Detailed information for the scripts in the *GCPBayes pipeline*

This part includes all concepts and information for each script used in the *GCPBayes pipeline* in order to provide more detailed information for developers who want to modify/add/remove of the pipeline.

Table of Contents

IMPORTANT NOTE 1: we recommend to check our GitHub webpage for any updates in the *GCPBayes pipeline* scripts including adding new strategies and functions:

https://github.com/CESP-ExpHer/GCPBayes-Pipeline

IMPORTANT NOTE 2: All scripts are available through our GitHub webpage:

https://github.com/CESP-ExpHer/GCPBayes-Pipeline/tree/main/0_Codes

IMPORTANT NOTE 3: All files used in the *GCPBayes pipeline* are available through our GitHub webpage:

https://github.com/CESP-ExpHer/GCPBayes-Pipeline/tree/main/0_Files

1. Standardization (Section A) (Python)

Consider there are two GWAS summary statistics files, two scripts are needed to be run for a standardization:

1.1. First GWAS Summary Statistics Data

For the first trait (we call it **reference**), a user needs to modify and run the following script:

(**Required program:** Python)
(**Script:** A1_code_reformatting_file_bcac_2020_all.py)

Note 1: A user should change the "**DEFINITION SECTION**" of the script based on the file name and path of the input file and the output path.

Note 2: The script is written based on the "BCAC_v.2020" GWAS summary statistics file (Zhang *et al.*, 2020). So, a user must just change the columns indices (and NOT their names in the script) based on their corresponding columns. Here are the defined columns in the script:

- ID (starts with rs... or chr:pos:Baseline_A:Effect_A) (Baseline_A=non-Effect Allele, Effect_A=Effect Allele)
- CHR (chromosome **as an integer**) (i.e. without "chr", "X", "Y", or any other characters)
- POS (base pair position)
- A1 (Effect Allele)
- A2 (non-Effect Allele)
- eaf (Effect Allele Frequency)
- info (info or r2)
- beta (beta value)
- se (standard error value)
- p (P_value)

In summary, two main steps would be performed using the script:

Filtering and calculation step:

- Removing entries with "NULL" or "NA" values for eaf, info, beta, se, p columns
- Removing entries if both beta and se values are zero
- Removing entries with alleles that are not composed of ATGC characters
- Removing ambiguous SNPs
- Removing duplicated entries
- Calculation of the Minor Allele Frequency

Saving files step:

- Creation of some files contain information about the filtering step (with "_SNP_ambiguous.txt", "_SNP_dupicated.txt", "_SNP_duplication_set.txt", "_SNP_weird_alleles.txt ", and "_Summary.txt" at the end of the files)
- Creation of a standardized file (with *"_reformatted.txt"* at the end of the file)

For running the script, use the following command in the terminal:

```
$ python A1_code_reformatting_file_bcac_2020_all.py
```

The main output would be "BCAC_2020_onco_ALL_reformatted.txt" file. This output file would be used as an input for the next part (1.2) and also as an input for B2 and D1 sections.

1.2. Second GWAS Summary Statistics Data

For the second trait, a standardization will be done based on the reference file (which created by the previous step). A user needs to modify and run the following script:

(**Required program:** Python)
(**Script:** A2_code_reformatting_file_ocac_bcac_2020_all.py)

Note 1: A user should change the "**DEFINITION SECTION**" of the script based on the files names and paths of the input files (reference and the second GWAS files) and the output path.

Note 2: The script is written based on the "OCAC" GWAS summary statistics file (Phelan *et al.*, 2017). So, a user must just change the columns indices (and NOT their names in the script) based on their corresponding columns. Here are the defined columns in the script:

- CHR (chromosome **as an integer**) (i.e. without "chr", "X", "Y", or any other characters)
- POS (base pair position)
- A1 (Effect Allele)
- A2 (non-Effect Allele)
- eaf (Effect Allele Frequency)
- neaf (non-Effect Allele Frequency)
- beta (beta value)
- se (standard error value)
- p (P_value)
- info (info or r2)
- N (number of samples)

In summary, three main steps would be performed using the script:

Filtering and calculation step:

- Removing entries with "NULL" or "NA" values for eaf, neaf, beta, se, p columns
- Removing entries if both beta and se values are zero
- Removing entries with alleles that are not composed of ATGC characters
- Removing ambiguous SNPs
- Removing duplicated entries
- Calculation of the Minor Allele Frequency
- Removing entries that do not exist in the reference file

Harmonization step of the alleles between two data sets (the reference and this data):

- If the alleles were switched, they will be corrected and the sign of beta value would be changed for the second data set

Saving files step:

- Creation of some files contain information about the filtering step (with "_SNP_removed_from_ref.txt , "_SNP_removed_from_file.txt", "_SNP_ambiguous.txt", "_SNP_dupicated.txt", "_SNP_duplication_set.txt", "_SNP_weird_alleles.txt ", and "_Summary.txt" at the end of the files)
- Creation of a standardized file for the second trait (with *"_reformatted.txt"* at the end of the file)

For running the script, use the following command in the terminal:

```
$ python A2_code_reformatting_file_ocac_bcac_2020_all.py
```

The main output would be "OCAC_BCAC_2020_onco_ALL_reformatted.txt" file. This output file could be also used as an input for B2 and D1 sections.

2. Annotation (Section B) (R + PLINK)

Based on Fig. 1 in the manuscript (overview of the pipeline), two annotation files are needed (one for Section D5 and the other for Section D2).

2.1. First Step (Section B1) (R)

The annotation file which is needed for Section D5, is a gene annotation for Human organism available through different public databases. We have explained the whole process (from downloading to cleaning and analyzing an annotation file) in our GitHub page:

https://github.com/CESP-ExpHer/Gene_Annotation/tree/main/1_hg19

We used the annotation file downloaded from GENCODE webpage (Frankish *et al.*, 2019).

We have done the cleaning procedures (explained in the GitHub page) and just used protein-coding genes for this study. The final file is available in our GitHub path:

https://github.com/CESP-ExpHer/GCPBayes-Pipeline/tree/main/0_Files

This output file would be used as an input for Section D5.

2.2. Second Step (Section B2) (R + PLINK)

The annotation file which is used for D2, is actually a GWAS summary statistics data (for one of the traits) with a gene column added based on the annotation file created from the above section. Here are the steps to create the final file:

Extraction of "protein_coding" genes from the downloaded annotation file
(**Required program:** R)

```
> library(dplyr)
> annot_all <- read.table(file = "annot_gencode_v38lift37_modified_gene_class.txt", header = TRUE)
> coding <- filter(annot_all, annot_all$gene_type == "protein_coding")
> write.table(coding, file = "annot_gencode_v38lift37_modified_gene_class_coding.txt",
      quote = FALSE, col.names = TRUE, sep = "\t")
```

Changing all chromosomes characters to numbers
(**Required program:** R)

```
> x <- substring(coding[, 1], 4, nchar(coding[, 1]))
> coding$chr <- x
> coding$chr <- gsub("X", "23", coding$chr)
> coding$chr <- gsub("x", "23", coding$chr)
> coding$chr <- gsub("Y", "24", coding$chr)
> coding$chr <- gsub("y", "24", coding$chr)
> coding$chr <- gsub("M", "25", coding$chr)
> coding$chr <- gsub("m", "25", coding$chr)
> coding$chr <- as.numeric(coding$chr)
> Annot_plink <- coding[, c("chr", "start", "end", "gene_name")]
> write.table(Annot_plink,
file = "annot_gencode_v38lift37_modified_gene_class_coding_chr_num_plink_input.txt", sep = "\t", row.names = FALSE, col.names = FALSE, quote = FALSE)
```

