**Practical Exercise – Polygenic Risk Scores**

In this practical, we will generate at polygenic risk score (PRS) for height and see how much variation in height it explains. This practical is a reduced set of analyses provided in the tutorial by Shing Wan Choi and colleagues, which can be found at the following link: <https://choishingwan.github.io/PRS-Tutorial/>. Please review the tutorial for more detailed information, and calculation of the PRS using the additional methods that were discussed in the lecture.

The practical is separated into four main sections, corresponding to the guide in the Nature Protocols paper discussed in the lecture (Choi et al. Nat Proc: 15, 2759-2772 (2020); <https://www.nature.com/articles/s41596-020-0353-1>):

1. Quality control (QC) of the base data
2. QC of the target data
3. Calculating and analysing the PRS
4. Visualising the PRS results

Please refer to the paper as you work through the practical.

Any command line code will be presented in red text, R code in green text and questions to answer in blue text.

1. **QC of base data**

The first step in Polygenic Risk Score (PRS) analyses is to generate or obtain the base data (GWAS summary statistics). Ideally these will correspond to the most powerful GWAS results available on the phenotype under study. In this example, we will use GWAS summary statistics from simulated height data (these will be provided in the folder with this document).

Height.gwas.txt.gz is compressed. To read its content, you can type:

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

which will display the first 10 lines of the file. The Height.gwas.txt.gz file contains the following columns:

* CHR: The chromosome in which the SNP resides
* BP: Chromosomal co-ordinate of the SNP
* SNP: SNP ID, usually in the form of rs-ID
* A1: The effect allele of the SNP
* A2: The non-effect allele of the SNP
* N: Number of samples used to obtain the effect size estimate
* SE: The standard error (SE) of the effect size esimate
* P: The P-value of association between the SNP genotypes and the base phenotype
* OR: The effect size estimate of the SNP, if the outcome is binary/case-control. If the outcome is continuous or treated as continuous then this will usually be BETA
* INFO: The imputation information score
* MAF: The minor allele frequency (MAF) of the SNP

QC checklist: Base data

1. Heritability check

We recommend that PRS analyses are performed on base data with a SNP-heritability estimate >0.05. The SNP-heritability of a GWAS can be estimated using e.g. LD Score Regression (LDSC). These height GWAS data are simulated to have a SNP-heritability much greater than 0.05 and so we can move on to the next QC step. For practice, try calculating the SNP-heritability using LDSC in your own time and ensure that it is greater than 0.05.

1. Effect allele

It is important to know which allele is the effect allele and which is the non-effect allele for PRS association results to be in the correct direction. Some GWAS results files do not make clear which allele is the effect allele and which is the non-effect allele. If the incorrect assumption is made in computing the PRS, then the effect of the PRS in the target data will be in the wrong direction. To avoid misleading conclusions the effect allele from the base (GWAS) data must be known.

1. Genome build

You must check that your base and target data are on the same genome build, and if they are not then use a tool such as LiftOver to make the builds consistent across the data sets. The height summary statistic are on the same genome build as the target data that we will be using.

1. Standard GWAS QC

Both the base and target data should be subjected to the standard stringent QC steps performed in GWAS. If the base data have been obtained as summary statistics from a public source, then the typical QC steps that you will be able to perform on them are to filter the SNPs according to imputation information (INFO) score and minor allele frequency (MAF). SNPs with low MAF or INFO are more likely to generate false positive results due to their lower statistical power (and higher probability of genotyping errors in the case of low MAF). Therefore, SNPs with low MAF and INFO are typically removed before performing downstream analyses. We recommend removing SNPs with MAF < 1% and INFO < 0.8 (with very large base sample sizes these thresholds could be reduced if sensitivity checks indicate reliable results). These SNP filters can be achieved using the following code:

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

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

gzip > Height.gz

The bash code above does the following:

Decompresses and reads the Height.gwas.txt.gz file

Prints the header line (NR==1)

Prints any line with MAF above 0.01 ($11 because the eleventh column of the file contains the MAF information)

Prints any line with INFO above 0.8 ($10 because the tenth column of the file contains the INFO information)

Compresses and writes the results to Height.gz

1. Mismatching SNPs

SNPs that have mismatching alleles reported in the base and target data are either resolvable by "strand-flipping" the alleles to their complementary alleles in e.g. the target data, such as for a SNP with A/C in the base data and G/T in the target, or non-resolvable, such as for a SNP with C/G in the base and C/T in the target. Most polygenic score software perform strand-flipping automatically for SNPs that are resolvable, and remove non-resolvable mismatching SNPs. Since we need the target data to know which SNPs have mismatching alleles, we will perform this strand-flipping in the target data.

