# PRS评分计算

## 参考：

1. <https://www.nature.com/articles/s41596-020-0353-1>

2.该文章作者写的教程<https://choishingwan.github.io/PRS-Tutorial/base/>

## 环境配置：

1. 要求>4.2.3版本的R，因此选用了最新版4.4.1版本的R

conda create -n PRS

conda activate PRS

conda install r-base=4.4

1. 下载了1.9版本的plink，plink官网上下到的

## 数据获取

1. Base data：[Height.gwas.txt.gz - Google 云端硬盘](https://drive.google.com/file/d/1RWjk49QNZj9zvJHc9X_wyZ51fdy6xQjv/view)
2. Target data：[EUR.zip - Google 云端硬盘](https://drive.google.com/file/d/1uhJR_3sn7RA8U5iYQbcmTp6vFdQiF4F2/view)

## 具体操作-base data

1. 遗传力检查

chip-heritability h2snp>0.05

GWAS的芯片遗传性可以使用LD分数回归（LDSC）等来估计。

1. 效应等位基因

要知道哪个等位基因是效应等位基因，哪个是非效应等位基因。

1. 基因组build

确保base data和target data有相同的基因组构建。

1. 标准GWAS质量控制

具有低次要等位基因频率（MAF）或插补信息评分（INFO）的SNP更有可能产生假阳性结果，因为它们的统计功效较低

建议删除 MAF < 1% 且 INFO < 0.8 的 SNP（对于非常大的基本样本量，如果灵敏度检查表明结果可靠，则可以降低这些阈值）

删除代码如下：

gunzip -c Height.gwas.txt.gz |\

awk 'NR==1 || ($11 > 0.01) && ($10 > 0.8) {print}' |\

gzip > Height.gz

1. SNP不匹配

由于我们需要目标数据来知道哪些 SNP 具有不匹配的等位基因，因此我们将在目标数据中执行这种链翻转。

1. 重复的 SNP

删去重复SNP的代码如下：

gunzip -c Height.gz |\

awk '{seen[$3]++; if(seen[$3]==1){ print}}' |\

gzip - > Height.nodup.gz

1. 不明确的SNP

对于MAF接近50%的SNP或当base和target数据来自不同群体时，其推断哪些等位基因位于同一条链上的准确性可能较低

建议删除所有模棱两可的SNP，以避免引入这种潜在的系统误差来源。

gunzip -c Height.nodup.gz |\

awk '!( ($4=="A" && $5=="T") || \

($4=="T" && $5=="A") || \

($4=="G" && $5=="C") || \

($4=="C" && $5=="G")) {print}' |\

gzip > Height.QC.gz

1. 性染色体

生理性别和报告性别不匹配的个体通常会被删除。

1. 样本重叠

base数据和target数据之间的样本重叠可能导致 PRS 与目标数据中测试的性状之间的关联大幅膨胀所以必须消除。

理想情况下，从base数据中删除重叠样本，并重新计算base GWAS。

1. 相关性

base数据和target数据之间密切相关的个体可能会导致结果过拟合，从而限制结果的普遍性。

## 具体操作-target data

1. 基因组构建

Base和target数据的基因组构建应当是相同的

1. 标准GWAS质量控制

去除低基因分型率的SNP， 低次要等位基因频率，脱离 Hardy-Weinberg 平衡，去除 基因分型率低的个体

plink \

--bfile EUR \

--maf 0.01 \

--hwe 1e-6 \

--geno 0.01 \

--mind 0.01 \

--write-snplist \

--make-just-fam \

--out EUR.QC

进行修剪pruning以去除高度相关的 SNP

plink \

--bfile EUR \

--keep EUR.QC.fam \

--extract EUR.QC.snplist \

--indep-pairwise 200 50 0.25 \

--out EUR.QC

这将生成两个文件：1）EUR.QC.prune.in 和 2）EUR.QC.prune.out。

EUR.QC.prune.in 内的所有 SNP 都具有成对的r2<0.25

然后可以使用以下方法计算杂合率

plink \

--bfile EUR \

--extract EUR.QC.prune.in \

--keep EUR.QC.fam \

--het \

--out EUR.QC

这将产生EUR.QC.het 文件，其中包含用于评估杂合性的 F 系数估计值。

我们将删除 F 系数与均值大于 3 个标准差 （SD） 单位的个体

{R}

library(data.table)

# Read in file

dat <- fread("EUR.QC.het")

# Get samples with F coefficient within 3 SD of the population mean

valid <- dat[F<=mean(F)+3\*sd(F) & F>=mean(F)-3\*sd(F)]