Creation of GWAS input file for PLINK based on BCAC_2020 GWAS reformatted file (the file created after the standardization step)
(**Required program:** R)

```
> library(vroom)
> input_path <- "~/BCAC_OCAC/"
> gwas <- vroom(file=paste0(input_path, "BCAC_2020_onco_ALL_reformatted.txt"))
> gwas <- as.data.frame(gwas)
> gwas <- gwas[, c(2, 1, 3, 8)]
> colnames(gwas) <- c("CHR", "SNP", "BP", "P")
> output_path <- "~/BCAC_OCAC/"
> vroom_write(gwas, file=paste0(output_path, "BCAC_2020_onco_ALL_reformatted.assoc"))
```

Annotation GWAS file with PLINK
(**Required program:** PLINK)

The following command needs to be run for this step. Note that the input GWAS and annotation files should be available in the working directory.

```
$ plink --annotate BCAC_2020_onco_ALL_reformatted.assoc attrib=snp129.attrib.gz ranges=annot_gencode_v38lift37_modified_gene_class_coding_chr_num_plink_input.txt
```

The output file would be "plink.annot".

Reformatting the PLINK Annotation output file (plink.annot)
(**Required program:** R)

```
> library(vroom)
> library(splitstackshape)
> input_path <- "~/BCAC_OCAC/"
> gwas <- vroom(file=paste0(input_path, "plink.annot"))
> gwas <- as.data.frame(gwas)
> colnames(gwas) <- c("CHR", "SNP", "BP", "P_ANNOT")
> gwas <- cSplit(gwas, "P_ANNOT", " ")
> colnames(gwas) <- c("CHR","SNP","BP","P", "ANNOT")
> output_path <- "~/BCAC_OCAC/"
> vroom_write(gwas, file=paste0(input_path, "Annot_BCAC_2020_onco_ALL_reformatted_coding.txt"))
```

This output file would be used as an input for Section D2.

In summary, after Section B, two following files are created which would be used as inputs for Sections D5 and D2:

annot_gencode_v38lift37_modified_gene_class_coding.txt

Annot_BCAC_2020_onco_ALL_reformatted_coding.txt

3. LD Clumping (Section C) (R)

For creation of an LD file, we have used LD clumping method available through the "ieugwasr" R package (*https://github.com/MRCIEU/ieugwasr*). Nevertheless, instead of using OpenGWAS API, we used local calculations which prevent getting error while working with a large set of data. LD clumping function needs two columns (SNP RS IDs and P-values). Since we are working on two traits, it is recommended to use P-values obtained from a SNP level association method. Here we used PLACO (Ray and Chatterjee, 2020) for obtaining the P-values. Another advantage is that by using PLACO a user could also have a SNP level-based pleiotropy analysis as well as gene level-based pleiotropy results derived from the GCPBayes package.

All three R scripts for this part are available in the following GitHub path:

https://github.com/CESP-ExpHer/GCPBayes-Pipeline/tree/main/0_Codes

For LD Clumping calculation, three steps need to be run:

3.1. First Step (finding shared SNPs between two traits)

Finding shared SNPs between two traits and calculation of *Z* column (required for PLACO analysis)

(**Required program:** R)
(**Script:** C1_code_find_shared_snps_one_pair.R)

Note 1: A user should change the "**DEFINITION SECTION**" of the script based on the file names and paths of the input files and the file names and path of the output path. There are also two threshold values (for r^2 and MAF) for filtering SNPs with values less than these values which a user could modify in this section.

Note 2: The script is written based on the BCAC and OCAC reformatted GWAS summary statistics data (derived after Section A). So, a user must just change the columns indices (and NOT their names in the script) based on their corresponding columns. Here are the defined columns in the script:

For BCAC and OCAC reformatted GWAS data:

- snp (RS ID)
- chr (chromosome)
- bp_hg19 (base pair position)
- Effect_A (Effect Allele)
- nonEffect_A (non-Effect Allele)
- beta (beta value)
- se (standard error)
- pval (P-value)
- info (r^2)
- EAF (Effect Allele Frequency)
- MAF (Minor Allele Frequency)

Note 3: Since running PLACO needs a *Z* column and this column is not available in the BCAC and OCAC reformatted files, in this script *Z* value would also be calculated based on beta and standard error values.

The output would be two files:

"BCAC_2020_ALL_Shared_OCAC_inc_Z.txt"

"OCAC_Shared_BCAC_2020_ALL_inc_Z.txt"

3.2. Second Step (running PLACO)

(**Required program:** R)
(**Script:** C2_code_run_PLACO_decor_one_pair.R)

Note 1: A user should change the "**DEFINITION SECTION**" of the script based on the file names and path of the input files and the file name and path of the output path. There is also a threshold value (pval_threshold) used for decorrelating the Z-scores based on the recommendation mentioned through the PLACO tutorial (Ray and Chatterjee, 2020).

Note 2: The script reads the output from the First step. So, a user does NOT need to change anything in terms of column names.

The output would be "output_PLACO_BCAC_2020_ALL_OCAC.txt" file.

3.3. Third Step (running local LD Clumping)

(**Required program:** R)
(**Script:** C3_code_ldclumping_local.R)

For more information about how this step work, a user could see the following webpage:

https://mrcieu.github.io/ieugwasr/articles/local_ld.html

Note 1: A user should change the "**DEFINITION SECTION**" of the script based on the file name and path of the input files and the file name and path of the output path.

Note 2: The script reads the output from the Second step. So, a user does NOT need to change anything in terms of column names.

Note 3: The "ld_clump" function needs some parameters to be defined:

- clump_kb (Clumping kb window) (Default is very strict, 10000)
- clump_r2 (Clumping r^2 threshold) (Default is very strict, 0.001) (the bigger r2, the more SNPs remain in the final file)
- clump_p (Clumping significant level for index variants) (Default = 1 i.e. no threshold)
- pop = "EUR" (Super-population to use as reference panel) (Default = "EUR")
- plink_bin = genetics.binaRies::get_plink_binary() (specify path to plink binary)
- bfile = "PATH" (path where super-population data (in .bed/.bim/.fam file formats) exists)

Note 4: A user should install "genetics.binaRies" package in R in order to "plink_bin" option works.

Note 5: A user should download the super-population data (in .bed/.bim/.fam file formats) from the following address. The super-population data are from the 1000 Genome Project (Auton *et al.*, 2015). Then extract it and put its path in front of the "bfile" option:

http://fileserve.mrcieu.ac.uk/ld/1kg.v3.tgz

The output file would be "output_ld_clumping_08_BCAC_2020_ALL_OCAC.txt" file.

4. Running Core (Section D)

4.1. Without LD Clumping STEP

(**Required program:** R)
(**Script:** D1_code_pipeline_annot_coding_withoutldclumping_extra_info.R)

According to Fig. 1 in the manuscript (overview of the *GCPBayes pipeline*), D1, D2, D3, and D5 steps would be run. The D4 part does NOT run because there is no LD clumping step considered in this case.

