Appendix 1

Example Script for Creating Polygenic Scores

Fazil Aliev ([faliev@vcu.edu](mailto:faliev@vcu.edu)) and Sally Kuo ([ickuo@vcu.edu](mailto:ickuo@vcu.edu))

Files and software needed:

1. Discovery sample GWAS based on preferably on the same ancestry individuals as target sample individuals for phenotype of interest. This file must have Chromosome, Marker/SNP name, position, effect and alternative alleles, GWAS beta or OR, p-value. Extra information like MAF, imputation quality also might be helpful for extra filtering and QC. Let’s assume that GWAS file name is discovery\_example.txt. We will use [discovery\_path]/discovery\_example.txt

to refer this file. We use brackets ([]) to refer discovery GWAS file path on the server. In most cases this file contains one line per marker after removing possible lines per marker - per covariate and one header line. Here is an example of top lines of discovery\_example.txt file:

CHR SNP BP A1 A2 INFO Beta SE P

8 rs62513865 101592213 t c 0.949 -1.006 0.0271 0.8086

8 rs79643588 106973048 a g 0.997 1.0178 0.0244 0.4606

8 rs17396518 108690829 t g 0.987 0.9612 0.0143 0.0046

8 rs983166 108681675 a c 0.998 0.9799 0.0139 0.1452

8 rs35107696 109712249 a at 0.999 1.0130 0.0165 0.4302

8 rs37704624 105176418 t ttc 1 1.0085 0.0157 0.5867

8 rs7014597 104152280 c g 0.993 1.0192 0.0182 0.2934

8 rs3134156 100479917 t c 0.998 0.9824 0.0190 0.3526

8 rs6980591 103144592 a c 0.997 1.0449 0.0167 0.0083

8 rs72670434 108166508 a t 0.985 1.0126 0.0147 0.3898

1. Target sample genotype information. This information can be in any of uncompressed or compressed formats that Plink (Chang et al., 2015; Purcell et al., 2007) accepts. Assume that Plink standard format .bed (.bim, .fam) is used and file names are [target\_path]/target\_example.bed, [target\_path]/target\_example.bim and [target\_path]/target\_example.fam which is referred in plink as

--bfile [target\_path]/target\_example*.*

For simplicity we also assume that markers of discovery and target files have the same allele codes, same map/positions. When maps are different any uplifting algorithm can match discovery and target marker maps.

1. Plink (https://zzz.bwh.harvard.edu/plink/plink2.shtml) software installed (Any of Plink 1 or Plink 2 versions can work but plink 2 is much faster).
2. PRS-CS (Ge et al., 2019) software installed (https://github.com/getian107/PRScs) (python also must be installed to use PRS-CS, but for P+T scores PRS-CS and python are not needed).

We explain here only commands based on the unix/linux environment, but commands for other software and programming language environment are also similar. Chromosome and position columns will not be used in analysis because chromosome number and marker positions will not be the same (originally or after possible uplifting) between the discovery and target samples. Accordingly, only marker name (SNP column above) will be used to match SNPs. In the example above allele names have small letters. Target sample usually uses capital letters. To match SNPs we need to capitalize allele letters. The command to capitalize possible allele letters is:

sed -i -e 's/g/G/g; s/c/C/g; s/t/T/g; s/a/A/g' [discovery\_path]/discovery\_example.txt

Results will be written to the same file. Allele letters in the file will be capitalized after running this command.

As we see in the above example rs7014597(c, g alleles*)* andrs72670434 (a, talleles),SNPs are palindromic and should be removed. Allele columns are located in columns 4 and 5. All palindromic SNPs (i.e., having allele pairs AT, TA, CG, GC) must be removed (with the exceptions when the strand in genotyping/imputation is the same across the discovery and target samples. Also SNPs rs35107696 and rs377046245 have alleles having more than one letters and must be removed. We will keep only SNPs with allele pairs "AC", "AG", "CA", "CT", "GA", "GT", "TC" and "TG", which will ensure that all palindromic SNPs have been removed and only biallelic SNPs are included. You might need to remove all markers with no p-value. Usually, in a GWAS results file we see period (.) or NA when p-value or Beta/OR values are missing. P-value column number is 9 and $9>0 restriction in the following command will filter out all lines with no positive p-value.

Let’s write both filters to one single command:

gawk -F " " '($4$5=="AC" || $4$5=="AG" || $4$5=="CA" || $4$5=="CT" || $4$5=="GA" || $4$5=="GT" || $4$5=="TC" || $4$5=="TG") && $9>0 || $1=="CHR" {print $0}' [discovery\_path]/discovery\_example.txt > [work\_path]/discovery\_cleaned.txt

Note that depending on your unix system you might need to replace gawk to awk. gawk is a powerful version of awk some unix systems might only have awk.

