   CHR   BP   P
# 其中SNP列为SNP名称，这一列可以没有
# CHR，BP分别为SNP所在的染色体和位置
# P表示该SNP的关联显著性p值
data<-read.table('input.txt',header=T)

# 取bonferroni矫正阈值
fdr=0.01/nrow(data)

manhattan(data, main = "Manhattan Plot",
          ylim = c(0, 10), cex = 0.6,cex.axis = 0.9,
          col = c("blue4", "orange3"),
          suggestiveline = -log10(1e-05), genomewideline = -log10(fdr))

qq(data$P, main = "Q-Q plot of GWAS p-values",
   xlim = c(0, 7), ylim = c(0,12),
   pch = 18, col = "blue4", cex = 1.5)
```

LD衰减-可以根据LD衰减距离来决定候选基因上下游的范围

https://cloud.tencent.com/developer/article/1677015
https://zhuanlan.zhihu.com/p/109164405

```
## calculate ld decay using poplddecay全部
/w/00/g/g01/user305/soft/PopLDdecay/bin/PopLDdecay -InVCF /w/00/g/g01/user305/Result/TRe


#画图
perl /w/00/g/g01/user305/soft/PopLDdecay/bin/Plot_OnePop.pl -inFile demo.impute.pld.stat

```

POPhelper

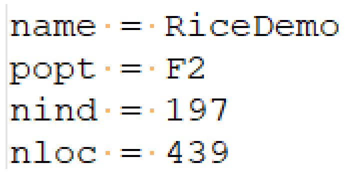
BLUP分析

https://www.jianshu.com/p/c72b3835be27

QTL定位

①文件格式

popt = RI7



```
name = RiceDemo
popt = F2
nind = 197
nloc = 439
```

②软件
https://github.com/bmansfeld/QTLseqr