For running, four input files are needed:

1. BCAC_2020_onco_ALL_reformatted.txt (See 1.1)
2. OCAC_BCAC_2020_onco_ALL_reformatted.txt (See 1.2)
3. annot_gencode_v38lift37_modified_gene_class_coding.txt (See 2.1)
4. Annot_BCAC_2020_onco_ALL_reformatted_coding.txt (See 2.2)

Note: A user should change the "**DEFINITION SECTION**" of the script based on file names and paths of input and output files. In addition, there are also two threshold values (for r^2 and *MAF*) for filtering SNPs with values less than these values which a user could modify in this section.

RUNNING SECTION: As it is mentioned in Fig. 1 of the manuscript, there are four steps during running this script. Here is a summary for each step:

1. **D1:** Keeping shared SNPs between two traits (BCAC_2020_onco_ALL_reformatted.txt and OCAC_BCAC_2020_onco_ALL_reformatted.txt), then sorting rows based on SNP IDs.

   Note: If a user created the reformatted GWAS files using our scripts (See Section 1), there is no need to rename columns. If NOT, a user should rename columns of input GWAS data (for both traits) considering the following column names:

   - snp (RS ID)
   - chr (chromosome)
   - bp_hg19 (base pair position)
   - Effect_A (Effect Allele)
   - nonEffect_A (non-Effect Allele)
   - beta (beta value)
   - se (standard error)
   - pval (P-value)
   - info (r^2)
   - MAF (Minor Allele Frequency)

2. **D2:** Comparing one of the GWAS summary statistics files (created in D1) with the annotation file (Annot_BCAC_2020_onco_ALL_reformatted_coding.txt) and adding the gene column annotated to each SNP.

3. **D3:** Merging the Information from the two GWAS summary statistics files (created in D1) and the file created at the end of D2.

4. **D5:** Reading two GWAS files (created in D3) and splitting rows in which more than one gene assigned to a SNP.
   Then, removing other non-gene information (such as missense, nonsense, frameshift, and splice) from the gene column.
   After that, the scripts adds some extra information about each gene (such as chromosome, start base-pair position, end base-pair position, gene length, and gene class) obtained from the Human gene annotation file (annot_gencode_v38lift37_modified_gene_class_coding.txt).
   Finally, it creates two files (begins with Matrices_... and Matrices_extra_info_... and are in "*.Rdata*" formats) which could be easily read by the GCPBayes package for exploring pleiotropy between GWAS data of two traits at a gene-level.

4.2. With LD Clumping STEP

(**Required program:** R)
(**Script:** D2_code_pipeline_annot_coding_ldclumping_extra_info.R)

According to Fig. 1 in the manuscript (overview of the *GCPBayes pipeline*), D1 to D5 steps would be run.

For running the pipeline including LD clumping STEP, five input files are needed:

1. BCAC_2020_onco_ALL_reformatted.txt (See 1.1)
2. OCAC_BCAC_2020_onco_ALL_reformatted.txt (See 1.2)
3. annot_gencode_v38lift37_modified_gene_class_coding.txt (See 2.1)
4. Annot_BCAC_2020_onco_ALL_reformatted_coding.txt (See 2.2)
5. output_ld_clumping_08_BCAC_2020_ALL_OCAC.txt (See 3)

Note: A user should change the "**DEFINITION SECTION**" of the script based on file names and paths of input and output files. In addition, there are also two threshold values (for r^2 and *MAF*) for filtering SNPs with values less than these values which a user could modify in this section.

RUNNING SECTION: As it is mentioned in Fig. 1 of the manuscript, this script ONLY has ONE more STEP than the previous script. So, it includes five steps and here is a summary for each step:

1. **D1:** Keeping shared SNPs between two traits (BCAC_2020_onco_ALL_reformatted.txt and OCAC_BCAC_2020_onco_ALL_reformatted.txt), then sorting rows based on SNP IDs.

   Note: If a user created the reformatted GWAS files using our scripts (See Section 1), there is no need to rename columns. If NOT, a user should rename columns of input GWAS data (for both traits) considering the following column names:

   - snp (RS ID)
   - chr (chromosome)
   - bp_hg19 (base pair position)
   - Effect_A (Effect Allele)
   - nonEffect_A (non-Effect Allele)
   - beta (beta value)
   - se (standard error)
   - pval (P-value)
   - info (r^2)

- MAF (Minor Allele Frequency)

2. **D2:** Comparing one of the GWAS summary statistics files (created in D1) with the annotation file (Annot_BCAC_2020_onco_ALL_reformatted_coding.txt) and adding the gene column annotated to each SNP.

3. **D3:** Merging the Information from the two GWAS summary statistics files (created in the D1) and the file created at the end of D2.

4. **D4:** Selection of the SNPs (from the two GWAS summary statistics files created in the D3) which are ONLY available in the LD file (output_ld_clumping_08_BCAC_2020_ALL_OCAC.txt)

5. **D5:** Reading two GWAS files (created in D4) and splitting rows in which more than one gene assigned to a SNP.
   Then, removing other non-gene information (such as missense, nonsense, frameshift, and splice) from the gene column.
   After that, the scripts adds some extra information about each gene (such as chromosome, start base-pair position, end base-pair position, gene length, and gene class) obtained from the Human gene annotation file (annot_gencode_v38lift37_modified_gene_class_coding.txt).
   Finally, it creates two files (begins with Matrices_... and Matrices_extra_info_... and are in ".*Rdata*" formats) which could be easily read by the GCPBayes package for exploring pleiotropy between GWAS data of two traits at a gene-level.

5. Running GCPBayes (Section E)

5.1. Genes with number of SNPs less than a threshold

(**Required program:** R)
(**Script:** E1_code_gcpbayes_less_extra_info.R)

For running the GCPBayes package, two input files are needed which were created by the pipeline (with or without LD Clumping step) (see Section 4):

1. Matrices_....Rdata (This file contains list of beta and standard error values of all SNPs related to each gene)
2. Matrices_extra_info_....Rdata (This file contains some information about each gene such as chromosome, start base-pair position, end base-pair position, gene length, and gene class)

NOTE: A user should change the "**DEFINITION SECTION**" of the script based on file names and paths of input and output files. In addition, there are also two threshold values (for *SNP_number_threshold* and *theta_threshold*). This script ONLY run GCPBayes for genes with number of SNPs less than *SNP_number_threshold* value (Default = 500). Also, *theta_threshold* value is used for detecting genes with potential pleiotropic signals (Default = 0.5). Obviously, a user could modify these two values in this section.

RUNNING SECTION: As it is mentioned before, this script ONLY run GCPBayes for genes with number of SNPs less than *SNP_number_threshold*. It uses "DS Method" for running (Baghfalaki *et al.*, 2021) with appropriate parameters that set based on our previous study (Baghfalaki *et*

al., 2021). Clearly, a user could change the parameters in the script ("*Launch of GCPBayes function*" section).