# print FID and IID for valid samples

fwrite(valid[,c("FID","IID")], "EUR.valid.sample", sep="\t")

q() # exit R

1. SNP不匹配

碱基数据和靶标数据中报告的等位基因不匹配的SNP可以通过将等位基因与靶数据中的互补等位基因进行链翻转来解析，例如对于碱基数据中具有A/C和靶标中具有G/T的SNP。

（大多数 PRS 软件会自动执行股线翻转，因此通常不需要此步骤。）

① 将 bim 文件、汇总统计量和 QC SNP 列表加载到 R 中

# magrittr allow us to do piping, which help to reduce the

# amount of intermediate data types

library(data.table)

library(magrittr)

# Read in bim file

bim <- fread("EUR.bim") %>%

# Note: . represents the output from previous step

# The syntax here means, setnames of the data read from

# the bim file, and replace the original column names by

# the new names

setnames(., colnames(.), c("CHR", "SNP", "CM", "BP", "B.A1", "B.A2")) %>%

# And immediately change the alleles to upper cases

.[,c("B.A1","B.A2"):=list(toupper(B.A1), toupper(B.A2))]

# Read in summary statistic data (require data.table v1.12.0+)

height <- fread("Height.QC.gz") %>%

# And immediately change the alleles to upper cases

.[,c("A1","A2"):=list(toupper(A1), toupper(A2))]

# Read in QCed SNPs

qc <- fread("EUR.QC.snplist", header=F)

② 识别需要链翻转的 SNP

# Merge summary statistic with target

info <- merge(bim, height, by=c("SNP", "CHR", "BP")) %>%

# And filter out QCed SNPs

.[SNP %in% qc[,V1]]

# Function for calculating the complementary allele

complement <- function(x){

switch (x,

"A" = "T",

"C" = "G",

"T" = "A",

"G" = "C",

return(NA)

)

}

# Get SNPs that have the same alleles across base and target

info.match <- info[A1 == B.A1 & A2 == B.A2, SNP]

# Identify SNPs that are complementary between base and target

com.snps <- info[sapply(B.A1, complement) == A1 &

sapply(B.A2, complement) == A2, SNP]

# Now update the bim file

bim[SNP %in% com.snps, c("B.A1", "B.A2") :=

list(sapply(B.A1, complement),

sapply(B.A2, complement))]

③识别需要在靶标中重新编码的 SNP

# identify SNPs that need recoding

recode.snps <- info[B.A1==A2 & B.A2==A1, SNP]

# Update the bim file

bim[SNP %in% recode.snps, c("B.A1", "B.A2") :=

list(B.A2, B.A1)]

# identify SNPs that need recoding & complement

com.recode <- info[sapply(B.A1, complement) == A2 &

sapply(B.A2, complement) == A1, SNP]

# Now update the bim file

bim[SNP %in% com.recode, c("B.A1", "B.A2") :=

list(sapply(B.A2, complement),

sapply(B.A1, complement))]

# Write the updated bim file

fwrite(bim[,c("SNP", "B.A1")], "EUR.a1", col.names=F, sep="\t")

④ 鉴定base和target中具有不同等位基因的 SNP（通常是由于基因组构建或插入缺失的差异）

mismatch <- bim[!(SNP %in% info.match |

SNP %in% com.snps |

SNP %in% recode.snps |

SNP %in% com.recode), SNP]

write.table(mismatch, "EUR.mismatch", quote=F, row.names=F, col.names=F)

q() # exit R

然后，我们可以使用 EUR.a1 文件来更新 A1 等位基因。

1. 重复SNP

确保删除目标数据中的任何重复 SNP

1. 性染色体

生理性别和报告性别不匹配的个体通常会被删除，个体的X染色体纯合度估计值（F统计量）<0.2，则称为女性，如果估计值>0.8，则称为男性。

在进行性别检查之前，应进行修剪pruning

plink \

--bfile EUR \

--extract EUR.QC.prune.in \

--keep EUR.valid.sample \

--check-sex \

--out EUR.QC

#开始检查

R

library(data.table)