Note that in the above command $4$5 pairs corresponds to the first and second allele columns (4 and 5) which must be changed properly when the allele columns in the discovery GWAS file are different. Additional filtering (by, for example, MAF, Hardy-Weinberg (H-W), INFO) can be performed at this step, if necessary. Results of the command will be written to [work\_path]/discovery\_cleaned.txtfile and look like:

CHR SNP BP A1 A2 INFO Beta SE P

8 rs62513865 101592213 T C 0.949 -1.006 0.0271 0.8086

8 rs79643588 106973048 A G 0.997 1.0178 0.0244 0.4606

8 rs17396518 108690829 T G 0.987 0.9612 0.0143 0.0046

8 rs983166 108681675 A C 0.998 0.9799 0.0138 0.1452

8 rs3134156 100479917 T C 0.998 0.9824 0.0190 0.3526

8 rs6980591 103144592 A C 0.997 1.0449 0.0167 0.0083

…

We can run some filtering based on INFO and missingness. If there is MAF and H-W information available in discovery sample, we can also add MAF and H-W filters.

gawk -F " " '$9 != "NA" && $6>0.8 {print $2 $4 $5 $7 $9}' [discovery\_path]/discovery\_cleaned.txt > output\_file.txt

$9 and $6 in this command refer to the P-value column (column #9) and the INFO column (column #6). $9 != "NA"means to drop lines with p-value ="NA".

{print $2 $4 $5 $7 $9}prints *SNP, A1, A2, BETA* and *P* columns, where column numbers are followed by the $ sign.

PRS-CS (Ge et al., 2019) creating method has two steps. First, we create new weights based on discovery GWAS and external panel information provided by PRS-CS software. Second we apply Plink *--score* method to new weights in target sample and create scores.

Need to make sure PRS-CS and python are installed properly.

python [PRS-CS directory]/PRScs.py --ref\_dir=[ref files directory]/[ancestry reference file] --bim\_prefix=[target\_path]/target\_example --sst\_file=[work\_path]/discovery\_cleaned.txt --chrom=[chromosomome no] --n\_gwas=[gwas sample size] --out\_dir=[output file name with PRS-CS new weights with full path] --seed=123

Here [ancestry reference file]is the file with the ancestry as the discovery ancestry if discovery sample has one ancestry. Corresponding ancestry files are provided by the software. File names start with ldblk\_1kg*\_* followed by ancestry. PRS-CS software provides an output file with the new weights for SNPs for each chromosome. Combined file with all chromosome outputs with new weights is used with Plink -- *score* method.

Chromosome files can be combined with

cat [output file name with PRS-CS new weights with full path]\* > [combined PRS-CS weights file]

Plink command to create final score looks like:

[Plink full path name]/plink --bfile [target\_path]/target\_example --allow-no-sex --out [final output PRS-CS file full name] --score [combined PRS-CS weights file] 2 4 6 no-sum

The last Plink command will create a final PRS-CS file [final output PRS-CS file full name].profilewhich can be used for further prediction analyses.

**Using single R script to prepare and send all commands to the system to create PRS-CS score**

It is possible to use R to send all above mentioned commands to the unix system. R code here creates PRS-CS without any other commands. By copying, pasting and running the script below you can create PRS-CS for any discovery GWAS file and target genotype file. Discovery GWAS file and three target sample genotype files (Plink format) must be located in the “data” directory. All necessary parameters are listed in the beginning of the script. The rest of the script works using parameters entered on the top part.

### This R script runs basic steps of PRS-CS creating on Linux/Unix systems

### Requires installed Plink, PRS-CS, R and Python

### (Most of systems have R and Python, please check first and install if needed)

### All chromosomes run at once and it is slow.

### Add --chrom=[chr\_no] option to run by chromosome for fast runs

### To run the script:

### 1) Copy this file to working directory (~/PRS-CS\_test),

### 2) Copy discovery GWAS file and targer genotype files to data\_dir

### (~/PRS-CS\_test/data)

### 3) Make sure that Plink and PRS-CS (R and Python) are installed on the system

### 4) Correct all parameters below before "### End of parameters" line and save

### 5) Use system command : cd ~/PRS-CS\_test

### and then type : [R\_ path\_if\_needed/]R --no-save < prs\_cs\_tesr.R

### Enter all parameters here:

### Working directory on root (put "/" at the end)

wd="~/PRS-CS\_test/"