OUTPUT: The outputs of the GCPBayes script would be three files*:

1. *results*: The results of the GCPBayes method for all genes. The file includes the following columns:
    - gene_index
    - log10BF
    - lBFDR
    - theta
    - PPA1
    - PPA2
    - gene name
    - SNP_numbers (total number of SNPs in the gene)
    - run_time (in seconds)
    - chr (chromosome)
    - start (start position base-pair of the gene)
    - end (end position base-pair of the gene)
    - gene_type (class of the gene)
    - gene_length (in bp)
    - gene_length_ratio
2. *pleiotropy*: The results of the GCPBayes method for the genes with theta value greater than *theta_threshold*. The file includes the following columns
    - Gene Index
    - Gene Name
    - #SNPs (total number of SNPs in the gene)
    - Theta
    - Pleiotropic Effect based on CI
    - Pleiotropic Effect based on Median
    - chr (chromosome)
    - start (start position base-pair of the gene)
    - end (end position base-pair of the gene)
    - gene type (class of the gene)
    - gene length
    - snp number gene length ratio
    - running time (in seconds)
    - SNP_ID (SNPs RS IDs that were significant in the gene)
    - Study1 (if the SNP was significant in the first trait)
    - Study2 (if the SNP was significant in the second trait)
    - Total (if the SNP was significant in both traits)
3. *errors*: This file includes all indices of the genes in which the GCPBayes method got an error.

* If there was no gene with error during running the GCPBayes function, the "errors" file would NOT be created and there will be just two output files.

5.2. Genes with number of SNPs more than a threshold

(**Required program:** R)

(**Script:** E2_code_gcpbayes_greater_extra_info.R)

All procedures and explanations are similar to the previous section, except here JUST genes which their number of SNPs are greater than *SNP_number_threshold* will be considered for running the GCPBayes method.

6. Visualization Scripts

6.1. Checking GWAS summary statistics input data (Section A)

(**Required program:** R)
(**Script:** code_analysis_A_checksumstats_BCAC.R)

INPUT FILE: BCAC_2020_onco_ALL_reformatted.txt (created from Section 1)

The script uses a R package recently introduced (called "*CheckSumStats*") to extracts some SNPs from the BCAC GWAS summary statistics input file and compare them with GWAS database (such as GWAS catalog and 1000 Genome). Here is the steps available in the script:

i. Extracting SNPs list based on EFO ID of Breast Carcinoma
ii. Checking allele frequency (NOTE: We also checked an output if a user selects the "effect" and "non-effect" alleles columns WRONG!)
iii. Checking the effect allele by comparing Z scores in the test and GWAS catalog datasets
iv. Checking the effect allele by comparing effect allele frequency (EAF) between the test dataset and the GWAS catalog
v. Checking for errors or analytical issues in the summary data
vi. Comparing the expected and reported effect sizes
vii. Plotting the relative bias, i.e. the percentage deviation of the expected from the reported effect size
viii. Checking whether the reported P values correspond to the reported effect sizes in the dataset
ix. Saving all outputs in one file

The output derived after running the script on BCAC GWAS data (created from Section 1) is shown in Fig. 1.

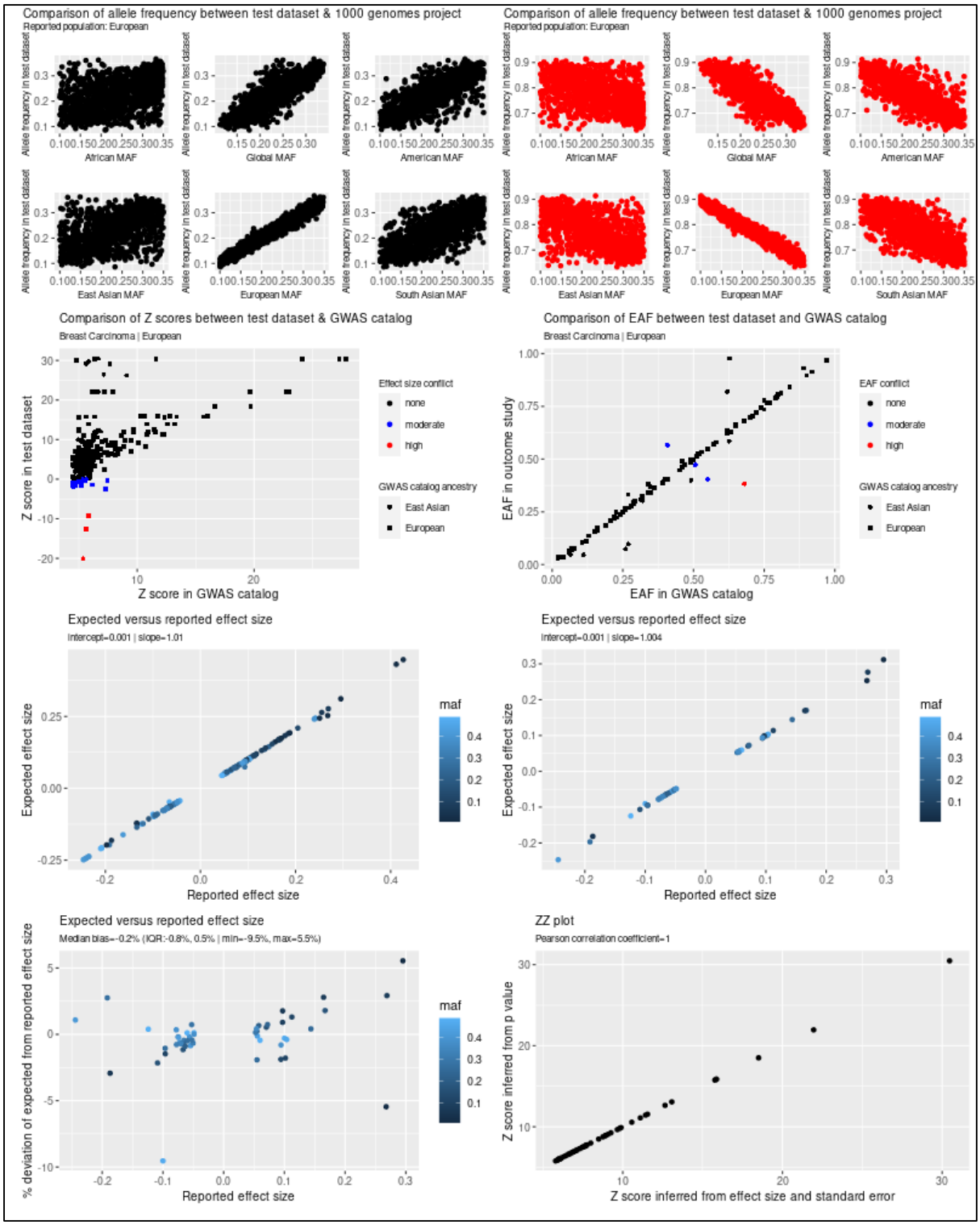

Fig. 1: output of using "*CheckSumStats*" package for checking BCAC GWAS data (created from Section 1). **NOTE:** The second plot including all data in red which is shown a situation if a user selects the "effect" and "non-effect" alleles columns WRONG.

Also, we performed similar checking on OCAC data as well:

(**Required program:** R)
(**Script:** code_analysis_A_checksumstats_OCAC.R)

INPUT FILE: OCAC_BCAC_2020_onco_ALL_reformatted.txt (created from Section 1)

Here is the steps available in the script:

i. Extracting SNPs list based on EFO ID of Ovarian Carcinoma
ii. Checking allele frequency
iii. Checking the effect allele by comparing Z scores in the test and GWAS catalog datasets
iv. Checking the effect allele by comparing effect allele frequency (EAF) between the test dataset and the GWAS catalog
v. Checking for errors or analytical issues in the summary data
vi. Comparing the expected and reported effect sizes
vii. Plotting the relative bias, i.e. the percentage deviation of the expected from the reported effect size
viii. Checking whether the reported P values correspond to the reported effect sizes in the dataset
ix. Saving all outputs in one file

The output derived after running the script on OCAC GWAS data (created from Section 1) is shown in Fig. 2.