1. Duplicate SNPs

If an error has occurred in the generation of the base data, then there may be duplicated SNPs in the base data file. Most PRS software do not allow duplicated SNPs in the base data input and thus they should be removed, using a command such as the one below:

gunzip -c Height.gz |**\**

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

gzip - > Height.nodup.gz

The above command does the following:

Decompresses and reads the Height.gz file

Count number of time SNP ID was observed, assuming the third column contain the SNP ID (seen[$3]++). If this it the first time seeing this SNP ID, print it.

Compresses and writes the results to Height.nodup.gz

How many duplicated SNPs are there?

1. Ambiguous SNPs

If the base and target data were generated using different genotyping chips and the chromosome strand (+/-) that was used for either is unknown, then it is not possible to pair-up the alleles of ambiguous SNPs (i.e. those with complementary alleles, either C/G or A/T SNPs) across the data sets, because it will be unknown whether the base and target data are referring to the same allele or not. While allele frequencies could be used to infer which alleles are on the same strand, the accuracy of this could be low for SNPs with MAF close to 50% or when the base and target data are from different populations. Therefore, we recommend removing all ambiguous SNPs to avoid introducing this potential source of systematic error.

Ambiguous SNPs can be removed in the base data and then there will be no such SNPs in the subsequent analyses, since analyses are performed only on SNPs that overlap between the base and target data.

Non-ambiguous SNPs can be retained using the following:

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

How many non-ambiguous SNPs were there?

1. Sample overlap and relatedness

Since the target data were simulated there are no overlapping samples between the base and target data here (see the relevant section of the paper for discussion of the importance of avoiding sample overlap).

Closely related individuals within and between the base and the target data may lead to overfitted results, limiting the generalizability of the results (see the relevant sections of the paper). Relatedness within the target data is tested in the Target Data section.

The Height.QC.gz base data are now ready for using in downstream analyses.

1. **QC of target data**

Target data consist of individual-level genotype-phenotype data, usually generated within your lab/department/collaboration. For this tutorial, we have simulated some genotype-phenotype data using the 1000 Genomes Project European samples. You can download the data here

Unzip the data as follow:

unzip EUR.zip

QC checklist: Target data

1. Sample size

We recommend that users only perform PRS analyses on target data of at least 100 individuals. The sample size of our target data here is 503 individuals.

1. Genome build

As stated in the base data section, the genome build for our base and target data is the same, as it should be.

1. Standard GWAS QC

The target data must be quality controlled to at least the standards implemented in GWAS studies, e.g. removing SNPs with low genotyping rate, low minor allele frequency, out of Hardy-Weinberg Equilibrium, removing individuals with low genotyping rate (see Marees et al; <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6001694/>).

The following plink command applies some of these QC metrics to the target data:

./plink **\**

--bfile EUR **\**

--maf 0.01 **\**

--hwe 1e-6 **\**

--geno 0.01 **\**

--mind 0.01 **\**

--write-snplist **\**

--make-just-fam **\**

--out EUR.QC

Each of the parameters corresponds to the following

| **Parameter** | **Value** | **Description** |
| --- | --- | --- |
| bfile | EUR | Informs plink that the input genotype files should have a prefix of EUR |
| maf | 0.01 | Removes all SNPs with minor allele frequency less than 0.01. Genotyping errors typically have a larger influence on SNPs with low MAF. Studies with large sample sizes could apply a lower MAF threshold |
| hwe | 1e-6 | Removes SNPs with low P-value from the Hardy-Weinberg Equilibrium Fisher's exact or chi-squared test. SNPs with significant P-values from the HWE test are more likely affected by genotyping error or the effects of natural selection. Filtering should be performed on the control samples to avoid filtering SNPs that are causal (under selection in cases). When phenotype information is included, plink will automatically perform the filtering in the controls. |
| geno | 0.01 | Excludes SNPs that are missing in a high fraction of subjects. A two-stage filtering process is usually performed (see [Marees et al](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6001694/)). |
| mind | 0.01 | Excludes individuals who have a high rate of genotype missingness, since this may indicate problems in the DNA sample or processing. (see [Marees et al](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6001694/) for more details). |
| make-just-fam | - | Informs plink to only generate the QC'ed sample name to avoid generating the .bed file. |
| write-snplist | - | Informs plink to only generate the QC'ed SNP list to avoid generating the .bed file. |
| out | EUR.QC | Informs plink that all output should have a prefix of EUR.QC |

How many SNPs and samples were filtered out of the data?

