# Benchmarking the Accuracy of Polygenic Risk Scores and their Generative Methods: Supplementary II

**Scott Kulm**[1,2,3], **Jason Mezey**[4,5,*], **and Olivier Elemento**[2,3,*]

[1]Department of Physiology and Biophysics, Weill Cornell Medicine, New York, NY

[2]Caryl and Israel Englander Institute of Precision Medicine, Weill Cornell Medicine, New York, NY

[3]Physiology, Biophysics and Systems Biology Graduate Program, Weill Cornell Medicine, New York, NY

[4]Department of Genetic Medicine, Weill Cornell Medicine, New York, NY

[5]Department of Computational Biology, Cornell University, Ithaca, NY

[*]corresponding authors

This supplementary index aims to provide a more detailed reporting of the methods implemented. Additional information can be found directly within the code, which has been uploaded to GitHub: https://github.com/kulmsc.

## Acquisition of Summary Statistics

All summary statistics were acquired from the GWAS Catalog (https://www.ebi.ac.uk/gwas/downloads/summary-statistics). All studies were sought that had relatively high sample size, studied relatively prevalent, binary, disease traits, and did not use UK Biobank data. In total 26 traits were chosen. Twenty-five of these traits are described in Supplementary Table 2, the other trait was for Tourette's Syndrome, which did not have a large enough sample size of cases within the UK Biobank to merit analysis. All summary statistics were downloaded directly from the FTP server.

A conversion script was deployed to regularize all of the various summary statistics. The first step of the conversion script is to determine various statistics that might not have been included. If the allele frequency was not provided, then allele frequency in the UK Biobank was assumed. If the standard error of the estimate was not included, then it was approximated from the p-value. If the sample size in each variant's test was not provided it was assumed to be the sample size for the larger publication. If the minimum effect was larger than zero, the reported effect was assumed to be an odds ratio value and therefore logged to create a beta value.These were all rough approximations, but they were rarely used in any downstream analysis. Next, the data was munged following the script that accompanies the Linkage Disequilibrium Scoring Regression tool set. An explanation of the QC steps taken by the munging script is described within its own documentation (https://github.com/bulik/ldsc). The output variants in the munged summary statistics were then used to subset the original summary statistics. The subsequent steps were performed on each chromosome individually. The summary statistics were subset to those within the imputed UK Biobank dataset. Then ambiguous variants were removed, where the effect and non-effect allele were A/T, T/A, C/G, or G/C. Next, the bases were flipped to match the identity of the UK Biobank reference. For example, if the effect allele was G, but the reference alleles were C and T, then the G would be flipped to C, its complementary base. Lastly, the UK Biobank allele frequency and imputed INFO score was included as columns on the summary statistics. A converted and standardized summary statistics file was then written. The first ten columns of these files match the format of the ".assoc.dosage" file generated from the PLINK software.

## Preparation of UK Biobank Genetic Data

The UK Biobank imputed data was employed to create small PLINK friendly files that could undergo scoring. First the sample QC information was used to subset individuals that were reported to not be heterozygosity outliers, not holding sex chromosome, aneuploidy, had matching genetic and reported sex, and were reported to be white, British individuals. These remaining individuals were then split into a training phase and testing phase. The polygenic risk scoring methods were solely applied to the training phase.

The remaining genetic data preparation was uniquely applied for each combination of polygenic risk scoring method, the

specified parameters, and the summary statistics. For each of these combination, the supplied imputed, .bgen files from the UK Biobank was freshly copied into a working environment. The variants in the summary statistic were further subset by selecting variants with an INFO score above 0.3, effect and non-effect alleles were both length one, p-value was greater than 0, and variant ID (rsID) did not repeat within the summary statistics. The subset variant IDs were used to subset the original .bgen file into a smaller one, then PLINK2 was used to extract the training phase individuals, finally PLINK 1.9 was used to only select variants whose allele frequency in the previous file was above 0.01. This series of programs was found to be the fastest. The final output genetic file was in PLINK format (bed, bim, fam). The summary statistic file was then subset to the variants within this final genetic file.

## Basic

The basic method involves including all of the variants of the original summary statistic file in the scoring process. This scoring process uses PLINK, with the flag "—score". The "sum" option was included such that the sub-scores across chromosomes could be added together.

## Clumping

Implementation of the clumping method was applied through the PLINK software. The "—clump" flag with a series of "—clump-p1" p-value and "—clump-p2" R2 thresholds. The variants in the output file were used to subset the primary summary statistics.