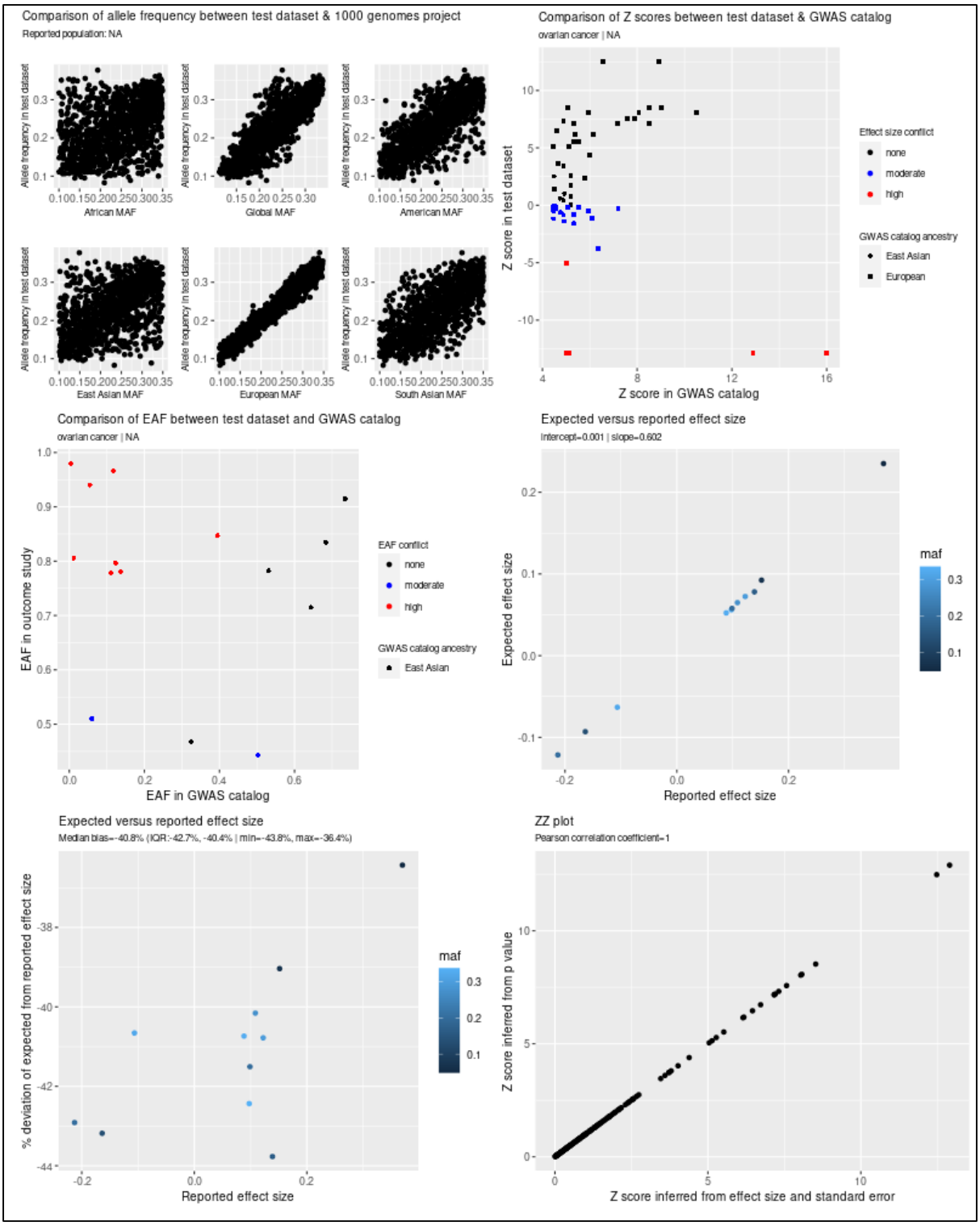

Fig. 2: Output of using "*CheckSumStats*" package for checking OCAC GWAS data (created from Section 1).

6.2. Pie chart of the Annotation file (Section B)

INPUT FILE: annot_gencode_v38lift37_modified_gene_class_coding.txt

A user could have an overview from the annotation file created in the Section B. There are two possibilities for the visualization of the file:

1. Running "code_analysis_B_annotation_file_coding.R" script using RStudio

   After running, an interactive pie chart would be created in which a user could find various information regarding each part of the plot (total number of genes, percentage, etc.) by moving mouse cursor on the section (Fig. 3-A). In addition, by clicking on every items of the legend, it is possible to show/hide that part on the pie chart (Fig. 3-B).

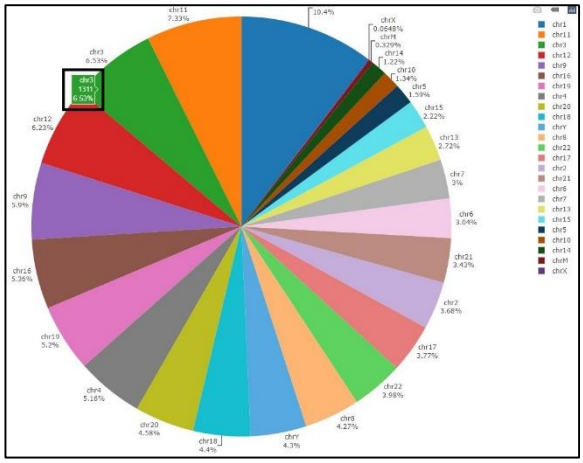

Fig. 3-A: Example of a pie chart of the annotation file used in our study as an interactive plot which shows a section details by moving a mouse cursor on it (for instance, the black box showed in the figure). Each chromosome is shown with a specific color.

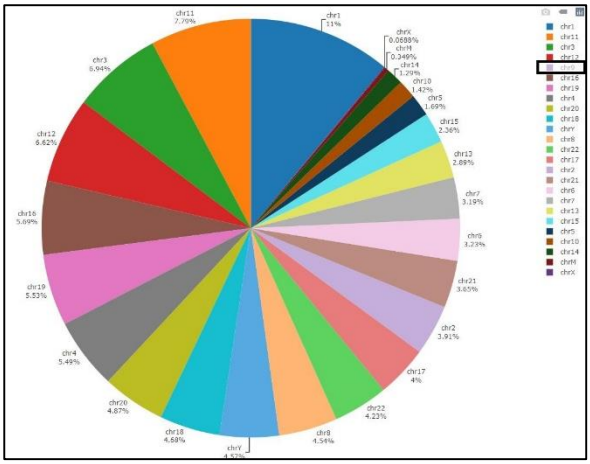

Fig. 3-B: Example of a pie chart of the annotation file used in our study as an interactive plot which hide "chromosome 9" by clicking on its legend (the black box in the figure). Each chromosome is shown with a specific color.

2. Running "code_analysis_B_annotation_file_coding.R" script using R command line (in UNIX or other Operating Systems (OS))

   A pie chart including an overview of genes distribution (here just coding-genes) among all chromosomes would be created in ".jpeg" and ".pdf" format.