Very high or low heterozygosity rates in individuals could be due to DNA contamination or to high levels of inbreeding. Therefore, samples with extreme heterozygosity are typically removed prior to downstream analyses.

First, we perform pruning to remove highly correlated SNPs:

./plink **\**

--bfile EUR **\**

--keep EUR.QC.fam **\**

--extract EUR.QC.snplist **\**

--indep-pairwise 200 50 0.25 **\**

--out EUR.QC

Each of the parameters corresponds to the following

| **Parameter** | **Value** | **Description** |
| --- | --- | --- |
| bfile | EUR | Informs plink that the input genotype files should have a prefix of EUR |
| keep | EUR.QC.fam | Informs plink that we only want to use samples in EUR.QC.fam in the analysis |
| extract | EUR.QC.snplist | Informs plink that we only want to use SNPs in EUR.QC.snplist in the analysis |
| indep-pairwise | 200 50 0.25 | Informs plink that we wish to perform pruning with a window size of 200 variants, sliding across the genome with step size of 50 variants at a time, and filter out any SNPs with LD r2 higher than 0.25 |
| out | EUR.QC | Informs plink that all output should have a prefix of EUR.QC |

This will generate two files 1) **EUR.QC.prune.in** and 2) **EUR.QC.prune.out**. All SNPs within **EUR.QC.prune.in** have a pairwise r2<0.25.

Heterozygosity rates can then be computed using plink:

./plink **\**

--bfile EUR **\**

--extract EUR.QC.prune.in **\**

--keep EUR.QC.fam **\**

--het **\**

--out EUR.QC

This will generate the EUR.QC.het file, which contains F coefficient estimates for assessing heterozygosity. We will remove individuals with F coefficients that are more than 3 standard deviation (SD) units from the mean, which can be performed using the following R command:

dat <- read.table("EUR.QC.het", header=T) *# Read in the EUR.het file, specify it has header*

m <- mean(dat$F) *# Calculate the mean*

s <- sd(dat$F) *# Calculate the SD*

valid <- subset(dat, F <= m+3\*s & F >= m-3\*s) *# Get any samples with F coefficient within 3 SD of the population mean*

write.table(valid[,c(1,2)], "EUR.valid.sample", quote=F, row.names=F) *# print FID and IID for valid samples*

q() *# exit R*

How many samples were excluded due to high or low heterozygosity rate?

1. Mismatching SNPs

SNPs that have mismatching alleles reported in the base and target data may be resolvable by strand-flipping the alleles to their complementary alleles in e.g. the target data, such as for a SNP with A/C in the base data and G/T in the target. Most PRS software will perform strand-flipping automatically. Check your software does this before calculating your PRS; if it does not, there is some R code in the online tutorial for performing strand-flipping (<https://choishingwan.github.io/PRS-Tutorial/target/>).

Run using the following R script to correct mismatching alleles:

Rscript Ambiguous\_SNPs.R

1. Duplicate SNPs

Make sure to remove any duplicate SNPs in your target data (these target data were simulated and so include no duplicated SNPs).

1. Sex chromosomes

Sometimes sample mislabelling can occur, which may lead to invalid results. One indication of a mislabelled sample is a difference between reported sex and that indicated by the sex chromosomes. While this may be due to a difference in sex and gender identity, it could also reflect mislabeling of samples or misreporting and, thus, individuals in which there is a mismatch between biological and reported sex are typically removed. A sex check can be performed in PLINK, in which individuals are called as females if their X chromosome homozygosity estimate (F statistic) is < 0.2 and as males if the estimate is > 0.8.

Before performing a sex check, pruning should be performed. A sex check can then easily be conducted using plink:

./plink **\**

--bfile EUR **\**

--extract EUR.QC.prune.in **\**

--keep EUR.valid.sample **\**

--check-sex **\**

--out EUR.QC

This will generate a file called EUR.QC.sexcheck containing the F-statistics for each individual. Individuals are typically called as being biologically male if the F-statistic is > 0.8 and biologically female if F < 0.2. We can read this file into R and exclude those individuals whose reported sex doesn’t match their biological sex:

valid <- read.table("EUR.valid.sample", header=T) *# Read in file*

dat <- read.table("EUR.QC.sexcheck", header=T)

valid <- subset(dat, STATUS=="OK" & FID %in% valid$FID)

write.table(valid[,c("FID", "IID")], "EUR.QC.valid", row.names=F, col.names=F, sep="\t", quote=F)

q() *# exit R*

How many samples were excluded due to mismatched sex information?