# Read in file

valid <- fread("EUR.valid.sample")

dat <- fread("EUR.QC.sexcheck")[FID%in%valid$FID]

fwrite(dat[STATUS=="OK",c("FID","IID")], "EUR.QC.valid", sep="\t")

q() # exit R

1. 样本重叠
2. 关联性

数据中密切相关的个体可能会导致结果过度拟合

在计算相关性之前，应进行修剪（请参阅此处）。 样本中具有一级或二级亲属（π>0.125）可以使用以下命令删除：

plink \

--bfile EUR \

--extract EUR.QC.prune.in \

--keep EUR.QC.valid \

--rel-cutoff 0.125 \

--out EUR.QC

1. 生成最终的质量控制target数据文件

plink \

--bfile EUR \

--make-bed \

--keep EUR.QC.rel.id \

--out EUR.QC \

--extract EUR.QC.snplist \

--exclude EUR.mismatch \

--a1-allele EUR.a1

## Plink计算PRS评分

* + - 1. 更新效果大小

library(data.table)

dat <- fread("Height.QC.gz")

fwrite(dat[,BETA:=log(OR)], "Height.QC.Transformed", sep="\t")

q() # exit R

* + - 1. Clumping

去除 SNP 的方式是仅保留弱相关的 SNP，但优先保留与所研究表型最相关的 SNP

plink \

--bfile EUR.QC \

--clump-p1 1 \

--clump-r2 0.1 \

--clump-kb 250 \

--clump Height.QC.Transformed \

--clump-snp-field SNP \

--clump-field P \

--out EUR

这一步非常缓慢和耗时。

这将生成 EUR.clumped，在执行聚类后包含索引 SNP。 我们可以通过执行以下命令来提取索引 SNP ID：

awk 'NR!=1{print $3}' EUR.clumped > EUR.valid.snp

* + - 1. 生成 PRS

awk '{print $3,$8}' Height.QC.Transformed > SNP.pvalue

# 这里计算对应于几个阈值的 PRS：

echo "0.001 0 0.001" > range\_list

echo "0.05 0 0.05" >> range\_list

echo "0.1 0 0.1" >> range\_list

echo "0.2 0 0.2" >> range\_list

echo "0.3 0 0.3" >> range\_list

echo "0.4 0 0.4" >> range\_list

echo "0.5 0 0.5" >> range\_list

#计算PRS

plink \

--bfile EUR.QC \

--score Height.QC.Transformed 3 4 12 header \

--q-score-range range\_list SNP.pvalue \

--extract EUR.valid.snp \

--out EUR

4. 考虑人口分层

种群结构是GWAS中混杂的主要来源，通常通过将主成分（PC）作为协变量来解释。我们可以将 PC 纳入我们的 PRS 分析中，以解释人口分层。

我们可以使用以下方法计算 PC

# First, we need to perform prunning

plink \

--bfile EUR.QC \

--indep-pairwise 200 50 0.25 \

--out EUR

# Then we calculate the first 6 PCs

plink \

--bfile EUR.QC \

--extract EUR.prune.in \

--pca 6 \

--out EUR

5. 寻找最合适的PRS

library(data.table)

library(magrittr)

p.threshold <- c(0.001,0.05,0.1,0.2,0.3,0.4,0.5)

phenotype <- fread("EUR.height")

pcs <- fread("EUR.eigenvec", header=F) %>%

setnames(., colnames(.), c("FID", "IID", paste0("PC",1:6)) )

covariate <- fread("EUR.cov")

pheno <- merge(phenotype, covariate) %>%

merge(., pcs)

null.r2 <- summary(lm(Height~., data=pheno[,-c("FID", "IID")]))$r.squared

prs.result <- NULL

for(i in p.threshold){

pheno.prs <- paste0("EUR.", i, ".profile") %>%

fread(.) %>%

.[,c("FID", "IID", "SCORE")] %>%

merge(., pheno, by=c("FID", "IID"))

model <- lm(Height~., data=pheno.prs[,-c("FID","IID")]) %>%

summary

model.r2 <- model$r.squared

prs.r2 <- model.r2-null.r2

prs.coef <- model$coeff["SCORE",]

prs.result %<>% rbind(.,

data.frame(Threshold=i, R2=prs.r2,

P=as.numeric(prs.coef[4]),

BETA=as.numeric(prs.coef[1]),

SE=as.numeric(prs.coef[2])))

}

print(prs.result[which.max(prs.result$R2),])

q() # exit R

## PRSice-2计算PRS评分

需要一个协变量文件，因此我们的协变量文件和 PCs 文件应该合并。

library(data.table)

covariate <- fread("EUR.cov")

pcs <- fread("EUR.eigenvec", header=F)

colnames(pcs) <- c("FID","IID", paste0("PC",1:6))

cov <- merge(covariate, pcs)

fwrite(cov,"EUR.covariate", sep="\t")

q()