   NOTE: since "plotly" package creates an interactive plot, for saving such plots in static file formats (like ".jpeg" or ".pdf"), a user needs to install "orca" on the OS (see https://github.com/plotly/orca#installation for more details).

   An example of a pie chart created from the protein-coding genes annotation file ("annot_gencode_v38lift37_modified_gene_class_coding.txt") is shown in Fig. 4.

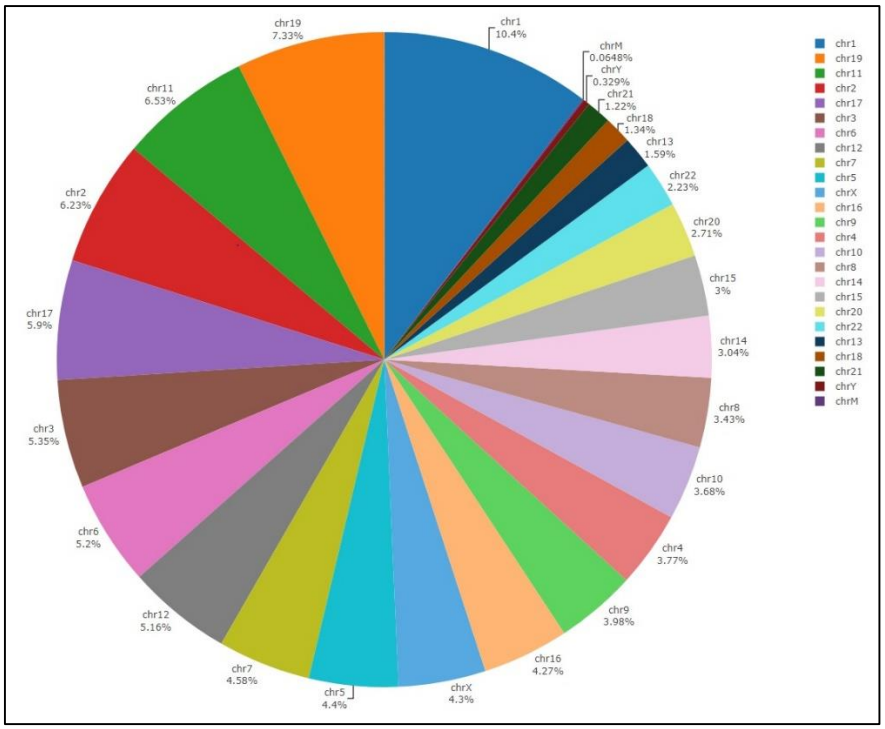

Fig. 4: Example of a pie chart of the annotation file used in our study as a static plot saved in ".jpeg" file format. Each chromosome is shown with a specific color.

6.3. Analysis of PLACO result (Section C)

INPUT FILE: output_PLACO_BCAC_2020_ALL_OCAC.txt

A user could run the "code_analysis_C_PLACO_results_one_pair.R" script which perform three analyses as follow:

1. Performing a global correlation analysis between two traits (using Z columns) based on two methods (spearman and pearson) and save the results in "…_cor_test.txt" file.
2. Finding significant SNPs from the PLACO result ($p < 5\times10^{-8}$) and save the results in "output_PLACO_BCAC_2020_ALL_OCAC_Sig.txt" file.
3. Creation of a Manhattan Plot for all chromosomes and drawing a horizontal dashed line in black (indicating p-value threshold 5×10^{-8}). In addition, SNPs with positive effect are colored in red while SNPs with negative effect are in blue. The plot would be saved as a ".png" file format. An example of the Manhattan Plot for PLACO output for BCAC and OCAC GWAS data (used in our study) is shown in Fig. 5.

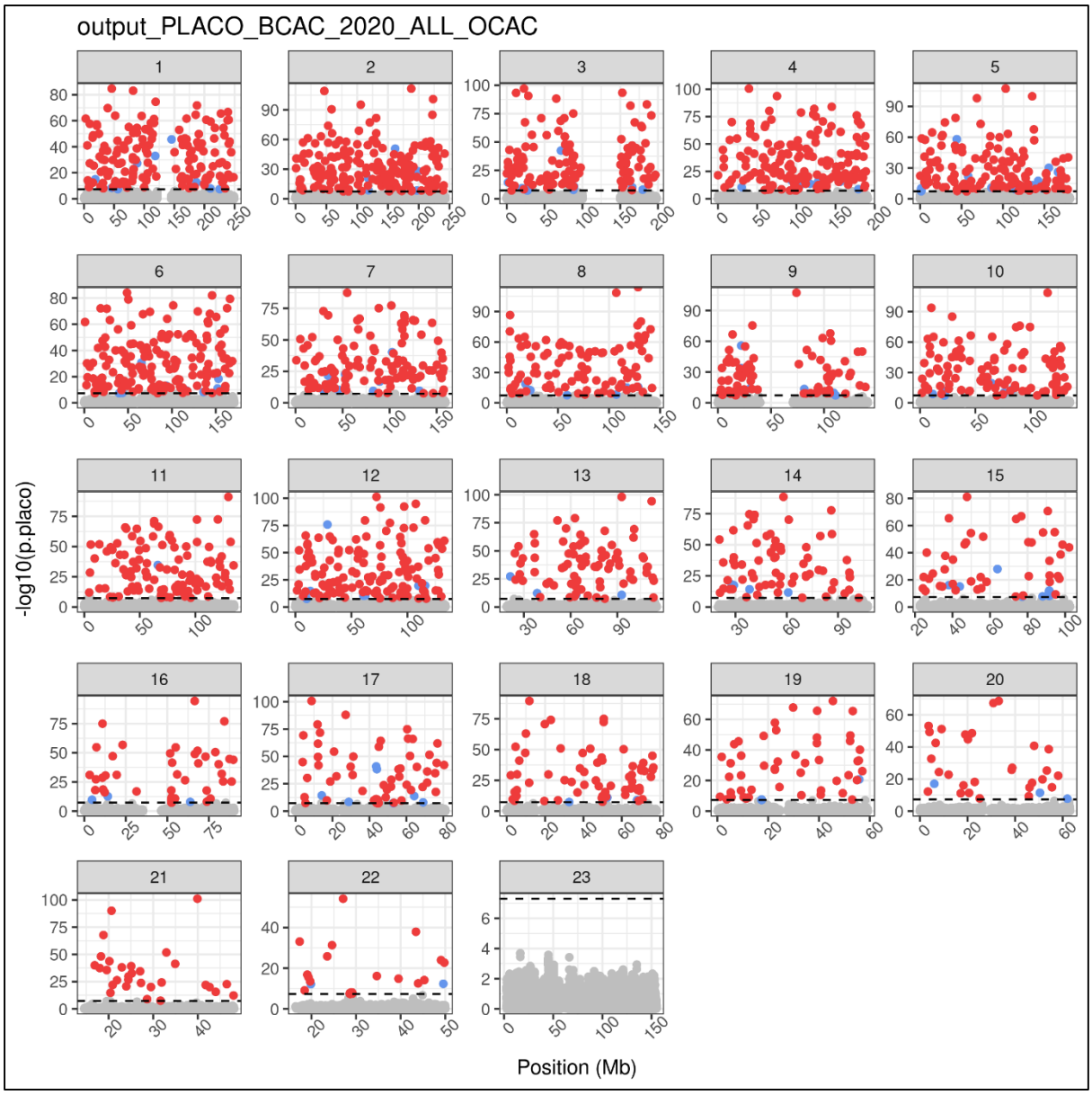

Fig. 5: Manhattan Plot for PLACO output for BCAC and OCAC GWAS data used in our study. Each box represents results for a specific chromosome. The horizontal dashed black line indicates p-value threshold 5×10^{-8}. SNPs with positive effect are colored in red while SNPs with negative effect are in blue.