Official Documentation: https://www.cog-genomics.org/plink/1.9/postproc

## LDpred

To implement the LDpred method the starting summary statistic file was first converted into the STANDARD format required by LDpred (columns of chromosome, position, reference allele, alternative allele, reference allele frequency, info, rsID, p-value, and effect of the alterative allele). Generating this file simply required reorganizing the starting summary statistic file. Next, meta-data was determined from a reference file, specifically the total number of SNPs across all chromosomes and the sample size was recorded. With this data the LD range was calculated as the number of SNPs divided by 4500. The "ldpred coord" step was first run, followed by a series of "ldpred gibbs" applications with various "f" values (the proportion of true causal variants). The adjusted effects produced by each application was substituted into the original summary statistic file. Throughout, version 1.0.6 was employed. Other starting summary statistic formats were attempted, although they all produced odd results and were therefore excluded. In addition, not sorting the output file from the gibbs step to match the order of the original summary statistics produced unusually good predictions. The results were even randomly shuffled to prove the strong prediction of the mismatched variant file. This work could not determine why this is.

Official Documentation: https://github.com/bvilhjal/ldpred

## WC-lasso

To implement WC-lasso (winners curse lasso), the original summary statistic file was first clumped with a p-value cut-off of 0.01 and R2 cut-off of 0.1. The vector of remaining effect sizes was then modified according to the equation:

$$\hat{\beta}_m^{lasso} = sign(\hat{\beta}_m)\left|\hat{\beta}_m\right| - \lambda I\left(\left|\hat{\beta}_m\right| > \lambda\right)$$

The lambda value ranged from 0.001 to 0.1. The modified summary statistics were used for scoring.

## WC-likelihood

The WC-likelihood (winners curse likelihood) method was implemented similar to WC-lasso, with the same initial clumping step. The following effect adjustment step however required minimizing a likelihood function which included cumulative normal distribution functions. The exact likelihood function can be found within the "winnersCurse.py" script on GitHub. The Python function "minimize" was used along with the "nelder-mead" method. The adjusted effects were again substituted into the original summary statistics and used for scoring.

## GraBLD

Implementation of the GraBLD method required access to phenotypic data. A maximum of 2,000 cases and 2,000 controls were randomly subset into a separate file. This genotypic data was fully normalized following a function provided within the GraBLD package, then used to conduct a GWAS. The resulting effects were then employed in the key GraB function alongside "annotation" data, which was just the original summary statistic file. The GraB function also required various random forest algorithm related parameters. The significance and depth parameters were altered, it was found that altering the other values could cause errors. The effects that were output from the GraB function were further adjusted by dividing by the output of the LDadj function. These final altered values were then substituted into the original summary statistics, removing any variants that no longer existed, and used for scoring. This process often times hit computational errors, possibly owing to the complexity of the random forest generated or the large amount of data processed that was all from a single source.

Official Documentation: https://github.com/GMELab/GraBLD

## lassosum

Implementation of the lassosum method began preparing phenotypic data, following the same steps described in the GraBLD method. Additional reference data was also prepared, extracted from the testing phase of the UK Biobank and limited to the variants under analysis. The primary lassosum function was then called with three different sets of parameters specifying different datasets being tested or used as reference. The first employed the training data as the refrence data, the second

had the training data as both the reference and testing data, and the third used the training data as the testing data and the additional reference data derived from the testing phase as the reference data. In each instance the EUR.hg19 file specified in the documentation was used to define the LD blocks. The lassosum pipeline was then called three times, with the options: pseudovalidate, validate and split validate. In order to enable the extraction of betas from this pipeline, the released code for lassosum was modified and is currently available on GitHub (https://github.com/kulmsc/modLassosum). The original summary statistics were subset to the variants included in the respective output file, the new betas values were substituted into the summary statistics and this updated summary statistic file was used for scoring.

Official Documentation: https://github.com/tshmak/lassosum

## PRScs

Implementation of the PRScs method began by reducing the number of columns in the original summary statistic file to a reduced file specified within the PRScs documentation. No other additional steps were necessary, as the next step was the main PRScs function call. The reference data employed was the European LD data listed within the PRScs documentation. The phi parameter was changed over three different function calls, whereas the a and b parameter were held steady. The beta values from the output were then substituted into the summary statistics, and this new file was used for scoring.

Official Documentation: https://github.com/getian107/PRScs

## Tweedie

Implementation of the Tweedie method began by an application of the clumping method, following the steps as described in the respective section, with the p-value threshold set at 0.05 and R2 at 0.25. The modified summary statistic file was then used within the main tweedy R script. In order to extract the beta values the published script was modified slightly, and is located at https://github.com/kulmsc/PRS-Ithaca/blob/master/tweedy.R. Three variations of the betas were created, one for each of the FDR, Tweedie, and FDR x Tweedie sub-methods. These betas values were subset into the original summary statistics, and this new file was used for scoring.