1. Relatedness

Closely related individuals in the target data may lead to overfitted results, limiting the generalisability of the results.

Before calculating the relatedness, pruning should be performed. Individuals that have a first or second degree relative in the sample (>0.125) can be removed with the following command:

./plink **\**

--bfile EUR **\**

--extract EUR.QC.prune.in **\**

--keep EUR.QC.valid **\**

--rel-cutoff 0.125 **\**

--out EUR.QC

How many related samples were excluded?

1. Generate final QC'ed target data file

After performing the full analysis, you can generate a QC'ed data set with the following command:

./plink **\**

--bfile EUR **\**

--make-bed **\**

--keep EUR.QC.rel.id **\**

--out EUR.QC **\**

--extract EUR.QC.snplist **\**

--exclude EUR.mismatch **\**

--a1-allele EUR.a1

Each of the parameters corresponds to the following

| **Parameter** | **Value** | **Description** |
| --- | --- | --- |
| bfile | EUR | Informs plink that the input genotype files should have a prefix of EUR |
| keep | EUR.QC.rel.id | Informs plink that we only want to keep samples in EUR.QC.rel.id |
| extract | EUR.QC.snplist | Informs plink that we only want to use SNPs in EUR.QC.snplist in the analysis |
| exclude | EUR.mismatch | Informs plink that we wish to remove any SNPs in EUR.mismatch |
| a1-allele | EUR.a1 | Fix all A1 alleles to those specified in EUR.a1 |
| out | EUR.QC | Informs plink that all output should have a prefix of EUR.QC |

1. **Calculating and analysing the PRS**

You will compute PRS using the popular genetic analyses tool plink - while plink is not a dedicated PRS software, you can perform every required steps of the C+T approach with plink. This multi-step process is a good way to learn the processes involved in computing PRS, which are typically performed automatically by PRS software.

1. Required Data

In the previous sections, we have generated the following files:

| **File Name** | **Description** |
| --- | --- |
| **Height.QC.gz** | The post-QCed summary statistic |
| **EUR.QC.bed** | The genotype file after performing some basic filtering |
| **EUR.QC.bim** | This file contains the SNPs that passed the basic filtering |
| **EUR.QC.fam** | This file contains the samples that passed the basic filtering |
| **EUR.height** | This file contains the phenotype of the samples |
| **EUR.cov** | This file contains the covariates of the samples |

1. Update Effect Size

When the effect size relates to disease risk and is thus given as an odds ratio (OR), rather than BETA (for continuous traits), then the PRS is computed as a product of ORs. To simplify this calculation, we take the natural logarithm of the OR so that the PRS can be computed using summation instead (which can be back-transformed afterwards). If the effect size is already reported as a BETA, then this step is not required. We can obtain the transformed summary statistics with R:

dat <- read.table(gzfile("Height.QC.gz"), header=T)

dat$BETA <- log(dat$OR)

write.table(dat, "Height.QC.Transformed", quote=F, row.names=F)

q()

1. Clumping

Linkage disequilibrium, which corresponds to the correlation between the genotypes of genetic variants across the genome, makes identifying the contribution from causal independent genetic variants extremely challenging. One way of approximately capturing the right level of causal signal is to perform clumping, which removes SNPs in ways that only weakly correlated SNPs are retained but preferentially retaining the SNPs most associated with the phenotype under study. Clumping can be performed using the following command in plink:

./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

Each of the new parameters corresponds to the following

| **Parameter** | **Value** | **Description** |
| --- | --- | --- |
| clump-p1 | 1 | P-value threshold for a SNP to be included as an index SNP. 1 is selected such that all SNPs are include for clumping |
| clump-r2 | 0.1 | SNPs having r2 higher than 0.1 with the index SNPs will be removed |
| clump-kb | 250 | SNPs within 250k of the index SNP are considered for clumping |
| clump | Height.QC.Transformed | Base data (summary statistic) file containing the P-value information |
| clump-snp-field | SNP | Specifies that the column SNP contains the SNP IDs |
| clump-field | P | Specifies that the column P contains the P-value information |

This will generate EUR.clumped, containing the index SNPs after clumping is performed. We can extract the index SNP ID by performing the following command:

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

$3 because the third column contains the SNP ID

Note: If your target data are small (e.g. N < 500) then you can use the 1000 Genomes Project samples for the LD calculation. Make sure to use the population that most closely reflects represents the base sample.