6.4. Analysis of GCPBayes input file (Section D)

INPUT FILE: Matrices_extra_info_output_pipeline_BCAC_ALL_OCAC_coding_withoutclumping.Rdata

A user could run the "code_analysis_D_gcpbayes_input_ggplot.R" script and obtain an overview about distribution of genes with different number of SNPs. We designed two different overviews for a user as follow:

1. A histogram (based on number of SNPs available for each gene) for all genes. For example, the histogram for GCPBayes input file for BCAC and OCAC GWAS data (used in our study) while the pipeline run without LD clumping step is shown in Fig. 6.

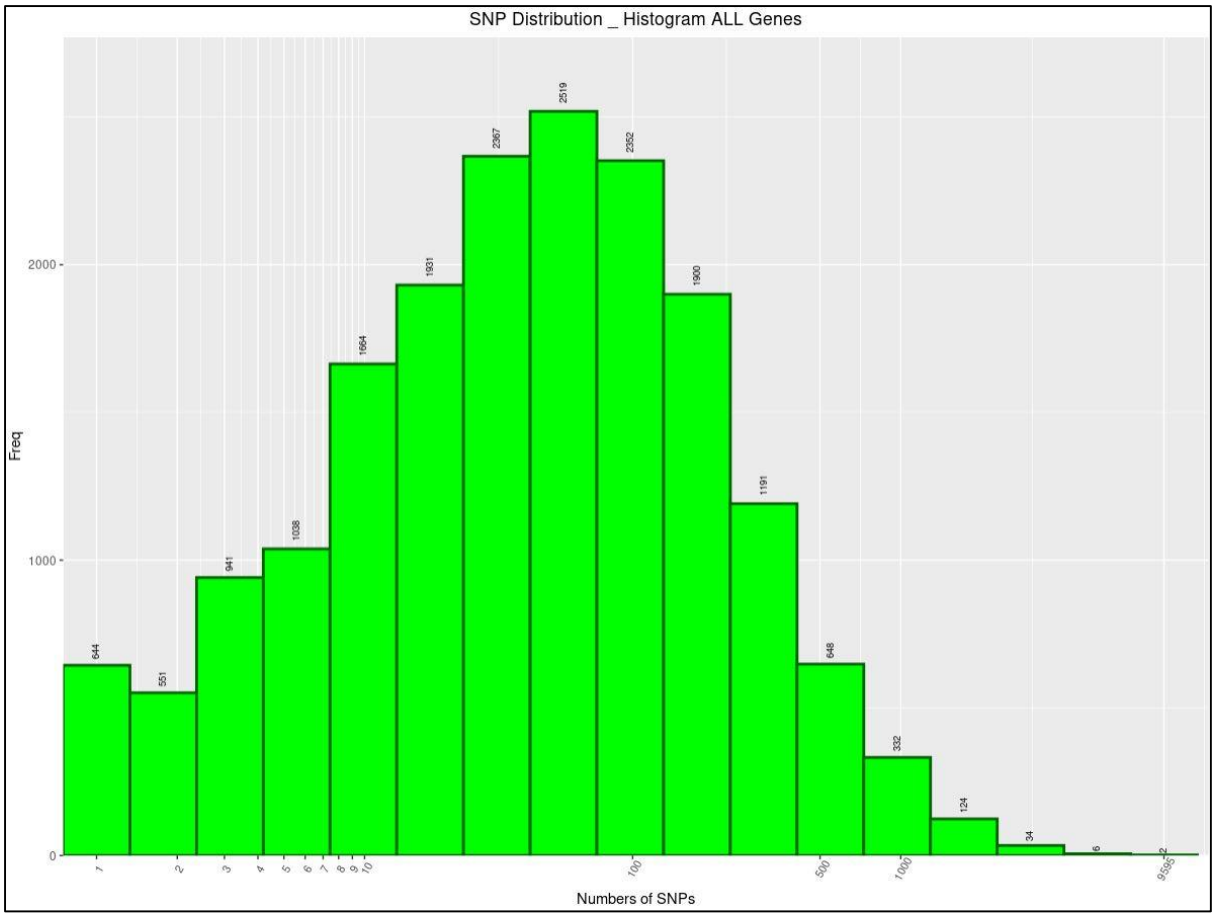

Fig. 6: Histogram for GCPBayes input file for BCAC and OCAC GWAS data. X-axis represents bins based on different number of SNPs and Y-axis is their frequencies. The bars labels represents total number of genes related to their corresponding bins.

a. Based on our experience, in a GWAS data, most of the genes contain a number of SNPs less than 50 (due to our practical experience with various GWAS datasets, especially when working with only protein-coding genes). Therefore, in order to have a deeper overview, a user could define a threshold value (Default = 50) to have a plot with a more detailed view (bar plot) to see the number of genes with SNPs less than the threshold value, as well as a histogram for genes in which the number of SNPs are more than the threshold value. For instance, two plots for GCPBayes input file for BCAC and OCAC GWAS data (used in our study) while the pipeline run without LD clumping step are shown in Fig. 7.

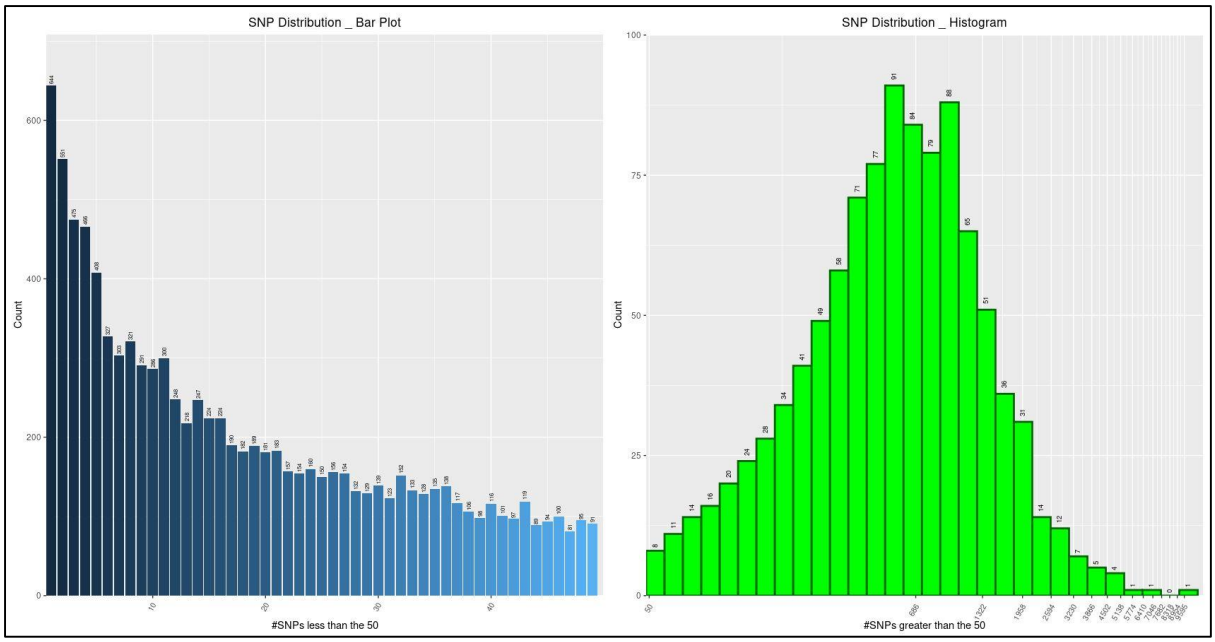

Fig. 7: division of the previous figure (Fig. 4) into two plots based on a threshold value (here = 50).
Left graph is a bar plot for genes which contain total number of SNPs less than 50. X-axis represents different number of SNPs and Y-axis is their frequencies. The bars labels represents total number of genes related to their corresponding number of SNPs.
Right graph is a histogram for genes which contain total number of SNPs equal to or greater than 50. X-axis represents bins based on different number of SNPs and Y-axis is their frequencies. The bars labels represents total number of genes related to their corresponding bins.