### Data directory: put discovery and target sample files here

data\_dir=paste0(wd,"data/")

### Directory, where PRS-CS is located (has "PRScs-master" directory inside) prs\_cs\_dir=paste0("[put\_PRS-CS\_directory\_here]","/")

phi=0 ## PRS-CS threshold phi parameter phi=0 uses ## auto, pho>0 uses phi as threshold

plink\_exe=paste0("put\_plink\_directory\_here","plink") ## Put full path for plink exe file

### Discovery GWAS file parameters:

discovery\_gwas="discovery\_gwas.txt" ## Discovery GWAS file name in working directory

n\_discovery\_gwas=10000 ## discovery GWAS sample size

gwas\_delimiter=" " ## gwas delimiter, use "," if csv

ancestry="EUR" ## discovery GWAS sample ancesty

or\_is\_used=0 ## or\_is\_used=0 if discovery GWAS has BETA

## or\_is\_used=1 is for OR(Odds Ratio)

snp\_column\_no=1 ## SNP/marker (rs) column number

ref\_allele\_column\_no=4 ## Reference (A1) column number

alt\_allele\_column\_no=5 ## Alternative (A2) column number

beta\_column\_no=7 ## Beta (OR if binary) cloumn number

p\_value\_column\_no=9 ## p-value column number

### Target sample genotype file parameters:

target\_gwas = "target\_file\_name" ## Target sample Plink format genotype file

## name w/o extension (.bed, .bim, .fam)

### End of parameters

############## Do not change below, all necessary parameters are above ########

### Setting working directory and creating data and temp directories

setwd(wd)

if (!file.exists(data\_dir)) dir.create(data\_dir,recursive=T)

temp\_dir=paste0(wd,"temp/") ## Tepm directory to write temp files

if (!file.exists(temp\_dir)) dir.create(temp\_dir,recursive=T)

### Selecting necessary lines from discovery GWAS file

Sys.setlocale('LC\_ALL','C'); ltr="\""; lsr="'"; uuu="$"; uu1="-"

### Depending on Linux version you might need to replace "gawk" below with "awk". ### Check your system's gawk/awk commands

### Keep only palindromic SNPs

comfaz=paste0(uuu,snp\_column\_no,",",uuu,ref\_allele\_column\_no,",",

uuu,alt\_allele\_column\_no,",",uuu,beta\_column\_no,",",uuu,p\_value\_column\_no)

a1a2=paste0(uuu,ref\_allele\_column\_no,uuu,alt\_allele\_column\_no) ## This is A1A2 column ($ref$alt)

command1=paste0("gawk ",uu1,"F ",ltr,gwas\_delimiter,ltr,

" '",uuu,p\_value\_column\_no,">0 && (",

a1a2,"==",ltr,"AC",ltr," || ",a1a2,"==",ltr,"AG",ltr," || ",

a1a2,"==",ltr,"CA",ltr," || ",a1a2,"==",ltr,"CT",ltr," || ",

a1a2,"==",ltr,"GA",ltr," || ",a1a2,"==",ltr,"GT",ltr," || ",

a1a2,"==",ltr,"TC",ltr," || ",a1a2,"==",ltr,"TG",ltr," || ",

a1a2,"==",ltr,"ac",ltr," || ",a1a2,"==",ltr,"ag",ltr," || ",

a1a2,"==",ltr,"ca",ltr," || ",a1a2,"==",ltr,"ct",ltr," || ",

a1a2,"==",ltr,"ga",ltr," || ",a1a2,"==",ltr,"gt",ltr," || ",

a1a2,"==",ltr,"tc",ltr," || ",a1a2,"==",ltr,"tg",ltr,

") {print ",comfaz,"}' ", data\_dir, discovery\_gwas," > ",

temp\_dir, "temp1.txt")

system(command1)

### Replace small allele letters with big letters

command2=paste0("sed -i -e 's/g/G/g; s/c/C/g; s/t/T/g; s/a/A/g' ",

temp\_dir, "temp1.txt")

system(command2)

### Read non-palindromic file with reduced columns and assign names to columns

merged0 = read.table(paste0(temp\_dir, "temp1.txt"), header=F)

names(merged0) = c("SNP", "A1", "A2", ifelse(or\_is\_used,"OR","BETA"), "P")

### Remove duplicate snps

merged1=merged0[!duplicated(merged0[,"SNP"]),]