1. Generate PRS

plink provides a convenient function --score and --q-score-range for calculating polygenic scores.

We will need three files:

1. The base data file: Height.QC.Transformed
2. A file containing SNP IDs and their corresponding P-values ($3 because SNP ID is located in the third column; $8 because the P-value is located in the eighth column)

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

1. A file containing the different P-value thresholds for inclusion of SNPs in the PRS. Here calculate PRS corresponding to a few thresholds for illustration purposes:

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

The format of the **range\_list** file should be as follows:

Name of threshold Lower bound Upper bound

Note: The threshold boundaries are inclusive. For example, for the 0.05 threshold, we include all SNPs with P-value from 0 to 0.05, **including** any SNPs with P-value equal to 0.05.

We can then calculate the PRS with the following plink command:

./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

The meaning of the new parameters are as follows:

| **Paramter** | **Value** | **Description** |
| --- | --- | --- |
| score | Height.QC.Transformed 3 4 12 header | We read from the **Height.QC.Transformed** file, assuming that the 3st column is the SNP ID; 4th column is the effective allele information; the 12th column is the effect size estimate; and that the file contains a header |
| q-score-range | range\_list SNP.pvalue | We want to calculate PRS based on the thresholds defined in **range\_list**, where the threshold values (P-values) were stored in **SNP.pvalue** |

The above command and range\_list will generate 7 files:

1. EUR.0.5.profile
2. EUR.0.4.profile
3. EUR.0.3.profile
4. EUR.0.2.profile
5. EUR.0.1.profile
6. EUR.0.05.profile
7. EUR.0.001.profile
8. Accounting for Population Stratification

Population structure is the principal source of confounding in GWAS and is usually accounted for by incorporating principal components (PCs) as covariates. We can incorporate PCs into our PRS analysis to account for population stratification.

Again, we can calculate the PCs using plink:

*# 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

Here the PCs have been stored in the **EUR.eigenvec** file and can be used as covariates in the regression model to account for population stratification.

1. Finding the "best-fit" PRS

The P-value threshold that provides the "best-fit" PRS under the C+T method is usually unknown. To approximate the "best-fit" PRS, we can perform a regression between PRS calculated at a range of P-value thresholds and then select the PRS that explains the highest phenotypic variance (please see Section 4.6 of our paper on overfitting issues). This can be achieved using R as follows:

p.threshold <- c(0.001,0.05,0.1,0.2,0.3,0.4,0.5) *# P-value thresholds used to generate PRS*

phenotype <- read.table("EUR.height", header=T) *# Read in the phenotype file*

pcs <- read.table("EUR.eigenvec", header=F) *# Read in the PCs*

colnames(pcs) <- c("FID", "IID", paste0("PC",1:6)) *# The default output from plink does not include a header. To make things simple, we will add the appropriate headers (1:6 because there are 6 PCs)*

covariate <- read.table("EUR.cov", header=T) *# Read in the covariates*

pheno <- merge(merge(phenotype, covariate, by=c("FID", "IID")), pcs, by=c("FID","IID")) *# Now merge the files*

null.model <- lm(Height~., data=pheno[,!colnames(pheno)%in%c("FID","IID")]) *# We can then calculate the null model (model without PRS) using a linear regression (as height is quantitative)*

null.r2 <- summary(null.model)$r.squared *# The R2 of the null model*

prs.result <- **NULL** *# Matrix to hold the analysis results of each PRS*

for(i in p.threshold){

*# Go through each p-value threshold*

prs <- read.table(paste0("EUR.",i,".profile"), header=T)

*# Merge the prs with the phenotype matrix. We only want the FID,*

*IID and PRS from the PRS file, therefore we only select the*

*relevant columns*

pheno.prs <- merge(pheno, prs[,c("FID","IID", "SCORE")],

by=c("FID", "IID"))

*# Now perform a linear regression on Height with PRS and the*

*covariates ignoring the FID and IID from our model*

model <- lm(Height~.,

data=pheno.prs[,!colnames(pheno.prs)%in%c("FID","IID")])