然后可以运行 PRSice-2 来获得 PRS 结果，如下所示：

Rscript PRSice.R \

--prsice PRSice\_linux \

--base Height.QC.gz \

--target EUR.QC \

--binary-target F \

--pheno EUR.height \

--cov EUR.covariate \

--base-maf MAF:0.01 \

--base-info INFO:0.8 \

--stat OR \

--or \

--out EUR

## LDpred-2计算PRS评分

准备工作&数据读取

# 1. prepare workspace and load bigsnpr

library(bigsnpr)

library(bigsnpr)

options(bigstatsr.check.parallel.blas = FALSE)

options(default.nproc.blas = NULL)

# 2. Read in the phenotype and covariate files

library(data.table)

library(magrittr)

phenotype <- fread("EUR.height")

covariate <- fread("EUR.cov")

pcs <- fread("EUR.eigenvec")

# rename columns

colnames(pcs) <- c("FID","IID", paste0("PC",1:6))

# generate required table

pheno <- merge(phenotype, covariate) %>%

merge(., pcs)

# 3. Obtain HapMap3 SNPs

info <- readRDS(runonce::download\_file(

"https://ndownloader.figshare.com/files/25503788",

fname = "map\_hm3\_ldpred2.rds"))

# 4. Load and transform the summary statistic file

# Read in the summary statistic file

sumstats <- bigreadr::fread2("Height.QC.gz")

# LDpred 2 require the header to follow the exact naming

names(sumstats) <-

c("chr",

"pos",

"rsid",

"a1",

"a0",

"n\_eff",

"beta\_se",

"p",

"OR",

"INFO",

"MAF")

# Transform the OR into log(OR)

sumstats$beta <- log(sumstats$OR)

# Filter out hapmap SNPs

sumstats <- sumstats[sumstats$rsid%in% info$rsid,]

计算LD矩阵

从Genome Wide bed files中开始计算

# Calculate the LD matrix from Genome Wide bed files

# Get maximum amount of cores

NCORES <- nb\_cores()

# Open a temporary file

tmp <- tempfile(tmpdir = "tmp-data")

on.exit(file.remove(paste0(tmp, ".sbk")), add = TRUE)

# Initialize variables for storing the LD score and LD matrix

corr <- NULL

ld <- NULL

# We want to know the ordering of samples in the bed file

fam.order <- NULL

# preprocess the bed file (only need to do once for each data set)

snp\_readBed("EUR.QC.bed")

# now attach the genotype object

obj.bigSNP <- snp\_attach("EUR.QC.rds")

# extract the SNP information from the genotype

map <- obj.bigSNP$map[-3]

names(map) <- c("chr", "rsid", "pos", "a1", "a0")

# perform SNP matching

info\_snp <- snp\_match(sumstats, map)

# Assign the genotype to a variable for easier downstream analysis

genotype <- obj.bigSNP$genotypes

# Rename the data structures

CHR <- map$chr

POS <- map$pos

# get the CM information from 1000 Genome

# will download the 1000G file to the current directory (".")

POS2 <- snp\_asGeneticPos(CHR, POS, dir = ".")

# calculate LD

for (chr in 1:22) {

# Extract SNPs that are included in the chromosome

ind.chr <- which(info\_snp$chr == chr)

ind.chr2 <- info\_snp$`\_NUM\_ID\_`[ind.chr]

# Calculate the LD

corr0 <- snp\_cor(

genotype,

ind.col = ind.chr2,

ncores = NCORES,

infos.pos = POS2[ind.chr2],

size = 3 / 1000

)

if (chr == 1) {

ld <- Matrix::colSums(corr0^2)

corr <- as\_SFBM(corr0, tmp)

} else {

ld <- c(ld, Matrix::colSums(corr0^2))

corr$add\_columns(corr0, nrow(corr))

}

}

# We assume the fam order is the same across different chromosomes

fam.order <- as.data.table(obj.bigSNP$fam)

# Rename fam order

setnames(fam.order,

c("family.ID", "sample.ID"),

c("FID", "IID"))

从Chromosome separated bed files中进行计算

# Calculate the LD matrix from Chromosome separated bed files

# Get maximum amount of cores

NCORES <- nb\_cores()

# Open a temporary file

tmp <- tempfile(tmpdir = "tmp-data")

on.exit(file.remove(paste0(tmp, ".sbk")), add = TRUE)

# Initialize variables for storing the LD score and LD matrix

corr <- NULL

ld <- NULL

# We want to know the ordering of samples in the bed file

info\_snp <- NULL

fam.order <- NULL

for (chr in 1:22) {

# preprocess the bed file (only need to do once for each data set)

# Assuming the file naming is EUR\_chr#.bed

snp\_readBed(paste0("EUR\_chr",chr,".bed"))