Official Documentation: https://sites.google.com/site/honcheongso/software/empirical-bayes-risk-prediction

## WC-2d

Implementation of the WC-2d (winners curse two dimensions) method roughly followed the steps outlined in the clumping method, except for specification of which regions clumping applies to. Specifically, variants identified as being conserved (listed within http://compbio.mit.edu/human-constraint/data/gff/) and pleiotropic (listed within supplementary table 2 of the respective publication) were both clumped with a higher p-value threshold. Only one of these varieties of variants were considered as important at a time. In total, 18 different parameter combinations were used with this method, as the p-value and R2 values for both the special and remaining regions were altered independently. The retained variants in these special regions, and the

retained variants within the remaining part of the genome were combined and used to subset the summary statistic. These files were finally used for scoring.

## AnnoPred

Implementation of the AnnoPred method began by preparing phenotypic data for the primary training data. Unlike the lassosum or GraBLD methods, all available phenotyping data was insert directly into the PLINK fam file. Next, the summary statistic file was reduced to match the columns, and the respective labels, specified by the AnnoPred documentation. The summary statistics were then split into groups of 2,500 variants, with each subset file independently used within the AnnoPred function call. The data was split because an analysis of any larger amount of data caused the AnnoPred function to frequently crash or error. Within each AnnoPred function call, the reference and validation data were set to the total phenotype-set file, and the annotation flag was set to tier 3. No user heritability was set. The original summary statistics were subset to those in each of the four output files created by the combination of first and second priors, and infinitesimal and non-infinitesimal genetic architecture assumptions. The output betas for each of these combinations were substituted into their respective summary statistics and used for scoring.

Official Documentation: https://github.com/yiminghu/AnnoPred

## LDpred-funct

Implementation of the LDpred-funct method required extensive use of external references and programs. First, the LD Score regression was applied with both the baseline_v1.1 and the baselineLD_v2.2 sets of reference data. The frequency file was set to 1000G.EUR.QC and the weights were set without HLA. Additional instructions can be found directly within the S-LDSC documentation: https://github.com/bulik/ldsc/wiki/Partitioned-Heritability. Continuing to follow the initial heritability preparation, the two calculations of partitioned heritability were then multiplied by two possible annotation files (both the phase3 EAS baselineLD v2.1 and 2.2). The final generated matrix contains rows of SNPs and columns of annotation types. A fuller description of these initial steps are provided in the LDpred-funct documentation. The remaining pieces needed for the main LDpred-funct function call are the phenotype file, created by the same script used within AnnoPred, and a summary statistic file adjusted to the LDpred-funct specifications. The parameters within this function call are the total heritability, also determined by normal LDSC, the sample size of the original GWAS, and the LD radius, which was held at 1000. The total heritability was a byproduct of the partitioned heritability calculations, included within the log file. The output from this function call were either the infinitesimal genetic architecture betas, which did not require any adjustment, and the non-infinitesimal betas, which were further adjusted according to which effect size bin the beta was in. Each of these sets of betas were substituted into their own summary statistic file and finally used for scoring. As perhaps the most complex polygenic risk score process, the actual code for running this analysis may serve as a more useful reference.

Official Documentation: https://github.com/carlaml/LDpred-funct

## sBLUP

Implementation of the sBLUP method required several pieces of meta-information, including: the total sample size, total number of SNPs, and the estimated heritability. This estimate was taken from the byproducts of the LDpred-funct method, specifically the LDSC version which included the baseline_v1.1 annotation. It should be noted that in all methods that required heritability, if the estimation was less than 0 it was changed to 0.01. Next, the original summary statistic file was down sized to only the variants included within the HapMap (https://www.broadinstitute.org/medical-and-population-genetics/hapmap-3), and the columns were re-arranged to the MA format used throughout the GCTA tool kit. The actual sblup option was then run within gcta, with the wind option (the LD distance parameter) set to 100. The output variants were then used to subset the original summary statistics, and the updated betas were substituted within. This new summary statistic file was used for scoring.

Official Documentation: https://cnsgenomics.com/software/gcta/SBLUP

## SBayesR

Implementation of the SBayesR method required minimal preparation, as the only step taken was reorganizing the summary statistics into the necessary MA format. The primary SBayesR computations were then run within the gctb toolkit. The ldm file was created from a subset of the UKBB training data and reduced to a spare format. Creation of these files took most of the time required to run this method, and were relatively large in terms of bytes. The output variants were then used to subset the original summary statistics, and the updated betas were substituted within. This new summary statistic file was used for scoring.

Official Documentation: https://cnsgenomics.com/software/gctb/SummaryBayesianAlphabet

## stackCT

Implementation of the stackCT method was almost entirely carried out within an R environment. The genomic information, which is typically kept within a PLINK format, was first read into R by using the snp_attach function of the bigsnpr package. Next, the summary statistics were read in and adjusted to the correct format. The genomic information and the summary statistics were then adjusted so that the columns of the genome file corresponded to the rows of the summary statistics. Over an array of possible training fractions, the snp_grid_clumping, snp_grid_PRS, and snp_grid_stacking functions were called in that order. The output variants and their respective effects were reorganized into the format of the original summary statistics and were used for scoring.

Official Documentation: https://privefl.github.io/bigsnpr/