*# model R2 is obtained as*

model.r2 <- summary(model)$r.squared

*# R2 of PRS is simply calculated as the model R2 minus the null R2*

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

*# We can also obtain the coeffcient and p-value of association of*

*PRS as follow*

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

prs.beta <- as.numeric(prs.coef[1])

prs.se <- as.numeric(prs.coef[2])

prs.p <- as.numeric(prs.coef[4])

*# We can then store the results*

prs.result <- rbind(prs.result, data.frame(Threshold=i, R2=prs.r2,

P=prs.p, BETA=prs.beta,SE=prs.se))

}

prs.result[which.max(prs.result$R2),] *# The best result*

Which P-value threshold generates the “best-fit” PRS?

How much phenotypic variance does the “best-fit” PRS explain?

1. **Visualising the PRS results**

The PRS results corresponding to a range of P-value thresholds obtained by application of the C+T PRS method can be visualised using R as follows:

*# We strongly recommend the use of ggplot2 (see code in online tutorial – I’ve included this code here in case some people are unable to install the ggplot2 package).*

*# Specify that we want to generate plot in EUR.height.bar.png*

png("EUR.height.bar.png", height=10, width=10, res=300, unit="in")

*# First, obtain the colorings based on the p-value*

col <- suppressWarnings(colorRampPalette(c("dodgerblue", "firebrick")))

*# We want the color gradient to match the ranking of p-values*

prs.result <- prs.result[order(-log10(prs.result$P)),]

prs.result$color <- col(nrow(prs.result))

prs.result <- prs.result[order(prs.result$Threshold),]

*# generate a pretty format for p-value output*

prs.result$print.p <- round(prs.result$P, digits = 3)

prs.result$print.p[!is.na(prs.result$print.p) & prs.result$print.p == 0 ] <- format(prs.result$P[!is.na(prs.result$print.p) & prs.result$print.p == 0 ], digits = 2)

prs.result$print.p <- sub("e", "\*x\*10^", prs.result$print.p)

*# Generate the axis labels*

xlab <- expression(italic(P) - value ~ threshold ~ (italic(P)[T]))

ylab <- expression(paste("PRS model fit: ", R ^ 2))

*# Setup the drawing area*

layout(t(1:2), widths=c(8.8,1.2))

par( cex.lab=1.5, cex.axis=1.25, font.lab=2,

oma=c(0,0.5,0,0),

mar=c(4,6,0.5,0.5))

*# Plotting the bars*

b <- barplot(height=prs.result$R2,

col=prs.result$color,

border=**NA**,

ylim=c(0, max(prs.result$R2)\*1.25),

axes = F, ann=F)

*# Plot the axis labels and axis ticks*

odd <- seq(0,nrow(prs.result)+1,2)

even <- seq(1,nrow(prs.result),2)

axis(side=1, at=b[odd], labels=prs.result$Threshold[odd], lwd=2)

axis(side=1, at=b[even], labels=prs.result$Threshold[even],lwd=2)

axis(side=1, at=c(0,b[1],2\*b[length(b)]-b[length(b)-1]), labels=c("","",""), lwd=2, lwd.tick=0)

*# Write the p-value on top of each bar*

text( parse(text=paste(

prs.result$print.p)),

x = b+0.1,

y = prs.result$R2+ (max(prs.result$R2)\*1.05-max(prs.result$R2)),

srt = 45)

*# Now plot the axis lines*

box(bty='L', lwd=2)

axis(2,las=2, lwd=2)

*# Plot the axis titles*

title(ylab=ylab, line=4, cex.lab=1.5, font=2 )

title(xlab=xlab, line=2.5, cex.lab=1.5, font=2 )

*# Generate plot area for the legend*

par(cex.lab=1.5, cex.axis=1.25, font.lab=2, mar=c(20,0,20,4))

prs.result <- prs.result[order(-log10(prs.result$P)),]

image(1, -log10(prs.result$P), t(seq\_along(-log10(prs.result$P))), col=prs.result$color, axes=F,ann=F)

axis(4,las=2,xaxs='r',yaxs='r', tck=0.2, col="white")

*# plot legend title*

title(bquote(atop(-log[10] ~ model, italic(P) - value),), line=2, cex=1.5, font=2, adj=0)

*# write the plot to file*

dev.off()

q() *# exit R*