6.5. Analysis of GCPBayes output file (Section E)

6.5.1. Pleiotropic genes karyotype plot

INPUT FILE: output_GCPBayes_BCAC_All_OCAC_coding_....._threshold_500_pleiotropy.txt

A user could run the "code_analysis_E_gcpbayes_output_karyotype.R" script which reads the GCPBayes output contains pleiotropic genes information, then show them based on their positions on a chromosome overview (called Karyotype plot). Fig. 8 demonstrates a Karyotype plot for GCPBayes output of BCAC and OCAC GWAS data (used in our study) while the pipeline run without LD clumping step.

NOTE: Since for drawing a Karyotype plot, the script uses the HUGO Gene Nomenclature Committee (HGNC) symbol (for each input gene symbol) extracted from Ensembl database, it would NOT be able to show the genes that their HGNC symbol did not find.

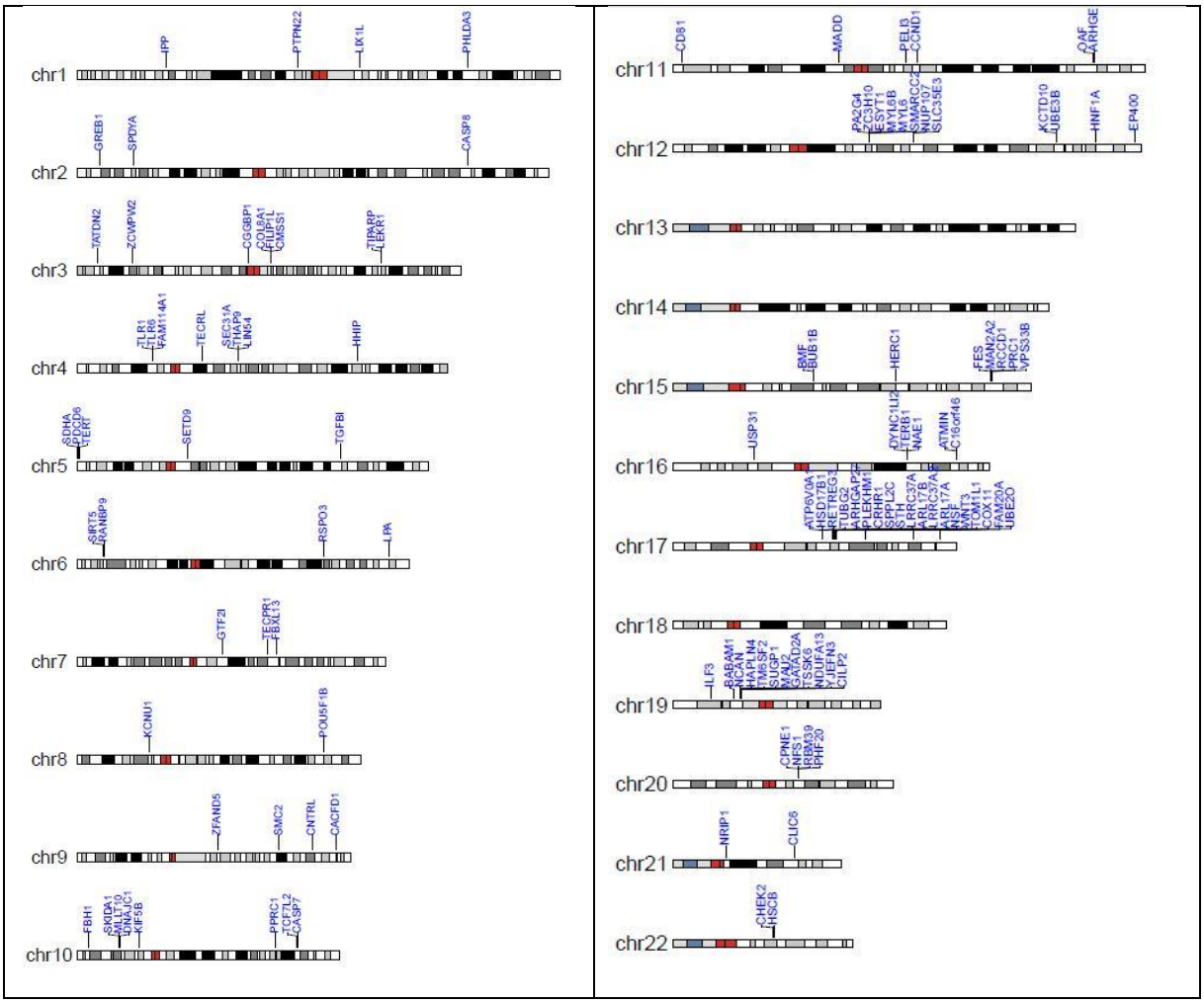

Fig. 8: Example of a Karyotype plot for GCPBayes output of BCAC and OCAC GWAS data derived without performing LD clumping step

6.5.2. Pleiotropic genes table overview

INPUT FILE: output_GCPBayes_BCAC_All_OCAC_coding_....._threshold_500_pleiotropy.txt

There is also a script called "code_analysis_E_gcpbayes_output_table_overview.R" which a user could perform on the GCPBayes pleiotropic genes output and has an overall information about potential pleiotropic genes (something similar to Table 1 in the manuscript).

7. References

Auton,A. *et al.* (2015) A global reference for human genetic variation. *Nature*, **526**, 68.

Baghfalaki,T. *et al.* (2021) Bayesian meta-analysis models for cross cancer genomic investigation of pleiotropic effects using group structure. *Stat. Med.*, **40**, 1498–1518.

Frankish,A. *et al.* (2019) GENCODE reference annotation for the human and mouse genomes. *Nucleic Acids Res.*, **47**, D766–D773.

Phelan,C.M. *et al.* (2017) Identification of 12 new susceptibility loci for different histotypes of epithelial ovarian cancer. *Nat. Genet.*, **49**, 680–691.

Ray,D. and Chatterjee,N. (2020) A powerful method for pleiotropic analysis under composite null hypothesis identifies novel shared loci between Type 2 Diabetes and Prostate Cancer. *PLOS Genet.*, **16**, e1009218.

Zhang,H. *et al.* (2020) Genome-wide association study identifies 32 novel breast cancer susceptibility loci from overall and subtype-specific analyses. *Nat. Genet. 2020 526*, **52**, 572–581.