### Now select only common snps b/w discovery and target

command3=paste0("awk '{print $2}' ",data\_dir, target\_gwas, ".bim > ",

temp\_dir, "tempbim.txt") ## Copy rs column of bim file

system(command3)

inputbim=read.table(paste0(temp\_dir, "tempbim.txt"), header=F) ## Read rs column

names(inputbim)[1]="SNP" ## Assign a name

### Merge and keep only common SNP names

merged=merge(merged1, inputbim, by="SNP", all.x=F, all.y=F)

### Create score file

write.table(merged, file=paste0(temp\_dir,"score\_file.txt"), row.names=F,

col.names=T, quote=F)

## Writing common SNPs

write.table(merged[,"SNP"],file=paste0(temp\_dir, "common.txt"), row.names=F,

col.names=F, quote=F)

### Using plink to extract subset of common discovery and target SNPs

command4=paste0(plink\_exe," --bfile ", data\_dir, target\_gwas,

" --allow-extra-chr --keep-allele-order --allow-no-sex --extract ",

temp\_dir, "common.txt --make-bed --out ", temp\_dir, "newtarget")

system(command4)

### Resolving strand issue between discovery and target samples

### (Plink’s --flip option)

### Read new/common target genotype file SNP, A1 and A2 columns

inputbim2=read.table(paste0(temp\_dir, "newtarget.bim"), header=F)

## Assign names A to A1 and B to A2

names(inputbim2)=c("cc", "SNP", "p1", "p2", "A", "B")

tempmg=merge(merged[,c("SNP", "A1", "A2")], inputbim2[,c("SNP", "A", "B")],

by="SNP", all.x=F, all.y=F)

tempmerge2=tempmg[as.character(tempmg$A1) !=

as.character(tempmg$A) && as.character(tempmg$A1) != as.character(tempmg$B),]

targetname=""

if (dim(tempmerge2)[1]>0) ## Check if there are flipped SNPs and print the list

{ write.table(tempmerge2[,c(sn2)], file=paste0(temp\_dir, "flip.txt"),

row.names=F, col.names=F, quote=F)

command5=paste0(plink\_exe, " --bfile ",

temp\_dir, "newtarget --allow-extra-chr --allow-no-sex --flip ",

temp\_dir, "flip.txt --make-bed --out ",temp\_dir,"newtargett")

targetname="t" ##for target bed file name

system(command5)

}

### Creating python PRS-CS command

prs\_cs\_command=paste0("python ",prs\_cs\_dir,"PRScs-master/PRScs.py ",

" --ref\_dir=", prs\_cs\_dir, "ldblk\_1kg\_", ifelse(ancestry=="AFR", "afr",

ifelse(ancestry=="EAS", "eas", "eur")),

" --bim\_prefix=", temp\_dir, "newtarget", targetname,

" --sst\_file=", temp\_dir, "score\_file.txt ",

" --n\_gwas=", n\_discovery\_gwas, ifelse(phi>0, paste0(" --phi=",phi," "), " "),

" --out\_dir=", temp\_dir, "prs\_cs\_out --seed=123459921")

### --ref\_dir=path\_to\_ref --bim\_prefix=path\_to\_bim --sst\_file=path\_to\_sumstats

system(prs\_cs\_command) ## Runs PRS-CS and creates new weights

### Results of PRS-CS are splitted by chromosome, combine into one file

command6=paste0("cat ",temp\_dir,"prs\_cs\_out\*phiauto\_chr\*.txt >",

temp\_dir,"prs\_estimates.txt")

system(command6)

### Using Plink with new weights to create final PRS-CS score file

plink\_command=paste0(plink\_exe," --bfile ", temp\_dir, "newtarget", targetname,

" --allow-no-sex --out ", wd, "New\_PRS-CS\_score --score ", temp\_dir,

"prs\_estimates.txt 2 4 6 no-sum")

system(plink\_command)

### Scores will be written to [working directory]/New\_PRS-CS\_score.profile

References:

Chang CC, Chow CC, Tellier LC, Vattikuti S, Purcell SM, Lee JJ. Second-generation PLINK: rising to the challenge of larger and richer datasets. *Gigascience*. 2015; s13742–015–0047–8<https://doi.org/10.1186/s13742-015-0047-8>.

Ge, T., Chen, C. Y., Ni, Y., Feng, Y. C. A., & Smoller, J. W. (2019). Polygenic prediction via Bayesian regression and continuous shrinkage priors. *Nature Communications*, *10*(1), 1-10.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., ... & Sham, P. C. (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. *The American Journal of Human Genetics*, *81*(3), 559-575.