# now attach the genotype object

obj.bigSNP <- snp\_attach(paste0("EUR\_chr",chr,".rds"))

# extract the SNP information from the genotype

map <- obj.bigSNP$map[-3]

names(map) <- c("chr", "rsid", "pos", "a1", "a0")

# perform SNP matching

tmp\_snp <- snp\_match(sumstats[sumstats$chr==chr,], map)

info\_snp <- rbind(info\_snp, tmp\_snp)

# Assign the genotype to a variable for easier downstream analysis

genotype <- obj.bigSNP$genotypes

# Rename the data structures

CHR <- map$chr

POS <- map$pos

# get the CM information from 1000 Genome

# will download the 1000G file to the current directory (".")

POS2 <- snp\_asGeneticPos(CHR, POS, dir = ".")

# calculate LD

# Extract SNPs that are included in the chromosome

ind.chr <- which(tmp\_snp$chr == chr)

ind.chr2 <- tmp\_snp$`\_NUM\_ID\_`[ind.chr]

# Calculate the LD

corr0 <- snp\_cor(

genotype,

ind.col = ind.chr2,

ncores = NCORES,

infos.pos = POS2[ind.chr2],

size = 3 / 1000

)

if (chr == 1) {

ld <- Matrix::colSums(corr0^2)

corr <- as\_SFBM(corr0, tmp)

} else {

ld <- c(ld, Matrix::colSums(corr0^2))

corr$add\_columns(corr0, nrow(corr))

}

# We assume the fam order is the same across different chromosomes

if(is.null(fam.order)){

fam.order <- as.data.table(obj.bigSNP$fam)

}

}

# Rename fam order

setnames(fam.order,

c("family.ID", "sample.ID"),

c("FID", "IID"))

3. 进行LD score回归

df\_beta <- info\_snp[,c("beta", "beta\_se", "n\_eff", "\_NUM\_ID\_")]

ldsc <- snp\_ldsc( ld,

length(ld),

chi2 = (df\_beta$beta / df\_beta$beta\_se)^2,

sample\_size = df\_beta$n\_eff,

blocks = NULL)

h2\_est <- ldsc[["h2"]]

* + - 1. 计算null R2

Quantitative trait：

# Reformat the phenotype file such that y is of the same order as the

# sample ordering in the genotype file

y <- pheno[fam.order, on = c("FID", "IID")]

# Calculate the null R2

# use glm for binary trait

# (will also need the fmsb package to calculate the pseudo R2)

null.model <- paste("PC", 1:6, sep = "", collapse = "+") %>%

paste0("Height~Sex+", .) %>%

as.formula %>%

lm(., data = y) %>%

summary

null.r2 <- null.model$r.squared

Binary trait：

library(fmsb)

# Reformat the phenotype file such that y is of the same order as the

# sample ordering in the genotype file

y <- pheno[fam.order, on = c("FID", "IID")]

# Calculate the null R2

# use glm for binary trait

# (will also need the fmsb package to calculate the pseudo R2)

null.model <- paste("PC", 1:6, sep = "", collapse = "+") %>%

paste0("Height~Sex+", .) %>%

as.formula %>%

glm(., data = y, family=binomial) %>%

summary

null.r2 <- fmsb::NagelkerkeR2(null.model)

infinitesimal model

beta\_inf <- snp\_ldpred2\_inf(corr, df\_beta, h2 = h2\_est)

grid model

# Prepare data for grid model

p\_seq <- signif(seq\_log(1e-4, 1, length.out = 17), 2)

h2\_seq <- round(h2\_est \* c(0.7, 1, 1.4), 4)

grid.param <-

expand.grid(p = p\_seq,

h2 = h2\_seq,

sparse = c(FALSE, TRUE))

# Get adjusted beta from grid model

beta\_grid <-

snp\_ldpred2\_grid(corr, df\_beta, grid.param, ncores = NCORES)

auto model

# Get adjusted beta from the auto model

multi\_auto <- snp\_ldpred2\_auto(

corr,

df\_beta,

h2\_init = h2\_est,

vec\_p\_init = seq\_log(1e-4, 0.9, length.out = NCORES),

ncores = NCORES

)

beta\_auto <- sapply(multi\_auto, function(auto)

auto$beta\_est)

* + - 1. 获得PRS评分

详情见[LDpred-2 - Basic Tutorial for Polygenic Risk Score Analyses (choishingwan.github.io)](https://choishingwan.github.io/PRS-Tutorial/ldpred/)

* + - 1. Get the final performance of the LDpred models